Решения заданий заключительного тура олимпиады школьников «Гранит науки» по профилю Информатика в 2021/2022 учебном году

ИНФОРМАТИКА

ВАРИАНТ 1.

1. С тех пор, как Иван изучил формулы в Excel, он ежедневно оттачивает свое мастерство. Сегодня он напечатал некоторое целое число M в ячейку A1, в соседнюю ячейку B1 он внес формулу =ОКРУГЛ(ОСТАТ(8;$A$1)+СТЕПЕНЬ(A1;2)-ЕСЛИ(A1>0;A1^2/3;A1/3);1) и растянул до ячейки E1. В соседней ячейке F1, он ввел формулу: =СЧЁТЕСЛИ(A1:E1;">20")+СУММЕСЛИ(A1:E1;">3";A1:E1). На следующей строке, начиная с ячейки A2, мальчик ввел формулу =ОКРУГЛ(ОСТАТ(B1;A1)-КОРЕНЬ(ABS(A1-B1))+ЕСЛИ(A1+B1<2;B1/A1;A1+3);2) и скопировал в диапазон B2:E2. В ячейку F2 он скопировал формулу из ячейки F1. На следующей строке он ввел число 2,5 в ячейку A3 и повторил действия, как для первой строки. А в четвертую (диапазон A4:E4) скопировал формулы со 2 строки, в ячейку F4 была вставлена формула из ячейки F3. В пятой строке Иван ввел некоторое число N в ячейку A5 и решил повторить действия, как для строки 1. В ячейку A6 он скопировал формулу из A2 и растянул с помощью автозаполнения до ячейки E6. В ячейку F6 была скопирована формула из F5. При этом мальчик получил следующий результат:

	A	B	C	D	E	F
1	?	2,7	4,9	16	170,7	192,6
2	4,86	6,42	5,87	17,26	190,92	226,33
3	2,5	4,2	11,8	92,8	5741,2	5852
4	5,9	7,84	16	101,04	5844,47	5977,25
5	?	6	24	384	98304	98721
6	4,27	4,76	8,03	74,08	98703,58	98796,72

Определите наименьшие целые числа M и N? Перечислите числа через точку с запятой. **(10 баллов)**

Решение:

	A	B	C	D	E	F
1	2	2,7	4,9	16	170,7	192,6
2	4,86	6,42	5,87	17,26	190,92	226,33
3	2,5	4,2	11,8	92,8	5741,2	5852
4	5,9	7,84	16	101,04	5844,47	5977,25
5	3	6	24	384	98304	98721
6	4,27	4,76	8,03	74,08	98703,58	98796,72

Ответ: 2; 3.

2. Два сладкоежки решили сыграть в игру «35 шоколадных конфет». Играющие по очереди могут взять от одной до четырёх конфет. Кто не может сделать ход (конфет не осталось), проигрывает. Другими словами, выигрывает взявший последнюю конфету. Кто выигрывает при безошибочной игре – игрок, делающий первый ход, или игрок, делающий второй ход? Поясните алгоритм графически. Напишите обобщенный алгоритм для любого количества конфет. **(10 баллов)**

Решение:

Используется Стратегия_Дополни до пяти. Всегда выигрывает Игрок_2 (игрок, делающий ход вторым). Суть Стратегия_Дополни до пяти:

Ход_1: Игрок_1 (игрок делающий ход первым) берет от 1 до 4 конфет (в данном случае конфет остаётся от 31 до 34).

Ход_2: Игрок_2 (игрок, делающий ход вторым) должен дополнять ход первого до пяти конфет (то есть, если Игрок_1 взял одну конфету, Игрок_2 должен взять 4 конфеты. Конфет в вазочке должно остаться 30).

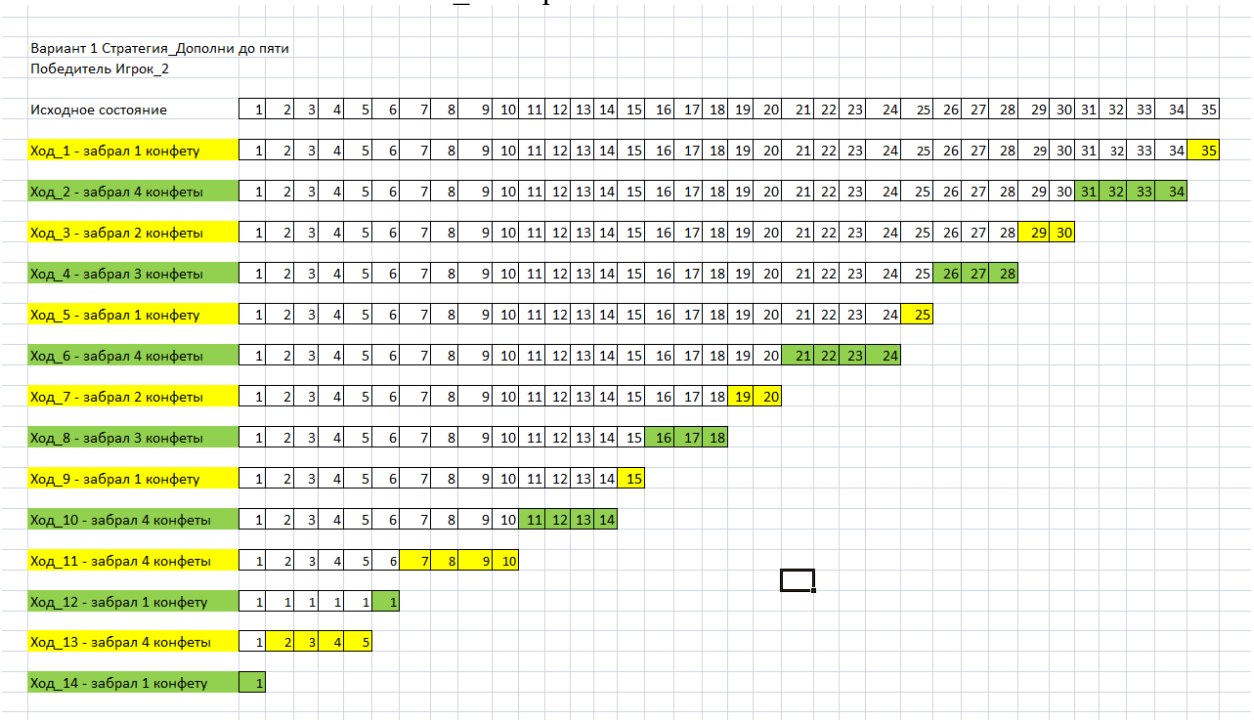
Ход_3: Игрок_1 берет от 1 до 4 конфет (в данном случае конфет остаётся от 26 до 29).

Ход_4: Игрок_2 должен дополнять ход первого до пяти конфет (Конфет в вазочке должно остаться 25). И так далее до Победа Игрок_2. Проиграть невозможно.

Иллюстрация на рисунке ниже. Алгоритм подходит в общем случае, для N конфет., если N делится на 5 без остатка, то второй может гарантировать себе выигрыш, дополняя ход противника до 5.

Алгоритм для Игрок_2 (обобщенный):

Шаг 1 Игрок _1 взял k конфет
Шаг 2 Игорок_2 взял (5-k) конфет
Если Остаток = 0 – значит СТОП_Выиграл.



3. Выражение $x \wedge y \oplus z \to z \wedge x \leftrightarrow y$ =*истина*. Перечислите выражения (выражение), удовлетворяющие этому ответу и тому же набору значений *x, y, z* из приведенных ниже:

1). $x \to \bar{y} \vee z \leftrightarrow x \vee y \oplus z$

2). $y \leftrightarrow x \vee y \wedge \bar{x} \oplus z$

3). $x \to \bar{y} \wedge z \leftrightarrow y \vee \bar{x} \oplus z$

4). $\bar{z} \vee y \leftrightarrow x \vee z \rightarrow y \wedge \bar{x} \oplus y$

5). $x \rightarrow y \vee z \oplus x \leftrightarrow \bar{y} \oplus z$

6). $y \vee x \leftrightarrow z \oplus x \rightarrow y \wedge x$

В случае, если выражений несколько перечислите их в ответе через запятую. В процессе решения необходимо расставить порядок действий для всех выражений, а также построить таблицы истинности. При расстановке действий использовать в первую очередь приоритет операций, а потом уже очередность появления. **(10 баллов)**

Решение:

1. На первом шаге необходимо определить табличные значения для основного примера, для каких наборов соответствует *истина*. Для этого необходимо расставить приоритет действий и построить таблицу истинности.
2. Расстановка действий:
   1 – конъюнкция *x*,*y*
   2 – конъюнкций *z*,*x*
   3 – исключающее или первого действия и *z*
   4 – имликация третьего действия из второго
   5 – равносильность 4 действия и *y*.
3. Построение таблицы истинности:

X	***y***	***z***	**1**	**2**	**3**	**4**	**5**
0	0	0	0	0	0	1	0
0	0	1	0	0	1	0	1
0	1	0	0	0	0	1	1
0	1	1	0	0	1	0	0
1	0	0	0	0	0	1	0
1	0	1	0	1	1	1	0
1	1	0	1	0	1	0	0
1	1	1	1	1	0	1	1

Исходя из полученных данных, значения *истина* получены для следующих наборов:

X	***y***	***z***
0	0	1
0	1	0
1	1	1

Эти значения будут подставлены в каждый из предлагаемых примеров для сравнения результатов.

Исследование 1 примера начинается с расстановки действий:

1). $x \rightarrow \bar{y} \vee z \leftrightarrow x \vee y \oplus z$

1 – отрицание *y*

2 – дизъюнкция первого действия и *z*

3 – дизъюнкция *x*, *y*

4 – исключающее или 3 действия и *z*

5 – импликация *x* из 2 действия

6 – равносильность результатов 5 и 4 действий.

Следующий шаг – построение таблицы истинности:

x	***y***	***z***	***1***	***2***	***3***	***4***	***5***	***6***
0	0	1	1	1	0	1	1	1
0	1	0	0	0	1	1	1	1
1	1	1	0	1	1	0	1	0

Пример не подходит.

Исследование примера 2 с расстановкой действий:

2). $y \leftrightarrow x \vee y \wedge \bar{x} \oplus z$

1 – отрицание *x*

2 – конъюнкция *y* и первого действия

3 – дизъюнкция *x* и результатов второго действия

4 – исключающее или 3 действия и *z*

5 – равносильность *y* и 4 действия

x	***Y***	***z***	***1***	***2***	***3***	***4***	***5***
0	0	1	1	0	0	1	0
0	1	0	1	1	1	1	1
1	1	1	0	0	1	0	0

Пример не подходит.

Исследование примера 3 с расстановкой действий:

3). $x \rightarrow \bar{y} \wedge z \leftrightarrow y \vee \bar{x} \oplus z$

1 – отрицание *y*

2 – отрицание *x*

3 – конъюнкция 1 действия и *z*

4 – дизъюнкций *y* и 2 действия

5 – исключающее или 4 действия и *z*

6 – импликация *x* и 3 действия

7 – равносильность 6 и 5 действий

x	***Y***	***z***	***1***	***2***	***3***	***4***	***5***	***6***	***7***
0	0	1	1	1	1	1	0	1	0
0	1	0	0	1	0	1	1	1	1
1	1	1	0	0	0	1	0	0	1

Пример не подходит.

Исследование 4 примера с расстановкой действий:

4). $\bar{z} \vee y \leftrightarrow x \vee z \rightarrow y \wedge \bar{x} \oplus y$

1 – отрицание *z*

2 – отрицание *x*

3 – конъюнкция y и результата 2 действия

4 – дизъюнкция первого действия и *y*

5 – дизъюнкция *x, z*

6 – исключающее или 2 действия и *y*

7 – импликация 5 действия из 6

8 – равносильность 4 действия и 7

x	***y***	***z***	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***
0	0	1	0	1	0	0	1	0	0	1
0	1	0	1	1	1	1	0	0	1	1
1	1	1	0	0	0	1	1	1	1	1

Пример подходит.

Исследование 5 примера с расстановкой действий:

5). $x \rightarrow y \vee z \oplus x \leftrightarrow \bar{y} \oplus z$

1 – отрицание *y*

2 – дизъюнкция *y, z*

3 – исключающее или 1 действия и *z*

4 – исключающее или результатов 2 действия и *x*

5 – импликация x и результатов 4 действия

6 – равносильность результатов 5 и 4 действий

x	***y***	***z***	***1***	***2***	***3***	***4***	***5***	***6***
0	0	1	1	1	0	1	1	0
0	1	0	0	1	0	1	1	0
1	1	1	0	1	1	0	0	0

Пример не подходит.

Исследование 6 примера с расстановкой действий:

6). $y \vee x \leftrightarrow z \oplus x \rightarrow y \wedge x$

1 – конъюнкция *y,x*

2 – дизъюнкция *y,x*

3 – исключающее или *z, x*

4 – импликация результатов 3 действия из 1

5 – равносильность 2 действия и 4

x	*Y*	*Z*	*1*	*2*	*3*	*4*	*5*
0	0	1	0	0	1	0	1
0	1	0	0	1	0	1	1
1	1	1	1	1	0	1	1

Пример подходит.

Ответ: 4, 6.

4. Максим устроился работать помощником сетевого администратора в IT-отдел Университета. В первый же день работы он получил ответственное задание. Отдел информационных технологий Университета использует диапазон частных IP-адресов, в котором настроены 6 сетей филиалов. Данные по распределению адресного пространства частично известны: в каждой сети компьютеру системного администратора назначен адрес 172.28.228.145, а размер сетевого префикса указан в таблице. При этом адресное пространство использовано максимально эффективно.

Максим обладает следующими знаниями:

- одному устройству должен соответствовать один IP-адрес вида XXX.XXX.XXX.XXX, при этом XXX – число в диапазоне [0:255], называемое октетом;
- устройства внутри одной сети должны беспрепятственно взаимодействовать друг с другом, взаимодействие устройств в разных сетях реализовано провайдером с помощью механизмов преобразования сетевых адресов;
- выделенный размер сети содержит минимально необходимое количество адресов;
- выделенный размер сети записывается как 2^n-2, где n – минимальное количество бит, которое может содержать все номера адресов сети;
- в каждом сегменте сети первый адрес зарезервирован под идентификатор сети, а последний – под широковещательную рассылку.

Название подсети	**IP-адрес ПК сетевого администратор**	**Адрес сети**	**Десятичная маска**	**Диапазон доступных адресов**	**Выделенный размер**	**Широковеща-тельный адрес**
Главный корпус	172.28.228.145/18					
Инженерный корпус	172.28.228.145/21					
Учебный центр №2	172.28.228.145/23					
Библиотека	172.28.228.145/25					
Пост охраны	172.28.228.145/29					
IT-отдел	172.28.228.145/30	172.28.228.144	255.255.255.252	172.28.228.145-172.28.228.146	2	172.28.228.147

Максим смог заполнить данные для IT-отдела. Заполнение остальной информации вызвало у помощника сетевого администратора затруднения. Необходимо помочь Максиму заполнить таблицу недостающими данными. **(20 баллов)**

Решение:

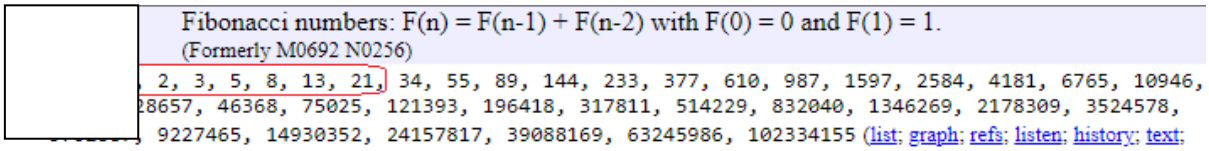
Название подсети	**IP-адрес ПК сетевого администратора**	**Адрес сети**	**Десятичная маска**	**Диапазон доступных адресов**	**Выделенный размер**	**Широковеща-тельный адрес**
Главный корпус	172.28.228.145/18	**172.28.**192.0	255.255.192.0	.192.1-.255.254	16 382	.255.255
Инженерный корпус	172.28.228.145/21	**172.28.**224.0	255.255.248.0	.224.1-.231.254	2 046	.231.255
Учебный центр №2	172.28.228.145/23	**172.28.**228.0	255.255.254.0	.228.1-.229.254	510	.229.255
Библиотека	172.28.228.145/25	**172.28.228.**128	255.255.255.128	.129-.254	126	.255
Пост охраны	172.28.228.145/29	**172.28.228**.144	255.255.255.248	.145-.150	6	.151
IT-отдел	**172.28.228.**145/30	**172.28.228**.144	255.255.255.252	**172.28.228.**145-**172.28.228.**146	2	**172.28.228.**147

*полужирным выделена общая часть адреса до точки (по аналогии с последней строкой таблицы)

5. Гном носит драгоценные камни в плетеной корзинке наверх из шахты в кладовую. В кладовую из шахты ведет лестница из 7 ступенек. Шаг у гнома небольшой, поэтому у корзины две ручки, и он может наступать на каждую ступеньку, или перескакивать через одну ступеньку и становиться сразу на следующую. Каждый раз гном поднимается по лестнице по-новому, шагая по одной, по две ступеньки, или где по одной, а где по две. Сколько раз гном сможет подняться по лестнице из 7 ступенек, ни разу не повторив последовательность шагов? Напишите алгоритм и программу, позволяющую вычислить, сколько раз гном сможет подняться по лестнице из произвольного целого N (N>2) количества ступенек (в пределах типа данных языка программирования), чтобы ни разу не повторить последовательность шагов. Учтите, что возможности компьютера, на котором будет выполняться программа, ограничены: допустимо использовать до 10 переменных, массивы ограничены 10 ячейками соответствующего типа, а стек вмещает 10 вызовов и не расширяется. Алгоритм должен раскрывать решение на каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте уникальные особенности языка программирования (классы, методы, объекты, попарный обмен между переменными, специальные функции и библиотеки и др., кроме счетчиков циклов) для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для 7 ступенек, раскрыв содержимое переменных на каждом шаге.**(25 баллов)**

Решение:

Алгоритм решения задачи: задача на вычисление последовательности чисел Фибоначчи.


Fibonacci numbers: F(n) = F(n-1) + F(n-2) with F(0) = 0 and F(1) = 1.
(Formerly M0692 N0256)
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946,
28657, 46368, 75025, 121393, 196418, 317811, 514229, 832040, 1346269, 2178309, 3524578,
9227465, 14930352, 24157817, 39088169, 63245986, 102334155 (list; graph; refs; listen; history; text;
https://oeis.org/A000045

В задаче требуется рассчитать число уникальных комбинаций шагов, которыми можно подняться по лестнице из **7** ступенек, используя шаги в 1, 2 или 1 и 2 ступеньки. **Ответом является 7-е** по счету число последовательности Фибоначчи (**a_7 = 21**), считая, что **a_1 = 1**, **a_2 = 2**.

Объяснение 1. Обозначим число комбинаций как a_n, где n – число ступенек, n>2.
a_1 = 1 способ –> на лестницу из одной ступеньки можно подняться только одним способом - шагом в 1 ступеньку.
a_2 = 2 способа –> 1+1 ступенька; сразу 2 ступеньки.
a_3 = 3 –> 1+1+1; 1+2; 2+1 ступеньки.
a_4 = 5 –> 1+1+1+1; 1+1+2; 1+2+1; 2+1+1; 2+2 ступеньки.
$a_4 = a_3 + a_2$; $a_3 = a_2 + a_1$; ==> **$a_n = a_{n-1} + a_{n-2}$;**
$a_5 = a_4 + a_3 = 5+3=8$; $a_6 = a_5 + a_4 = 8+5=13$; $a_7 = a_6 + a_5 = 13+8=21$.
Ответ: a_7 = 21 способ.
Объяснение 2. Обозначим число комбинаций как a_n, где n – число ступенек, n>2. Можно начать подъем с шага в 1 ступеньку. Тогда останется a_{n-1} комбинаций подъема по лестнице. Если начать подъем с шага в 2 ступеньки, то останется a_{n-2} комбинаций подъема по лестнице. Первый шаг входит в любую из комбинаций шагов. Таким образом количество комбинаций определяется суммой количества оставшихся ступенек для каждого варианта первого шага и размером первого шага, т.е. **$a_n = a_{n-1} + a_{n-2}$.**

Задачу можно решить, например *рекурсией* (*неправильно*, ограничение по стеку вызовов при больших N), **динамическим программированием (ожидаемое решение)**, с помощью умножения матриц (допустимое, но избыточное решение), формулы Бине (допустимое, но избыточное решение). Ниже приведен пример решения с помощью динамического программирования на Python 3.82 и C.

```
    ция Фибоначи с обменом
    омит память, хранит только два
    ыдущих значения
ib_ult(n):#Для вывода по шагам
    #Значение функции на два и один шаг назад
    back2=1; back1=2
    if n<2:
        print("Ступенек не может быть меньше 2-х!")
        return 0
    for i in range(2,n-1):
        next=back1+back2
        print("Шаг ",i-1," back2=",back2," ",\
              "back1=",back1," ","next=",next,sep='')
        back2=back1#обмен содержимым переменных
        back1=next#обмен содержимым переменных
    print("Шаг ",i," back2=",back2," ",\
          "back1=",back1," ","next=",back1+back2,sep='')
    return back1+back2
fibNumb=int(input("Введите количество ступенек:\n"))
print("Ответ: необходимо подняться ",fib_ult(fibNumb)," раз.",sep='')
```

```
Введите количество ступенек:
7
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Ответ: необходимо подняться 21 раз.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int fibNumb;//Порядковый номер в ряу Фибоначчи
unsigned long long answ;//Ответ
unsigned long long fib_ultim(int n);//Сигнатура

int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");                // Очистка консоли
    printf("Введите количество ступенек:\n");
    scanf("%d",&fibNumb);
    answ=fib_ultim(fibNumb-1);//Так как нумерация массивов с нуля
    if (answ==0){//Выход если ступенек меньше 2-х
        getchar();//Задержка для отображения вывода
        printf("Нажмите любую кнопку для завершения.\n");
        int ch = getch();
        return (EXIT_SUCCESS);
    }
    printf("Ответ: необходимо подняться %llu раз.  \n",answ);
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}
```

```
nsigned long long fib_ultim(int n)//Семантика

    int i;//счетчик
    //Значение функции на два и один шаг назад
    unsigned long long back2=1, back1=2, next;
    if (n<2){//Выход если ступенек меньше 2-х
        printf("Ступенек не может быть меньше 2-х!\n");
        return 0;
    }
    for (i=2; i<n; i++)//при n<2 цикл не выполняется
    {
        next=back1+back2;
        printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
                i-1, back2,back1,next);
        back2=back1;//обмен содержимым переменных
        back1=next;//обмен содержимым переменных
    }
    printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
                i-1, back2,back1,back1+back2);
    return back1+back2;
}
```

```
Введите количество ступенек:
7
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Ответ: необходимо подняться 21 раз.
Нажмите любую кнопку для завершения.
```

6. Гном вынес все драгоценные камни в плетеной корзинке с двумя ручками наверх из шахты в кладовую по лестнице. Теперь гном решил застелить пол шахты квадратными кварцевыми плитками, инкрустировав перекрестие плиток драгоценным камнем. Стык пола со стенами гном решил не украшать. Сторона плитки – 1 шаг гнома. Гном обмерил шахту шагами и составил её план. На плане гном отметил границы шахты линиями, соединяющими углы стен. Координаты углов шахты в шагах гнома: {3,2; 2,5; 4,9; 8,8; 8,5; 6,3}. Сколько понадобится драгоценных камней гному, чтобы инкрустировать перекрестие плиток? Напишите алгоритм и программу, позволяющую вычислить, сколько понадобится драгоценных камней гному, чтобы инкрустировать перекрестие плиток в шахте с произвольными целыми координатами углов. Алгоритм должен раскрывать решение на каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте специальные функции и библиотеки для реализации основных этапов решения. Считывание заранее рассчитанных значений не

засчитывается за решение задачи. Допустимо задать готовые координаты в коде основной функции. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для координат углов шахты в шагах гнома {3,2; 2,5; 4,9; 8,8; 8,5; 6,3}, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача из области вычислительной геометрии на вычисление количества точек с целочисленными координатами, лежащих внутри (но не на границе) многоугольника, заданного координатами своих вершин.

В задаче требуется рассчитать количество точек с целыми координатами, попадающими внутрь многоугольника, не считая точки, лежащие точно на ребрах многоугольника. Ответ можно проверить визуально. Ожидается увидеть алгоритм решения в виде рисунка/рисунков и формул Пика и НОД. **Ответ**: **25** точек.

Теория_1. Для любого многоугольника с целочисленными координатами вершин справедлива **формула Пика**: S = n + m/2 — 1, где S – площадь многоугольника, n – количество целых точек лежащих строго внутри многоугольника, m – количество целых точек, лежащих на границе многоугольника. Площадь многоугольника S вычисляется методом трапеций (**ожидаемое решение**) или по формуле Гаусса (“шнуровки”, **допустимое равнозначное решение**). Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1, где катет_1,2 - катеты прямоугольного треугольника, образованного на каждом ребре многоугольника (гипотенуза). Искомое количество целых точек n, лежащих строго внутри многоугольника, вычисляется как n = S - m/2 + 1.

Теория_2. Площадь многоугольника. Для вычисления площади многоугольника необходимо “замкнуть” периметр, добавив последнюю_вершину = первая_вершина. Формулы площадей показаны ниже.

$$S_{Гаусс} = \frac{1}{2}\left|\sum_{i=1}^{n-1} x_i y_{i+1} + x_n y_1 - \sum_{i=1}^{n-1} x_{i+1} y_i - x_1 y_n\right|$$

$$S_{трап} = \frac{1}{2}\left|\sum_{i=1}^{n-1} (x_{i+1} - x_i)(y_{i+1} + y_i)\right|$$

Теория_3. Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1. Катеты прямоугольного треугольника для каждого ребра многоугольника вычисляются как разности координат вершин, образующих ребро. Для вертикального или горизонтального ребра один из катетов будет равен нулю. Необходимо последовательно перебрать все вершины за исключением “замыкающей”, вычислить НОД катетов и просуммировать их. Алгоритм вычисления показан ниже.

```
lineNodes(polygon) := "Количество узлов, лежащих на ребрах многоугольника."
                      "Сумма НОД катетов прямоугольных трегольников"
                      "для каждого ребра-гипотенузы"
                      "Замкнутый контур. Последняя_вершина=первая_вершина"
                      s ← 0
                      for i ∈ ORIGIN .. rows(polygon) − 1
                        s ← s + gcd(polygon_{i+1,1} − polygon_{i,1}, polygon_{i+1,2} − polygon_{i,2})
                      s
```

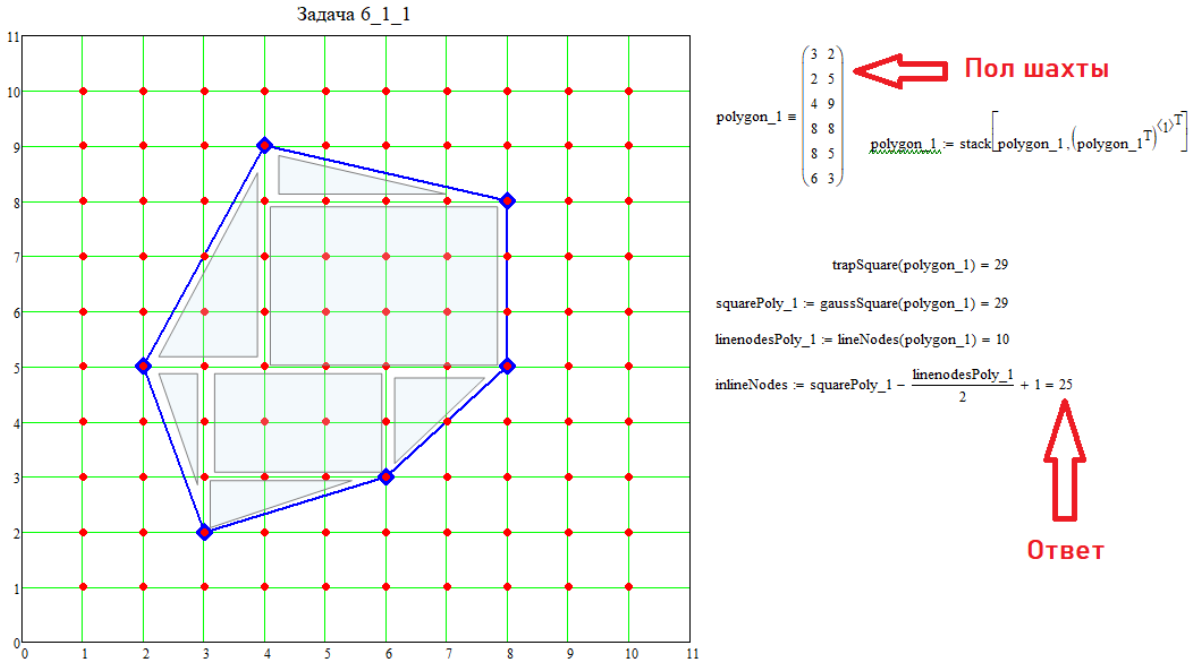
Алгоритм вычисления площади многоугольника **методом трапеций**:

```
trapSquare(polygon) := "Площадь многоугольника методом трапеций"
                       "Замкнутый контур. Последняя_вершина=первая_вершина"
                       s ← 0
                       for i ∈ ORIGIN .. rows(polygon) − 1
                           a ← polygon_{i+1,1} − polygon_{i,1}
                           b ← polygon_{i+1,2} + polygon_{i,2}
                           s ← s + a·b
                       s ← 0.5 |s|
                       s
```

Алгоритм вычисления площади многоугольника **методом Гаусса**:

```
gaussSquare(polygon) := "Площадь многоугольника методом Гаусса ("шнуровки")"
                        "Замкнутый контур. Последняя_вершина=первая_вершина"
                        s ← 0
                        for i ∈ ORIGIN .. rows(polygon) − 1
                            s ← s + polygon_{i+1,1}·polygon_{i,2}
                            s ← s − polygon_{i,1}·polygon_{i+1,2}
                        s ← s + polygon_{rows(polygon),1}·polygon_{1,2}
                        s ← s − polygon_{1,1}·polygon_{rows(polygon),2}
                        s ← 0.5 |s|
                        s
```

Графическое решение и ответ показаны ниже.



Ниже приведен пример решения на Python 3.82 и C.

```python
#Формула Пика
#НОД
def NOD(a,b):
    while b != 0:
        a, b = b, a % b
    return a
#Количество узлов, лежащих
#на ребрах многоугольника
def lineNodes(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        dX=polygon[0][i+1]-polygon[0][i]
        dY=polygon[1][i+1]-polygon[1][i]
        s=s+NOD(abs(dX),abs(dY))
    return s

#Площадь многоугольника методом трапеций
def trapSquare(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        a=polygon[0][i+1]-polygon[0][i]
        b=polygon[1][i+1]+polygon[1][i]
        s=s+a*b
    s=abs(s)/2
    return s

#Площадь многоугольника методом Гаусса
def gaussSquare(polygon):
    s=0; last=len(polygon[0])-1
    for i in range(last):
        s=s+polygon[0][i+1]*polygon[1][i]
        s=s-polygon[0][i]*polygon[1][i+1]
    s=s+polygon[0][last]*polygon[1][0]
    s=s-polygon[0][0]*polygon[1][last]
    s=0.5*abs(s)
    return s
```

```
#Вычисление по формуле Пика
#S = n + m/2 – 1
def inlineNodes(polygon):
    S=trapSquare(polygon)
    print("Площадь многоугольника",S)
    m=lineNodes(polygon)
    print("Количество точек на ребрах",m,"шт.")
    n=S-m/2+1
    return int(n)


#Координаты углов многоугольника
#двумерный список (x,y)
Plgn=[[3,2,4,8,8,6],[2,5,9,8,5,3]]
#"Замыкание" многоугольника
Plgn[0].append(Plgn[0][0])
Plgn[1].append(Plgn[1][0])

print("Количество точек внутри многоугольника" \
      ,inlineNodes(Plgn),"шт.")
```

```
Площадь многоугольника 29
Количество точек на ребрах 10 шт.
Количество точек внутри многоугольника 25 шт.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int NOD(int a, int b);//НОД
int lineNodes(int *X, int *Y, int len);//Количество точек на ребрах мнгоуголника
int inlineNodes(int *X, int *Y, int len);//Вычисление по формуле Пика
float trapSquare(int *X, int *Y, int len);//S многогольника методом трапеций
float gaussSquare(int *X, int *Y, int len);//S многогольника методом Гаусса


int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");              // Очистка консоли
    //Сразу задаем "замкнутый" массив координат углов
    int pointsX[]={3,2,4,8,8,6,3};
    int pointsY[]={2,5,9,8,5,3,2};

    //Количество углов в массиве координат
    int n=sizeof(pointsX)/sizeof(pointsX[0]);
    //printf("%f\n",(float)5/2);
    printf("Количество точек внутри многоугольника %i шт.\n", \
            inlineNodes(pointsX,pointsY,n));
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}
```

```
30  //Вычисление НОД
31  int NOD(int a, int b){
32      int t;
33      while (b != 0){
34          t = b;
35          b = a % b;
36          a = t;
37      }
38      return a;
39  }
40  //Вычисление площади методом трапеций
41  float trapSquare(int *X, int *Y, int len){
42      float s=0;
43      for (int i=0; i<len-1; i++){
44          int a= X[i+1]- X[i];
45          int b= Y[i+1]+ Y[i];
46          s=s+a*b;
47      }
48      s=(float)abs(s)/2;
49      return s;
50  }
51  //Вычисление площади методом Гаусса
52  float gaussSquare(int *X, int *Y, int len){
53      float s=0; int last=len-1;
54      for (int i=0; i<last; i++){
55          s=s+X[i+1]*Y[i];
56          s=s-X[i]*Y[i+1];
57      }
58      s=s+X[last]*Y[0];
59      s=s-X[0]*Y[last];
60      s=(float)abs(s)/2;
61      return s;
62  }
```

```
63  //Количество точек на ребрах многоугольника
64  int lineNodes(int *X, int *Y, int len){
65      int s=0;
66      for (int i=0; i<len-1; i++){
67          int dX=X[i+1]-X[i];
68          int dY=Y[i+1]-Y[i];
69          s=s+NOD(abs(dX),abs(dY));
70      }
71      return s;
72  }
73  //Вычисление по формуле Пика
74  //S = n + m/2 — 1
75  int inlineNodes(int *X, int *Y, int len){
76      float S=trapSquare(X,Y,len);
77      printf("Площадь многоугольника %.2f\n",S);
78      int m=lineNodes(X,Y,len);
79      printf("Количество точек на ребрах %i шт.\n",m);
80      int n=S-m/2+1;//Точек внутри многоугольника
81      return n;
82  }
```

```
Площадь многоугольника 29
Количество точек на ребрах 10 шт.
Количество точек внутри многоугольника 25 шт.
```

ИНФОРМАТИКА

ВАРИАНТ 2.

1. С тех пор, как Иван изучил формулы в Excel, он ежедневно оттачивает свое мастерство. Сегодня он напечатал число 1,5 в ячейку A1, в соседнюю ячейку B1 он внес формулу =ОКРУГЛ(ОСТАТ(8;$A$1)+СТЕПЕНЬ(A1;2)-ЕСЛИ(A1>0;A1^2/3;A1/3);1) и растянул до ячейки E1. В соседней ячейке F1, он ввел формулу: =СЧЁТЕСЛИ(A1:E1;">20")+СУММЕСЛИ(A1:E1;">3";A1:E1). На следующей строке, начиная с ячейки A2, мальчик ввел формулу =ОКРУГЛ(ОСТАТ(B1;A1)-КОРЕНЬ(ABS(A1-B1))+ЕСЛИ(A1+B1<2;B1/A1;A1+3);2) и скопировал в диапазон B2:E2. В ячейку F2 он скопировал формулу из ячейки F1. На следующей строке он ввел целое число M в ячейку A3 и повторил действия, как для первой строки. А в четвертую (диапазон A4:E4) скопировал формулы со 2 строки, в ячейку F4 была вставлена формула из ячейки F3. В пятой строке Иван ввел некоторое целое число N в ячейку A5 и решил повторить действия, как для строки 1. В ячейку A6 он скопировал формулу из A2 и растянул с помощью автозаполнения до ячейки E6. В ячейку F6 была скопирована формула из F5. При этом мальчик получил следующий результат:

	A	B	C	D	E	F
1	1,5	2	3,2	7,3	36	47,5
2	4,29	5,1	5,08	11,74	47,11	74,32
3	?	3,2	7,3	36	864,5	913
4	5,1	5,08	11,74	10,72	909,04	942,68
5	?	11,2	84,1	4715,7	14825218	14830036
6	7,52	11,36	25,14	4642,06	14829970	14834659

Определите наименьшие целые числа M и N? Перечислите числа через точку с запятой. **(10 баллов)**

Решение:

Решение:

	A	B	C	D	E	F
1	1,5	2	3,2	7,3	36	47,5
2	4,29	5,1	5,08	11,74	47,11	74,32
3	2	3,2	7,3	36	864,5	913
4	5,1	5,08	11,74	10,72	909,04	942,68
5	4	11,2	84,1	4715,7	14825218	14830036
6	7,52	11,36	25,14	4642,06	14829970	14834659

Ответ: 2;4.

2. Два сладкоежки решили сыграть в игру «30 шоколадных конфет». Играющие по очереди могут взять от одной до четырёх конфет. Кто не может сделать ход (конфет не осталось), проигрывает. Другими словами, выигрывает взявший последнюю конфету. Кто выигрывает при безошибочной игре – игрок, делающий первый ход, или игрок, делающий второй ход? Поясните алгоритм графически. Напишите обобщенный алгоритм для любого количества конфет. **(10 баллов)**

Решение:

Используется Стратегия_Дополни до пяти. Всегда выигрывает Игрок_2 (игрок, делающий ход вторым). Суть Стратегия_Дополни до пяти:

Ход_1: Игрок_1 (игрок, делающий ход первым) берет от 1 до 4 конфет (в данном случае конфет остаётся от 26 до 29).

Ход_2: Игрок_2 (игрок, делающий ход вторым) должен дополнять ход первого до пяти конфет (то есть, если Игрок_1 взял одну конфету, Игрок_2 должен взять 4 конфеты. Конфет в вазочке должно остаться 25).

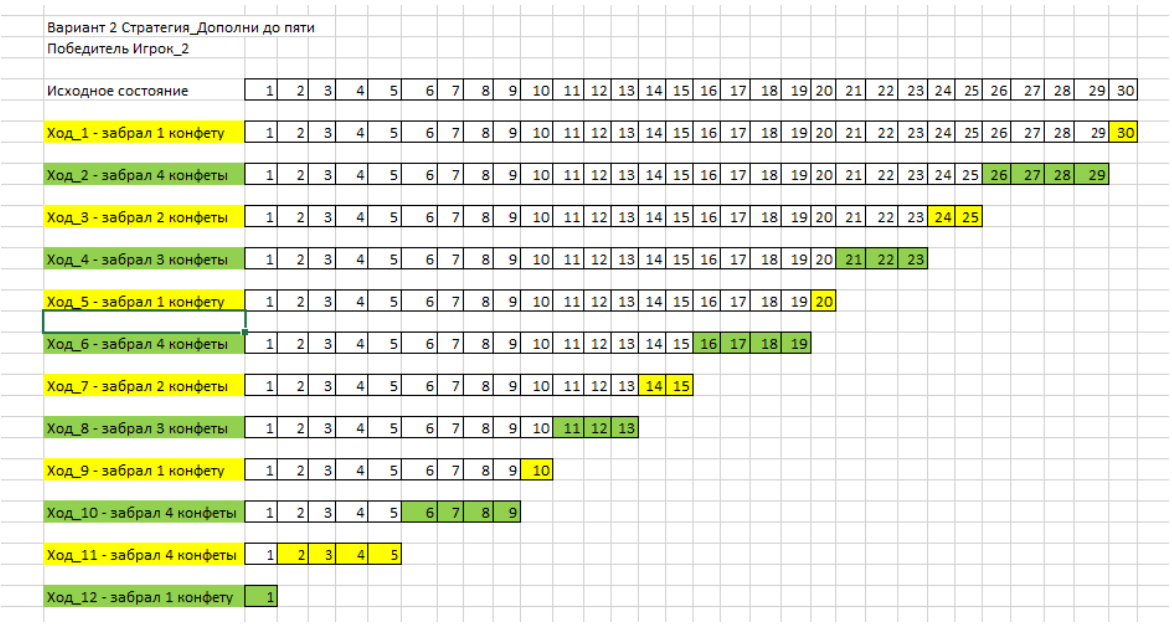
Ход_3: Игрок_1 берет от 1 до 4 конфет (в данном случае конфет остаётся от 21 до 24).

Ход_4: Игрок_2 должен дополнять ход первого до пяти конфет (Конфет в вазочке должно остаться 20). И так далее до Победа Игрок_2. Проиграть невозможно.

Иллюстрация на рисунке ниже. Алгоритм подходит в общем случае, для N конфет., если N делится на 5 без остатка, то второй может гарантировать себе выигрыш, дополняя ход противника до 5.

Алгоритм для Игрок_2 (обобщенный):

Шаг 1 Игрок _1 взял k конфет
Шаг 2 Игорок_2 взял (5-k) конфет
Если Остаток = 0 – значит СТОП_Выиграл.



3. Выражение $x \wedge y \oplus z \leftrightarrow z \wedge x \vee y$ =*ложь*. Перечислите выражения (выражение), удовлетворяющие этому ответу и тому же набору значений *x, y, z* из приведенных ниже:

1). $x \leftrightarrow \bar{y} \vee z \leftrightarrow x \wedge y \oplus z$

2). $y \oplus x \to y \wedge \bar{x} \oplus z$

3). $x \wedge \bar{y} \to z \leftrightarrow y \vee \bar{x} \oplus z$

4). $\bar{z} \oplus y \leftrightarrow x \vee z \to y \wedge \bar{x} \oplus y$

5). $x \to \bar{y} \vee z \wedge x \leftrightarrow \bar{y} \oplus z$

6). $\bar{x} \vee y \to z \leftrightarrow z \wedge x \oplus \bar{y}$

В случае, если выражений несколько перечислите их в ответе через запятую. В процессе решения необходимо расставить порядок действий для всех выражений, а также построить таблицы истинности. При расстановке действий использовать в первую очередь приоритет операций, а потом уже очередность появления.
(10 баллов)

Решение:

1. На первом шаге необходимо определить табличные значения для основного примера, для каких наборов соответствует *истина*. Для этого необходимо расставить приоритет действий и построить таблицу истинности.
2. Расстановка действий:

1 – конъюнкция *x*, *y*
2 – конъюнкций *z*, *x*
3 – дизъюнкция результатов второго действия и *y*
4 – исключающее или первого действия и *z*
5 – эквивалентность результатов действий 4 и 3.

3. Построение таблицы истинности:

x	***y***	***z***	**1**	**2**	**3**	**4**	**5**
0	0	0	0	0	0	0	1
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	0	0	1	1	1
1	0	0	0	0	0	0	1
1	0	1	0	1	1	1	1
1	1	0	1	0	1	1	1
1	1	1	1	1	1	0	0

Исходя из полученных данных, значения *истина* получены для следующих наборов:

x	***y***	***z***
0	0	1
0	1	0
1	1	1

Эти значения будут подставлены в каждый из предлагаемых примеров для сравнения результатов.

Исследование 1 примера начинается с расстановки действий:

1). $x \leftrightarrow \bar{y} \vee z \leftrightarrow x \wedge y \oplus z$

1 – отрицание y

2 – конъюнкция x, y

3 – дизъюнкция результатов первого действия и z

4 – исключающее или результатов второго действия и z

5 – эквивалентность x и результатов третьего действия

6 – эквивалентность результатов 5 и 4 действий.

Следующий шаг – построение таблицы истинности:

x	***y***	***Z***	***1***	***2***	***3***	***4***	***5***	***6***
0	0	1	1	0	1	1	0	0
0	1	0	0	0	0	0	1	0
1	1	1	0	1	1	0	1	0

Пример подходит.

Исследование примера 2 с расстановкой действий:

2). $y \oplus x \rightarrow y \wedge \bar{x} \oplus z$

1 – отрицание x

2 – конъюнкция y и результатов первого действия

3 – исключающее или y и x

4 – исключающее или результатов 2 действия и z

5 – импликация результатов 3 и 4 действий

x	***y***	***Z***	***1***	***2***	***3***	***4***	***5***
0	0	1	1	0	0	1	1
0	1	0	1	1	1	1	1
1	1	1	0	0	0	1	1

Пример не подходит.

Исследование примера 3 с расстановкой действий:

3). $x \wedge \bar{y} \rightarrow z \leftrightarrow y \vee \bar{x} \oplus z$

1 – отрицание y

2 – отрицание x

3 – конъюнкция результатов 1 действия и x

4 – дизъюнкций y и результатов 2 действия

5 – исключающее или результатов 4 действия и z

6 – импликация результатов 3 действия из z

7 – эквивалентность результатов 6 и 5 действий

x	***Y***	***z***	***1***	***2***	***3***	***4***	***5***	***6***	***7***
0	0	1	1	1	0	1	0	1	0
0	1	0	0	1	0	1	1	1	1
1	1	1	0	0	0	1	0	1	0

Пример не подходит.

Исследование 4 примера с расстановкой действий:

4). $\bar{z} \oplus y \leftrightarrow x \vee z \rightarrow y \wedge \bar{x} \oplus y$

1 – отрицание *z*

2 – отрицание *x*

3 – конъюнкция *y* и результатов 2 действия

4 – дизъюнкция *x, z*

5 –исключающее или результатов первого действия и *y*

6 – исключающее или результатов 3 действия и *y*

7 – импликация результатов 4 действия и 6

8 – эквивалентность результатов 5 действия и 7

x	***Y***	***z***	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***
0	0	1	0	1	0	1	0	0	0	1
0	1	0	1	1	1	0	0	0	1	0
1	1	1	0	0	0	1	1	1	1	1

Пример не подходит.

Исследование 5 примера с расстановкой действий:

5). $x \rightarrow \bar{y} \vee z \wedge x \leftrightarrow \bar{y} \oplus z$

1 – отрицание *y* (оно же 2 действие)

3 – конъюнкция *z, x*

4 – дизъюнкция 1 действия и z

5 – исключающее или результатов 2 действия и *z*

6 – импликация *x* из результатов 4 действия

7 – эквивалентность результатов 6 и 5 действий

x	***y***	***z***	***1, 2***	***3***	***4***	***5***	***6***	***7***
0	0	1	1	0	1	0	1	0
0	1	0	0	0	0	0	1	0
1	1	1	0	1	1	1	1	1

Пример не подходит.

Исследование 6 примера с расстановкой действий:

6). $\bar{x} \vee y \rightarrow z \leftrightarrow z \wedge x \oplus \bar{y}$

1 – отрицание *x*

2 – отрицание *y*

3 – конъюнкция *z*, *x*

4 – дизъюнкция результатов 1 действия и y

5 – исключающее или результатов 3 и 2 действий

6 – импликация результатов 4 действия и *z*

7 – эквивалентность результатов 5 и 6 действий.

x	***y***	***z***	***1***	***2***	***3***	***4***	***5***	***6***	***7***
0	0	1	1	1	0	1	1	1	1
0	1	0	1	0	0	1	0	0	1
1	1	1	0	0	1	1	1	1	1

Пример не подходит.

Ответ: 1.

4. Максим устроился работать помощником сетевого администратора в IT-отдел Университета. В первый же день работы он получил ответственное задание. Отдел информационных технологий Университета использует диапазон частных IP-адресов, в котором настроены 6 сетей филиалов. Данные по распределению адресного пространства частично известны: в каждой сети компьютеру системного администратора назначен адрес 172.26.228.145, а размер сетевого префикса указан в таблице. При этом адресное пространство использовано максимально эффективно.

Максим обладает следующими знаниями:

- одному устройству должен соответствовать один IP-адрес вида XXX.XXX.XXX.XXX, при этом XXX – число в диапазоне [0:255], называемое октетом;
- устройства внутри одной сети должны беспрепятственно взаимодействовать друг с другом, взаимодействие устройств в разных сетях реализовано провайдером с помощью механизмов преобразования сетевых адресов;
- выделенный размер сети содержит минимально необходимое количество адресов;
- выделенный размер сети записывается как 2^n-2, где n – минимальное количество бит, которое может содержать все номера адресов сети;
- в каждом сегменте сети первый адрес зарезервирован под идентификатор сети, а последний – под широковещательную рассылку.

Название подсети	**IP-адрес ПК сетевого администратор**	**Адрес сети**	**Десятичная маска**	**Диапазон доступных адресов**	**Выделенный размер**	**Широковеща-тельный адрес**
Главный корпус	172.26.228.145/18					
Инженерный корпус	172.26.228.145/21					
Учебный центр №2	172.26.228.145/23					

Библиотека	172.26.228.145/25					
Пост охраны	172.26.228.145/29					
IT-отдел	172.26.228.145/30	172.26.228.144	255.255.255.252	172.26.228.145-172.26.228.146	2	172.26.228.147

Максим смог заполнить данные для IT-отдела. Заполнение остальной информации вызвало у помощника сетевого администратора затруднения. Необходимо помочь Максиму заполнить таблицу недостающими данными.
(**20 баллов**)
Решение:

Название подсети	**IP-адрес ПК сетевого администратора**	**Адрес сети**	**Десятичная маска**	**Диапазон доступных адресов**	**Выделенный размер**	**Широковеща-тельный адрес**
Главный корпус	172.26.228.145/18	**172.26.**192.0	255.255.192.0	.192.1-.255.254	16 382	.255.255
Инженерный корпус	172.26.228.145/21	**172.26.**224.0	255.255.248.0	.224.1-.231.254	2 046	.231.255
Учебный центр №2	172.26.228.145/23	**172.26.**228.0	255.255.254.0	.228.1-.229.254	510	.229.255
Библиотека	172.26.228.145/25	**172.26.228.**128	255.255.255.128	.129-.254	126	.255
Пост охраны	172.26.228.145/29	**172.26.228**.144	255.255.255.248	.145-.150	6	.151
IT-отдел	**172.26.228**.145/30	**172.26.228**.144	255.255.255.252	**172.26.228.**145-**172.26.228**.146	2	**172.26.228**.147

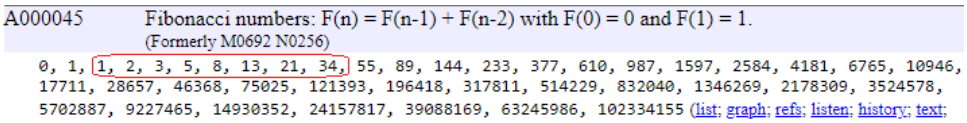
*полужирным выделена общая часть адреса до точки (по аналогии с последней строкой таблицы)

5. Гном носит драгоценные камни в плетеной корзинке наверх из шахты в кладовую. В кладовую из шахты ведет лестница из 8 ступенек. Шаг у гнома небольшой, поэтому у корзины две ручки, и он может наступать на каждую ступеньку, или перескакивать через одну ступеньку и становиться сразу на следующую. Каждый раз гном поднимается по лестнице по-новому, шагая по одной, по две ступеньки, или где по одной, а где по две. Сколько раз гном сможет подняться по лестнице из 8 ступенек, ни разу не повторив последовательность шагов? Напишите алгоритм и программу, позволяющую вычислить, сколько раз гном сможет подняться по лестнице из произвольного целого N (N>2) количества ступенек (в пределах типа данных языка программирования), чтобы ни разу не повторить последовательность шагов. Учтите, что возможности компьютера, на котором будет выполняться программа, ограничены: допустимо использовать до 10 переменных, массивы ограничены 10 ячейками соответствующего типа, а стек вмещает 10 вызовов и не расширяется. Алгоритм должен раскрывать решение на каждом шаге. Код

программы должен состоять из простых и понятных команд. Не используйте уникальные особенности языка программирования (классы, методы, объекты, попарный обмен между переменными, специальные функции и библиотеки и др., кроме счетчиков циклов) для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для 8 ступенек, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача на вычисление последовательности чисел Фибоначчи.


A000045 Fibonacci numbers: F(n) = F(n-1) + F(n-2) with F(0) = 0 and F(1) = 1.
(Formerly M0692 N0256)

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75025, 121393, 196418, 317811, 514229, 832040, 1346269, 2178309, 3524578, 5702887, 9227465, 14930352, 24157817, 39088169, 63245986, 102334155 (list; graph; refs; listen; history; text;

https://oeis.org/A000045

В задаче требуется рассчитать число уникальных комбинаций шагов, которыми можно подняться по лестнице из **8** ступенек, используя шаги в 1, 2 или 1 и 2 ступеньки. **Ответом является 8-е** по счету число последовательности Фибоначчи (**$a_8 = 34$**), считая, что **$a_1 = 1$**, **$a_2 = 2$**.

Объяснение 1. Обозначим число комбинаций как a_n, где n – число ступенек, n>2.
$a_1 = 1$ способ –> на лестницу из одной ступеньки можно подняться только одним способом - шагом в 1 ступеньку.
$a_2 = 2$ способа –> 1+1 ступенька; сразу 2 ступеньки.
$a_3 = 3$ –> 1+1+1; 1+2; 2+1 ступеньки.
$a_4 = 5$ –> 1+1+1+1; 1+1+2; 1+2+1; 2+1+1; 2+2 ступеньки.
$a_4 = a_3 + a_2$; $a_3 = a_2 + a_1$; ==> **$a_n = a_{n-1} + a_{n-2}$;**
$a_5 = a_4 + a_3 = 5+3=8$; $a_6 = a_5 + a_4 = 8+5=13$; $a_7 = a_6 + a_5 = 13+8=21$;
$a_8 = a_7 + a_6 = 21+13=34$.
Ответ: $a_8 = 34$ способа.
Объяснение 2. Обозначим число комбинаций как a_n, где n – число ступенек, n>2. Можно начать подъем с шага в 1 ступеньку. Тогда останется a_{n-1} комбинаций подъема по лестнице. Если начать подъем с шага в 2 ступеньки, то останется a_{n-2} комбинаций подъема по лестнице. Первый шаг входит в любую из комбинаций шагов. Таким образом количество комбинаций определяется суммой количества оставшихся ступенек для каждого варианта первого шага и размером первого шага, т.е. **$a_n = a_{n-1} + a_{n-2}$.**

Задачу можно решить, например *рекурсией* (*неправильно*, ограничение по стеку вызовов при больших N), **динамическим программированием (ожидаемое решение)**, с помощью умножения матриц (допустимое, но избыточное решение), формулы Бине (допустимое, но избыточное решение). Ниже приведен пример решения с помощью динамического программирования на Python 3.82 и C.

```
#Функция Фибоначи с обменом
#Экономит память, хранит только два
#предыдущих значения
def fib_ult(n):#Для вывода по шагам
    #Значение функции на два и один шаг назад
    back2=1; back1=2
    if n<2:
        print("Ступенек не может быть меньше 2-х!")
        return 0
    for i in range(2,n-1):
        next=back1+back2
        print("Шаг ",i-1," back2=",back2," ",\
              "back1=",back1," ","next=",next,sep='')
        back2=back1#обмен содержимым переменных
        back1=next#обмен содержимым переменных
    print("Шаг ",i," back2=",back2," ",\
          "back1=",back1," ","next=",back1+back2,sep='')
    return back1+back2
fibNumb=int(input("Введите количество ступенек:\n"))
print("Ответ: необходимо подняться ",fib_ult(fibNumb)," раз.",sep='')
```

```
Введите количество ступенек:
8
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Шаг 6 back2=13 back1=21 next=34
Ответ: необходимо подняться 34 раз.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int fibNumb;//Порядковый номер в ряу фибоначчи
unsigned long long answ;//Ответ
unsigned long long fib_ultim(int n);//Сигнатура

int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");                // Очистка консоли
    printf("Введите количество ступенек:\n");
    scanf("%d",&fibNumb);
    answ=fib_ultim(fibNumb-1);//Так как нумерация массивов с нуля
    if (answ==0){//Выход если ступенек меньше 2-х
        getchar();//Задержка для отображения вывода
        printf("Нажмите любую кнопку для завершения.\n");
        int ch = getch();
        return (EXIT_SUCCESS);
    }
    printf("Ответ: необходимо подняться %llu раз.  \n",answ);
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}

unsigned long long fib_ultim(int n)//Семантика
{
    int i;//счетчик
    //Значение функции на два и один шаг назад
    unsigned long long back2=1, back1=2, next;
    if (n<2){//Выход если ступенек меньше 2-х
        printf("Ступенек не может быть меньше 2-х!\n");
        return 0;
    }
    for (i=2; i<n; i++)//при n<2 цикл не выполняется
    {
        next=back1+back2;
        printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
                i-1, back2,back1,next);
        back2=back1;//обмен содержимым переменных
        back1=next;//обмен содержимым переменных
    }
    printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
                i-1, back2,back1,back1+back2);
    return back1+back2;
}
```

```
Введите количество ступенек:
8
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Шаг 6 back2=13 back1=21 next=34
Ответ: необходимо подняться 34 раз.
Нажмите любую кнопку для завершения.
```

6. Гном вынес все драгоценные камни в плетеной корзинке с двумя ручками наверх из шахты в кладовую по лестнице. Теперь гном решил застелить пол шахты квадратными кварцевыми плитками, инкрустировав перекрестие плиток драгоценным камнем. Стык пола со стенами гном решил не украшать. Сторона плитки – 1 шаг гнома. Гном обмерил шахту шагами и составил её план. На плане гном отметил границы шахты линиями, соединяющими углы стен. Координаты углов шахты в шагах гнома: {7,1; 3,2; 2,5; 4,8; 7,8; 8,5}. Сколько понадобится драгоценных камней гному, чтобы инкрустировать перекрестие плиток? Напишите алгоритм и программу, позволяющую вычислить, сколько понадобится драгоценных камней гному, чтобы инкрустировать перекрестие плиток в шахте с произвольными целыми координатами углов. Алгоритм должен раскрывать решение на каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте специальные функции и библиотеки для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Допустимо задать готовые координаты в коде основной функции. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для координат углов шахты в шагах гнома {7,1; 3,2; 2,5; 4,8; 7,8; 8,5}, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача из области вычислительной геометрии на вычисление количества точек с целочисленными координатами, лежащих внутри (но не на границе) многоугольника, заданного координатами своих вершин.

В задаче требуется рассчитать количество точек с целыми координатами, попадающими внутрь многоугольника, не считая точки, лежащие точно на ребрах многоугольника. Ответ можно проверить визуально. Ожидается увидеть алгоритм решения в виде рисунка/рисунков и формул Пика и НОД. **Ответ**: **28** точек.

Теория_1. Для любого многоугольника с целочисленными координатами вершин справедлива **формула Пика**: S = n + m/2 — 1, где S – площадь многоугольника, n – количество целых точек лежащих строго внутри многоугольника, m – количество целых точек, лежащих на границе многоугольника. Площадь многоугольника S вычисляется

методом трапеций (**ожидаемое решение**) или по формуле Гаусса (“шнуровки”, **допустимое равнозначное решение**). Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1, где катет_1,2 - катеты прямоугольного треугольника, образованного на каждом ребре многоугольника (гипотенуза). Искомое количество целых точек n, лежащих строго внутри многоугольника, вычисляется как n = S - m/2 + 1.

Теория_2. Площадь многоугольника. Для вычисления площади многоугольника необходимо “замкнуть” периметр, добавив последнюю_вершину = первая_вершина. Формулы площадей показаны ниже.

$$S_{Гаусс} = \frac{1}{2}\left|\sum_{i=1}^{n-1} x_i y_{i+1} + x_n y_1 - \sum_{i=1}^{n-1} x_{i+1} y_i - x_1 y_n\right|$$

$$S_{трап} = \frac{1}{2}\left|\sum_{i=1}^{n-1} (x_{i+1} - x_i)(y_{i+1} + y_i)\right|$$

Теория_3. Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1. Катеты прямоугольного треугольника для каждого ребра многоугольника вычисляются как разности координат вершин, образующих ребро. Для вертикального или горизонтального ребра один из катетов будет равен нулю. Необходимо последовательно перебрать все вершины за исключением “замыкающей”, вычислить НОД катетов и просуммировать их. Алгоритм вычисления показан ниже.

```
lineNodes(polygon) := "Количество узлов, лежащих на ребрах многоугольника."
                      "Сумма НОД катетов прямоугольных трегольников"
                      "для каждого ребра-гипотенузы"
                      "Замкнутый контур. Последняя_вершина=первая_вершина"
                      s ← 0
                      for i ∈ ORIGIN.. rows(polygon) − 1
                        s ← s + gcd(polygon_{i+1,1} − polygon_{i,1}, polygon_{i+1,2} − polygon_{i,2})
                      s
```

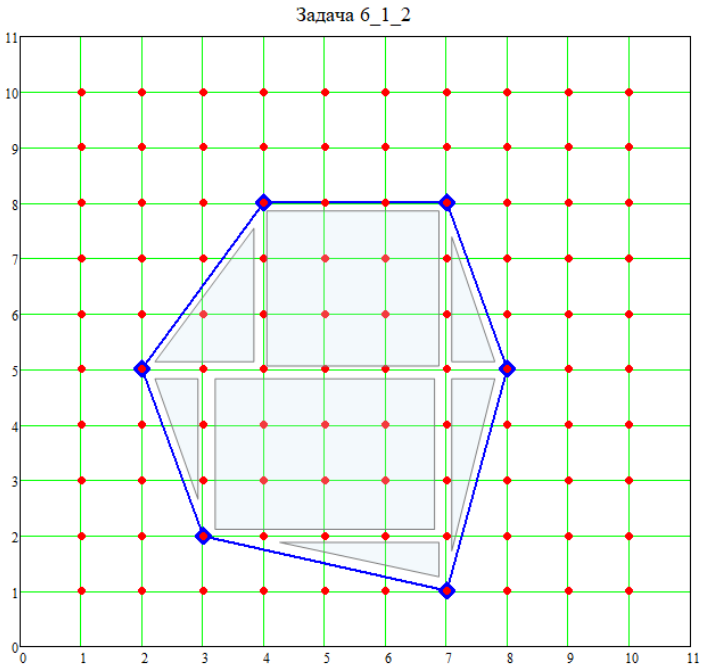
Алгоритм вычисления площади многоугольника **методом трапеций**:

```
trapSquare(polygon) := "Площадь многоугольника методом трапеций"
                       "Замкнутый контур. Последняя_вершина=первая_вершина"
                       s ← 0
                       for i ∈ ORIGIN.. rows(polygon) − 1
                         a ← polygon_{i+1,1} − polygon_{i,1}
                         b ← polygon_{i+1,2} + polygon_{i,2}
                         s ← s + a·b
                       s ← 0.5 |s|
                       s
```

Алгоритм вычисления площади многоугольника **методом Гаусса**:

```
gaussSquare(polygon) := | "Площадь многоугольника методом Гаусса ("шнуровки")"
                        | "Замкнутый контур. Последняя_вершина=первая_вершина"
                        | s ← 0
                        | for i ∈ ORIGIN.. rows(polygon) − 1
                        |     s ← s + polygon_{i+1,1}·polygon_{i,2}
                        |     s ← s − polygon_{i,1}·polygon_{i+1,2}
                        | s ← s + polygon_{rows(polygon),1}·polygon_{1,2}
                        | s ← s − polygon_{1,1}·polygon_{rows(polygon),2}
                        | s ← 0.5 |s|
                        | s
```

Графическое решение и ответ показаны ниже.



$$polygon\_1 \equiv \begin{pmatrix} 7 & 1 \\ 3 & 2 \\ 2 & 5 \\ 4 & 8 \\ 7 & 8 \\ 8 & 5 \end{pmatrix}$$

Пол шахты

$$polygon\_1 := stack\left[polygon\_1, \left(polygon\_1^{T}\right)^{\langle 1 \rangle^{T}}\right]$$

$$trapSquare(polygon\_1) = 31$$

$$squarePoly\_1 := gaussSquare(polygon\_1) = 31$$

$$linenodesPoly\_1 := lineNodes(polygon\_1) = 8$$

$$inlineNodes := squarePoly\_1 - \frac{linenodesPoly\_1}{2} + 1 = 28$$

Ответ

Ниже приведен пример решения на Python 3.82 и C.

```python
#Формула Пика
#НОД
def NOD(a,b):
    while b != 0:
        a, b = b, a % b
    return a
#Количество узлов, лежащих
#на ребрах многоугольника
def lineNodes(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        dX=polygon[0][i+1]-polygon[0][i]
        dY=polygon[1][i+1]-polygon[1][i]
        s=s+NOD(abs(dX),abs(dY))
    return s

#Площадь многоугольника методом трапеций
def trapSquare(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        a=polygon[0][i+1]-polygon[0][i]
        b=polygon[1][i+1]+polygon[1][i]
        s=s+a*b
    s=abs(s)/2
    return s

#Площадь многоугольника методом Гаусса
def gaussSquare(polygon):
    s=0; last=len(polygon[0])-1
    for i in range(last):
        s=s+polygon[0][i+1]*polygon[1][i]
        s=s-polygon[0][i]*polygon[1][i+1]
    s=s+polygon[0][last]*polygon[1][0]
    s=s-polygon[0][0]*polygon[1][last]
    s=0.5*abs(s)
    return s
```

```
#Вычисление по формуле Пика
#S = n + m/2 – 1
def inlineNodes(polygon):
    S=trapSquare(polygon)
    print("Площадь многоугольника",S)
    m=lineNodes(polygon)
    print("Количество точек на ребрах",m,"шт.")
    n=S-m/2+1
    return int(n)


#Координаты углов многоугольника
#двумерный список (x,y)
Plgn=[[7,3,2,4,7,8],[1,2,5,8,8,5]]
#"Замыкание" многоугольника
Plgn[0].append(Plgn[0][0])
Plgn[1].append(Plgn[1][0])

print("Количество точек внутри многоугольника" \
      ,inlineNodes(Plgn),"шт.")
```

```
Площадь многоугольника 31
Количество точек на ребрах 8 шт.
Количество точек внутри многоугольника 28 шт.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int NOD(int a, int b);//НОД
int lineNodes(int *X, int *Y, int len);//Количество точек на ребрах мнгоуголника
int inlineNodes(int *X, int *Y, int len);//Вычисление по формуле Пика
float trapSquare(int *X, int *Y, int len);//S многогольника методом трапеций
float gaussSquare(int *X, int *Y, int len);//S многогольника методом Гаусса


int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");                // Очистка консоли
    //Сразу задаем "замкнутый" массив координат углов
    int pointsX[]={7,3,2,4,7,8,7};
    int pointsY[]={1,2,5,8,8,5,1};

    //Количество углов в массиве координат
    int n=sizeof(pointsX)/sizeof(pointsX[0]);
    //printf("%f\n",(float)5/2);
    printf("Количество точек внутри многоугольника %i шт.\n", \
            inlineNodes(pointsX,pointsY,n));
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}
```

```
30    //Вычисление НОД
31    int NOD(int a, int b){
32        int t;
33        while (b != 0){
34            t = b;
35            b = a % b;
36            a = t;
37        }
38        return a;
39    }
40    //Вычисление площади методом трапеций
41    float trapSquare(int *X, int *Y, int len){
42        float s=0;
43        for (int i=0; i<len-1; i++){
44            int a= X[i+1]- X[i];
45            int b= Y[i+1]+ Y[i];
46            s=s+a*b;
47        }
48        s=(float)abs(s)/2;
49        return s;
50    }
51    //Вычисление площади методом Гаусса
52    float gaussSquare(int *X, int *Y, int len){
53        float s=0; int last=len-1;
54        for (int i=0; i<last; i++){
55            s=s+X[i+1]*Y[i];
56            s=s-X[i]*Y[i+1];
57        }
58        s=s+X[last]*Y[0];
59        s=s-X[0]*Y[last];
60        s=(float)abs(s)/2;
61        return s;
62    }
```

```
63    //Количество точек на ребрах многоугольника
64    int lineNodes(int *X, int *Y, int len){
65        int s=0;
66        for (int i=0; i<len-1; i++){
67            int dX=X[i+1]-X[i];
68            int dY=Y[i+1]-Y[i];
69            s=s+NOD(abs(dX),abs(dY));
70        }
71        return s;
72    }
73    //Вычисление по формуле Пика
74    //S = n + m/2 — 1
75    int inlineNodes(int *X, int *Y, int len){
76        float S=trapSquare(X,Y,len);
77        printf("Площадь многоугольника %.2f\n",S);
78        int m=lineNodes(X,Y,len);
79        printf("Количество точек на ребрах %i шт.\n",m);
80        int n=S-m/2+1;//Точек внутри многоугольника
81        return n;
82    }
```

```
c_pik
Площадь многоугольника 31.00
Количество точек на ребрах 8 шт.
Количество точек внутри многоугольника 28 шт.
```

ИНФОРМАТИКА

ВАРИАНТ 3.

1. С тех пор, как Иван изучил формулы в Excel, он ежедневно оттачивает свое мастерство. Сегодня он напечатал число 2 в ячейку A1, в соседнюю ячейку B1 он внес формулу =ОКРУГЛ(ОСТАТ(8;$A$1)+СТЕПЕНЬ(A1;2)-ЕСЛИ(A1>0;A1^2/3;A1/3);1) и растянул до ячейки E1. В соседней ячейке F1, он ввел формулу: =СЧЁТЕСЛИ(A1:E1;">20")+СУММЕСЛИ(A1:E1;">3";A1:E1). На следующей строке, начиная с ячейки A2, мальчик ввел формулу =ОКРУГЛ(ОСТАТ(B1;A1)-КОРЕНЬ(ABS(A1-B1))+ЕСЛИ(A1+B1<2;B1/A1;A1+3);2) и скопировал в диапазон B2:E2. В ячейку F2 он скопировал формулу из ячейки F1. На следующей строке он ввел целое число M в ячейку A3 и повторил действия, как для первой строки. А в четвертую (диапазон A4:E4) скопировал формулы со 2 строки, в ячейку F4 была вставлена формула из ячейки F3. В пятой строке Иван ввел некоторое целое число N в ячейку A5 и решил повторить действия, как для строки 1. В ячейку A6 он скопировал формулу из A2 и растянул с помощью автозаполнения до ячейки E6. В ячейку F6 была скопирована формула из F5. При этом мальчик получил следующий результат:

	A	B	C	D	E	F
1	2	2,7	4,9	16	170,7	192,6
2	4,86	6,42	5,87	17,26	190,92	226,33
3	?	6	24	384	98304	98721
4	4,27	4,76	8,03	74,08	98703,58	98796,72
5	?	10,7	76,3	3881,1	10041958	10045933
6	7,11	7	83,72	2268,21	10045873	10048242

Определите наименьшие целые числа M и N? Перечислите числа через точку с запятой. **(10 баллов)**

Решение:

	A	B	C	D	E	F
1	2	2,7	4,9	16	170,7	192,6
2	4,86	6,42	5,87	17,26	190,92	226,33
3	3	6	24	384	98304	98721
4	4,27	4,76	8,03	74,08	98703,58	98796,72
5	4	10,7	76,3	3881,1	10041958	10045933
6	7,11	7	83,72	2268,21	10045873	10048242
7						

Ответ: 3;4

2. Два сладкоежки решили сыграть в игру «27 шоколадных конфет». Играющие по очереди могут взять от одной до четырёх конфет. Кто не может сделать ход (конфет не осталось), проигрывает. Другими словами, выигрывает взявший последнюю конфету. Кто выигрывает при безошибочной игре – игрок, делающий первый ход, или игрок, делающий второй ход? Поясните алгоритм графически. Напишите обобщенный алгоритм для любого количества конфет. **(10 баллов)**

Решение:

Используется Стратегия_Дополни до пяти. Всегда выигрывает Игрок_1 (игрок, делающий ход первым). Суть Стратегия_Дополни до пяти:

Ход_1: Игрок_1(игрок, делающий ход первым) берет такое количество конфет, чтобы конфет осталось кратно 5 (в данном случае 2 конфеты, чтобы осталось 25).

Ход_2: Игрок_2 (игрок, делающий ход вторым) берет от 1 до 4 конфет (в данном случае конфет остаётся от 21 до 24).

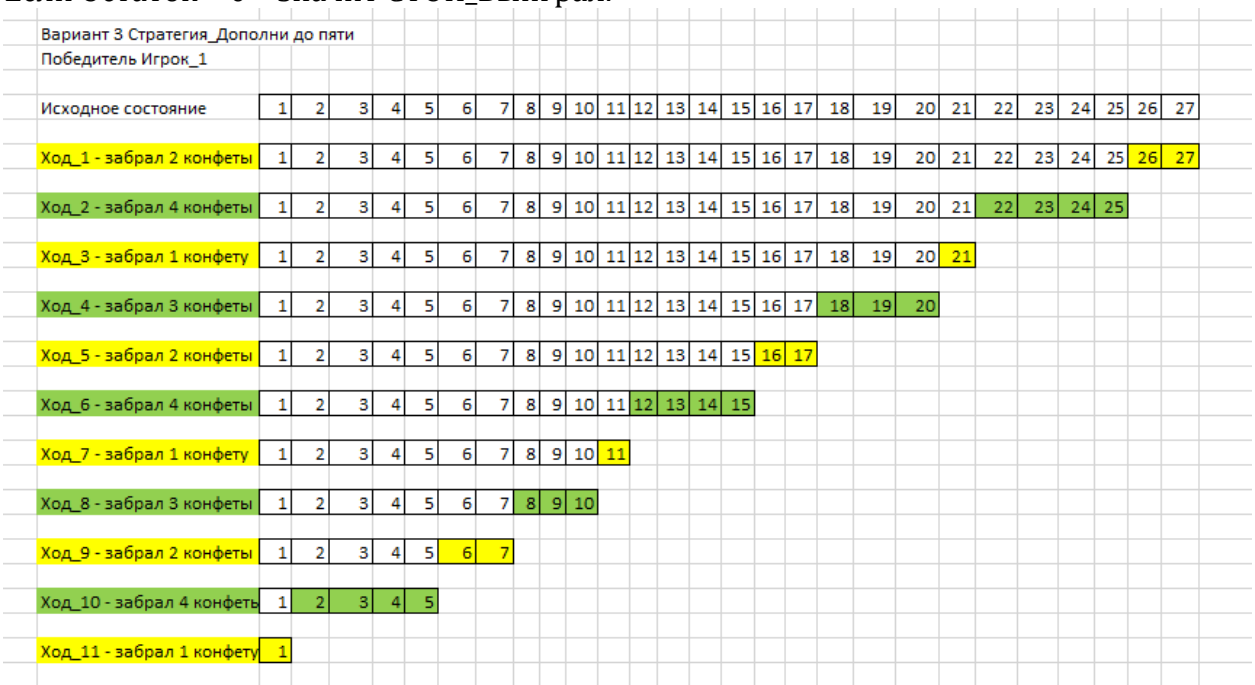
Ход_3: Игрок_1 должен дополнять ход первого до пяти конфет (Конфет в вазочке должно остаться 20). И так далее до Победа Игрок_1. Проиграть невозможно

Иллюстрация на рисунке ниже. Алгоритм подходит в общем случае, для N конфет., если N не делится на 5 без остатка, то первый может гарантировать себе выигрыш, дополняя ход противника до 5.

Алгоритм для Игрок_1 (обобщенный):

Шаг 1 Игрок _1 взял k конфет, чтобы (N-k) делилось на 5 без остатка
Шаг 2 Игрок_2 взял r конфет
Шаг 3 Игрок_1 взял (5-r) конфет

Если Остаток = 0 – значит СТОП_Выиграл.



3. Выражение $x \to \bar{y} \wedge z \vee \bar{x} \leftrightarrow z$ =ложь. Перечислите выражения (выражение), удовлетворяющие этому ответу и тому же набору значений *x, y, z* из приведенных ниже:

1). $x \oplus \bar{y} \to z \leftrightarrow x \wedge y \vee z$

2). $x \oplus y \leftrightarrow \bar{y} \vee z \to y \vee x$

3). $y \vee z \oplus \bar{x} \vee y \leftrightarrow x \rightarrow z$

4). $y \oplus z \rightarrow x \leftrightarrow y \wedge z \vee \bar{y}$

5). $x \vee \bar{y} \wedge z \oplus x \leftrightarrow \bar{y} \rightarrow z$

6). $y \rightarrow \bar{x} \leftrightarrow z \oplus x \vee y \wedge x$

В случае, если выражений несколько перечислите их в ответе через запятую. В процессе решения необходимо расставить порядок действий для всех выражений, а также построить таблицы истинности. При расстановке действий использовать в первую очередь приоритет операций, а потом уже очередность появления. **(10 баллов)**

Решение:

1. Необходимо расставить порядок действий для исходного примера:

1-е действие – отрицание *y*

2-е действие –отрицание *x*

3-е действие – конъюнкция результатов первого действия и *z*

4 – е действие – дизъюнкция результатов третьего и второго действий

5- действие – импликация *x* и результатов четвертого действия

6 – е действие – эквивалентность *z* и результатов пятого действия.

Построение таблицы истинности:

X	***y***	***z***	***1***	***2***	***3***	***4***	***5***	***6***
0	0	0	1	1	0	1	1	0
0	0	1	1	1	1	1	1	1
0	1	0	0	1	0	1	1	0
0	1	1	0	1	0	1	1	1
1	0	0	1	0	0	0	0	1
1	0	1	1	0	1	1	1	1
1	1	0	0	0	0	0	0	1
1	1	1	0	0	0	0	0	0

Значение «ложь» получено при следующих наборов значений:

x	***Y***	***z***
0	0	0
0	1	0
1	1	1

Исследование примеров и наборов значений:

1). $x \oplus \bar{y} \rightarrow z \leftrightarrow x \wedge y \vee z$

1-е действие – отрицание *y*,

2-е действие – конъюнкция *x, y*

3-е действие – дизъюнкция результатов второго действия и *z*

4-е действие – исключающее или *x* и результатов первого действия

5-е действие – импликация результатов четвертого действия и *z*

6-е действие – эквивалентность результатов 5-го действия и 3-го.

x	***y***	***z***	***1***	***2***	***3***	***4***	***5***	***6***
0	0	0	1	0	0	1	0	1
0	1	0	0	0	1	1	1	1
1	1	1	0	1	1	1	1	1

Данный пример не подходит.

2). Следующий шаг – исследование примера $x \oplus y \leftrightarrow \bar{y} \vee z \rightarrow y \vee x$

1-е действие – отрицание *y*

2-е действие – дизъюнкция результатов первого действия и *z*

3-е действие – дизъюнкция *y* и *x*

4-е действие – исключающее или *x, y*

5-е действие – импликация результатов второго и третьего действий

6-действие – эквивалентность результатов 4-го и 5-го действий

x	***y***	***Z***	***1***	***2***	***3***	***4***	***5***	***6***
0	0	0	1	1	0	0	0	1
0	1	0	0	0	1	1	1	1
1	1	1	0	1	1	0	1	0

Данный пример не подходит.

3). Необходимо исследовать следующий пример $y \vee z \oplus \bar{x} \vee y \leftrightarrow x \rightarrow z$:

1-е действие – отрицание *x*

2-е действие – дизъюнкция y, z

3-е действие – дизъюнкция результатов первого действия и y

4-е – действие исключающее или результатов 2 и 3 действий

5-е действие – импликация x, z

6-е действие – эквивалентность результатов 4-го действия и 5

x	***y***	***z***	***1***	***2***	***3***	***4***	***5***	***6***
0	0	0	1	0	1	1	1	1
0	1	0	1	1	0	0	1	1
1	1	1	0	1	1	1	1	1

Пример не подходит

4). Исследование четвертого примера $y \oplus z \rightarrow x \leftrightarrow y \wedge z \vee \bar{y}$

1-е действие – отрицание y

2-е действие – конъюнкция y, z

3-е действие – дизъюнкция результатов второго и первого действий

4-е действие – исключающее или y, z

5-е действие – импликация результатов четвертого действия и *x*

6-действие – эквивалентность результатов пятого действия третьего

x	***y***	***z***	***1***	***2***	***3***	***4***	***5***	***6***
0	0	0	1	0	1	0	1	1
0	1	0	0	0	0	1	0	1
1	1	1	0	1	1	0	1	1

Пример не подходит.

5). Исследование пятого примера $x \vee \bar{y} \wedge z \oplus x \leftrightarrow \bar{y} \rightarrow z$

1-е действие – отрицание *y* (оно же 2)

3-е действие – конъюнкция *z* и результатов первого действия

4-е действие – дизъюнкция *x* и результатов второго действия

5-е действие – исключающее или результатов 4-го действия и x

6-е действие – импликация результатов второго действия и *z*

7-е действие – эквивалентность результатов 5 и 6 действий.

x	***y***	***Z***	***1, 2***	***3***	***4***	***5***	***6***	***7***
0	0	0	1	0	0	0	0	1
0	1	0	0	0	0	0	1	0
1	1	1	0	0	1	0	1	0

Пример не подходит

6). Необходимо исследовать 6-ой пример $y \rightarrow \bar{x} \leftrightarrow z \oplus x \vee y \wedge x$

1-е действие – отрицание *x*

2-е действие – конъюнкция *y, x*

3-действие – дизъюнкция результатов второго действия и *x*

4-е действие – исключающее или *z* и результатов 3-го действия

5-е действие – импликация *y* и результатов 1-го действия

6-е действие – эквивалентность результатов 5-го и 4-го действий.

x	***y***	***Z***	***1***	***2***	***3***	***4***	***5***	***6***
0	0	0	1	0	0	1	1	0
0	1	0	1	0	0	1	1	0
1	1	1	0	1	1	0	0	0

Пример 6 подходит.

Ответ: 6.

4. Максим устроился работать помощником сетевого администратора в IT-отдел Университета. В первый же день работы он получил ответственное задание. Отдел

информационных технологий Университета использует диапазон частных IP-адресов, в котором настроены 6 сетей филиалов. Данные по распределению адресного пространства частично известны: в каждой сети компьютеру системного администратора назначен адрес 172.22.228.145, а размер сетевого префикса указан в таблице. При этом адресное пространство использовано максимально эффективно.

Максим обладает следующими знаниями:

- одному устройству должен соответствовать один IP-адрес вида XXX.XXX.XXX.XXX, при этом XXX – число в диапазоне [0:255], называемое октетом;
- устройства внутри одной сети должны беспрепятственно взаимодействовать друг с другом, взаимодействие устройств в разных сетях реализовано провайдером с помощью механизмов преобразования сетевых адресов;
- выделенный размер сети содержит минимально необходимое количество адресов;
- выделенный размер сети записывается как 2^n-2, где n – минимальное количество бит, которое может содержать все номера адресов сети;
- в каждом сегменте сети первый адрес зарезервирован под идентификатор сети, а последний – под широковещательную рассылку.

Название подсети	**IP-адрес ПК сетевого администратор**	**Адрес сети**	**Десятичная маска**	**Диапазон доступных адресов**	**Выделенный размер**	**Широковеща-тельный адрес**
Главный корпус	172.22.228.145/18					
Инженерный корпус	172.22.228.145/21					
Учебный центр №2	172.22.228.145/23					
Библиотека	172.22.228.145/25					
Пост охраны	172.22.228.145/29					
IT-отдел	172.22.228.145/30	172.22.228.144	255.255.255.252	172.22.228.145-172.22.228.146	2	172.22.228.147

Максим смог заполнить данные для IT-отдела. Заполнение остальной информации вызвало у помощника сетевого администратора затруднения. Необходимо помочь Максиму заполнить таблицу недостающими данными.
(**20 баллов**)

Решение:

Название подсети	**IP-адрес ПК сетевого администратора**	**Адрес сети**	**Десятичная маска**	**Диапазон доступных адресов**	**Выделенный размер**	**Широковеща-тельный адрес**

Главный корпус	172.22.228.145/18	**172.22.**192.0	255.255.192.0	.192.1-.255.254	16 382	.255.255
Инженерный корпус	172.22.228.145/21	**172.22.**224.0	255.255.248.0	.224.1-.231.254	2 046	.231.255
Учебный центр №2	172.22.228.145/23	**172.22.**228.0	255.255.254.0	.228.1-.229.254	510	.229.255
Библиотека	172.22.228.145/25	**172.22.228.**128	255.255.255.128	.129-.254	126	.255
Пост охраны	172.22.228.145/29	**172.22.228**.144	255.255.255.248	.145-.150	6	.151
IT-отдел	**172.22.228**.145/30	**172.22.228**.144	255.255.255.252	**172.22.228**.145-**172.22.228**.146	2	**172.22.228**.147

*полужирным выделена общая часть адреса до точки (по аналогии с последней строкой таблицы)

5. Гном носит драгоценные камни в плетеной корзинке наверх из шахты в кладовую. В кладовую из шахты ведет лестница из 9 ступенек. Шаг у гнома небольшой, поэтому у корзины две ручки, и он может наступать на каждую ступеньку, или перескакивать через одну ступеньку и становиться сразу на следующую. Каждый раз гном поднимается по лестнице по-новому, шагая по одной, по две ступеньки, или где по одной, а где по две. Сколько раз гном сможет подняться по лестнице из 9 ступенек, ни разу не повторив последовательность шагов? Напишите алгоритм и программу, позволяющую вычислить, сколько раз гном сможет подняться по лестнице из произвольного целого N (N>2) количества ступенек (в пределах типа данных языка программирования), чтобы ни разу не повторить последовательность шагов. Учтите, что возможности компьютера, на котором будет выполняться программа, ограничены: допустимо использовать до 10 переменных, массивы ограничены 10 ячейками соответствующего типа, а стек вмещает 10 вызовов и не расширяется. Алгоритм должен раскрывать решение на каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте уникальные особенности языка программирования (классы, методы, объекты, попарный обмен между переменными, специальные функции и библиотеки и др., кроме счетчиков циклов) для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для 9 ступенек, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача на вычисление последовательности чисел Фибоначчи.

A000045 Fibonacci numbers: F(n) = F(n-1) + F(n-2) with F(0) = 0 and F(1) = 1.
(Formerly M0692 N0256)

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75025, 121393, 196418, 317811, 514229, 832040, 1346269, 2178309, 3524578, 5702887, 9227465, 14930352, 24157817, 39088169, 63245986, 102334155 (list; graph; refs; listen; history; text;

https://oeis.org/A000045

В задаче требуется рассчитать число уникальных комбинаций шагов, которыми можно подняться по лестнице из **9** ступенек, используя шаги в 1, 2 или 1 и 2 ступеньки. **Ответом является 9-е** по счету число последовательности Фибоначчи (**$a_9 = 55$**), считая, что **$a_1 = 1$**, **$a_2 = 2$**.

Объяснение 1. Обозначим число комбинаций как a_n, где n – число ступенек, n>2.
$a_1 = 1$ способ –> на лестницу из одной ступеньки можно подняться только одним способом - шагом в 1 ступеньку.
$a_2 = 2$ способа –> 1+1 ступенька; сразу 2 ступеньки.
$a_3 = 3$ –> 1+1+1; 1+2; 2+1 ступеньки.
$a_4 = 5$ –> 1+1+1+1; 1+1+2; 1+2+1; 2+1+1; 2+2 ступеньки.
$a_4 = a_3 + a_2$; $a_3 = a_2 + a_1$; ==> **$a_n = a_{n-1} + a_{n-2}$;**
$a_5 = a_4 + a_3 = 5+3=8$; $a_6 = a_5 + a_4 = 8+5=13$; $a_7 = a_6 + a_5 = 13+8=21$;
$a_8 = a_7 + a_6 = 21+13=34$; $a_9 = a_8 + a_7 = 34+21=55$.
Ответ: $a_9 = 55$ способов.
Объяснение 2. Обозначим число комбинаций как a_n, где n – число ступенек, n>2. Можно начать подъем с шага в 1 ступеньку. Тогда останется a_{n-1} комбинаций подъема по лестнице. Если начать подъем с шага в 2 ступеньки, то останется a_{n-2} комбинаций подъема по лестнице. Первый шаг входит в любую из комбинаций шагов. Таким образом количество комбинаций определяется суммой количества оставшихся ступенек для каждого варианта первого шага и размером первого шага, т.е. **$a_n = a_{n-1} + a_{n-2}$.**

Задачу можно решить, например *рекурсией* (*неправильно*, ограничение по стеку вызовов при больших N), **динамическим программированием (ожидаемое решение)**, с помощью умножения матриц (допустимое, но избыточное решение), формулы Бине (допустимое, но избыточное решение). Ниже приведен пример решения с помощью динамического программирования на Python 3.82 и C.

```
#Функция Фибоначи с обменом
#Экономит память, хранит только два
#предыдущих значения
def fib_ult(n):#Для вывода по шагам
    #Значение функции на два и один шаг назад
    back2=1; back1=2
    if n<2:
        print("Ступенек не может быть меньше 2-х!")
        return 0
    for i in range(2,n-1):
        next=back1+back2
        print("Шаг ",i-1," back2=",back2," ",\
              "back1=",back1," ","next=",next,sep='')
        back2=back1#обмен содержимым переменных
        back1=next#обмен содержимым переменных
    print("Шаг ",i," back2=",back2," ",\
          "back1=",back1," ","next=",back1+back2,sep='')
    return back1+back2
fibNumb=int(input("Введите количество ступенек:\n"))
print("Ответ: необходимо подняться ",fib_ult(fibNumb)," раз.",sep='')
```

```
Введите количество ступенек:
9
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Шаг 6 back2=13 back1=21 next=34
Шаг 7 back2=21 back1=34 next=55
Ответ: необходимо подняться 55 раз.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int fibNumb;//Порядковый номер в ряу Фибоначчи
unsigned long long answ;//Ответ
unsigned long long fib_ultim(int n);//Сигнатура

int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");                // Очистка консоли
    printf("Введите количество ступенек:\n");
    scanf("%d",&fibNumb);
    answ=fib_ultim(fibNumb-1);//Так как нумерация массивов с нуля
    if (answ==0){//Выход если ступенек меньше 2-х
        getchar();//Задержка для отображения вывода
        printf("Нажмите любую кнопку для завершения.\n");
        int ch = getch();
        return (EXIT_SUCCESS);
    }
    printf("Ответ: необходимо подняться %llu раз.  \n",answ);
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}
```

```
30    unsigned long long fib_ultim(int n)//Семантика
31    {
32        int i;//счетчик
33        //Значение функции на два и один шаг назад
34        unsigned long long back2=1, back1=2, next;
35        if (n<2){//Выход если ступенек меньше 2-х
36            printf("Ступенек не может быть меньше 2-х!\n");
37            return 0;
38        }
39        for (i=2; i<n; i++)//при n<2 цикл не выполняется
40        {
41            next=back1+back2;
42            printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
43                    i-1, back2,back1,next);
44            back2=back1;//обмен содержимым переменных
45            back1=next;//обмен содержимым переменных
46        }
47        printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
48                i-1, back2,back1,back1+back2);
49        return back1+back2;
50    }
```

```
Введите количество ступенек:
9
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Шаг 6 back2=13 back1=21 next=34
Шаг 7 back2=21 back1=34 next=55
Ответ: необходимо подняться 55 раз.
Нажмите любую кнопку для завершения.
```

6. Гном вынес все драгоценные камни в плетеной корзинке с двумя ручками наверх из шахты в кладовую по лестнице. Теперь гном решил застелить пол шахты квадратными кварцевыми плитками, инкрустировав перекрестие плиток драгоценным камнем. Стык пола со стенами гном решил не украшать. Сторона плитки – 1 шаг гнома. Гном обмерил шахту шагами и составил её план. На плане гном отметил границы шахты линиями, соединяющими углы стен. Координаты углов шахты в шагах гнома: {1,1; 3,3; 2,5; 2,8; 6,6; 5,2}. Сколько понадобится драгоценных камней гному, чтобы инкрустировать перекрестие плиток? Напишите

алгоритм и программу, позволяющую вычислить, сколько понадобится драгоценных камней гному, чтобы инкрустировать перекрестие плиток в шахте с произвольными целыми координатами углов. Алгоритм должен раскрывать решение на каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте специальные функции и библиотеки для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Допустимо задать готовые координаты в коде основной функции. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для координат углов шахты в шагах гнома {1,1; 3,3; 2,5; 2,8; 6,6; 5,2}, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача из области вычислительной геометрии на вычисление количества точек с целочисленными координатами, лежащих внутри (но не на границе) многоугольника, заданного координатами своих вершин.

В задаче требуется рассчитать количество точек с целыми координатами, попадающими внутрь многоугольника, не считая точки, лежащие точно на ребрах многоугольника. Ответ можно проверить визуально. Ожидается увидеть алгоритм решения в виде рисунка/рисунков и формул Пика и НОД. **Ответ**: **14** точек.

Теория_1. Для любого многоугольника с целочисленными координатами вершин справедлива **формула Пика**: S = n + m/2 — 1, где S – площадь многоугольника, n – количество целых точек лежащих строго внутри многоугольника, m – количество целых точек, лежащих на границе многоугольника. Площадь многоугольника S вычисляется методом трапеций (**ожидаемое решение**) или по формуле Гаусса (“шнуровки”, **допустимое равнозначное решение**). Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1, где катет_1,2 - катеты прямоугольного треугольника, образованного на каждом ребре многоугольника (гипотенуза). Искомое количество целых точек n, лежащих строго внутри многоугольника, вычисляется как n = S - m/2 + 1.

Теория_2. Площадь многоугольника. Для вычисления площади многоугольника необходимо “замкнуть” периметр, добавив последнюю_вершину = первая_вершина. Формулы площадей показаны ниже.

$$S_{Гаусс} = \frac{1}{2}\left|\sum_{i=1}^{n-1} x_i y_{i+1} + x_n y_1 - \sum_{i=1}^{n-1} x_{i+1} y_i - x_1 y_n\right|$$

$$S_{трап} = \frac{1}{2}\left|\sum_{i=1}^{n-1} (x_{i+1} - x_i)(y_{i+1} + y_i)\right|$$

Теория_3. Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1. Катеты прямоугольного треугольника для каждого ребра многоугольника вычисляются как разности координат вершин, образующих ребро. Для вертикального или горизонтального ребра один из катетов будет равен нулю. Необходимо последовательно перебрать все вершины за исключением “замыкающей”, вычислить НОД катетов и просуммировать их. Алгоритм вычисления показан ниже.

```
lineNodes(polygon) :=
    "Количество узлов, лежащих на ребрах многоугольника."
    "Сумма НОД катетов прямоугольных трегольников"
    "для каждого ребра-гипотенузы"
    "Замкнутый контур. Последняя_вершина=первая_вершина"
    s ← 0
    for i ∈ ORIGIN .. rows(polygon) − 1
        s ← s + gcd(polygon_{i+1,1} − polygon_{i,1}, polygon_{i+1,2} − polygon_{i,2})
    s
```

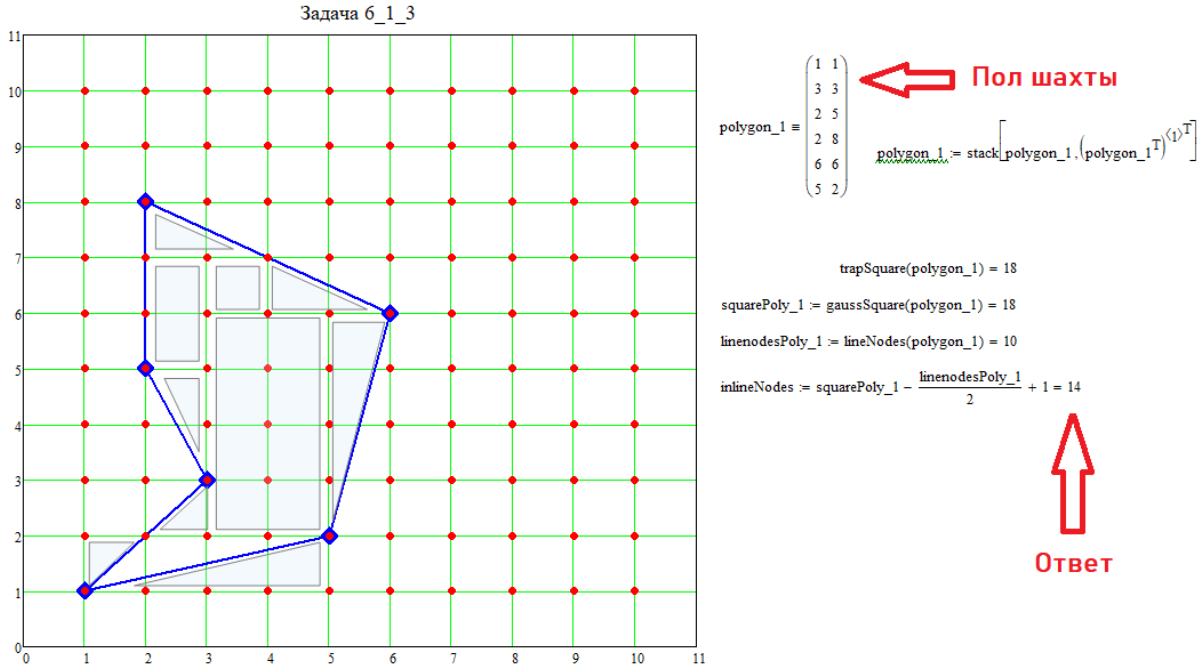
Алгоритм вычисления площади многоугольника **методом трапеций**:

```
trapSquare(polygon) :=
    "Площадь многоугольника методом трапеций"
    "Замкнутый контур. Последняя_вершина=первая_вершина"
    s ← 0
    for i ∈ ORIGIN .. rows(polygon) − 1
        a ← polygon_{i+1,1} − polygon_{i,1}
        b ← polygon_{i+1,2} + polygon_{i,2}
        s ← s + a·b
    s ← 0.5 |s|
    s
```

Алгоритм вычисления площади многоугольника **методом Гаусса**:

```
gaussSquare(polygon) :=
    "Площадь многоугольника методом Гаусса ("шнуровки")"
    "Замкнутый контур. Последняя_вершина=первая_вершина"
    s ← 0
    for i ∈ ORIGIN .. rows(polygon) − 1
        s ← s + polygon_{i+1,1}·polygon_{i,2}
        s ← s − polygon_{i,1}·polygon_{i+1,2}
    s ← s + polygon_{rows(polygon),1}·polygon_{1,2}
    s ← s − polygon_{1,1}·polygon_{rows(polygon),2}
    s ← 0.5 |s|
    s
```

Графическое решение и ответ показаны ниже.



Ниже приведен пример решения на Python 3.82 и C.

```python
#Формула Пика
#НОД
def NOD(a,b):
    while b != 0:
        a, b = b, a % b
    return a
#Количество узлов, лежащих
#на ребрах многоугольника
def lineNodes(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        dX=polygon[0][i+1]-polygon[0][i]
        dY=polygon[1][i+1]-polygon[1][i]
        s=s+NOD(abs(dX),abs(dY))
    return s

#Площадь многоугольника методом трапеций
def trapSquare(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        a=polygon[0][i+1]-polygon[0][i]
        b=polygon[1][i+1]+polygon[1][i]
        s=s+a*b
    s=abs(s)/2
    return s

#Площадь многоугольника методом Гаусса
def gaussSquare(polygon):
    s=0; last=len(polygon[0])-1
    for i in range(last):
        s=s+polygon[0][i+1]*polygon[1][i]
        s=s-polygon[0][i]*polygon[1][i+1]
    s=s+polygon[0][last]*polygon[1][0]
    s=s-polygon[0][0]*polygon[1][last]
    s=0.5*abs(s)
    return s
```

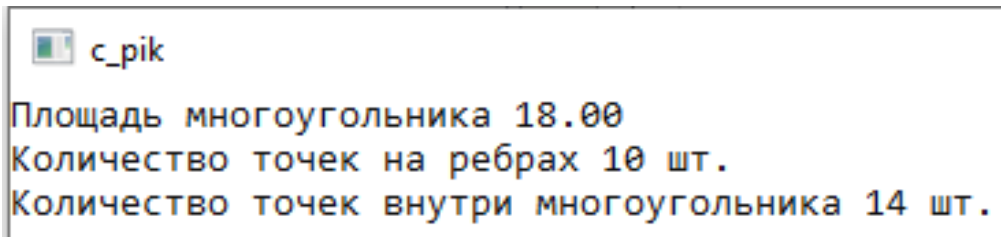
```
#Вычисление по формуле Пика
#S = n + m/2 – 1
def inlineNodes(polygon):
    S=trapSquare(polygon)
    print("Площадь многоугольника",S)
    m=lineNodes(polygon)
    print("Количество точек на ребрах",m,"шт.")
    n=S-m/2+1
    return int(n)


#Координаты углов многоугольника
#двумерный список (x,y)
Plgn=[[1,3,2,2,6,5],[1,3,5,8,6,2]]
#"Замыкание" многоугольника
Plgn[0].append(Plgn[0][0])
Plgn[1].append(Plgn[1][0])

print("Количество точек внутри многоугольника" \
      ,inlineNodes(Plgn),"шт.")
```

```
Площадь многоугольника 18
Количество точек на ребрах 10 шт.
Количество точек внутри многоугольника 14 шт.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int NOD(int a, int b);//НОД
int lineNodes(int *X, int *Y, int len);//Количество точек на ребрах многоугольника
int inlineNodes(int *X, int *Y, int len);//Вычисление по формуле Пика
float trapSquare(int *X, int *Y, int len);//S многогольника методом трапеций
float gaussSquare(int *X, int *Y, int len);//S многогольника методом Гаусса


int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");              // Очистка консоли
    //Сразу задаем "замкнутый" массив координат углов
    int pointsX[]={1,3,2,2,6,5,1};
    int pointsY[]={1,3,5,8,6,2,1};

    //Количество углов в массиве координат
    int n=sizeof(pointsX)/sizeof(pointsX[0]);
    printf("Количество точек внутри многоугольника %i шт.\n", \
            inlineNodes(pointsX,pointsY,n));
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}
```

```
30    //Вычисление НОД
31    int NOD(int a, int b){
32        int t;
33        while (b != 0){
34            t = b;
35            b = a % b;
36            a = t;
37        }
38        return a;
39    }
40    //Вычисление площади методом трапеций
41    float trapSquare(int *X, int *Y, int len){
42        float s=0;
43        for (int i=0; i<len-1; i++){
44            int a= X[i+1]- X[i];
45            int b= Y[i+1]+ Y[i];
46            s=s+a*b;
47        }
48        s=(float)abs(s)/2;
49        return s;
50    }
51    //Вычисление площади методом Гаусса
52    float gaussSquare(int *X, int *Y, int len){
53        float s=0; int last=len-1;
54        for (int i=0; i<last; i++){
55            s=s+X[i+1]*Y[i];
56            s=s-X[i]*Y[i+1];
57        }
58        s=s+X[last]*Y[0];
59        s=s-X[0]*Y[last];
60        s=(float)abs(s)/2;
61        return s;
62    }
```

```
63    //Количество точек на ребрах многоугольника
64    int lineNodes(int *X, int *Y, int len){
65        int s=0;
66        for (int i=0; i<len-1; i++){
67            int dX=X[i+1]-X[i];
68            int dY=Y[i+1]-Y[i];
69            s=s+NOD(abs(dX),abs(dY));
70        }
71        return s;
72    }
73    //Вычисление по формуле Пика
74    //S = n + m/2 — 1
75    int inlineNodes(int *X, int *Y, int len){
76        float S=trapSquare(X,Y,len);
77        printf("Площадь многоугольника %.2f\n",S);
78        int m=lineNodes(X,Y,len);
79        printf("Количество точек на ребрах %i шт.\n",m);
80        int n=S-m/2+1;//Точек внутри многоугольника
81        return n;
82    }
```


c_pik
Площадь многоугольника 18.00
Количество точек на ребрах 10 шт.
Количество точек внутри многоугольника 14 шт.

ИНФОРМАТИКА

ВАРИАНТ 4.

1. С тех пор, как Иван изучил формулы в Excel, он ежедневно оттачивает свое мастерство. Сегодня он напечатал целое число M в ячейку A1, в соседнюю ячейку B1 он внес формулу =ОКРУГЛ(ОСТАТ(8;$A$1)+СТЕПЕНЬ(A1;2)-ЕСЛИ(A1>0;A1^2/3;A1/3);1) и растянул до ячейки E1. В соседней ячейке F1, он ввел формулу: =СЧЁТЕСЛИ(A1:E1;">20")+СУММЕСЛИ(A1:E1;">3";A1:E1). На следующей строке, начиная с ячейки A2, мальчик ввел формулу =ОКРУГЛ(ОСТАТ(B1;A1)-КОРЕНЬ(ABS(A1-B1))+ЕСЛИ(A1+B1<2;B1/A1;A1+3);2) и скопировал в диапазон B2:E2. В ячейку F2 он скопировал формулу из ячейки F1. На следующей строке он ввел целое число N в ячейку A3 и повторил действия, как для первой строки. А в четвертую (диапазон A4:E4) скопировал формулы со 2 строки, в ячейку F4 была вставлена формула из ячейки F3. В пятой строке Иван ввел число 3 в ячейку A5 и решил повторить действия, как для строки 1. В ячейку A6 он скопировал формулу из A2 и растянул с помощью автозаполнения до ячейки E6. В ячейку F6 была скопирована формула из F5. При этом мальчик получил следующий результат:

	A	B	C	D	E	F
1	?	2,7	4,9	16	170,7	192,6
2	4,86	6,42	5,87	17,26	190,92	226,33
3	?	10,7	76,3	3881,1	10041958	10045933
4	7,11	7	83,72	2268,21	10045873	10048242
5	3	6	24	384	98304	98721
6	4,27	4,76	8,03	74,08	98703,58	98796,72

Определите наименьшие целые числа M и N? Перечислите числа через точку с запятой. **(10 баллов)**

Решение:

	A	B	C	D	E	F
1	2	2,7	4,9	16	170,7	192,6
2	4,86	6,42	5,87	17,26	190,92	226,33
3	4	10,7	76,3	3881,1	10041958	10045933
4	7,11	7	83,72	2268,21	10045873	10048242
5	3	6	24	384	98304	98721
6	4,27	4,76	8,03	74,08	98703,58	98796,72

Ответ: 2;4.

2. Два сладкоежки решили сыграть в игру «33 шоколадных конфеты». Играющие по очереди могут взять от одной до четырёх конфет. Кто не может сделать ход (конфет не осталось), проигрывает. Другими словами, выигрывает взявший последнюю конфету. Кто выигрывает при безошибочной игре – игрок, делающий первый ход, или игрок, делающий второй ход? Поясните алгоритм графически. Напишите обобщенный алгоритм для любого количества конфет. **(10 баллов)**

Решение:

Используется Стратегия_Дополни до пяти. Всегда выигрывает Игрок_1 (игрок, делающий ход первым). Суть Стратегия_Дополни до пяти:

Ход_1: Игрок_1(игрок, делающий ход первым) берет такое количество конфет, чтобы конфет осталось кратно 5 (в данном случае 3 конфеты, чтобы осталось 30).

Ход_2: Игрок_2 (игрок, делающий ход вторым) берет от 1 до 4 конфет (в данном случае конфет остаётся от 26 до 29).

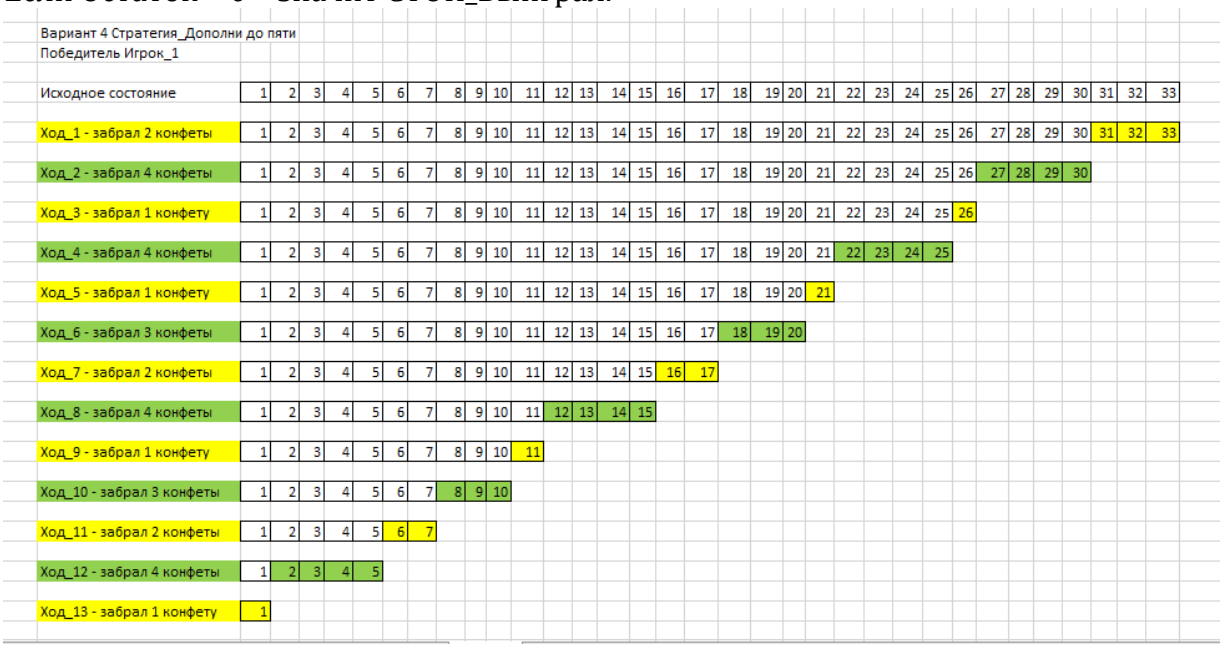
Ход_3: Игрок_1 должен дополнять ход первого до пяти конфет (Конфет в вазочке должно остаться 25). И так далее до Победа Игрок_1. Проиграть невозможно

Иллюстрация на рисунке ниже. Алгоритм подходит в общем случае, для N конфет., если N не делится на 5 без остатка, то первый может гарантировать себе выигрыш, дополняя ход противника до 5.

Алгоритм для Игрок_1 (обобщенный):

Шаг 1 Игрок _1 взял k конфет, чтобы (N-k) делилось на 5 без остатка
Шаг 2 Игрок_2 взял r конфет
Шаг 3 Игрок_1 взял (5-r) конфет

Если Остаток = 0 – значит СТОП_Выиграл.



3. Выражение $x \leftrightarrow \bar{y} \rightarrow z \leftrightarrow x \vee \bar{y}$ =*ложь*. Перечислите выражения (выражение), удовлетворяющие этому ответу и тому же набору значений *x, y, z* из приведенных ниже:

1). $x \rightarrow \bar{y} \vee z \leftrightarrow x \wedge y \oplus z$

2). $x \rightarrow y \leftrightarrow \bar{y} \oplus z \wedge y \vee x$

3). $y \to z \wedge \bar{x} \vee y \leftrightarrow x \wedge z$

4). $y \oplus z \to x \vee y \wedge z \to \bar{y}$

5). $x \vee \bar{y} \vee z \to x \leftrightarrow \bar{y} \oplus z$

6). $y \wedge x \leftrightarrow z \oplus x \to y \vee x$

В случае, если выражений несколько перечислите их в ответе через запятую. В процессе решения необходимо расставить порядок действий для всех выражений, а также построить таблицы истинности. При расстановке действий использовать в первую очередь приоритет операций, а потом уже очередность появления. **(10 баллов)**

Решение:

Необходимо исследовать пример $x \leftrightarrow \bar{y} \to z \leftrightarrow x \vee \bar{y}$, чтобы понять какие наборы значений дадут в результате «ложь».

1-е действие – отрицание y (оно же 2-е),

3-е действие – дизъюнкция x и результатов 2 действия

4-е действие – импликация результатов 1-го действия и z

5-е действие – эквивалентность x и результатов 4-го действия

6-е действие – эквивалентность 5-го действия и результатов 3-го

x	y	z	1, 2	3	4	5	6
0	0	0	1	1	0	1	1
0	0	1	1	1	1	0	0
0	1	0	0	0	1	0	1
0	1	1	0	0	1	0	1
1	0	0	1	1	0	0	0
1	0	1	1	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	0	1	1	1	1

Для следующих наборов значений соответствует набор «ложь»:

X	y	Z
0	0	1
1	0	0

1). Необходимо исследовать наборы значений примера $x \to \bar{y} \vee z \leftrightarrow x \wedge y \oplus z$:

1-е действие – отрицание y

2-е действие – конъюнкция x, y

3-е действие – дизъюнкция результатов первого действия и z

4-е действие – исключающее или результатов 2-го действия и z

5-е действие – импликация x и результатов 3-го действия

6-е действия – эквивалентность действий 5 и 4.

x	y	z	1	2	3	4	5	6
0	0	1	1	0	1	1	1	1
1	0	0	1	0	1	0	1	0

Пример не подходит

2). Исследование второго примера $x \to y \leftrightarrow \bar{y} \oplus z \wedge y \vee x$:

1-е действие – отрицание y

2-е действие – конъюнкция z, y

3-е действие – дизъюнкция результатов 2-го действия и x

4-е действие – исключающее или результатов 1-го и 3-го действий

5-е действие – импликация x, y

6-е действие – эквивалентность результатов 5-го и 4-го действий.

x	y	z	1	2	3	4	5	6
0	0	1	1	0	0	0	1	0
1	0	0	1	0	1	1	0	0

Пример подходит.

3). $y \to z \wedge \bar{x} \vee y \leftrightarrow x \wedge z$

1-е действие – отрицание x

2-е действие – конъюнкция z и результатов 1-го действия

3-е действие – конъюнкция x, z

4-е действие - дизъюнкция 2-го действия и y

5-е действие – импликация y и результатов 4-го действие

6-е действие – эквивалентность 5-го действия и 3-го

x	y	z	1	2	3	4	5	6
0	0	1	1	1	0	1	1	0
1	0	0	1	0	0	0	1	0

Пример подходит

4). $y \oplus z \to x \vee y \wedge z \to \bar{y}$

1-е действие – отрицание y

2-е действие – конъюнкция y, z

3-е действие – дизъюнкция результатов 2-го действия и x

4-е действие – исключающее или y, z

5-е действие – импликация результатов 4-го действия и 3-го

6-е действие – импликация результатов 5-го действия и 1-го

x	y	z	1	2	3	4	5	6
0	0	1	1	0	0	1	0	1

1	0	0	1	0	1	0	1	1

Пример не подходит.

5). $x \vee \bar{y} \vee z \to x \leftrightarrow \bar{y} \oplus z$

1-е действие – отрицание y (оно же и 2-е)

3-е действие – дизъюнкция x и первого действия

4-е действие – дизъюнкция результатов 3-го действия и z

5-е действие – исключающее или результатов 2-го действия и z

6-е действие – импликация результатов 5-го действия и z

7-е действие – эквивалентность результатов 6-го действия и 5-го

x	y	z	1, 2	3	4	5	6	7
0	0	1	1	1	1	0	0	1
1	0	0	1	1	1	1	1	1

Пример не подходит

6). $y \wedge x \leftrightarrow z \oplus x \to y \vee x$

1-е действие – конъюнкция y, x

2-е действие – дизъюнкция y, x

3-е действие – исключающее или z, x

4-е действие – импликация результатов 3 и 4 действия

5-е действие – эквивалентность результатов 1-го и 4-го действий

x	y	z	1	2	3	4	5
0	0	1	0	0	1	0	1
1	0	0	0	1	1	1	0

Пример не подходит.

Ответ: 2, 3.

4. Максим устроился работать помощником сетевого администратора в IT-отдел Университета. В первый же день работы он получил ответственное задание. Отдел информационных технологий Университета использует диапазон частных IP-адресов, в котором настроены 6 сетей филиалов. Данные по распределению адресного пространства частично известны: в каждой сети компьютеру системного администратора назначен адрес 172.24.228.145, а размер сетевого префикса указан в таблице. При этом адресное пространство использовано максимально эффективно.
Максим обладает следующими знаниями:

- одному устройству должен соответствовать один IP-адрес вида XXX.XXX.XXX.XXX, при этом XXX – число в диапазоне [0:255], называемое октетом;

- устройства внутри одной сети должны беспрепятственно взаимодействовать друг с другом, взаимодействие устройств в разных сетях реализовано провайдером с помощью механизмов преобразования сетевых адресов;
- выделенный размер сети содержит минимально необходимое количество адресов;
- выделенный размер сети записывается как 2^n-2, где n – минимальное количество бит, которое может содержать все номера адресов сети;
- в каждом сегменте сети первый адрес зарезервирован под идентификатор сети, а последний – под широковещательную рассылку.

Название подсети	**IP-адрес ПК сетевого администратор**	**Адрес сети**	**Десятичная маска**	**Диапазон доступных адресов**	**Выделенный размер**	**Широковеща-тельный адрес**
Главный корпус	172.24.228.145/18					
Инженерный корпус	172.24.228.145/21					
Учебный центр №2	172.24.228.145/23					
Библиотека	172.24.228.145/25					
Пост охраны	172.24.228.145/29					
IT-отдел	172.24.228.145/30	172.24.228.144	255.255.255.252	172.24.228.145-172.24.228.146	2	172.24.228.147

Максим смог заполнить данные для IT-отдела. Заполнение остальной информации вызвало у помощника сетевого администратора затруднения. Необходимо помочь Максиму заполнить таблицу недостающими данными.

(**20 баллов**)

Решение:

Название подсети	**IP-адрес ПК сетевого администратора**	**Адрес сети**	**Десятичная маска**	**Диапазон доступных адресов**	**Выделенный размер**	**Широковеща-тельный адрес**
Главный корпус	172.24.228.145/18	**172.24.**192.0	255.255.192.0	.192.1-.255.254	16 382	.255.255
Инженерный корпус	172.24.228.145/21	**172.24.**224.0	255.255.248.0	.224.1-.231.254	2 046	.231.255
Учебный центр №2	172.24.228.145/23	**172.24.**228.0	255.255.254.0	.228.1-.229.254	510	.229.255
Библиотека	172.24.228.145/25	**172.24.228.**128	255.255.255.128	.129-.254	126	.255
Пост охраны	172.24.228.145/29	**172.24.228**.144	255.255.255.248	.145-.150	6	.151

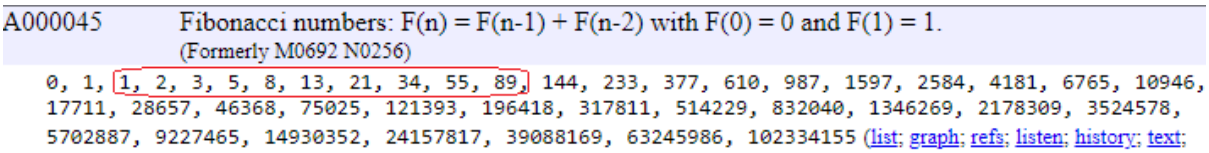
IT-отдел	**172.24.228.**145/30	**172.24.228**.144	255.255.255.252	**172.24.228.**145-**172.24.228.**146	2	**172.24.228**.147

*полужирным выделена общая часть адреса до точки (по аналогии с последней строкой таблицы)

5. Гном носит драгоценные камни в плетеной корзинке наверх из шахты в кладовую. В кладовую из шахты ведет лестница из 10 ступенек. Шаг у гнома небольшой, поэтому у корзины две ручки, и он может наступать на каждую ступеньку, или перескакивать через одну ступеньку и становиться сразу на следующую. Каждый раз гном поднимается по лестнице по-новому, шагая по одной, по две ступеньки, или где по одной, а где по две. Сколько раз гном сможет подняться по лестнице из 10 ступенек, ни разу не повторив последовательность шагов? Напишите алгоритм и программу, позволяющую вычислить, сколько раз гном сможет подняться по лестнице из произвольного целого N (N>2) количества ступенек (в пределах типа данных языка программирования), чтобы ни разу не повторить последовательность шагов. Учтите, что возможности компьютера, на котором будет выполняться программа, ограничены: допустимо использовать до 10 переменных, массивы ограничены 10 ячейками соответствующего типа, а стек вмещает 10 вызовов и не расширяется. Алгоритм должен раскрывать решение на каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте уникальные особенности языка программирования (классы, методы, объекты, попарный обмен между переменными, специальные функции и библиотеки и др., кроме счетчиков циклов) для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для 10 ступенек, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача на вычисление последовательности чисел Фибоначчи.


A000045 Fibonacci numbers: F(n) = F(n-1) + F(n-2) with F(0) = 0 and F(1) = 1.
(Formerly M0692 N0256)

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75025, 121393, 196418, 317811, 514229, 832040, 1346269, 2178309, 3524578, 5702887, 9227465, 14930352, 24157817, 39088169, 63245986, 102334155 (list; graph; refs; listen; history; text;

https://oeis.org/A000045

В задаче требуется рассчитать число уникальных комбинаций шагов, которыми можно подняться по лестнице из **10** ступенек, используя шаги в 1, 2 или 1 и 2 ступеньки. **Ответом является 10-е** по счету число последовательности Фибоначчи ($\mathbf{a_{10} = 89}$), считая, что $\mathbf{a_1 = 1}$, $\mathbf{a_2 = 2}$.

Объяснение 1. Обозначим число комбинаций как a_n, где n – число ступенек, n>2.
$\mathbf{a_1 = 1}$ **способ** –> на лестницу из одной ступеньки можно подняться только одним способом - шагом в 1 ступеньку.
$\mathbf{a_2 = 2}$ **способа** –> 1+1 ступенька; сразу 2 ступеньки.
$a_3 = 3$ –> 1+1+1; 1+2; 2+1 ступеньки.
$a_4 = 5$ –> 1+1+1+1; 1+1+2; 1+2+1; 2+1+1; 2+2 ступеньки.
$a_4 = a_3 + a_2$; $a_3 = a_2 + a_1$; ==> $\mathbf{a_n = a_{n-1} + a_{n-2}}$;
$a_5 = a_4 + a_3 = 5+3=8$; $a_6 = a_5 + a_4 = 8+5=13$; $a_7 = a_6 + a_5 = 13+8=21$;
$a_8 = a_7 + a_6 = 21+13=34$; $a_9 = a_8 + a_7 = 34+21=55$; $a_{10} = a_9 + a_8 = 55+34=89$.
Ответ: $\mathbf{a_{10} = 89}$ **способов.**
Объяснение 2. Обозначим число комбинаций как a_n, где n – число ступенек, n>2. Можно начать подъем с шага в 1 ступеньку. Тогда останется a_{n-1} комбинаций подъема по

лестнице. Если начать подъем с шага в 2 ступеньки, то останется a_{n-2} комбинаций подъема по лестнице. Первый шаг входит в любую из комбинаций шагов. Таким образом количество комбинаций определяется суммой количества оставшихся ступенек для каждого варианта первого шага и размером первого шага, т.е. $\mathbf{a_n = a_{n-1} + a_{n-2}}$.

Задачу можно решить, например *рекурсией* (*неправильно*, ограничение по стеку вызовов при больших N), **динамическим программированием (ожидаемое решение)**, с помощью умножения матриц (допустимое, но избыточное решение), формулы Бине (допустимое, но избыточное решение). Ниже приведен пример решения с помощью динамического программирования на Python 3.82 и C.

```
#Функция Фибоначи с обменом
#Экономит память, хранит только два
#предыдущих значения
def fib_ult(n):#Для вывода по шагам
    #Значение функции на два и один шаг назад
    back2=1; back1=2
    if n<2:
        print("Ступенек не может быть меньше 2-х!")
        return 0
    for i in range(2,n-1):
        next=back1+back2
        print("Шаг ",i-1," back2=",back2," ",\
              "back1=",back1," ","next=",next,sep='')
        back2=back1#обмен содержимым переменных
        back1=next#обмен содержимым переменных
    print("Шаг ",i," back2=",back2," ",\
          "back1=",back1," ","next=",back1+back2,sep='')
    return back1+back2
fibNumb=int(input("Введите количество ступенек:\n"))
print("Ответ: необходимо подняться ",fib_ult(fibNumb)," раз.",sep='')
```

```
Введите количество ступенек:
10
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Шаг 6 back2=13 back1=21 next=34
Шаг 7 back2=21 back1=34 next=55
Шаг 8 back2=34 back1=55 next=89
Ответ: необходимо подняться 89 раз.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int fibNumb;//Порядковый номер в ряу Фибоначчи
unsigned long long answ;//Ответ
unsigned long long fib_ultim(int n);//Сигнатура

int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");                // Очистка консоли
    printf("Введите количество ступенек:\n");
    scanf("%d",&fibNumb);
    answ=fib_ultim(fibNumb-1);//Так как нумерация массивов с нуля
    if (answ==0){//Выход если ступенек меньше 2-х
        getchar();//Задержка для отображения вывода
        printf("Нажмите любую кнопку для завершения.\n");
        int ch = getch();
        return (EXIT_SUCCESS);
    }
    printf("Ответ: необходимо подняться %llu раз.  \n",answ);
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}
```

```
30    unsigned long long fib_ultim(int n)//Семантика
31    {
32        int i;//счетчик
33        //Значение функции на два и один шаг назад
34        unsigned long long back2=1, back1=2, next;
35        if (n<2){//Выход если ступенек меньше 2-х
36            printf("Ступенек не может быть меньше 2-х!\n");
37            return 0;
38        }
39        for (i=2; i<n; i++)//при n<2 цикл не выполняется
40        {
41            next=back1+back2;
42            printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
43                    i-1, back2,back1,next);
44            back2=back1;//обмен содержимым переменных
45            back1=next;//обмен содержимым переменных
46        }
47        printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
48                    i-1, back2,back1,back1+back2);
49        return back1+back2;
50    }
```

```
Введите количество ступенек:
10
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Шаг 6 back2=13 back1=21 next=34
Шаг 7 back2=21 back1=34 next=55
Шаг 8 back2=34 back1=55 next=89
Ответ: необходимо подняться 89 раз.
Нажмите любую кнопку для завершения.
```

6. Гном вынес все драгоценные камни в плетеной корзинке с двумя ручками наверх из шахты в кладовую по лестнице. Теперь гном решил застелить пол шахты квадратными кварцевыми плитками, инкрустировав перекрестие плиток драгоценным камнем. Стык пола со стенами гном решил не украшать. Сторона плитки – 1 шаг гнома. Гном обмерил шахту шагами и составил её план. На плане гном отметил границы шахты линиями, соединяющими углы стен. Координаты

углов шахты в шагах гнома: {5,2; 2,1; 4,4; 3,7; 9,7; 8,1}. Сколько понадобится драгоценных камней гному, чтобы инкрустировать перекрестие плиток? Напишите алгоритм и программу, позволяющую вычислить, сколько понадобится драгоценных камней гному, чтобы инкрустировать перекрестие плиток в шахте с произвольными целыми координатами углов. Алгоритм должен раскрывать решение на каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте специальные функции и библиотеки для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Допустимо задать готовые координаты в коде основной функции. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для координат углов шахты в шагах гнома {5,2; 2,1; 4,4; 3,7; 9,7; 8,1}, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача из области вычислительной геометрии на вычисление количества точек с целочисленными координатами, лежащих внутри (но не на границе) многоугольника, заданного координатами своих вершин.

В задаче требуется рассчитать количество точек с целыми координатами, попадающими внутрь многоугольника, не считая точки, лежащие точно на ребрах многоугольника. Ответ можно проверить визуально. Ожидается увидеть алгоритм решения в виде рисунка/рисунков и формул Пика и НОД. **Ответ**: **24** точки.

Теория_1. Для любого многоугольника с целочисленными координатами вершин справедлива **формула Пика**: S = n + m/2 — 1, где S – площадь многоугольника, n – количество целых точек лежащих строго внутри многоугольника, m – количество целых точек, лежащих на границе многоугольника. Площадь многоугольника S вычисляется методом трапеций (**ожидаемое решение**) или по формуле Гаусса (“шнуровки”, **допустимое равнозначное решение**). Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1, где катет_1,2 - катеты прямоугольного треугольника, образованного на каждом ребре многоугольника (гипотенуза). Искомое количество целых точек n, лежащих строго внутри многоугольника, вычисляется как n = S - m/2 + 1.

Теория_2. Площадь многоугольника. Для вычисления площади многоугольника необходимо “замкнуть” периметр, добавив последнюю_вершину = первая_вершина. Формулы площадей показаны ниже.

$$S_{Гаусс} = \frac{1}{2}\left|\sum_{i=1}^{n-1} x_i y_{i+1} + x_n y_1 - \sum_{i=1}^{n-1} x_{i+1} y_i - x_1 y_n\right|$$

$$S_{трап} = \frac{1}{2}\left|\sum_{i=1}^{n-1} (x_{i+1} - x_i)(y_{i+1} + y_i)\right|$$

Теория_3. Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1. Катеты прямоугольного треугольника для каждого ребра многоугольника вычисляются как разности координат вершин, образующих ребро. Для вертикального или горизонтального ребра один из катетов будет равен нулю. Необходимо

последовательно перебрать все вершины за исключением “замыкающей”, вычислить НОД катетов и просуммировать их. Алгоритм вычисления показан ниже.

```
lineNodes(polygon) := "Количество узлов, лежащих на ребрах многоугольника."
                      "Сумма НОД катетов прямоугольных трегольников"
                      "для каждого ребра-гипотенузы"
                      "Замкнутый контур. Последняя_вершина=первая_вершина"
                      s ← 0
                      for i ∈ ORIGIN .. rows(polygon) − 1
                        s ← s + gcd(polygon_{i+1,1} − polygon_{i,1}, polygon_{i+1,2} − polygon_{i,2})
                      s
```

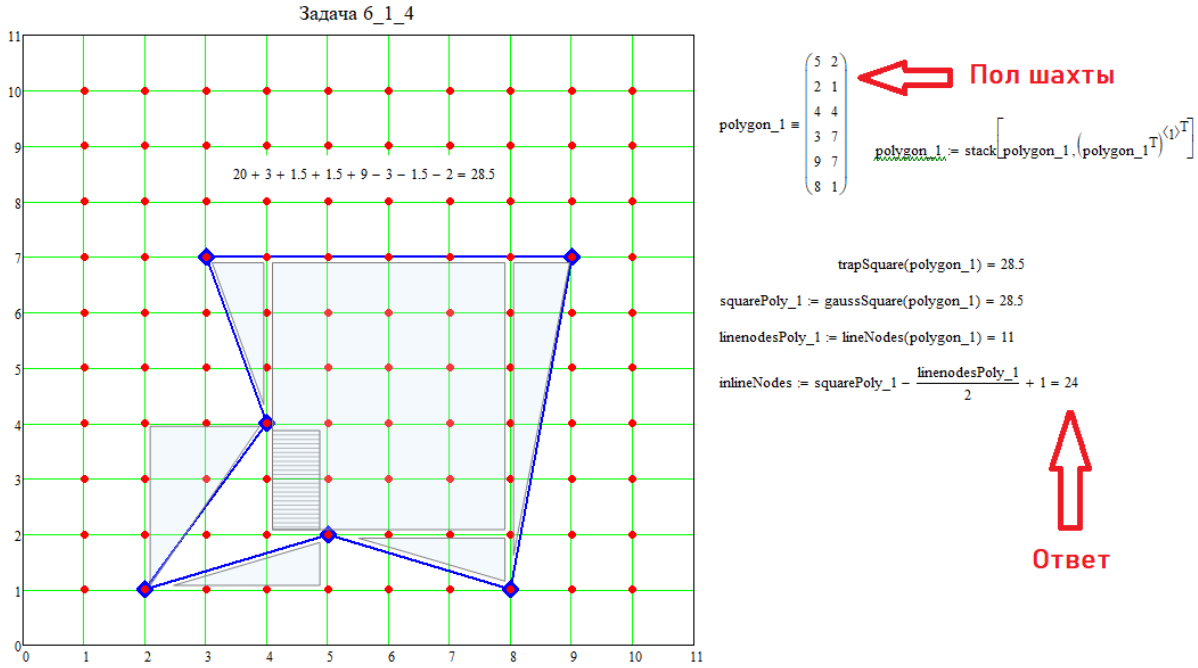
Алгоритм вычисления площади многоугольника **методом трапеций**:

```
trapSquare(polygon) := "Площадь многоугольника методом трапеций"
                       "Замкнутый контур. Последняя_вершина=первая_вершина"
                       s ← 0
                       for i ∈ ORIGIN .. rows(polygon) − 1
                         a ← polygon_{i+1,1} − polygon_{i,1}
                         b ← polygon_{i+1,2} + polygon_{i,2}
                         s ← s + a·b
                       s ← 0.5 |s|
                       s
```

Алгоритм вычисления площади многоугольника **методом Гаусса**:

```
gaussSquare(polygon) := "Площадь многоугольника методом Гаусса ("шнуровки")"
                        "Замкнутый контур. Последняя_вершина=первая_вершина"
                        s ← 0
                        for i ∈ ORIGIN .. rows(polygon) − 1
                          s ← s + polygon_{i+1,1}·polygon_{i,2}
                          s ← s − polygon_{i,1}·polygon_{i+1,2}
                        s ← s + polygon_{rows(polygon),1}·polygon_{1,2}
                        s ← s − polygon_{1,1}·polygon_{rows(polygon),2}
                        s ← 0.5 |s|
                        s
```

Графическое решение и ответ показаны ниже.



Ниже приведен пример решения на Python 3.82 и C.

```python
#Формула Пика
#НОД
def NOD(a,b):
    while b != 0:
        a, b = b, a % b
    return a
#Количество узлов, лежащих
#на ребрах многоугольника
def lineNodes(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        dX=polygon[0][i+1]-polygon[0][i]
        dY=polygon[1][i+1]-polygon[1][i]
        s=s+NOD(abs(dX),abs(dY))
    return s

#Площадь многоугольника методом трапеций
def trapSquare(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        a=polygon[0][i+1]-polygon[0][i]
        b=polygon[1][i+1]+polygon[1][i]
        s=s+a*b
    s=abs(s)/2
    return s

#Площадь многоугольника методом Гаусса
def gaussSquare(polygon):
    s=0; last=len(polygon[0])-1
    for i in range(last):
        s=s+polygon[0][i+1]*polygon[1][i]
        s=s-polygon[0][i]*polygon[1][i+1]
    s=s+polygon[0][last]*polygon[1][0]
    s=s-polygon[0][0]*polygon[1][last]
    s=0.5*abs(s)
    return s
```

```
#Вычисление по формуле Пика
#S = n + m/2 – 1
def inlineNodes(polygon):
    S=trapSquare(polygon)
    print("Площадь многоугольника",S)
    m=lineNodes(polygon)
    print("Количество точек на ребрах",m,"шт.")
    n=S-m/2+1
    return int(n)


#Координаты углов многоугольника
#двумерный список (x,y)
Plgn=[[1,3,2,2,6,5],[1,3,5,8,6,2]]
#"Замыкание" многоугольника
Plgn[0].append(Plgn[0][0])
Plgn[1].append(Plgn[1][0])

print("Количество точек внутри многоугольника" \
      ,inlineNodes(Plgn),"шт.")
```

```
Площадь многоугольника 28.5
Количество точек на ребрах 11 шт.
Количество точек внутри многоугольника 24 шт.
```

ИНФОРМАТИКА

ВАРИАНТ 5.

1. Современные преподаватели ведут электронный журнал. В одном университете приняли бальную систему оценивания на экзамене. Преподаватель по информатике решил сделать образцовый журнал, отвечающий установленным требованиям, описанным ниже:

1. Студент за семестр должен набрать минимум 60 баллов. Эти баллы формируются из сданных за семестр работ, а также максимум 20 баллов можно получить за две контрольные работы.
2. Практическая работа считается сданной вовремя, если она сдана в рамках 2 недель после занятия, на котором была объяснена. За нее студент получает 6 баллов, если сдает невовремя получает 5, в противном случае работа считается не сданной и студент ничего не получает.
3. Суммарные баллы и составляют оценку.

Преподаватель переписал список группы и сданные работы за семестр по датам. Чтобы автоматизировать процесс, он решил записать в ячейки электронной таблицы готовые формулы:

для определения баллов за одну работу преподаватель ввел формулу: =ЕСЛИ(ПОИСКПОЗ("*"&AA$2&"*";$C3:$Z3;0)-ПОИСКПОЗ("*"&AA$2&"*";$C$1:$Z$1;0)<=2;AA$1;AA$1-1)) и протянул формулу до конца календарного плана (столбцы U:AF) и до конца списка студентов (до 12 строки), эта формула поможет понять, сколько баллов студент получил за каждую сданную практическую работу.

За каждую контрольную работу студент получает не более 10 баллов. Максимально за контрольные студент может получить 20 баллов (баллы за контрольную работу указаны в диапазоне ячеек AG:AH).

Известно, что «отлично» получает студент, набравший от 90 баллов суммарно за семестр, «хорошо», если сумма баллов меньше или равна 89 и больше или равна 80, а «удовлетворительно», если сумма больше или равна 60 и меньше или равна 79. В противном случае оценка «неуд». Подсчет оценки был реализован с помощью формулы:
=ЕСЛИ(СУММ(U4:AH4)<=90;5;ЕСЛИ(И(СУММ(U4:AH4)<=89;СУММ(U4:AH4)>=80);4;ЕСЛИ(И(СУММ(U4:AH4)>=79;СУММ(U4:AH4)<=60);3;"неуд"))), затем формулу скопировали до конца журнала. Таким образом, для каждого студента удалось посчитать итоговый балл. К сожалению, преподаватель торопился и допустил ошибку в одной из формул.

Укажите правильную формулу и определите, кто из студентов получил оценку «отлично», зная график сдач их работ и баллы за контрольную работу

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1			**л1**	**л1**	**л2**	**л3**	**л4**	**л5**	**л5**	**КР**	**л6**	**л7**	**л8**	**л9**	**л10**	**л11**	**л12**	**КР**		
2	**№**	**Фамилия, Имя**	02.сен	09.сен	16.сен	23.сен	30.сен	07.окт	14.окт	21.окт	28.окт	04.ноя	11.ноя	18.ноя	25.ноя	02.дек	09.дек	16.дек	23.дек	30.дек
3	1	Абрикосова Алина	л1				л2	л3	н	н	л4		л5, л6	л7		л8	л9	л10	н	л11, л12
4	2	Баранов Владимир			л1	л2		л3, л4	л5		л6		н	л7	л8	л9	н	л10, л11		л12
5	3	Войтенко Иван		л1		л2, л3		л4		н	л6, л7			л5, л9	л8	н			л10	л11, л12
6	4	Галкина Наталья			л1, л2	л4	н		л5				л3, л6	л7, л8		н	л9	л10		л11, л12
7	5	Дмитриев Андрей			л1	н		л4, л2, л3					н	н	л6	л5, л7	н		л8	
8	6	Егоров Роман			л3		л4	н	л5	л2, л6, л1	н		л7				л10, л8, л9		л11	л12
9	7	Жукова Ирина			л1		н	л4		л2	л3		л6, л7		л5			л9		
10	8	Зайцвев Виктория			л1	л2	л3		н	л4, л6, л7				н		л9, л10	л8, л5		л11	л11, л12
11	9	Иванов Тимофей		н	л1	л2	л3	л4		л5, л6			л7		л8, л9	л10	н	н	л11	л12
12	10	Куликов Николай			л1		н	л4		л2	л3		л6, л7	н	л9	л5	л10, л8		н	л11, л12

Рисунок 1 – график сдач работ студентов

	A	B	AG	AH
1				
2	№	Фамилия, Имя	КР1	КР2
3	1	Абрикосова Алина	10	9
4	2	Баранов Владимир	8	8
5	3	Войтенко Иван	7	6
6	4	Галкина Наталья	5	8
7	5	Дмитриев Андрей	6	9
8	6	Егоров Роман	5	7
9	7	Жукова Ирина	9	8
10	8	Зайцвев Виктория	8	10
11	9	Иванов Тимофей	10	10
12	10	Куликов Николай	7	10

Рисунок 2 – баллы за контрольную работу

В ответе укажите правильную формулу, фамилию (фамилии) студентов, получивших «отлично», и конечное количество баллов за семестр.**(10 баллов)**

Решение:

	A	B	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF
1			*б*	*б*	*б*	*б*	*б*	*б*	*б*	*б*	*б*	*б*	*б*	*б*
2	№	Фамилия, Имя	л1	л2	л3	л4	л5	л6	л7	л8	л9	л10	л11	л12
3	1	Абрикосова Алина	6	6	6	5	5	6	6	5	5	5	5	5
4	2	Баранов Владимир	=ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&U$2&"*";$C4:$T4;0)-ПОИСКПОЗ("*"&U$2&"*";$C$1:$T$1;0)<=2;U$1;U$1-1);0)											
5	3	Войтенко Иван	6	6	6	6	5	6	6	6	6	5	5	5
6	4	Галкина Наталья	6	6	5	6	6	6	6	6	5	5	5	5
7	5	Дмитриев Андрей	6	5	6	6	5	5	5	5	0	0	0	0
8	6	Егоров Роман	5	5	6	6	6	6	6	5	5	6	5	5
9	7	Жукова Ирина	6	5	5	6	5	6	6	0	5	0	0	0
10	8	Зайцвев Виктория	6	6	6	5	5	6	6	5	6	6	5	5
11	9	Иванов Тимофей	6	6	6	6	6	6	6	6	6	6	5	5
12	10	Куликов Николай	6	5	5	6	5	6	6	5	6	6	5	5

	A	B	AI	AJ	AK	AL	AM	AN	AO	AP	AQ	AR	AS	AT	AU	AV
1																
2	№	Фамилия, Имя	Оценка	баллы												
3	1	Абрикосова Алина	=ЕСЛИ(СУММ(U3:AH3)>=90;5;ЕСЛИ(И(СУММ(U3:AH3)<=89;СУММ(U3:AH3)>=80);4;ЕСЛИ(И(СУММ(U3:AH3)<=79;СУММ(U3:AH3)>=60);3;"неуд")))													
4	2	Баранов Владимир	4	86												
5	3	Войтенко Иван	4	81												
6	4	Галкина Наталья	4	80												
7	5	Дмитриев Андрей	неуд	58												
8	6	Егоров Роман	3	78												
9	7	Жукова Ирина	3	61												
10	8	Зайцвев Виктория	4	85												
11	9	Иванов Тимофей	5	90												
12	10	Куликов Николай	4	83												

	A	B	AI	AJ
1				
2	№	Фамилия, Имя	Оценка	баллы
3	1	Абрикосова Алина	4	84
4	2	Баранов Владимир	4	86
5	3	Войтенко Иван	4	81
6	4	Галкина Наталья	4	80
7	5	Дмитриев Андрей	неуд	58
8	6	Егоров Роман	3	78
9	7	Жукова Ирина	3	61
10	8	Зайцвев Виктория	4	85
11	9	Иванов Тимофей	5	90
12	10	Куликов Николай	4	83
13				

Правильным также может считаться ответ с формулой, в которой указаны строки с 3 до 12. Например: =ЕСЛИ(СУММ(U3:AH3)>=90;5;ЕСЛИ(И(СУММ(U3:AH3)<=89;СУММ(U3:AH3)>=80);4;ЕСЛИ(И(СУММ(U3:AH3)<=79;СУММ(U3:AH3)>=60);3;"неуд"))) или =ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&U$2&"*";$C5:$T5;0)-ПОИСКПОЗ("*"&U$2&"*";$C$1:$T$1;0)<=2;U$1;U$1-1);0), также могут быть указаны столбцы с V по AF, например: =ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&V$2&"*";$C5:$T5;0)-ПОИСКПОЗ("*"&V$2&"*";$C$1:$T$1;0)<=2;V$1;V$1-1);0),

Ответ: Иванов, 90, =ЕСЛИ(СУММ(U4:AH4)>=90;5;ЕСЛИ(И(СУММ(U4:AH4)<=89;СУММ(U4:AH4)>=80);4;ЕСЛИ(И(СУММ(U4:AH4)<=79;СУММ(U4:AH4)>=60);3;"неуд"))) или =ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&U$2&"*";$C4:$T4;0)-ПОИСКПОЗ("*"&U$2&"*";$C$3:$T$3;0)<=2;U$1;U$1-1);0). Ответ также можно считать правильным, если найдены ошибки в обеих в формулах.

2. Два сладкоежки решили сыграть в игру «Поедание конфет». На столе лежат две вазочки конфет: в одной 15, в другой 12. Игроки ходят по очереди. За один ход можно взять любое число конфет (1; 2; 3 и т.д.) из одной из вазочек (по выбору игрока). Кто не может сделать ход (конфет не осталось), проигрывает. Кто выигрывает при безошибочной игре – игрок, делающий первый ход, или игрок, делающий второй ход? Поясните алгоритм графически. Напишите обобщенный алгоритм для любого количества конфет. **(10 баллов)**

Решение:

Используется Стратегия_Уравняй. Всегда выигрывает Игрок_1 (игрок, делающий ход первым). Суть Стратегия_Уравняй:

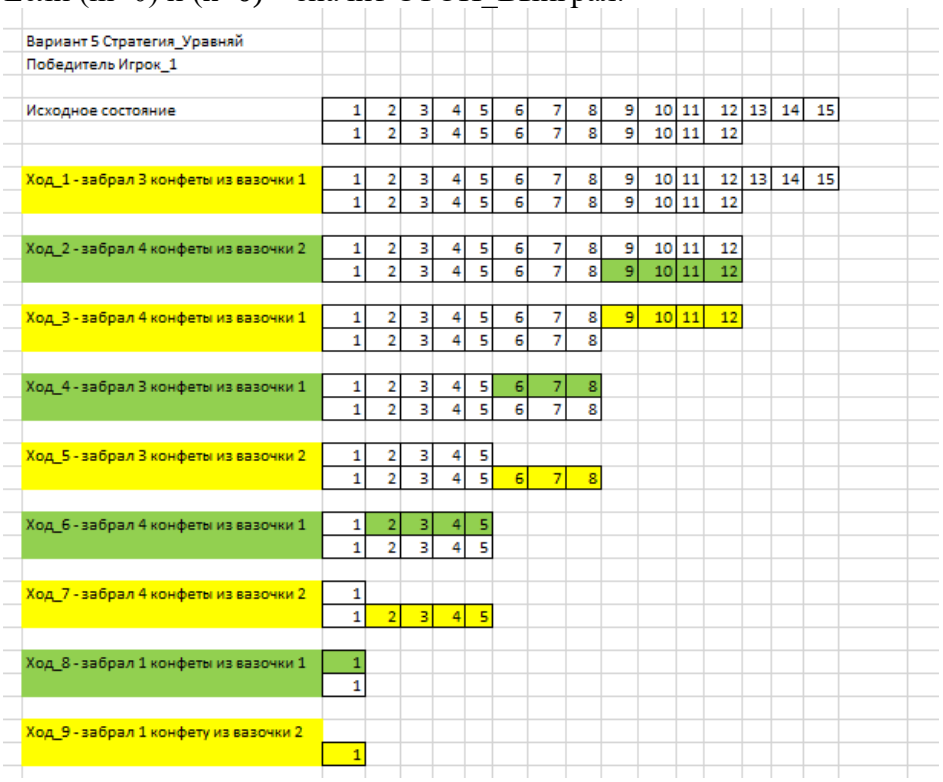
Ход_1: Игрок_1 уравнивает количество конфет в вазочках, взяв три конфеты из вазочки с большим количеством конфет (в данном случае конфет остаётся по 12 в каждой вазочке).

Ход_2: Игрок_2 (игрок, делающий ход вторым) берёт любое количество конфет из одной вазочки (то есть в одной из вазочек остаётся от 0 до 11 конфет).

Ход_3: Игрок_1 уравнивает количество конфет в вазочках, взяв такое же количество конфет как и Игрок_1 в Ход_2 из другой вазочки с большим количеством конфет.
И так далее до Победа Игрок_1. Проиграть невозможно.

Иллюстрация на рисунке ниже. Алгоритм подходит в общем случае, если m ≠ n, то выигрывает первый: он должен своим ходом уравнять кучки, а потом поддерживать равенство.

Алгоритм для Игрок_1 (обобщенный):
m-количество конфет в первой вазочке
n- количество конфет во второй вазочке
Если m>n
Шаг 1 Игрок _1 взял (m-n) конфет
Шаг 2 Игрок_2 взял любое количество конфет
Если m<n
Шаг 1 Игрок _1 взял (n-m) конфет
Шаг 2 Игрок_2 взял любое количество конфет
Если (m=0) и (n=0) – значит СТОП_Выиграл.



3. С недавних пор Аня научилась работать с таблицами истинности. Для того чтобы оттачивать свой навык, она решила каждый день строить таблицу для одного логического выражения. Но этого ей показалось мало, и она решила каждый полученный результат закодировать в виде 4 цветов в двоичной системе счисления:

0_{10} – белый

1_{10} – черный

2_{10} – синий

3_{10} – красный

Например, комбинация, 01000011 расшифровывается, как черный (01), белый (00), белый(00), красный (11).

При решении примера была получена следующая итоговая последовательность:

Черный-черный-синий-белый.

Известно, что переменные в примере были расставлены в следующем порядке: *x,y,z,x.* Также были получены промежуточные результаты:

F_1=синий-синий-синий-синий

F_2=белый-синий-белый-синий

F_3=красный-красный-белый-синий

И F_4 – результирующий, где Fn – это действия по порядку.

Напишите недостающие логические операции между переменными. В ответе предоставьте полное логическое выражение. Сопроводите ответ необходимыми теоретическими обоснованиями, расшифруйте результат каждого действия с помощью таблиц истинности. **(10 баллов)**

Решение:

1. Значение F1=синий-синий-синий-синий (10101010), следовательно, первое действие – отрицание *z*.
2. Действие F2=белый-синий-белый-синий (00100010), очевидно результат конъюнкции *y* с результатом первого действия.
3. В результате F3 получено красный-красный-белый-синий (11110010), судя по значениям, это результат импликации *x* и результата предыдущего действия.
4. Результирующей является комбинация: черный-черный-синий-белый (01011000), что является эквивалентностью значения предыдущего действия и *z*.

Ответ: $x \to y \& \bar{z} \leftrightarrow z$

4. Владимир устроился работать помощником системного администратора в IT-отдел Университета. В первый же день работы Владимир получил ответственное задание. Отдел информационных технологий Университета использует диапазон IP-адресов 172.28.192.0/18, в котором настроено адресное пространство сетей филиалов из 6 подсетей. Данные по количеству узлов в подсетях известны. При этом адресное пространство упорядочено по размеру.

Владимир обладает следующими знаниями:

- одному устройству должен соответствовать один IP-адрес вида XXX.XXX.XXX.XXX, при этом XXX – число в диапазоне [0:255], называемое октетом;
- маска с префиксом записывается в виде /18;

- устройства внутри одной подсети должны беспрепятственно взаимодействовать друг с другом и должны группироваться по их расположению;
- в каждом сегменте сети первый адрес зарезервирован под идентификатор подсети, а последний – под широковещательную рассылку;

Название подсети	Адрес сети	Требуемый размер сети	Маска с префиксом	Десятичная маска	Диапазон доступных адресов	Широковещательный адрес
Главный корпус		4096				
Инженерный корпус		2048				
Учебный центр №2		512				
Библиотека		128				
Пост охраны		64				
IT-отдел		4				

Заполнение информации вызвало у помощника системного администратора затруднения. Необходимо помочь Владимиру заполнить таблицу недостающими данными.

(**20 баллов**)

Решение:

Название подсети	Адрес сети	Требуемый размер сети	Маска с префиксом	Десятичная маска	Диапазон доступных адресов	Широковещательный адрес
Главный корпус	172.28.192.0	4096	/19	255.255.224.0	172.28.192.1 - 172.28.223.254	172.28.223.255
Инженерный корпус	172.28.224.0	2048	/20	255.255.240.0	172.28.224.1 - 172.28.239.254	172.28.239.255
Учебный центр №2	172.28.240.0	512	/22	255.255.252.0	172.28.240.1 - 172.28.243.254	172.28.243.255
Библиотека	172.28.244.0	128	/24	255.255.255.0	172.28.244.1 - 172.28.244.254	172.28.244.255
Пост охраны	172.28.245.0	64	/25	255.255.255.128	172.28.245.1 - 172.28.245.126	172.28.245.127
IT-отдел	172.28.245.128	4	/29	255.255.255.248	172.28.245.129 - 172.28.245.134	172.28.245.135

5. В пруду вдоль берега растут кувшинки. Листья кувшинок образуют узор в виде треугольного зигзага из 7 листьев. Лягушка может прыгнуть с берега только на первый или второй лист кувшинки и дальше до последнего, пока листья не закончатся. Каждый раз она прыгает по кувшинкам по-новому, чередуя прыжки по каждому или через лист. Сколько раз лягушка сможет прыгнуть с берега по 7 листьям, чтобы ни разу не повторить последовательность прыжков? Напишите алгоритм и программу, позволяющую вычислить, сколько раз лягушка сможет прыгать по узору из произвольного целого N (N>2) количества листьев кувшинки (в пределах типа данных языка программирования), чтобы ни разу не повторить последовательность прыжков. Учтите, что возможности компьютера, на котором будет выполняться программа, ограничены: допустимо использовать до 10 переменных, массивы ограничены 10 ячейками соответствующего типа, а стек вмещает 10 вызовов и не расширяется. Алгоритм должен раскрывать решение на каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте уникальные особенности языка программирования (классы, методы, объекты, попарный обмен между переменными, специальные функции и библиотеки и др., кроме счетчиков циклов) для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для 7 листьев, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача на вычисление последовательности чисел Фибоначчи.

A000045 Fibonacci numbers: F(n) = F(n-1) + F(n-2) with F(0) = 0 and F(1) = 1.
(Formerly M0692 N0256)

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75025, 121393, 196418, 317811, 514229, 832040, 1346269, 2178309, 3524578, 5702887, 9227465, 14930352, 24157817, 39088169, 63245986, 102334155 (list; graph; refs; listen; history; text;

https://oeis.org/A000045

В задаче требуется рассчитать число уникальных комбинаций прыжков, которыми можно прыгать по ряду из **7** листьев, используя прыжки длиною в 1, 2 или 1 и 2 листа кувшинки. **Ответом является 7-е** по счету число последовательности Фибоначчи (**a_7 = 21**), считая, что **a_1 = 1**, **a_2 = 2**.

Объяснение 1. Обозначим число комбинаций как a_n, где n – число ступенек, n>2.
a_1 = 1 способ –> по ряду из одного листа кувшинки можно прыгать только одним способом - прыжком в 1 лист.
a_2 = 2 способа –> 1+1 лист; сразу 2 листа.
a_3 = 3 –> 1+1+1; 1+2; 2+1 листа.
a_4 = 5 –> 1+1+1+1; 1+1+2; 1+2+1; 2+1+1; 2+2 листа.
$a_4 = a_3 + a_2$; $a_3 = a_2 + a_1$; ==> **$a_n = a_{n-1} + a_{n-2}$;**
$a_5 = a_4 + a_3 = 5+3=8$; $a_6 = a_5 + a_4 = 8+5=13$; $a_7 = a_6 + a_5 = 13+8=21$.
Ответ: a_7 = 21 способ.
Объяснение 2. Обозначим число комбинаций как a_n, где n – число листьев кувшинки, n>2. Можно начать прыжки с прыжка в 1 лист. Тогда останется a_{n-1} комбинаций прыжков по ряду листьев. Если начать подъем с прыжка в 2 листа, то останется a_{n-2} комбинаций

прыжков по ряду листьев. Первый прыжок входит в любую из комбинаций прыжков. Таким образом количество комбинаций определяется суммой количества оставшихся листьев для каждого варианта первого прыжка и размером первого прыжка, т.е. $\mathbf{a_n = a_{n-1} + a_{n-2}}$.

Задачу можно решить, например *рекурсией* (*неправильно*, ограничение по стеку вызовов при больших N), **динамическим программированием (ожидаемое решение)**, с помощью умножения матриц (допустимое, но избыточное решение), формулы Бине (допустимое, но избыточное решение). Ниже приведен пример решения с помощью динамического программирования на Python 3.82 и C.

```
#Функция Фибоначи с обменом
#Экономит память, хранит только два
#предыдущих значения
def fib_ult(n):#Для вывода по шагам
    #Значение функции на два и один шаг назад
    back2=1; back1=2
    if n<2:
        print("Листьев кувшинки не может быть меньше 2-х!")
        return 0
    for i in range(2,n-1):
        next=back1+back2
        print("Шаг ",i-1," back2=",back2," ",\
              "back1=",back1," ","next=",next,sep='')
        back2=back1#обмен содержимым переменных
        back1=next#обмен содержимым переменных
    print("Шаг ",i," back2=",back2," ",\
          "back1=",back1," ","next=",back1+back2,sep='')
    return back1+back2
fibNumb=int(input("Введите количество листьев кувшинки:\n"))
print("Ответ: необходимо прыгать с берега ",fib_ult(fibNumb)," раз.",sep='')
```

```
Введите количество листьев кувшинки:
7
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Ответ: необходимо прыгать с берега 21 раз.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int fibNumb;//Порядковый номер в ряду Фибоначчи
unsigned long long answ;//Ответ
unsigned long long fib_ultim(int n);//Сигнатура

int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");                // Очистка консоли
    printf("Введите количество листьев кувшинки:\n");
    scanf("%d",&fibNumb);
    answ=fib_ultim(fibNumb-1);//Так как нумерация массивов с нуля
    if (answ==0){//Выход если листьев меньше 2-х
        getchar();//Задержка для отображения вывода
        printf("Нажмите любую кнопку для завершения.\n");
        int ch = getch();
        return (EXIT_SUCCESS);
    }
    printf("Ответ: необходимо прыгать с берега %llu раз.  \n",answ);
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}
```

```c
unsigned long long fib_ultim(int n)//Семантика
{
    int i;//счетчик
    //Значение функции на два и один шаг назад
    unsigned long long back2=1, back1=2, next;
    if (n<2){//Выход если листьев меньше 2-х
        printf("Листьев кувшинки не может быть меньше 2-х!\n");
        return 0;
    }
    for (i=2; i<n; i++)//при n<2 цикл не выполняется
    {
        next=back1+back2;
        printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
                i-1, back2,back1,next);
        back2=back1;//обмен содержимым переменных
        back1=next;//обмен содержимым переменных
    }
    printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
                i-1, back2,back1,back1+back2);
    return back1+back2;
}
```

```
Введите количество листьев кувшинки:
7
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Ответ: необходимо прыгать с берега 21 раз.
Нажмите любую кнопку для завершения.
```

6. На карьере Марсианский автономный робот проводит взрывные работы. Для этого внутри границ карьера высверливаются отверстия под заряды. Заряды располагаются по углам прямоугольника со стороной 1 шаг робота и делаются по всей поверхности карьера. Робот может установить заряд только внутри карьера, так как по краям установке мешают стенки карьера. Карта карьера передается роботу со спутника в виде координат углов стенок в шагах робота. Спутник передал следующее сообщение: {3,3; 1,5; 3,8; 7,8; 8,4; 6,2}. Сколько отверстий под заряды высверлит робот в карьере? Напишите алгоритм и программу, позволяющую вычислить, сколько отверстий под заряды высверлит робот в карьере с произвольными целыми координатами углов стенок карьера. Алгоритм должен раскрывать решение на каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте специальные функции и библиотеки для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Допустимо задать готовые координаты в коде основной функции. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для координат углов стенок карьера в шагах робота {3,3; 1,5; 3,8; 7,8; 8,4; 6,2}, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача из области вычислительной геометрии на вычисление количества точек с целочисленными координатами, лежащих внутри (но не на границе) многоугольника, заданного координатами своих вершин.

В задаче требуется рассчитать количество точек с целыми координатами, попадающими внутрь многоугольника, не считая точки, лежащие точно на ребрах многоугольника. Ответ можно проверить визуально. Ожидается увидеть алгоритм решения в виде рисунка/рисунков и формул Пика и НОД. **Ответ**: **25** точек.

Теория_1. Для любого многоугольника с целочисленными координатами вершин справедлива **формула Пика**: S = n + m/2 — 1, где S – площадь многоугольника, n – количество целых точек лежащих строго внутри многоугольника, m – количество целых точек, лежащих на границе многоугольника. Площадь многоугольника S вычисляется методом трапеций (**ожидаемое решение**) или по формуле Гаусса (“шнуровки”, **допустимое равнозначное решение**). Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1, где катет_1,2 - катеты прямоугольного треугольника, образованного на каждом ребре многоугольника

(гипотенуза). Искомое количество целых точек n, лежащих строго внутри многоугольника, вычисляется как n = S - m/2 + 1.

Теория_2. Площадь многоугольника. Для вычисления площади многоугольника необходимо “замкнуть” периметр, добавив последнюю_вершину = первая_вершина. Формулы площадей показаны ниже.

$$S_{Гаусс} = \frac{1}{2}\left|\sum_{i=1}^{n-1} x_i y_{i+1} + x_n y_1 - \sum_{i=1}^{n-1} x_{i+1} y_i - x_1 y_n\right|$$

$$S_{трап} = \frac{1}{2}\left|\sum_{i=1}^{n-1} (x_{i+1} - x_i)(y_{i+1} + y_i)\right|$$

Теория_3. Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1. Катеты прямоугольного треугольника для каждого ребра многоугольника вычисляются как разности координат вершин, образующих ребро. Для вертикального или горизонтального ребра один из катетов будет равен нулю. Необходимо последовательно перебрать все вершины за исключением “замыкающей”, вычислить НОД катетов и просуммировать их. Алгоритм вычисления показан ниже.

```
lineNodes(polygon) := "Количество узлов, лежащих на ребрах многоугольника."
                      "Сумма НОД катетов прямоугольных трегольников"
                      "для каждого ребра-гипотенузы"
                      "Замкнутый контур. Последняя_вершина=первая_вершина"
                      s ← 0
                      for i ∈ ORIGIN .. rows(polygon) − 1
                        s ← s + gcd(polygon_{i+1,1} − polygon_{i,1}, polygon_{i+1,2} − polygon_{i,2})
                      s
```

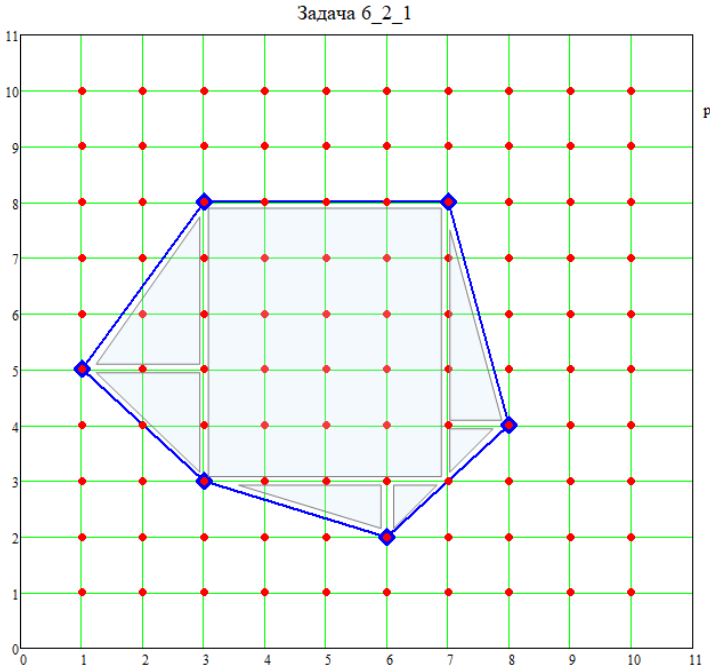
Алгоритм вычисления площади многоугольника **методом трапеций**:

```
trapSquare(polygon) := "Площадь многоугольника методом трапеций"
                       "Замкнутый контур. Последняя_вершина=первая_вершина"
                       s ← 0
                       for i ∈ ORIGIN .. rows(polygon) − 1
                         a ← polygon_{i+1,1} − polygon_{i,1}
                         b ← polygon_{i+1,2} + polygon_{i,2}
                         s ← s + a·b
                       s ← 0.5 |s|
                       s
```

Алгоритм вычисления площади многоугольника **методом Гаусса**:

```
gaussSquare(polygon) := "Площадь многоугольника методом Гаусса ("шнуровки")"
                        "Замкнутый контур. Последняя_вершина=первая_вершина"
                        s ← 0
                        for i ∈ ORIGIN .. rows(polygon) − 1
                            s ← s + polygon[i+1,1]·polygon[i,2]
                            s ← s − polygon[i,1]·polygon[i+1,2]
                        s ← s + polygon[rows(polygon),1]·polygon[1,2]
                        s ← s − polygon[1,1]·polygon[rows(polygon),2]
                        s ← 0.5·|s|
                        s
```

Графическое решение и ответ показаны ниже.



$$polygon\_1 = \begin{pmatrix} 3 & 3 \\ 1 & 5 \\ 3 & 8 \\ 7 & 8 \\ 8 & 4 \\ 6 & 2 \end{pmatrix}$$

Границы карьера

polygon_1 := stack[polygon_1, (polygon_1^T)^⟨1⟩^T]

trapSquare(polygon_1) = 29.5

squarePoly_1 := gaussSquare(polygon_1) = 29.5

linenodesPoly_1 := lineNodes(polygon_1) = 11

$$inlineNodes := squarePoly\_1 - \frac{linenodesPoly\_1}{2} + 1 = 25$$

Ответ

Ниже приведен пример решения на Python 3.82 и C.

```
#Формула Пика
#НОД
def NOD(a,b):
    while b != 0:
        a, b = b, a % b
    return a
#Количество узлов, лежащих
#на ребрах многоугольника
def lineNodes(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        dX=polygon[0][i+1]-polygon[0][i]
        dY=polygon[1][i+1]-polygon[1][i]
        s=s+NOD(abs(dX),abs(dY))
    return s

#Площадь многоугольника методом трапеций
def trapSquare(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        a=polygon[0][i+1]-polygon[0][i]
        b=polygon[1][i+1]+polygon[1][i]
        s=s+a*b
    s=abs(s)/2
    return s

#Площадь многоугольника методом Гаусса
def gaussSquare(polygon):
    s=0; last=len(polygon[0])-1
    for i in range(last):
        s=s+polygon[0][i+1]*polygon[1][i]
        s=s-polygon[0][i]*polygon[1][i+1]
    s=s+polygon[0][last]*polygon[1][0]
    s=s-polygon[0][0]*polygon[1][last]
    s=0.5*abs(s)
    return s
```

```
#Вычисление по формуле Пика
#S = n + m/2 – 1
def inlineNodes(polygon):
    S=trapSquare(polygon)
    print("Площадь многоугольника",S)
    m=lineNodes(polygon)
    print("Количество точек на ребрах",m,"шт.")
    n=S-m/2+1
    return int(n)


#Координаты углов многоугольника
#двумерный список (x,y)
Plgn=[[3,1,3,7,8,6],[3,5,8,8,4,2]]
#"Замыкание" многоугольника
Plgn[0].append(Plgn[0][0])
Plgn[1].append(Plgn[1][0])

print("Количество точек внутри многоугольника" \
      ,inlineNodes(Plgn),"шт.")
```

```
Площадь многоугольника 29.5
Количество точек на ребрах 11 шт.
Количество точек внутри многоугольника 25 шт.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int NOD(int a, int b);//НОД
int lineNodes(int *X, int *Y, int len);//Количество точек на ребрах мнгоуголника
int inlineNodes(int *X, int *Y, int len);//Вычисление по формуле Пика
float trapSquare(int *X, int *Y, int len);//S многогольника методом трапеций
float gaussSquare(int *X, int *Y, int len);//S многогольника методом Гаусса


int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");              // Очистка консоли
    //Сразу задаем "замкнутый" массив координат углов
    int pointsX[]={3,1,3,7,8,6,3};
    int pointsY[]={3,5,8,8,4,2,3};

    //Количество углов в массиве координат
    int n=sizeof(pointsX)/sizeof(pointsX[0]);
    printf("Количество точек внутри многоугольника %i шт.\n", \
            inlineNodes(pointsX,pointsY,n));
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}

```

```
30  //Вычисление НОД
31  int NOD(int a, int b){
32      int t;
33      while (b != 0){
34          t = b;
35          b = a % b;
36          a = t;
37      }
38      return a;
39  }
40  //Вычисление площади методом трапеций
41  float trapSquare(int *X, int *Y, int len){
42      float s=0;
43      for (int i=0; i<len-1; i++){
44          int a= X[i+1]- X[i];
45          int b= Y[i+1]+ Y[i];
46          s=s+a*b;
47      }
48      s=(float)abs(s)/2;
49      return s;
50  }
51  //Вычисление площади методом Гаусса
52  float gaussSquare(int *X, int *Y, int len){
53      float s=0; int last=len-1;
54      for (int i=0; i<last; i++){
55          s=s+X[i+1]*Y[i];
56          s=s-X[i]*Y[i+1];
57      }
58      s=s+X[last]*Y[0];
59      s=s-X[0]*Y[last];
60      s=(float)abs(s)/2;
61      return s;
62  }
```

```
63    //Количество точек на ребрах многоугольника
64    int lineNodes(int *X, int *Y, int len){
65        int s=0;
66        for (int i=0; i<len-1; i++){
67            int dX=X[i+1]-X[i];
68            int dY=Y[i+1]-Y[i];
69            s=s+NOD(abs(dX),abs(dY));
70        }
71        return s;
72    }
73    //Вычисление по формуле Пика
74    //S = n + m/2 — 1
75    int inlineNodes(int *X, int *Y, int len){
76        float S=trapSquare(X,Y,len);
77        printf("Площадь многоугольника %.2f\n",S);
78        int m=lineNodes(X,Y,len);
79        printf("Количество точек на ребрах %i шт.\n",m);
80        int n=S-m/2+1;//Точек внутри многоугольника
81        return n;
82    }
```

c_pik

```
Площадь многоугольника 29.50
Количество точек на ребрах 11 шт.
Количество точек внутри многоугольника 25 шт.
```

ИНФОРМАТИКА

ВАРИАНТ 6

1. Современные преподаватели ведут электронный журнал. В одном университете приняли бальную систему оценивания на экзамене. Преподаватель по информатике решил сделать образцовый журнал, отвечающий установленным требованиям, описанным ниже:

1. Студент за семестр должен набрать минимум 60 баллов. Эти баллы формируются из сданных за семестр работ, а также максимум 20 баллов можно получить за две контрольные работы.
2. Практическая работа считается сданной вовремя, если она сдана в рамках 2 недель после занятия, на котором была объяснена. За нее студент получает 6 баллов, если сдает не вовремя, получает 5, в противном случае работа считается не сданной и студент ничего не получает.
3. Суммарные баллы и составляют оценку.

Преподаватель переписал список группы и сданные работы за семестр по датам. Чтобы автоматизировать процесс, он решил записать в ячейки электронной таблицы готовые формулы:

для определения баллов за одну работу преподаватель ввел формулу: =ЕСЛИ(ПОИСКПОЗ("*"&AA$2&"*";$C3:$Z3;0)-ПОИСКПОЗ("*"&AA$2&"*";$C$1:$Z$1;0)<=2;AA$1;AA$1-1)) и протянул формулу до конца календарного плана (столбцы U:AF) и до конца списка студентов (до 12 строки), эта формула поможет понять, сколько баллов студент получил за каждую сданную работу.

За каждую контрольную работу студент получает не более 10 баллов. Максимально за контрольные студент может получить 20 баллов (баллы за контрольную работу указаны в диапазоне ячеек AG:AH).

Известно, что «отлично» получает студент, набравший от 90 баллов суммарно за семестр, «хорошо», если сумма баллов меньше или равна 89 и больше или равна 80, а «удовлетворительно», если сумма больше или равна 60 и меньше или равна 79. В противном случае оценка «неуд». Подсчет оценки был реализован с помощью формулы:
=ЕСЛИ(СУММ(U4:AH4)>=90;5;ЕСЛИ(ИЛИ(СУММ(U4:AH4)<=89;СУММ(U4:AH4)>=80);4;ЕСЛИ(ИЛИ(СУММ(U4:AH4)<=79;СУММ(U4:AH4)>=60);3;"неуд"))), затем формулу скопировали до конца журнала. Таким образом, для каждого студента удалось посчитать итоговый балл. К сожалению, преподаватель торопился и допустил ошибку в одной из формул.

Укажите правильную формулу и определите, кто из студентов получил оценку «неудовлетворительно», зная график сдач их работ и баллы за контрольную работу

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1			**л1**	**л1**	**л2**	**л3**	**л4**	**л5**	**л5**	**КР**	**л6**	**л7**	**л8**	**л9**	**л10**	**л11**	**л12**	**КР**		
2	**№**	**Фамилия, Имя**	02.сен	09.сен	16.сен	23.сен	30.сен	07.окт	14.окт	21.окт	28.окт	04.ноя	11.ноя	18.ноя	25.ноя	02.дек	09.дек	16.дек	23.дек	30.дек
3	1	Курилов Роман	л1				л2	л3	н	н	л4		л5, л6	л7		л8	л9	л10	н	л11, л12
4	2	Михайлова Анна			л1	л2		л3, л4	л5		л6		н	л7	л8	л9	н	л10, л11		л12
5	3	Николаев Владимир		л1		л2, л3		л4		н	л6, л7			л5, л9	л8	н			л10	
6	4	Оводов Давид			л1, л2	л4	н		л5				л3, л6	л7, л8		н	л9	л10		л11, л12
7	5	Петров Игорь			л1	н		л4, л2, л3					н	н	л6	л5, л7	н		л8	
8	6	Пименова Лилия			л3		л4	н	л5	л2, л6, л1	н		л7				л10, л8, л9		л11	л12
9	7	Романов Дмитрий			л1		н	л4		л2	л3		л6, л7		л5			л9		
10	8	Славина Ирина			л1	л2	л3		н	л4, л6, л7				н		л9, л10	л8, л5		л11	л11, л12
11	9	Тимофеева Инна		н	л1	л2	л3	л4		л5, л6			л7		л8, л9	л10	н	н	л11	л12
12	10	Хахалев Евгений			л1		н	л4		л2	л3		л6, л7	н	л9	л5	л10, л8		н	л11, л12

Рисунок 1 – график сдач работ студентов

	A	B	AG	AH
1				
2	№	Фамилия, Имя	КР1	КР2
3	1	Курилов Роман	10	9
4	2	Михайлова Анна	8	8
5	3	Николаев Владимир	7	6
6	4	Оводов Давид	5	8
7	5	Петров Игорь	6	9
8	6	Пименова Лилия	5	7
9	7	Романов Дмитрий	9	8
10	8	Славина Ирина	8	10
11	9	Тимофеева Инна	10	10
12	10	Хахалев Евгений	7	10

Рисунок 2 – баллы за контрольную работу

В ответе укажите правильную формулу, фамилию (фамилии) студентов, получивших «неудовлетворительно», и конечное количество баллов за семестр. (**10 баллов)**

Решение:

	A	B	AI	AJ
1				
2	№	Фамилия, Имя	Оценка	Баллы
3	1	Курилов Роман	4	84
4	2	Михайлова Анна	4	86
5	3	Николаев Владимир	3	71
6	4	Оводов Давид	4	80
7	5	Петров Игорь	неуд	58
8	6	Пименова Лилия	3	78
9	7	Романов Дмитрий	3	61
10	8	Славина Ирина	4	85
11	9	Тимофеева Инна	5	90
12	10	Хахалев Евгений	4	83

	A	B	AI	AJ	AK	AL	AM	AN	AO	AP	AQ	AR	AS	AT	AU	AV
1																
2	№	Фамилия, Имя	Оценка	Баллы												
3	1	Курилов Роман	=ЕСЛИ(СУММ(U3:AH3)>=90;5;ЕСЛИ(И(СУММ(U3:AH3)<=89;СУММ(U3:AH3)>=80);4;ЕСЛИ(И(СУММ(U3:AH3)<=79;СУММ(U3:AH3)>=60);3;"неуд")))													
4	2	Михайлова Анна	4	86												
5	3	Николаев Владимир	3	71												
6	4	Оводов Давид	4	80												
7	5	Петров Игорь	неуд	59												
8	6	Пименова Лилия	4	81												
9	7	Романов Дмитрий	3	61												
10	8	Славина Ирина	4	88												
11	9	Тимофеева Инна	5	92												
12	10	Хахалев Евгений	4	84												

	A	B	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF
1			б	б	б	б	б	б	б	б	б	б	б	б
2	№	Фамилия, Имя	л1	л2	л3	л4	л5	л6	л7	л8	л9	л10	л11	л12
3	1	Курилов Роман	=ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&U$2&"*";$C3:$T3;0)-ПОИСКПОЗ("*"&U$2&"*";$C$1:$T$1;0)<=2;U$1;U$1-1);0)											
4	2	Михайлова Анна	6	6	6	6	6	6	6	6	6	5	6	5
5	3	Николаев Владимир	6	6	6	6	5	6	6	6	6	5	0	0
6	4	Оводов Давид	6	6	5	6	6	6	6	6	5	5	5	5
7	5	Петров Игорь	6	5	6	6	5	5	5	5	0	0	0	0
8	6	Пименова Лилия	5	5	6	6	6	6	6	5	5	6	5	5
9	7	Романов Дмитрий	6	5	5	6	5	6	6	0	5	0	0	0
10	8	Славина Ирина	6	6	6	5	5	6	6	5	6	6	5	5
11	9	Тимофеева Инна	6	6	6	6	6	6	6	6	6	6	5	5
12	10	Хахалев Евгений	6	5	5	6	5	6	6	5	6	6	5	5

Правильным также может считаться ответ с формулой, в которой указаны строки с 3 до 12. Например: =ЕСЛИ(СУММ(U3:AH3)>=90;5;ЕСЛИ(И(СУММ(U3:AH3)<=89;СУММ(U3:AH3)>=80);4;ЕСЛИ(И(СУММ(U3:AH3)<=79;СУММ(U3:AH3)>=60);3;"неуд"))) или =ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&U$2&"*";$C5:$T5;0)-ПОИСКПОЗ("*"&U$2&"*";$C$1:$T$1;0)<=2;U$1;U$1-1);0), также могут быть указаны столбцы с V по AF, например: =ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&V$2&"*";$C5:$T5;0)-ПОИСКПОЗ("*"&V$2&"*";$C$1:$T$1;0)<=2;V$1;V$1-1);0),

Ответ: Петров, 58, =ЕСЛИ(СУММ(U4:AH4)>=90;5;ЕСЛИ(И(СУММ(U4:AH4)<=89;СУММ(U4:AH4)>=80);4;ЕСЛИ(И(СУММ(U4:AH4)<=79;СУММ(U4:AH4)>=60);3;"неуд"))) или =ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&U$2&"*";$C4:$T4;0)-ПОИСКПОЗ("*"&U$2&"*";$C$1:$T$1;0)<=2;U$1;U$1-1);0). Ответ также можно считать правильным, если найдены ошибки в обеих в формулах.

2. Два сладкоежки решили сыграть в игру «Поедание конфет». На столе лежат две вазочки конфет: в одной 19, в другой 8. Игроки ходят по очереди. За один ход можно взять любое число конфет (1; 2; 3; и т.д.) из одной из вазочек (по выбору игрока). Кто не может сделать ход (конфет не осталось), проигрывает. Кто выигрывает при безошибочной игре – игрок, делающий первый ход, или игрок, делающий второй ход? Поясните алгоритм графически. Напишите обобщенный алгоритм для любого количества конфет. **(10 баллов)**

Решение:

Используется Стратегия_Уравняй. Всегда выигрывает Игрок_1 (игрок, делающий ход первым). Суть Стратегия_Уравняй:

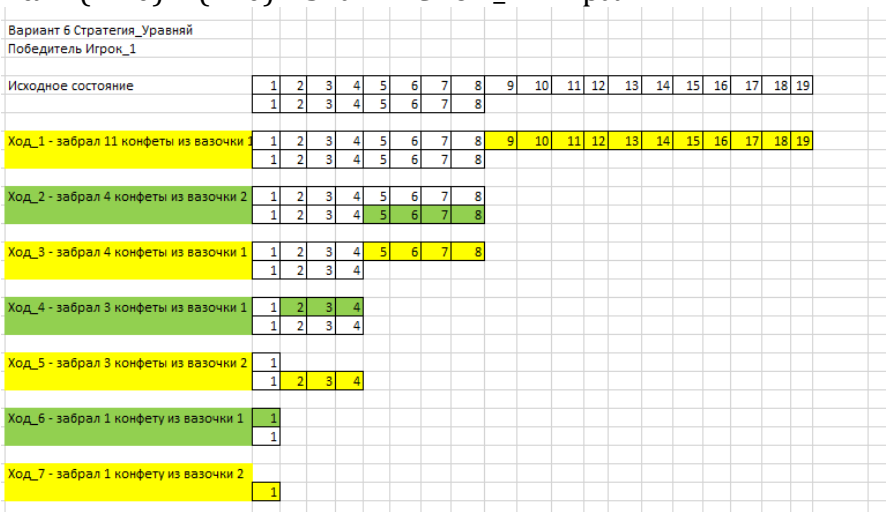
Ход_1: Игрок_1 уравнивает количество конфет в вазочках, взяв одиннадцать конфет из вазочки с большим количеством конфет (в данном случае конфет остаётся по 8 в каждой вазочке).

Ход_2: Игрок_2 (игрок, делающий ход вторым) берёт любое количество конфет из одной вазочки (то есть в одной из вазочек остаётся от 0 до 7 конфет).

Ход_3: Игрок_1 уравнивает количество конфет в вазочках, взяв такое же количество конфет как и Игрок_1 в Ход_2 из другой вазочки с большим количеством конфет.
И так далее до Победа Игрок_1. Проиграть невозможно.

Иллюстрация на рисунке ниже. Алгоритм подходит в общем случае, если m ≠ n, то выигрывает первый: он должен своим ходом уравнять кучки, а потом поддерживать равенство.

Алгоритм для Игрок_1 (обобщенный):
m-количество конфет в первой вазочке
n- количество конфет во второй вазочке
Если m>n
Шаг 1 Игрок _1 взял (m-n) конфет
Шаг 2 Игрок_2 взял любое количество конфет
Если m<n
Шаг 1 Игрок _1 взял (n-m) конфет
Шаг 2 Игрок_2 взял любое количество конфет
Если (m=0) и (n=0) – значит СТОП_Выиграл.



3. С недавних пор Аня научилась работать с таблицами истинности. Для того чтобы оттачивать свой навык, она решила каждый день строить таблицу для одного логического выражения. Но этого ей показалось мало, и она решила каждый полученный результат закодировать в виде 4 цветов в двоичной системе счисления:

0_{10} – белый

1_{10} – черный

2_{10} – синий

3_{10} – красный

Например, комбинация, 01000011 расшифровывается, как черный (01), белый (00), белый(00), красный (11).

При решении примера была получена следующая итоговая последовательность:

Красный – белый – белый - красный.

Известно, что переменные в примере были расставлены в следующем порядке: *y,x,z,x.* Также были получены промежуточные результаты:

F_1=красный-красный-белый-белый

F_2=красный-красный-черный-черный

F_3=белый-белый-красный-красный

И F_4 – результирующий, где Fn – это действия по порядку.

Напишите недостающие логические операции между переменными. В ответе предоставьте полное логическое выражение. Сопроводите ответ необходимыми теоретическими обоснованиями, расшифруйте результат каждого действия с помощью таблиц истинности. **(10 баллов)**

Решение:

1. Значение F1=красный-красный-белый-белый (11110000), следовательно, первое действие – отрицание *x*.
2. Действие F2=красный-красный-черный-черный (11110101), очевидно результат дизъюнкции *z* с результатом первого действия.
3. В результате F3 получено белый-белый-красный-красный (00001111), судя по значениям, это результат импликации результата предыдущего действия и x.
4. Результирующей является комбинация: красный-белый-белый-красный (11000011), что является эквивалентностью y и значения предыдущего действия.

Ответ: $y \leftrightarrow \bar{x} \vee z \rightarrow x$

4. Владимир устроился работать помощником системного администратора в IT-отдел Университета. В первый же день работы Владимир получил ответственное задание. Отдел информационных технологий Университета использует диапазон IP-адресов 172.26.192.0/18, в котором настроено адресное пространство сетей филиалов из 6 подсетей. Данные по количеству узлов в подсетях известны. При этом адресное пространство упорядочено по размеру.

Владимир обладает следующими знаниями:

- одному устройству должен соответствовать один IP-адрес вида XXX.XXX.XXX.XXX, при этом XXX – число в диапазоне [0:255], называемое октетом;
- маска с префиксом записывается в виде /18;
- устройства внутри одной подсети должны беспрепятственно взаимодействовать друг с другом и должны группироваться по их расположению;
- в каждом сегменте сети первый адрес зарезервирован под идентификатор подсети, а последний – под широковещательную рассылку;

Название подсети	Адрес сети	Требуемый размер сети	Маска с префиксом	Десятичная маска	Диапазон доступных адресов	Широковещательный адрес
Главный корпус		4096				

Инженерный корпус		2048				
Учебный центр №2		512				
Библиотека		128				
Пост охраны		64				
IT-отдел		4				

Заполнение информации вызвало у помощника системного администратора затруднения. Необходимо помочь Владимиру заполнить таблицу недостающими данными.(**20 баллов**)

Решение:

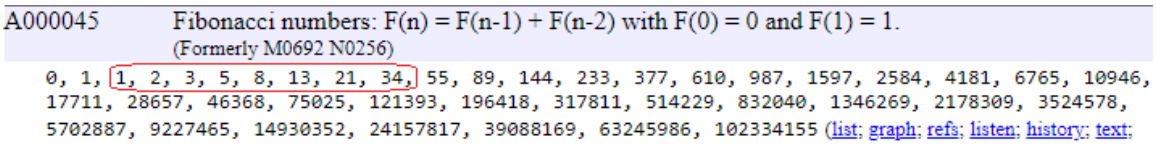
Название подсети	Адрес сети	Требуемый размер сети	Маска с префиксом	Десятичная маска	Диапазон доступных адресов	Широковещательный адрес
Главный корпус	172.26.192.0	4096	/19	255.255.224.0	172.26.192.1 - 172.26.223.254	172.26.223.255
Инженерный корпус	172.26.224.0	2048	/20	255.255.240.0	172.26.224.1 - 172.26.239.254	172.26.239.255
Учебный центр №2	172.26.240.0	512	/22	255.255.252.0	172.26.240.1 - 172.26.243.254	172.26.243.255
Библиотека	172.26.244.0	128	/24	255.255.255.0	172.26.244.1 - 172.26.244.254	172.26.244.255
Пост охраны	172.26.245.0	64	/25	255.255.255.128	172.26.245.1 - 172.26.245.126	172.26.245.127
IT-отдел	172.26.245.128	4	/29	255.255.255.248	172.26.245.129 - 172.26.245.134	172.26.245.135

5. В пруду вдоль берега растут кувшинки. Листья кувшинок образуют узор в виде треугольного зигзага из 8 листьев. Лягушка может прыгнуть с берега только на первый или второй лист кувшинки и дальше до последнего, пока листья не закончатся. Каждый раз она прыгает по кувшинкам по-новому, чередуя прыжки по каждому или через лист. Сколько раз лягушка сможет прыгнуть с берега по 8 листьям, чтобы ни разу не повторить последовательность прыжков? Напишите алгоритм и программу, позволяющую вычислить, сколько раз лягушка сможет прыгать по узору из произвольного целого N (N>2) количества листьев кувшинки (в пределах типа данных языка программирования), чтобы ни разу не повторить последовательность прыжков. Учтите, что возможности компьютера, на котором будет выполняться программа, ограничены: допустимо использовать до 10 переменных, массивы ограничены 10 ячейками соответствующего типа, а стек вмещает 10 вызовов и не расширяется. Алгоритм должен раскрывать решение на

каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте уникальные особенности языка программирования (классы, методы, объекты, попарный обмен между переменными, специальные функции и библиотеки и др., кроме счетчиков циклов) для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для 8 листьев, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача на вычисление последовательности чисел Фибоначчи.


A000045 Fibonacci numbers: F(n) = F(n-1) + F(n-2) with F(0) = 0 and F(1) = 1.
(Formerly M0692 N0256)

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75025, 121393, 196418, 317811, 514229, 832040, 1346269, 2178309, 3524578, 5702887, 9227465, 14930352, 24157817, 39088169, 63245986, 102334155 (list; graph; refs; listen; history; text;

https://oeis.org/A000045

В задаче требуется рассчитать число уникальных комбинаций прыжков, которыми можно прыгать по ряду из **8** листьев, используя прыжки длиною в 1, 2 или 1 и 2 листа кувшинки. **Ответом является 8-е** по счету число последовательности Фибоначчи (**a_8 = 34**), считая, что **a_1 = 1**, **a_2 = 2**.

Объяснение 1. Обозначим число комбинаций как a_n, где n – число ступенек, n>2.
a_1 = 1 способ –> по ряду из одного листа кувшинки можно прыгать только одним способом - прыжком в 1 лист.
a_2 = 2 способа –> 1+1 лист; сразу 2 листа.
a_3 = 3 –> 1+1+1; 1+2; 2+1 листа.
a_4 = 5 –> 1+1+1+1; 1+1+2; 1+2+1; 2+1+1; 2+2 листа.
$a_4 = a_3 + a_2$; $a_3 = a_2 + a_1$; ==> **$a_n = a_{n-1} + a_{n-2}$;**
$a_5 = a_4 + a_3$ = 5+3=8; $a_6 = a_5 + a_4$ = 8+5=13; $a_7 = a_6 + a_5$ = 13+8=21;
$a_8 = a_7 + a_6$ = 21+13=34.
Ответ: a_8 = 34 способа.
Объяснение 2. Обозначим число комбинаций как a_n, где n – число листьев кувшинки, n>2. Можно начать прыжки с прыжка в 1 лист. Тогда останется a_{n-1} комбинаций прыжков по ряду листьев. Если начать подъем с прыжка в 2 листа, то останется a_{n-2} комбинаций прыжков по ряду листьев. Первый прыжок входит в любую из комбинаций прыжков. Таким образом количество комбинаций определяется суммой количества оставшихся листьев для каждого варианта первого прыжка и размером первого прыжка, т.е. **$a_n = a_{n-1} + a_{n-2}$.**

Задачу можно решить, например *рекурсией* (*неправильно*, ограничение по стеку вызовов при больших N), **динамическим программированием (ожидаемое решение)**, с помощью умножения матриц (допустимое, но избыточное решение), формулы Бине (допустимое, но избыточное решение). Ниже приведен пример решения с помощью динамического программирования на Python 3.82 и C.

```
#Функция Фибоначи с обменом
#Экономит память, хранит только два
#предыдущих значения
def fib_ult(n):#Для вывода по шагам
    #Значение функции на два и один шаг назад
    back2=1; back1=2
    if n<2:
        print("Листьев кувшинки не может быть меньше 2-х!")
        return 0
    for i in range(2,n-1):
        next=back1+back2
        print("Шаг ",i-1," back2=",back2," ",\
              "back1=",back1," ","next=",next,sep='')
        back2=back1#обмен содержимым переменных
        back1=next#обмен содержимым переменных
    print("Шаг ",i," back2=",back2," ",\
          "back1=",back1," ","next=",back1+back2,sep='')
    return back1+back2
fibNumb=int(input("Введите количество листьев кувшинки:\n"))
print("Ответ: необходимо прыгать с берега ",fib_ult(fibNumb)," раз.",sep='')
```

```
Введите количество листьев кувшинки:
8
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Шаг 6 back2=13 back1=21 next=34
Ответ: необходимо прыгать с берега 34 раз.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int fibNumb;//Порядковый номер в ряду Фибоначчи
unsigned long long answ;//Ответ
unsigned long long fib_ultim(int n);//Сигнатура

int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");                // Очистка консоли
    printf("Введите количество листьев кувшинки:\n");
    scanf("%d",&fibNumb);
    answ=fib_ultim(fibNumb-1);//Так как нумерация массивов с нуля
    if (answ==0){//Выход если листьев меньше 2-х
        getchar();//Задержка для отображения вывода
        printf("Нажмите любую кнопку для завершения.\n");
        int ch = getch();
        return (EXIT_SUCCESS);
    }
    printf("Ответ: необходимо прыгать с берега %llu раз.  \n",answ);
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}
```

```c
unsigned long long fib_ultim(int n)//Семантика
{
    int i;//счетчик
    //Значение функции на два и один шаг назад
    unsigned long long back2=1, back1=2, next;
    if (n<2){//Выход если листьев меньше 2-х
        printf("Листьев кувшинки не может быть меньше 2-х!\n");
        return 0;
    }
    for (i=2; i<n; i++)//при n<2 цикл не выполняется
    {
        next=back1+back2;
        printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
                i-1, back2,back1,next);
        back2=back1;//обмен содержимым переменных
        back1=next;//обмен содержимым переменных
    }
    printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
                i-1, back2,back1,back1+back2);
    return back1+back2;
}
```

```
Введите количество листьев кувшинки:
8
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Шаг 6 back2=13 back1=21 next=34
Ответ: необходимо прыгать с берега 34 раз.
Нажмите любую кнопку для завершения.
```

6. На карьере Марсианский автономный робот проводит взрывные работы. Для этого внутри границ карьера высверливаются отверстия под заряды. Заряды располагаются по углам прямоугольника со стороной 1 шаг робота и делаются по всей поверхности карьера. Робот может установить заряд только внутри карьера, так как по краям установке мешают стенки карьера. Карта карьера передается роботу со спутника в виде координат углов стенок в шагах робота. Спутник передал следующее сообщение: {1,3; 2,8; 5,10; 7,7; 7,4; 4,2}. Сколько отверстий под заряды высверлит робот в карьере? Напишите алгоритм и программу, позволяющую вычислить, сколько отверстий под заряды высверлит робот в карьере с произвольными целыми координатами углов стенок карьера. Алгоритм должен раскрывать решение на каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте специальные функции и библиотеки для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Допустимо задать готовые координаты в коде основной функции. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для координат углов стенок карьера в шагах робота {1,3; 2,8; 5,10; 7,7; 7,4; 4,2}, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача из области вычислительной геометрии на вычисление количества точек с целочисленными координатами, лежащих внутри (но не на границе) многоугольника, заданного координатами своих вершин.

В задаче требуется рассчитать количество точек с целыми координатами, попадающими внутрь многоугольника, не считая точки, лежащие точно на ребрах многоугольника. Ответ можно проверить визуально. Ожидается увидеть алгоритм решения в виде рисунка/рисунков и формул Пика и НОД. **Ответ**: **30** точек.

Теория_1. Для любого многоугольника с целочисленными координатами вершин справедлива **формула Пика**: S = n + m/2 — 1, где S – площадь многоугольника, n – количество целых точек лежащих строго внутри многоугольника, m – количество целых

точек, лежащих на границе многоугольника. Площадь многоугольника S вычисляется методом трапеций (**ожидаемое решение**) или по формуле Гаусса (“шнуровки”, **допустимое равнозначное решение**). Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1, где катет_1,2 - катеты прямоугольного треугольника, образованного на каждом ребре многоугольника (гипотенуза). Искомое количество целых точек n, лежащих строго внутри многоугольника, вычисляется как n = S - m/2 + 1.

Теория_2. Площадь многоугольника. Для вычисления площади многоугольника необходимо “замкнуть” периметр, добавив последнюю_вершину = первая_вершина. Формулы площадей показаны ниже.

$$S_{Гаусс} = \frac{1}{2}\left|\sum_{i=1}^{n-1} x_i y_{i+1} + x_n y_1 - \sum_{i=1}^{n-1} x_{i+1} y_i - x_1 y_n\right|$$

$$S_{трап} = \frac{1}{2}\left|\sum_{i=1}^{n-1} (x_{i+1} - x_i)(y_{i+1} + y_i)\right|$$

Теория_3. Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1. Катеты прямоугольного треугольника для каждого ребра многоугольника вычисляются как разности координат вершин, образующих ребро. Для вертикального или горизонтального ребра один из катетов будет равен нулю. Необходимо последовательно перебрать все вершины за исключением “замыкающей”, вычислить НОД катетов и просуммировать их. Алгоритм вычисления показан ниже.

```
lineNodes(polygon) := "Количество узлов, лежащих на ребрах многоугольника."
                      "Сумма НОД катетов прямоугольных трегольников"
                      "для каждого ребра-гипотенузы"
                      "Замкнутый контур. Последняя_вершина=первая_вершина"
                      s ← 0
                      for i ∈ ORIGIN .. rows(polygon) − 1
                        s ← s + gcd(polygon_{i+1,1} − polygon_{i,1}, polygon_{i+1,2} − polygon_{i,2})
                      s
```

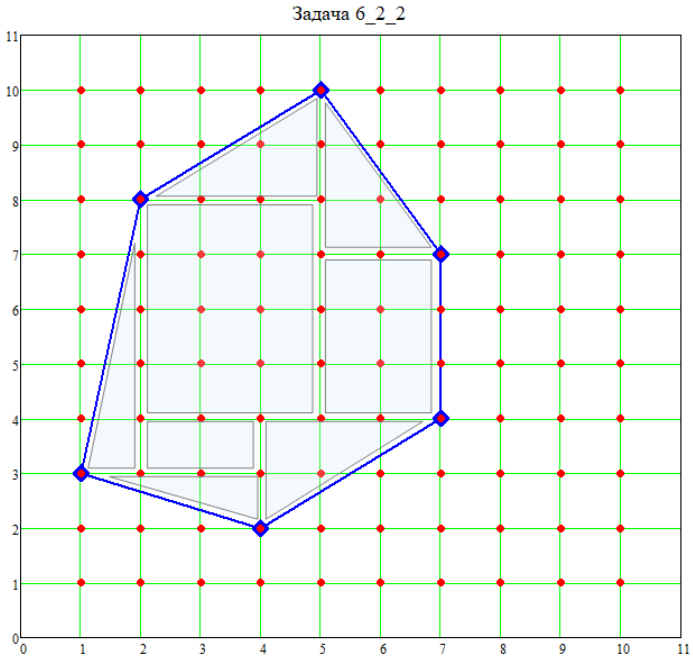
Алгоритм вычисления площади многоугольника **методом трапеций**:

```
trapSquare(polygon) := "Площадь многоугольника методом трапеций"
                       "Замкнутый контур. Последняя_вершина=первая_вершина"
                       s ← 0
                       for i ∈ ORIGIN .. rows(polygon) − 1
                         a ← polygon_{i+1,1} − polygon_{i,1}
                         b ← polygon_{i+1,2} + polygon_{i,2}
                         s ← s + a·b
                       s ← 0.5 |s|
                       s
```

Алгоритм вычисления площади многоугольника **методом Гаусса**:

```
gaussSquare(polygon) :=
    "Площадь многоугольника методом Гаусса ("шнуровки")"
    "Замкнутый контур. Последняя_вершина=первая_вершина"
    s ← 0
    for i ∈ ORIGIN .. rows(polygon) − 1
        s ← s + polygon_{i+1,1} · polygon_{i,2}
        s ← s − polygon_{i,1} · polygon_{i+1,2}
    s ← s + polygon_{rows(polygon),1} · polygon_{1,2}
    s ← s − polygon_{1,1} · polygon_{rows(polygon),2}
    s ← 0.5 |s|
    s
```

Графическое решение и ответ показаны ниже.



$$polygon\_1 \equiv \begin{pmatrix} 1 & 3 \\ 2 & 8 \\ 5 & 10 \\ 7 & 7 \\ 7 & 4 \\ 4 & 2 \end{pmatrix}$$

Границы карьера

$$polygon\_1 := stack\left[polygon\_1, \left(polygon\_1^{T}\right)^{\langle 1 \rangle^{T}}\right]$$

$$trapSquare(polygon\_1) = 33$$

$$squarePoly\_1 := gaussSquare(polygon\_1) = 33$$

$$linenodesPoly\_1 := lineNodes(polygon\_1) = 8$$

$$inlineNodes := squarePoly\_1 - \frac{linenodesPoly\_1}{2} + 1 = 30$$

Ответ

Ниже приведен пример решения на Python 3.82 и C.

```python
#Формула Пика
#НОД
def NOD(a,b):
    while b != 0:
        a, b = b, a % b
    return a
#Количество узлов, лежащих
#на ребрах многоугольника
def lineNodes(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        dX=polygon[0][i+1]-polygon[0][i]
        dY=polygon[1][i+1]-polygon[1][i]
        s=s+NOD(abs(dX),abs(dY))
    return s

#Площадь многоугольника методом трапеций
def trapSquare(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        a=polygon[0][i+1]-polygon[0][i]
        b=polygon[1][i+1]+polygon[1][i]
        s=s+a*b
    s=abs(s)/2
    return s

#Площадь многоугольника методом Гаусса
def gaussSquare(polygon):
    s=0; last=len(polygon[0])-1
    for i in range(last):
        s=s+polygon[0][i+1]*polygon[1][i]
        s=s-polygon[0][i]*polygon[1][i+1]
    s=s+polygon[0][last]*polygon[1][0]
    s=s-polygon[0][0]*polygon[1][last]
    s=0.5*abs(s)
    return s
#Вычисление по формуле Пика
#S = n + m/2 – 1
def inlineNodes(polygon):
    S=trapSquare(polygon)
    print("Площадь многоугольника",S)
    m=lineNodes(polygon)
    print("Количество точек на ребрах",m,"шт.")
    n=S-m/2+1
    return int(n)


#Координаты углов многоугольника
#двумерный список (x,y)
Plgn=[[1,2,5,7,7,4],[3,8,10,7,4,2]]
#"Замыкание" многоугольника
Plgn[0].append(Plgn[0][0])
Plgn[1].append(Plgn[1][0])

print("Количество точек внутри многоугольника" \
      ,inlineNodes(Plgn),"шт.")
```

```
Площадь многоугольника 33.0
Количество точек на ребрах 8 шт.
Количество точек внутри многоугольника 30 шт.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int NOD(int a, int b);//НОД
int lineNodes(int *X, int *Y, int len);//Количество точек на ребрах мнгоуголника
int inlineNodes(int *X, int *Y, int len);//Вычисление по формуле Пика
float trapSquare(int *X, int *Y, int len);//S многогольника методом трапеций
float gaussSquare(int *X, int *Y, int len);//S многогольника методом Гаусса


int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");                // Очистка консоли
    //Сразу задаем "замкнутый" массив координат углов
    int pointsX[]={1,2,5,7,7,4,1};
    int pointsY[]={3,8,10,7,4,2,3};

    //Количество углов в массиве координат
    int n=sizeof(pointsX)/sizeof(pointsX[0]);
    printf("Количество точек внутри многоугольника %i шт.\n", \
            inlineNodes(pointsX,pointsY,n));
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}

```

```
30  //Вычисление НОД
31  int NOD(int a, int b){
32      int t;
33      while (b != 0){
34          t = b;
35          b = a % b;
36          a = t;
37      }
38      return a;
39  }
40  //Вычисление площади методом трапеций
41  float trapSquare(int *X, int *Y, int len){
42      float s=0;
43      for (int i=0; i<len-1; i++){
44          int a= X[i+1]- X[i];
45          int b= Y[i+1]+ Y[i];
46          s=s+a*b;
47      }
48      s=(float)abs(s)/2;
49      return s;
50  }
51  //Вычисление площади методом Гаусса
52  float gaussSquare(int *X, int *Y, int len){
53      float s=0; int last=len-1;
54      for (int i=0; i<last; i++){
55          s=s+X[i+1]*Y[i];
56          s=s-X[i]*Y[i+1];
57      }
58      s=s+X[last]*Y[0];
59      s=s-X[0]*Y[last];
60      s=(float)abs(s)/2;
61      return s;
62  }
```

```c
//Количество точек на ребрах многоугольника
int lineNodes(int *X, int *Y, int len){
    int s=0;
    for (int i=0; i<len-1; i++){
        int dX=X[i+1]-X[i];
        int dY=Y[i+1]-Y[i];
        s=s+NOD(abs(dX),abs(dY));
    }
    return s;
}
//Вычисление по формуле Пика
//S = n + m/2 — 1
int inlineNodes(int *X, int *Y, int len){
    float S=trapSquare(X,Y,len);
    printf("Площадь многоугольника %.2f\n",S);
    int m=lineNodes(X,Y,len);
    printf("Количество точек на ребрах %i шт.\n",m);
    int n=S-m/2+1;//Точек внутри многоугольника
    return n;
}
```

c_pik

```
Площадь многоугольника 33.00
Количество точек на ребрах 8 шт.
Количество точек внутри многоугольника 30 шт.
```

ИНФОРМАТИКА

ВАРИАНТ 7

1. Современные преподаватели ведут электронный журнал. В одном университете приняли бальную систему оценивания на экзамене. Преподаватель по информатике решил сделать образцовый журнал, отвечающий установленным требованиям, описанным ниже:

1. Студент за семестр должен набрать минимум 60 баллов. Эти баллы формируются из сданных за семестр работ, а также максимум 20 баллов можно получить за две контрольные работы.
2. Практическая работа считается сданной вовремя, если она сдана в рамках 2 недель после занятия, на котором была объяснена. За нее студент получает 6 баллов, если сдает не вовремя, получает 5, в противном случае работа считается не сданной и студент ничего не получает.
3. Суммарные баллы и составляют оценку.

Преподаватель переписал список группы и сданные работы за семестр по датам. Чтобы автоматизировать процесс, он решил записать в ячейки электронной таблицы готовые формулы:

для определения баллов за одну работу преподаватель ввел формулу: =ЕСЛИ(ПОИСКПОЗ("*"&AA$2&"*";$C3:$Z3;0)+ПОИСКПОЗ("*"&AA$2&"*";$C$1:$Z$1;0)<=2;AA$1;AA$1-1) и протянул формулу до конца календарного плана (столбцы U:AF) и до конца списка студентов (до 12 строки), эта формула поможет понять, сколько баллов студент получил за каждую сданную работу.

За каждую контрольную работу студент получает не более 10 баллов. Максимально за контрольные студент может получить 20 баллов (баллы за контрольную работу указаны в диапазоне ячеек AG:AH).

Известно, что «отлично» получает студент, набравший от 90 баллов суммарно за семестр, «хорошо», если сумма баллов меньше или равна 89 и больше или равна 80, а «удовлетворительно», если сумма больше или равна 60 и меньше или равна 79. В противном случае оценка «неуд». Подсчет оценки был реализован с помощью формулы:
=ЕСЛИ(СУММ(U4:AH4)>=90;5;ЕСЛИ(И(СУММ(U4:AH4)<=89;СУММ(U4:AH4)>=80);4;ЕСЛИ(И(СУММ(U4:AH4)>=79;СУММ(U4:AH4)<=60);3;"неуд"))), затем формулу скопировали до конца журнала. Таким образом, для каждого студента удалось посчитать итоговый балл. К сожалению, преподаватель торопился и допустил ошибку в одной из формул.

Укажите правильную формулу и определите, кто из студентов получил оценку «удовлетворительно», зная график сдач их работ и баллы за контрольную работу:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1			**л1**	**л1**	**л2**	**л3**	**л4**	**л5**	**л5**	**КР**	**л6**	**л7**	**л8**	**л9**	**л10**	**л11**	**л12**	**КР**		
2	**№**	**Фамилия, Имя**	02.сен	09.сен	16.сен	23.сен	30.сен	07.окт	14.окт	21.окт	28.окт	04.ноя	11.ноя	18.ноя	25.ноя	02.дек	09.дек	16.дек	23.дек	30.дек
3	1	Авдеев Дмитрий			л1	н	л2	л3	н	н	л4		л5, л6	л7		л8	л9	л10	н	л11, л12
4	2	Баранов Роберт			л1	л2		л3, л4	л5		л6	н	н	л7	л8	л9	н	л10, л11		л12
5	3	Воробева Дарья		л1		л2, л3		л4		н	л6, л7			л5, л9	л8	н			л10	
6	4	Германова Полина			л1, л2	л4	н			л5			л3, л6	л7, л8		н	л9	л10		л11, л12
7	5	Дементьев Никита			л1	н			л4, л2, л3			н	н	н	л6	л5, л7	н		л8	
8	6	Емельянов Олег			л3		л4	н	л5	л2, л6, л1	н		л7				л10, л8, л9		л11	л12
9	7	Жорин Григорий			л1		н	л4		л2	л3		л6, л7	л5				л9		
10	8	Иванова Валерия			л1	л2	л3		н		л4, л6, л7	н		н		л9, л10	л8, л5		л11	л11, л12
11	9	Кузьмина Наталья		н	л1	л2	л3	л4			л5, л6		л7	л8, л9		л10	н	н	л11	л12
12	10	Михайленко Константин			л1		н	л4		л2	л3		л6, л7	н	л9	л5	л10, л8		н	л11, л12

Рисунок 1 – график сдач работ студентов

	A	B	AG	AH
1				
2	№	Фамилия, Имя	КР1	КР2
3	1	Авдеев Дмитрий	10	9
4	2	Баранов Роберт	6	10
5	3	Воробева Дарья	7	6
6	4	Германова Полина	5	10
7	5	Дементьев Никита	6	9
8	6	Емельянов Олег	5	5
9	7	Жорин Григорий	8	8
10	8	Иванова Валерия	8	10
11	9	Кузьмина Наталья	10	10
12	10	Михайленко Константин	7	10

Рисунок 2 – баллы за контрольную работу

В ответе укажите правильную формулу, фамилию (фамилии) студентов, получивших «удовлетворительно», и конечное количество баллов за семестр.**(10 баллов)**

Решение:

	A	B	AI	AJ
1				
2	№	Фамилия, Имя	Оценка	Баллы
3	1	Авдеев Дмитрий	4	84
4	2	Баранов Роберт	4	86
5	3	Воробева Дарья	3	71
6	4	Германова Полина	4	82
7	5	Дементьев Никита	неуд	57
8	6	Емельянов Олег	3	76
9	7	Жорин Григорий	3	60
10	8	Иванова Валерия	4	85
11	9	Кузьмина Наталья	4	89
12	10	Михайленко Константин	4	83

	A	B	U	V	W	X	Y	Z	AA	AB	AC	AD
1			6	6	6	6	6	6	6	6	6	6
2	№	Фамилия, Имя	л1	л2	л3	л4	л5	л6	л7	л8	л9	л10
3	1	Авдеев Дмитрий	=ЕСЛИ(ПОИСКПОЗ("*"&U$2&"*";$C3:$T3;0)-ПОИСКПОЗ("*"&U$2&"*";$C$1:$T$1;0)<=2;U$1;U$1-1)									
4	2	Баранов Роберт	6	6	6	6	6	6	6	6	6	5
5	3	Воробева Дарья	6	6	6	6	5	6	6	6	6	5
6	4	Германова Полина	6	6	5	6	6	6	6	6	5	5
7	5	Дементьев Никита	6	5	6	6	5	5	5	5	0	0
8	6	Емельянов Олег	5	5	6	6	6	6	6	5	6	6
9	7	Жорин Григорий	6	5	5	6	5	6	6	0	5	0
10	8	Иванова Валерия	6	6	6	5	5	6	6	5	6	6
11	9	Кузьмина Наталья	6	6	6	6	6	6	6	6	6	6
12	10	Михайленко Константин	6	5	5	6	5	6	6	5	6	6

	A	B	AI	AJ
1				
2	**№**	**Фамилия, Имя**	**Оценка**	**Баллы**
3	1	Авдеев Дмитрий	4	84
4	2	Баранов Роберт	=ЕСЛИ(СУММ(U4:АН	
5	3	Воробева Дарья	3	71
6	4	Германова Полина	4	82
7	5	Дементьев Никита	неуд	57
8	6	Емельянов Олег	3	76
9	7	Жорин Григорий	3	60
10	8	Иванова Валерия	4	85
11	9	Кузьмина Наталья	4	89
12	10	Михайленко Константин	4	83

Правильным также может считаться ответ с формулой, в которой указаны строки с 3 до 12. Например: =ЕСЛИ(СУММ(U3:АН3)>=90;5;ЕСЛИ(И(СУММ(U3:АН3)<=89;СУММ(U3:АН3)>=80);4;ЕСЛИ(И(СУММ(U3:АН3)<=79;СУММ(U3:АН3)>=60);3;"неуд"))) или =ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&U$2&"*";$C5:$T5;0)-ПОИСКПОЗ("*"&U$2&"*";$C$1:$T$1;0)<=2;U$1;U$1-1);0), также могут быть указаны столбцы с V по AF, например: =ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&V$2&"*";$C5:$T5;0)-ПОИСКПОЗ("*"&V$2&"*";$C$1:$T$1;0)<=2;V$1;V$1-1);0),

Ответ: Воробьева, Емельянов, Жорин, 71, 76, 60, =ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&U$2&"*";$C3:$T3;0)-ПОИСКПОЗ("*"&U$2&"*";$C$1:$T$1;0)<=2;U$1;U$1-1);0) или =ЕСЛИ(СУММ(U4:АН4)>=90;5;ЕСЛИ(И(СУММ(U4:АН4)<=89;СУММ(U4:АН4)>=80);4;ЕСЛИ(И(СУММ(U4:АН4)<=79;СУММ(U4:АН4)>=60);3;"неуд"))). Ответ также считается правильным, если ошибки найдены в обеих формулах.

2. Два сладкоежки решили сыграть в игру «Поедание конфет». На столе лежат две вазочки по 19 конфет в каждой. Игроки ходят по очереди. За один ход можно взять любое число конфет (1; 2; 3; и т.д.) из одной из вазочек (по выбору игрока). Кто не может сделать ход (конфет не осталось), проигрывает. Кто выигрывает при безошибочной игре – игрок, делающий первый ход, или игрок, делающий второй ход? Поясните алгоритм графически. Напишите обобщенный алгоритм для любого количества конфет. **(10 баллов)**

Решение:

Используется Стратегия_Уравняй. Всегда выигрывает Игрок_2 (игрок, делающий ход вторым). Суть Стратегия_Уравняй:

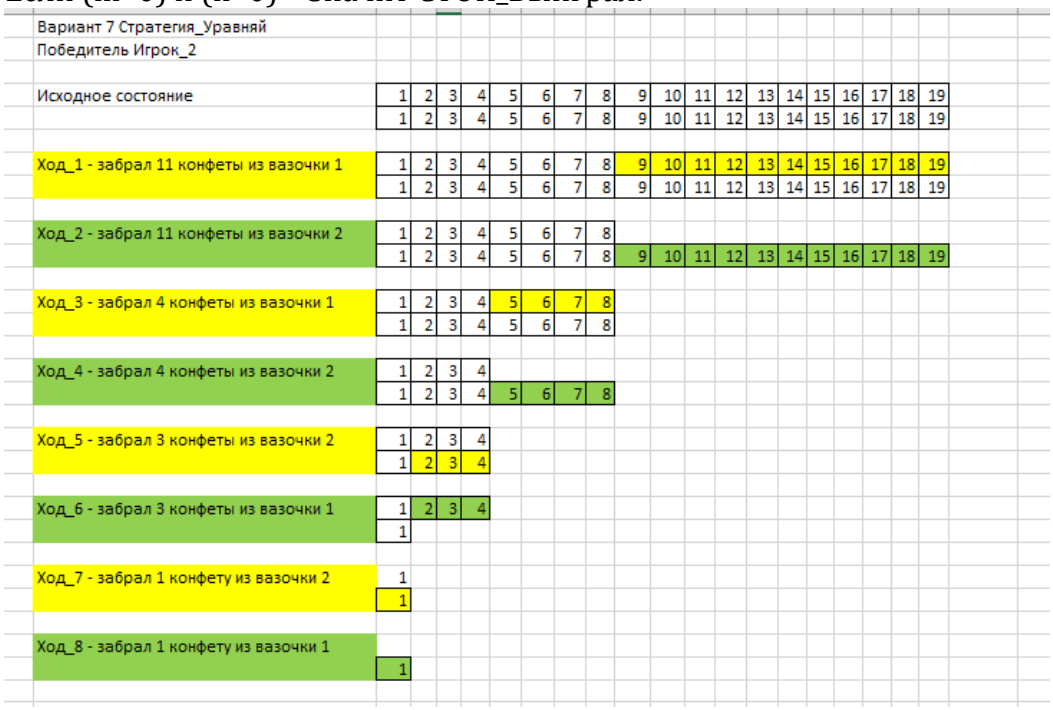
Ход_1: Игрок_1 (игрок, делающий ход первым) берёт любое количество конфет из одной вазочки (то есть в одной вазочке остаётся 19 конфет, а в другой остаётся от 0 до 18 конфет).

Ход_2: Игрок_2 уравнивает количество конфет в вазочках, взяв такое же количество конфет как и Игрок_1 в Ход_1 из другой вазочки с большим количеством конфет.

И так далее до Победа Игрок_1. Проиграть невозможно.

Иллюстрация на рисунке ниже. Алгоритм подходит в общем случае, если m = n, то выигрывает второй: он должен своим ходом уравнять кучки, а потом поддерживать равенство.

Алгоритм для Игрок_2 (обобщенный):
m-количество конфет в первой вазочке
n- количество конфет во второй вазочке
Если m=n
Шаг 1 Игрок _1 взял k конфет из вазочки 1
Шаг 2 Игрок_2 взял k конфет из вазочки 2
Если (m=0) и (n=0) – значит СТОП_Выиграл.



3. С недавних пор Аня научилась работать с таблицами истинности. Для того чтобы оттачивать свой навык, она решила каждый день строить таблицу для одного логического выражения. Но этого ей показалось мало, и она решила каждый полученный результат закодировать в виде 4 цветов в двоичной системе счисления:

0_{10} – белый

1_{10} – черный

2_{10} – синий

3_{10} – красный

Например, комбинация, 01000011 расшифровывается, как черный (01), белый (00), белый(00), красный (11).

При решении примера была получена следующая итоговая последовательность:

Красный-красный-синий-красный.

Известно, что переменные в примере были расставлены в следующем порядке: *z,x,y,x.* Также были получены промежуточные результаты:

F_1=красный-красный-белый-белый

F_2=белый-белый-белый-красный

F_3=красный-красный-белый-красный

И F_4 – результирующий, где Fn – это действия по порядку.

Напишите недостающие логические операции между переменными. В ответе предоставьте полное логическое выражение. Сопроводите ответ необходимыми теоретическими обоснованиями, расшифруйте результат каждого действия с помощью таблиц истинности. **(10 баллов)**

Решение:

1. Значение F1=красный-красный-белый-белый (11110000), следовательно, первое действие – отрицание x.
2. Действие F2=белый-белый-белый-красный (00000011), очевидно результат конъюнкции y с x.
3. В результате F3 получено красный-красный-белый-красный (11110011), судя по значениям, это результат дизъюнкции результатов первого и второго действий.
4. Результирующей является комбинация: красный-красный-синий-красный (11111011), что является импликацией z и значений предыдущего действия.

Ответ: $z \to x \wedge y \vee \bar{x}$

4. Владимир устроился работать помощником системного администратора в IT-отдел Университета. В первый же день работы Владимир получил ответственное задание. Отдел информационных технологий Университета использует диапазон IP-адресов 172.24.192.0/18, в котором настроено адресное пространство сетей филиалов из 6 подсетей. Данные по количеству узлов в подсетях известны. При этом адресное пространство упорядочено по размеру.

Владимир обладает следующими знаниями:

- одному устройству должен соответствовать один IP-адрес вида XXX.XXX.XXX.XXX, при этом XXX – число в диапазоне [0:255], называемое октетом;
- маска с префиксом записывается в виде /18;
- устройства внутри одной подсети должны беспрепятственно взаимодействовать друг с другом и должны группироваться по их расположению;
- в каждом сегменте сети первый адрес зарезервирован под идентификатор подсети, а последний – под широковещательную рассылку;

Название подсети	**Адрес сети**	**Требуемый размер сети**	**Маска с префиксом**	**Десятичная маска**	**Диапазон доступных адресов**	**Широковещательный адрес**
Главный корпус		4096				

Инженерный корпус		2048				
Учебный центр №2		512				
Библиотека		128				
Пост охраны		64				
IT-отдел		4				

Заполнение информации вызвало у помощника системного администратора затруднения. Необходимо помочь Владимиру заполнить таблицу недостающими данными.

(**20 баллов**)

Решение:

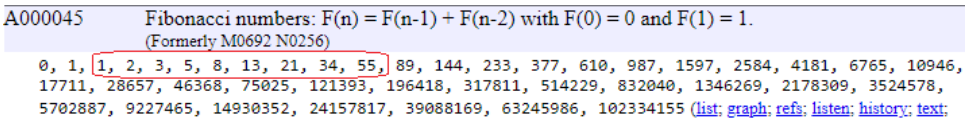
Название подсети	Адрес сети	Требуемый размер сети	Маска с префиксом	Десятичная маска	Диапазон доступных адресов	Широковещательный адрес
Главный корпус	172.24.192.0	4096	/19	255.255.224.0	172.24.192.1 - 172.24.223.254	172.24.223.255
Инженерный корпус	172.24.224.0	2048	/20	255.255.240.0	172.24.224.1 - 172.24.239.254	172.24.239.255
Учебный центр №2	172.24.240.0	512	/22	255.255.252.0	172.24.240.1 - 172.24.243.254	172.24.243.255
Библиотека	172.24.244.0	128	/24	255.255.255.0	172.24.244.1 - 172.24.244.254	172.24.244.255
Пост охраны	172.24.245.0	64	/25	255.255.255.128	172.24.245.1 - 172.24.245.126	172.24.245.127
IT-отдел	172.24.245.128	4	/29	255.255.255.248	172.24.245.129 - 172.24.245.134	172.24.245.135

5. В пруду вдоль берега растут кувшинки. Листья кувшинок образуют узор в виде треугольного зигзага из 9 листьев. Лягушка может прыгнуть с берега только на первый или второй лист кувшинки и дальше до последнего, пока листья не закончатся. Каждый раз она прыгает по кувшинкам по-новому, чередуя прыжки по каждому или через лист. Сколько раз лягушка сможет прыгнуть с берега по 9 листьям, чтобы ни разу не повторить последовательность прыжков? Напишите алгоритм и программу, позволяющую вычислить, сколько раз лягушка сможет прыгать по узору из произвольного целого N (N>2) количества листьев кувшинки (в пределах типа данных языка программирования), чтобы ни разу не повторить последовательность прыжков. Учтите, что возможности компьютера, на котором будет выполняться программа, ограничены: допустимо использовать до 10 переменных, массивы ограничены 10 ячейками соответствующего типа, а стек вмещает 10 вызовов и не расширяется. Алгоритм должен раскрывать решение на

каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте уникальные особенности языка программирования (классы, методы, объекты, попарный обмен между переменными, специальные функции и библиотеки и др., кроме счетчиков циклов) для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для 9 листьев, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача на вычисление последовательности чисел Фибоначчи.


A000045 Fibonacci numbers: F(n) = F(n-1) + F(n-2) with F(0) = 0 and F(1) = 1.
(Formerly M0692 N0256)
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75025, 121393, 196418, 317811, 514229, 832040, 1346269, 2178309, 3524578, 5702887, 9227465, 14930352, 24157817, 39088169, 63245986, 102334155 (list; graph; refs; listen; history; text;

https://oeis.org/A000045

В задаче требуется рассчитать число уникальных комбинаций прыжков, которыми можно прыгать по ряду из **9** листьев, используя прыжки длиною в 1, 2 или 1 и 2 листа кувшинки. **Ответом является 9-е** по счету число последовательности Фибоначчи ($\mathbf{a_9 = 55}$), считая, что $\mathbf{a_1 = 1}$, $\mathbf{a_2 = 2}$.

Объяснение 1. Обозначим число комбинаций как a_n, где n – число ступенек, n>2.
$\mathbf{a_1 = 1}$ **способ** –> по ряду из одного листа кувшинки можно прыгать только одним способом - прыжком в 1 лист.
$\mathbf{a_2 = 2}$ **способа** –> 1+1 лист; сразу 2 листа.
$a_3 = 3$ –> 1+1+1; 1+2; 2+1 листа.
$a_4 = 5$ –> 1+1+1+1; 1+1+2; 1+2+1; 2+1+1; 2+2 листа.
$a_4 = a_3 + a_2$; $a_3 = a_2 + a_1$; ==> $\mathbf{a_n = a_{n-1} + a_{n-2}}$;
$a_5 = a_4 + a_3 = 5+3=8$; $a_6 = a_5 + a_4 = 8+5=13$; $a_7 = a_6 + a_5 = 13+8=21$;
$a_8 = a_7 + a_6 = 21+13=34$; $a_9 = a_8 + a_7 = 34+21=55$.
Ответ: a_9 = 55 способов.
Объяснение 2. Обозначим число комбинаций как a_n, где n – число листьев кувшинки, n>2. Можно начать прыжки с прыжка в 1 лист. Тогда останется a_{n-1} комбинаций прыжков по ряду листьев. Если начать подъем с прыжка в 2 листа, то останется a_{n-2} комбинаций прыжков по ряду листьев. Первый прыжок входит в любую из комбинаций прыжков. Таким образом количество комбинаций определяется суммой количества оставшихся листьев для каждого варианта первого прыжка и размером первого прыжка, т.е. $\mathbf{a_n = a_{n-1} + a_{n-2}}$.

Задачу можно решить, например *рекурсией* (*неправильно*, ограничение по стеку вызовов при больших N), **динамическим программированием (ожидаемое решение)**, с помощью умножения матриц (допустимое, но избыточное решение), формулы Бине (допустимое, но избыточное решение). Ниже приведен пример решения с помощью динамического программирования на Python 3.82 и C.

```python
#Функция Фибоначи с обменом
#Экономит память, хранит только два
#предыдущих значения
def fib_ult(n):#Для вывода по шагам
    #Значение функции на два и один шаг назад
    back2=1; back1=2
    if n<2:
        print("Листьев кувшинки не может быть меньше 2-х!")
        return 0
    for i in range(2,n-1):
        next=back1+back2
        print("Шаг ",i-1," back2=",back2," ",\
              "back1=",back1," ","next=",next,sep='')
        back2=back1#обмен содержимым переменных
        back1=next#обмен содержимым переменных
    print("Шаг ",i," back2=",back2," ",\
          "back1=",back1," ","next=",back1+back2,sep='')
    return back1+back2
fibNumb=int(input("Введите количество листьев кувшинки:\n"))
print("Ответ: необходимо прыгать с берега ",fib_ult(fibNumb)," раз.",sep='')
```

```
Введите количество листьев кувшинки:
9
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Шаг 6 back2=13 back1=21 next=34
Шаг 7 back2=21 back1=34 next=55
Ответ: необходимо прыгать с берега 55 раз.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int fibNumb;//Порядковый номер в ряду Фибоначчи
unsigned long long answ;//Ответ
unsigned long long fib_ultim(int n);//Сигнатура

int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");                // Очистка консоли
    printf("Введите количество листьев кувшинки:\n");
    scanf("%d",&fibNumb);
    answ=fib_ultim(fibNumb-1);//Так как нумерация массивов с нуля
    if (answ==0){//Выход если листьев меньше 2-х
        getchar();//Задержка для отображения вывода
        printf("Нажмите любую кнопку для завершения.\n");
        int ch = getch();
        return (EXIT_SUCCESS);
    }
    printf("Ответ: необходимо прыгать с берега %llu раз.  \n",answ);
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}
```

```c
unsigned long long fib_ultim(int n)//Семантика
{
    int i;//счетчик
    //Значение функции на два и один шаг назад
    unsigned long long back2=1, back1=2, next;
    if (n<2){//Выход если листьев меньше 2-х
        printf("Листьев кувшинки не может быть меньше 2-х!\n");
        return 0;
    }
    for (i=2; i<n; i++)//при n<2 цикл не выполняется
    {
        next=back1+back2;
        printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
                i-1, back2,back1,next);
        back2=back1;//обмен содержимым переменных
        back1=next;//обмен содержимым переменных
    }
    printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
                i-1, back2,back1,back1+back2);
    return back1+back2;
}
```

```
Введите количество листьев кувшинки:
9
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Шаг 6 back2=13 back1=21 next=34
Шаг 7 back2=21 back1=34 next=55
Ответ: необходимо прыгать с берега 55 раз.
Нажмите любую кнопку для завершения.
```

6. На карьере Марсианский автономный робот проводит взрывные работы. Для этого внутри границ карьера высверливаются отверстия под заряды. Заряды располагаются по углам прямоугольника со стороной 1 шаг робота и делаются по всей поверхности карьера. Робот может установить заряд только внутри карьера, так как по краям установке мешают стенки карьера. Карта карьера передается роботу со спутника в виде координат углов стенок в шагах робота. Спутник передал следующее сообщение: {7,1; 4,3; 3,7; 6,7; 9,6; 9,3}. Сколько отверстий под заряды высверлит робот в карьере? Напишите алгоритм и программу, позволяющую вычислить, сколько отверстий под заряды высверлит робот в карьере с произвольными целыми координатами углов стенок карьера. Алгоритм должен раскрывать решение на каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте специальные функции и библиотеки для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Допустимо задать готовые координаты в коде основной функции. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для координат углов стенок карьера в шагах робота {7,1; 4,3; 3,7; 6,7; 9,6; 9,3}, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача из области вычислительной геометрии на вычисление количества точек с целочисленными координатами, лежащих внутри (но не на границе) многоугольника, заданного координатами своих вершин.

В задаче требуется рассчитать количество точек с целыми координатами, попадающими внутрь многоугольника, не считая точки, лежащие точно на ребрах многоугольника. Ответ можно проверить визуально. Ожидается увидеть алгоритм решения в виде рисунка/рисунков и формул Пика и НОД. **Ответ**: **21** точка.

Теория_1. Для любого многоугольника с целочисленными координатами вершин справедлива **формула Пика**: S = n + m/2 — 1, где S – площадь многоугольника, n – количество целых точек лежащих строго внутри многоугольника, m – количество целых точек, лежащих на границе многоугольника. Площадь многоугольника S вычисляется методом трапеций (**ожидаемое решение**) или по формуле Гаусса (“шнуровки”, **допустимое равнозначное решение**). Количество целых точек m, лежащих на границе

многоугольника, вычисляется как НОД(катет_1, катет_2) + 1, где катет_1,2 - катеты прямоугольного треугольника, образованного на каждом ребре многоугольника (гипотенуза). Искомое количество целых точек n, лежащих строго внутри многоугольника, вычисляется как n = S - m/2 + 1.

Теория_2. Площадь многоугольника. Для вычисления площади многоугольника необходимо “замкнуть” периметр, добавив последнюю_вершину = первая_вершина. Формулы площадей показаны ниже.

$$S_{Гаусс} = \frac{1}{2}\left|\sum_{i=1}^{n-1} x_i y_{i+1} + x_n y_1 - \sum_{i=1}^{n-1} x_{i+1} y_i - x_1 y_n\right|$$

$$S_{трап} = \frac{1}{2}\left|\sum_{i=1}^{n-1} (x_{i+1} - x_i)(y_{i+1} + y_i)\right|$$

Теория_3. Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1. Катеты прямоугольного треугольника для каждого ребра многоугольника вычисляются как разности координат вершин, образующих ребро. Для вертикального или горизонтального ребра один из катетов будет равен нулю. Необходимо последовательно перебрать все вершины за исключением “замыкающей”, вычислить НОД катетов и просуммировать их. Алгоритм вычисления показан ниже.

```
lineNodes(polygon) := "Количество узлов, лежащих на ребрах многоугольника."
                      "Сумма НОД катетов прямоугольных трегольников"
                      "для каждого ребра-гипотенузы"
                      "Замкнутый контур. Последняя_вершина=первая_вершина"
                      s ← 0
                      for i ∈ ORIGIN .. rows(polygon) − 1
                        s ← s + gcd(polygon_{i+1,1} − polygon_{i,1}, polygon_{i+1,2} − polygon_{i,2})
                      s
```

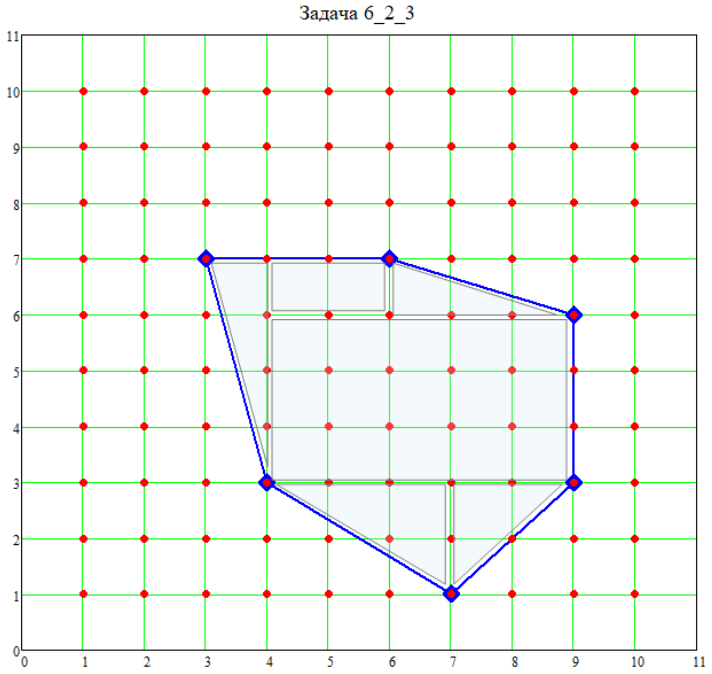
Алгоритм вычисления площади многоугольника **методом трапеций**:

```
trapSquare(polygon) := "Площадь многоугольника методом трапеций"
                       "Замкнутый контур. Последняя_вершина=первая_вершина"
                       s ← 0
                       for i ∈ ORIGIN .. rows(polygon) − 1
                         a ← polygon_{i+1,1} − polygon_{i,1}
                         b ← polygon_{i+1,2} + polygon_{i,2}
                         s ← s + a·b
                       s ← 0.5 |s|
                       s
```

Алгоритм вычисления площади многоугольника **методом Гаусса**:

```
gaussSquare(polygon) := "Площадь многоугольника методом Гаусса ("шнуровки")"
                        "Замкнутый контур. Последняя_вершина=первая_вершина"
                        s ← 0
                        for i ∈ ORIGIN .. rows(polygon) − 1
                            s ← s + polygon_{i+1,1} · polygon_{i,2}
                            s ← s − polygon_{i,1} · polygon_{i+1,2}
                        s ← s + polygon_{rows(polygon),1} · polygon_{1,2}
                        s ← s − polygon_{1,1} · polygon_{rows(polygon),2}
                        s ← 0.5 |s|
                        s
```

Графическое решение и ответ показаны ниже.



$$polygon\_1 \equiv \begin{pmatrix} 7 & 1 \\ 4 & 3 \\ 3 & 7 \\ 6 & 7 \\ 9 & 6 \\ 9 & 3 \end{pmatrix}$$

Границы карьера

$$polygon\_1 := stack\left[polygon\_1, \left(polygon\_1^{T}\right)^{\langle 1 \rangle^{T}}\right]$$

$$trapSquare(polygon\_1) = 25.5$$

$$squarePoly\_1 := gaussSquare(polygon\_1) = 25.5$$

$$linenodesPoly\_1 := lineNodes(polygon\_1) = 11$$

$$inlineNodes := squarePoly\_1 - \frac{linenodesPoly\_1}{2} + 1 = 21$$

Ответ

Ниже приведен пример решения на Python 3.82 и C.

```python
#Формула Пика
#НОД
def NOD(a,b):
    while b != 0:
        a, b = b, a % b
    return a
#Количество узлов, лежащих
#на ребрах многоугольника
def lineNodes(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        dX=polygon[0][i+1]-polygon[0][i]
        dY=polygon[1][i+1]-polygon[1][i]
        s=s+NOD(abs(dX),abs(dY))
    return s

#Площадь многоугольника методом трапеций
def trapSquare(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        a=polygon[0][i+1]-polygon[0][i]
        b=polygon[1][i+1]+polygon[1][i]
        s=s+a*b
    s=abs(s)/2
    return s

#Площадь многоугольника методом Гаусса
def gaussSquare(polygon):
    s=0; last=len(polygon[0])-1
    for i in range(last):
        s=s+polygon[0][i+1]*polygon[1][i]
        s=s-polygon[0][i]*polygon[1][i+1]
    s=s+polygon[0][last]*polygon[1][0]
    s=s-polygon[0][0]*polygon[1][last]
    s=0.5*abs(s)
    return s
#Вычисление по формуле Пика
#S = n + m/2 – 1
def inlineNodes(polygon):
    S=trapSquare(polygon)
    print("Площадь многоугольника",S)
    m=lineNodes(polygon)
    print("Количество точек на ребрах",m,"шт.")
    n=S-m/2+1
    return int(n)


#Координаты углов многоугольника
#двумерный список (x,y)
Plgn=[[7,4,3,6,9,9],[1,3,7,7,6,3]]
#"Замыкание" многоугольника
Plgn[0].append(Plgn[0][0])
Plgn[1].append(Plgn[1][0])

print("Количество точек внутри многоугольника" \
      ,inlineNodes(Plgn),"шт.")
```

```
Площадь многоугольника 25.5
Количество точек на ребрах 11 шт.
Количество точек внутри многоугольника 21 шт.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int NOD(int a, int b);//НОД
int lineNodes(int *X, int *Y, int len);//Количество точек на ребрах многоугольника
int inlineNodes(int *X, int *Y, int len);//Вычисление по формуле Пика
float trapSquare(int *X, int *Y, int len);//S многогольника методом трапеций
float gaussSquare(int *X, int *Y, int len);//S многогольника методом Гаусса


int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");              // Очистка консоли
    //Сразу задаем "замкнутый" массив координат углов
    int pointsX[]={7,4,3,6,9,9,7};
    int pointsY[]={1,3,7,7,6,3,1};

    //Количество углов в массиве координат
    int n=sizeof(pointsX)/sizeof(pointsX[0]);
    printf("Количество точек внутри многоугольника %i шт.\n", \
            inlineNodes(pointsX,pointsY,n));
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}

```

```
30    //Вычисление НОД
31    int NOD(int a, int b){
32        int t;
33        while (b != 0){
34            t = b;
35            b = a % b;
36            a = t;
37        }
38        return a;
39    }
40    //Вычисление площади методом трапеций
41    float trapSquare(int *X, int *Y, int len){
42        float s=0;
43        for (int i=0; i<len-1; i++){
44            int a= X[i+1]- X[i];
45            int b= Y[i+1]+ Y[i];
46            s=s+a*b;
47        }
48        s=(float)abs(s)/2;
49        return s;
50    }
51    //Вычисление площади методом Гаусса
52    float gaussSquare(int *X, int *Y, int len){
53        float s=0; int last=len-1;
54        for (int i=0; i<last; i++){
55            s=s+X[i+1]*Y[i];
56            s=s-X[i]*Y[i+1];
57        }
58        s=s+X[last]*Y[0];
59        s=s-X[0]*Y[last];
60        s=(float)abs(s)/2;
61        return s;
62    }
```

```
63      //Количество точек на ребрах многоугольника
64      int lineNodes(int *X, int *Y, int len){
65          int s=0;
66          for (int i=0; i<len-1; i++){
67              int dX=X[i+1]-X[i];
68              int dY=Y[i+1]-Y[i];
69              s=s+NOD(abs(dX),abs(dY));
70          }
71          return s;
72      }
73      //Вычисление по формуле Пика
74      //S = n + m/2 — 1
75      int inlineNodes(int *X, int *Y, int len){
76          float S=trapSquare(X,Y,len);
77          printf("Площадь многоугольника %.2f\n",S);
78          int m=lineNodes(X,Y,len);
79          printf("Количество точек на ребрах %i шт.\n",m);
80          int n=S-m/2+1;//Точек внутри многоугольника
81          return n;
82      }
```

c_pik

```
Площадь многоугольника 25.50
Количество точек на ребрах 11 шт.
Количество точек внутри многоугольника 21 шт.
```

ИНФОРМАТИКА

ВАРИАНТ 8

1. Современные преподаватели ведут электронный журнал. В одном университете приняли бальную систему оценивания на экзамене. Преподаватель по информатике решил сделать образцовый журнал, отвечающий установленным требованиям, описанным ниже:

1. Студент за семестр должен набрать минимум 60 баллов. Эти баллы формируются из сданных за семестр работ, а также максимум 20 баллов можно получить за две контрольные работы.
2. Практическая работа считается сданной вовремя, если она сдана в рамках 2 недель после занятия, на котором была объяснена. За нее студент получает 6 баллов, если сдает невовремя, получает 5, в противном случае работа считается не сданной и студент ничего не получает.
3. Суммарные баллы и составляют оценку.

Преподаватель переписал список группы и сданные работы за семестр по датам. Чтобы автоматизировать процесс, он решил записать в ячейки электронной таблицы готовые формулы:

для определения баллов за одну работу преподаватель ввел формулу: =ЕСЛИ(ПОИСКПОЗ("*"AA$2"*";$C3:$Z3;0)-ПОИСКПОЗ("*"AA$2"*";$C$1:$Z$1;0)<=2;AA$1;AA$1-1) и протянул формулу до конца календарного плана (столбцы U:AF) и до конца списка студентов (до 12 строки), эта формула поможет понять, сколько баллов студент получил за каждую сданную работу.

За каждую контрольную работу студент получает не более 10 баллов. Максимально за контрольные студент может получить 20 баллов (баллы за контрольную работу указаны в диапазоне ячеек AG:AH).

Известно, что «отлично» получает студент, набравший от 90 баллов суммарно за семестр, «хорошо», если сумма баллов меньше или равна 89 и больше или равна 80, а «удовлетворительно», если сумма больше или равна 60 и меньше или равна 79. В противном случае оценка «неуд». Подсчет оценки был реализован с помощью формулы: =ЕСЛИ(СУММ(U4:AH4)>=90;5;ЕСЛИ(И(СУММ(U4:AH4)<=89;СУММ(U4:AH4)>=80);4;ЕСЛИ(И(СУММ(U4:AH4)>=79;СУММ(U4:AH4)<=60);3;"неуд"))), затем формулу скопировали до конца журнала. Таким образом, для каждого студента удалось посчитать итоговый балл. К сожалению, преподаватель торопился и допустил ошибку в одной из формул.

Укажите правильную формулу и определите, кто из студентов получил оценку «хорошо», зная график сдач их работ и баллы за контрольную работу:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1			**л1**	**л1**	**л2**	**л3**	**л4**	**л5**	**л5**	**КР**	**л6**	**л7**	**л8**	**л9**	**л10**	**л11**	**л12**	**КР**		
2	**№**	**Фамилия, Имя**	03.сен	10.сен	17.сен	24.сен	01.окт	08.окт	15.окт	22.окт	29.окт	05.ноя	12.ноя	19.ноя	26.ноя	03.дек	10.дек	17.дек	24.дек	31.дек
3	1	Назаров Дмитрий			л1	н	л2	л3	н	н	л4		л5, л6	л7		л8	л9	л10	н	л11, л12
4	2	Овчинникова Алиса			л1	л2		л3, л4	л5		л6	н	н	л7	л8	л9	н	л10, л11		л12
5	3	Примаков Владимир		л1		л2, л3		л4		н	л6, л7			л5, л9	л8	н			л10	л11, л12
6	4	Стефанова Виктория			л1, л2	л4	н			л5			л3, л6		л7, л8	н	л9	л10		л11, л12
7	5	Тимофеева Полина			л1	н			л4, л2, л3			н	н	н	л6		л5, л7		л8	
8	6	Уваров Георгий			л3		л4	н	л5	л2, л6, л1	н			л7			л10, л8, л9		л11	л12
9	7	Федоров Владислав			л1		н	л4	л3	л2			л6, л7	л5				л9		
10	8	Чашкина Дарья			л1	л2	л3		н		л4, л6, л7	н		н		л9, л10	л8, л5		л11	л11, л12
11	9	Шишкин Григорий		н	л1	л2	л3	л4				л5, л6	л7		л8, л9	л10	н	н	л11	л12
12	10	Яковлева Жанна			л1		н	л4		л2	л3		л6, л7	н	л9	л5	л10, л8		н	л11, л12

Рисунок 1 – график сдач работ студентов

	A	B	AG	AH
1				
2	№	Фамилия, Имя	КР1	КР2
3	1	Назаров Дмитрий	10	9
4	2	Овчинникова Алиса	10	10
5	3	Примаков Владимир	8	6
6	4	Стефанова Виктория	8	10
7	5	Тимофеева Полина	6	9
8	6	Уваров Георгий	5	5
9	7	Федоров Владислав	10	8
10	8	Чашкина Дарья	8	10
11	9	Шишкин Григорий	9	10
12	10	Яковлева Жанна	10	10

Рисунок 2 – баллы за контрольную работу

В ответе укажите правильную формулу, фамилию (фамилии) студентов, получивших «хорошо», и конечное количество баллов за семестр.**(10 баллов)**

Решение:

	A	B	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF
1			*б*	*б*	*б*	*б*	*б*	*б*	*б*	*б*	*б*	*б*	*б*	*б*
2	№	Фамилия, Имя	*л1*	*л2*	*л3*	*л4*	*л5*	*л6*	*л7*	*л8*	*л9*	*л10*	*л11*	*л12*
3	1	Назаров Дмитрий	=ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&U$2&"*";$C3:$T3;0)-ПОИСКПОЗ("*"&U$2&"*";$C$1:$T$1;0)<=2;U$1;U$1-1);0)											
4	2	Овчинникова Алиса	6	6	6	6	6	6	6	6	6	5	6	5
5	3	Примаков Владимир	6	6	6	6	5	6	6	6	6	5	5	5
6	4	Стефанова Виктория	6	6	5	6	6	6	5	6	5	5	5	5
7	5	Тимофеева Полина	6	5	5	6	5	5	5	5	0	0	0	0
8	6	Уваров Георгий	5	5	6	6	6	6	6	5	5	6	5	5
9	7	Федоров Владислав	6	5	5	6	5	6	6	0	5	0	0	0
10	8	Чашкина Дарья	6	6	6	5	5	6	6	5	6	6	5	5
11	9	Шишкин Григорий	6	6	6	6	5	6	6	6	6	6	5	5
12	10	Яковлева Жанна	6	5	5	6	5	6	6	5	6	6	5	5

	A	B	AI	AJ	AK	AL	AM	AN	AO	AP	AQ	AR	AS	AT	AU
1															
2	№	Фамилия, Имя	Оценка	Итоговый балл											
3	1	Назаров Дмитрий	=ЕСЛИ(СУММ(U3:AH3)>=90;5;ЕСЛИ(И(СУММ(U3:AH3)<=89;СУММ(U3:AH3)>=80);4;ЕСЛИ(И(СУММ(U3:AH3)<=79;СУММ(U3:AH3)>=60);3;"неуд")))												
4	2	Овчинникова Алиса	5	90											
5	3	Примаков Владимир	4	82											
6	4	Стефанова Виктория	4	84											
7	5	Тимофеева Полина	неуд	57											
8	6	Уваров Георгий	3	76											
9	7	Федоров Владислав	3	62											
10	8	Чашкина Дарья	4	85											
11	9	Шишкин Григорий	4	88											
12	10	Яковлева Жанна	4	86											

	A	B	AI	AJ
1				
2	**№**	**Фамилия, Имя**	**Оценка**	**Итоговый балл**
3	1	Назаров Дмитрий	4	84
4	2	Овчинникова Алиса	5	90
5	3	Примаков Владимир	4	82
6	4	Стефанова Виктория	4	84
7	5	Тимофеева Полина	неуд	57
8	6	Уваров Георгий	3	76
9	7	Федоров Владислав	3	62
10	8	Чашкина Дарья	4	85
11	9	Шишкин Григорий	4	88
12	10	Яковлева Жанна	4	86

Правильным также может считаться ответ с формулой, в которой указаны строки с 3 до 12. Например: =ЕСЛИ(СУММ(U3:AH3)>=90;5;ЕСЛИ(И(СУММ(U3:AH3)<=89;СУММ(U3:AH3)>=80);4;ЕСЛИ(И(СУММ(U3:AH3)<=79;СУММ(U3:AH3)>=60);3;"неуд"))) или =ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&U$2&"*";$C5:$T5;0)-ПОИСКПОЗ("*"&U$2&"*";$C$1:$T$1;0)<=2;U$1;U$1-1);0), также могут быть указаны столбцы с V по AF, например: =ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&V$2&"*";$C5:$T5;0)-ПОИСКПОЗ("*"&V$2&"*";$C$1:$T$1;0)<=2;V$1;V$1-1);0),

Ответ: Назаров, Примаков, Стефанова, Чашкина, Шишкин, Яковлева, 84, 82, 84, 85, 88, 86.
=ЕСЛИ(СУММ(U4:AH4)>=90;5;ЕСЛИ(И(СУММ(U4:AH4)<=89;СУММ(U4:AH4)>=80);4;ЕСЛИ(И(СУММ(U4:AH4)<=79;СУММ(U4:AH4)>=60);3;"неуд"))) или =ЕСЛИОШИБКА(ЕСЛИ(ПОИСКПОЗ("*"&U$2&"*";$C3:$T3;0)-ПОИСКПОЗ("*"&U$2&"*";$C$1:$T$1;0)<=2;U$1;U$1-1);0). Ответ также считается правильным, если ошибки найдены в обеих формулах.

2. Два сладкоежки решили сыграть в игру «Поедание конфет». На столе лежат две вазочки по 14 конфет в каждой. Игроки ходят по очереди. За один ход можно взять любое число конфет (1; 2; 3; и т.д.) из одной из вазочек (по выбору игрока). Кто не может сделать ход (конфет не осталось), проигрывает. Кто выигрывает при безошибочной игре – игрок, делающий первый ход, или игрок, делающий второй ход? Поясните алгоритм графически. Напишите обобщенный алгоритм для любого количества конфет. **(10 баллов)**

Решение:

Используется Стратегия_Уравняй. Всегда выигрывает Игрок_2 (игрок, делающий ход вторым). Суть Стратегия_Уравняй:

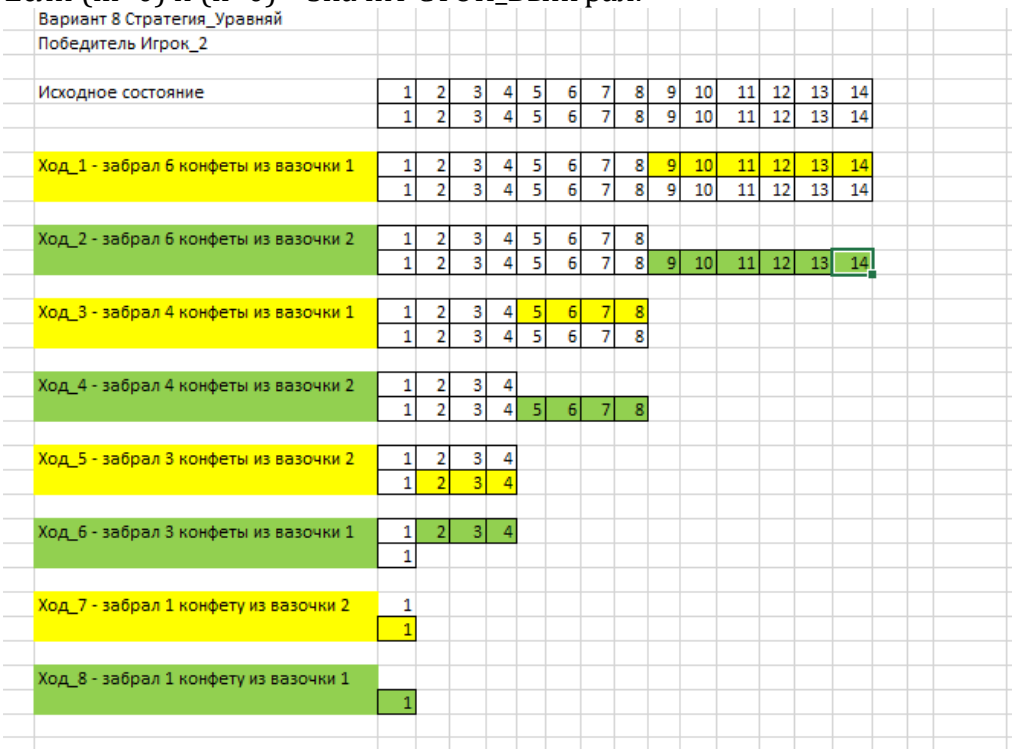
Ход_1: Игрок_1 (игрок, делающий ход первым) берёт любое количество конфет из одной вазочки (то есть в одной вазочке остаётся 14 конфет, а в другой остаётся от 0 до 13 конфет).

Ход_2: Игрок_2 уравнивает количество конфет в вазочках, взяв такое же количество конфет как и Игрок_1 в Ход_1 из другой вазочки с большим количеством конфет.

И так далее до Победа Игрок_1. Проиграть невозможно.

Иллюстрация на рисунке ниже. Алгоритм подходит в общем случае, если m = n, то выигрывает второй: он должен своим ходом уравнять кучки, а потом поддерживать равенство.

Алгоритм для Игрок_2 (обобщенный):
m-количество конфет в первой вазочке
n- количество конфет во второй вазочке
Если m=n
Шаг 1 Игрок _1 взял k конфет из вазочки 1
Шаг 2 Игрок_2 взял k конфет из вазочки 2
Если (m=0) и (n=0) – значит СТОП_Выиграл.



3. С недавних пор Аня научилась работать с таблицами истинности. Для того, чтобы оттачивать свой навык, она решила каждый день строить таблицу для одного логического выражения. Но этого ей показалось мало, и она решила каждый полученный результат разделять на 4 цвета в двоичной системе счисления:

0_{10} – белый

1_{10} – черный

2_{10} – синий

3_{10} – красный

Например комбинация, 01000011 расшифровывается, как черный, белый, белый, красный.

В итоге у нее получилась следующая комбинация:

Белый-белый-черный-белый

Известно, что переменные в примере были расставлены в следующем порядке: *x, y, z, x*. Также были получены промежуточные результаты:

F_1=красный-белый-красный-белый

F_2=черный-белый-черный-белый

F_3=красный-красный-черный-белый

И F_4 – результирующий, где Fn – это действия по порядку.

Напишите недостающие логические операции между переменными. В ответе предоставьте полное логическое выражение. **(10 баллов)**

Решение:

1. Значение F1=красный-белый-красный-белый (11001100), следовательно, первое действие – отрицание y.
2. Действие F2=черный-белый-черный-белый (01000100), очевидно конъюнкция результатов первого действия и z.
3. Результат F3= красный-красный-черный-белый (11110100) соответствует импликации x и результатов 2 действия.
4. Конечное значение (белый-белый-черный-белый) результат эквивалентности результатов третьего действия и x.

Ответ: $x \to \bar{y} \wedge z \leftrightarrow x$

4. Владимир устроился работать помощником системного администратора в IT-отдел Университета. В первый же день работы Владимир получил ответственное задание. Отдел информационных технологий Университета использует диапазон IP-адресов 172.22.192.0/18, в котором настроено адресное пространство сетей филиалов из 6 подсетей. Данные по количеству узлов в подсетях известны. При этом адресное пространство упорядочено по размеру.

Владимир обладает следующими знаниями:

- одному устройству должен соответствовать один IP-адрес вида XXX.XXX.XXX.XXX, при этом XXX – число в диапазоне [0:255], называемое октетом;
- маска с префиксом записывается в виде /18;
- устройства внутри одной подсети должны беспрепятственно взаимодействовать друг с другом и должны группироваться по их расположению;
- в каждом сегменте сети первый адрес зарезервирован под идентификатор подсети, а последний – под широковещательную рассылку;

Название подсети	**Адрес сети**	**Требуемый размер сети**	**Маска с префиксом**	**Десятичная маска**	**Диапазон доступных адресов**	**Широковещательный адрес**
Главный корпус		4096				

Инженерный корпус		2048				
Учебный центр №2		512				
Библиотека		128				
Пост охраны		64				
IT-отдел		4				

Заполнение информации вызвало у помощника системного администратора затруднения. Необходимо помочь Владимиру заполнить таблицу недостающими данными.

(**20 баллов**)

Решение:

Название подсети	**Адрес сети**	**Требуемый размер сети**	**Маска с префиксом**	**Десятичная маска**	**Диапазон доступных адресов**	**Широковещательный адрес**
Главный корпус	172.22.192.0	4096	/19	255.255.224.0	172.22.192.1 - 172.22.223.254	172.22.223.255
Инженерный корпус	172.22.224.0	2048	/20	255.255.240.0	172.22.224.1 - 172.22.239.254	172.22.239.255
Учебный центр №2	172.22.240.0	512	/22	255.255.252.0	172.22.240.1 - 172.22.243.254	172.22.243.255
Библиотека	172.22.244.0	128	/24	255.255.255.0	172.22.244.1 - 172.22.244.254	172.22.244.255
Пост охраны	172.22.245.0	64	/25	255.255.255.128	172.22.245.1 - 172.22.245.126	172.22.245.127
IT-отдел	172.22.245.128	4	/29	255.255.255.248	172.22.245.129 - 172.22.245.134	172.22.245.135

5. В пруду вдоль берега растут кувшинки. Листья кувшинок образуют узор в виде треугольного зигзага из 10 листьев. Лягушка может прыгнуть с берега только на первый или второй лист кувшинки и дальше до последнего, пока листья не закончатся. Каждый раз она прыгает по кувшинкам по-новому, чередуя прыжки по каждому или через лист. Сколько раз лягушка сможет прыгнуть с берега по 10 листьям, чтобы ни разу не повторить последовательность прыжков? Напишите алгоритм и программу, позволяющую вычислить, сколько раз лягушка сможет прыгать по узору из произвольного целого N (N>2) количества листьев кувшинки (в пределах типа данных языка программирования), чтобы ни разу не повторить последовательность прыжков. Учтите, что возможности компьютера, на котором будет выполняться программа, ограничены: допустимо использовать до 10 переменных, массивы ограничены 10 ячейками соответствующего типа, а стек вмещает 10 вызовов и не расширяется. Алгоритм должен раскрывать решение на каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте уникальные особенности языка программирования (классы, методы, объекты, попарный обмен между переменными, специальные функции и библиотеки и др., кроме счетчиков циклов) для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для 10 листьев, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача на вычисление последовательности чисел Фибоначчи.

A000045 Fibonacci numbers: F(n) = F(n-1) + F(n-2) with F(0) = 0 and F(1) = 1.
(Formerly M0692 N0256)

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, 1597, 2584, 4181, 6765, 10946, 17711, 28657, 46368, 75025, 121393, 196418, 317811, 514229, 832040, 1346269, 2178309, 3524578, 5702887, 9227465, 14930352, 24157817, 39088169, 63245986, 102334155 (list; graph; refs; listen; history; text;

https://oeis.org/A000045

В задаче требуется рассчитать число уникальных комбинаций прыжков, которыми можно прыгать по ряду из **10** листьев, используя прыжки длиною в 1, 2 или 1 и 2 листа кувшинки. **Ответом является 10-е** по счету число последовательности Фибоначчи (**a_{10} = 89**), считая, что **a_1 = 1**, **a_2 = 2**.

Объяснение 1. Обозначим число комбинаций как a_n, где n – число ступенек, n>2.
a_1 = 1 способ –> по ряду из одного листа кувшинки можно прыгать только одним способом - прыжком в 1 лист.
a_2 = 2 способа –> 1+1 лист; сразу 2 листа.
a_3 = 3 –> 1+1+1; 1+2; 2+1 листа.
a_4 = 5 –> 1+1+1+1; 1+1+2; 1+2+1; 2+1+1; 2+2 листа.
$a_4 = a_3 + a_2$; $a_3 = a_2 + a_1$; ==> **$a_n = a_{n-1} + a_{n-2}$;**
$a_5 = a_4 + a_3 = 5+3=8$; $a_6 = a_5 + a_4 = 8+5=13$; $a_7 = a_6 + a_5 = 13+8=21$;
$a_8 = a_7 + a_6 = 21+13=34$; $a_9 = a_8 + a_7 = 34+21=55$; $a_{10} = a_9 + a_8 = 55+34=89$.
Ответ: a_{10} = 89 способов.
Объяснение 2. Обозначим число комбинаций как a_n, где n – число листьев кувшинки, n>2. Можно начать прыжки с прыжка в 1 лист. Тогда останется a_{n-1} комбинаций прыжков по

ряду листьев. Если начать подъем с прыжка в 2 листа, то останется a_{n-2} комбинаций прыжков по ряду листьев. Первый прыжок входит в любую из комбинаций прыжков. Таким образом количество комбинаций определяется суммой количества оставшихся листьев для каждого варианта первого прыжка и размером первого прыжка, т.е. $\mathbf{a_n = a_{n-1} + a_{n-2}}$.

Задачу можно решить, например *рекурсией* (*неправильно*, ограничение по стеку вызовов при больших N), **динамическим программированием (ожидаемое решение)**, с помощью умножения матриц (допустимое, но избыточное решение), формулы Бине (допустимое, но избыточное решение). Ниже приведен пример решения с помощью динамического программирования на Python 3.82 и C.

```
#Функция Фибоначи с обменом
#Экономит память, хранит только два
#предыдущих значения
def fib_ult(n):#Для вывода по шагам
    #Значение функции на два и один шаг назад
    back2=1; back1=2
    if n<2:
        print("Листьев кувшинки не может быть меньше 2-х!")
        return 0
    for i in range(2,n-1):
        next=back1+back2
        print("Шаг ",i-1," back2=",back2," ",\
              "back1=",back1," ","next=",next,sep='')
        back2=back1#обмен содержимым переменных
        back1=next#обмен содержимым переменных
    print("Шаг ",i," back2=",back2," ",\
          "back1=",back1," ","next=",back1+back2,sep='')
    return back1+back2
fibNumb=int(input("Введите количество листьев кувшинки:\n"))
print("Ответ: необходимо прыгать с берега ",fib_ult(fibNumb)," раз.",sep='')
```

```
Введите количество листьев кувшинки:
10
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Шаг 6 back2=13 back1=21 next=34
Шаг 7 back2=21 back1=34 next=55
Шаг 8 back2=34 back1=55 next=89
Ответ: необходимо прыгать с берега 89 раз.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int fibNumb;//Порядковый номер в ряду Фибоначчи
unsigned long long answ;//Ответ
unsigned long long fib_ultim(int n);//Сигнатура

int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");                // Очистка консоли
    printf("Введите количество листьев кувшинки:\n");
    scanf("%d",&fibNumb);
    answ=fib_ultim(fibNumb-1);//Так как нумерация массивов с нуля
    if (answ==0){//Выход если листьев меньше 2-х
        getchar();//Задержка для отображения вывода
        printf("Нажмите любую кнопку для завершения.\n");
        int ch = getch();
        return (EXIT_SUCCESS);
    }
    printf("Ответ: необходимо прыгать с берега %llu раз.  \n",answ);
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}
```

```c
unsigned long long fib_ultim(int n)//Семантика
{
    int i;//счетчик
    //Значение функции на два и один шаг назад
    unsigned long long back2=1, back1=2, next;
    if (n<2){//Выход если листьев меньше 2-х
        printf("Листьев кувшинки не может быть меньше 2-х!\n");
        return 0;
    }
    for (i=2; i<n; i++)//при n<2 цикл не выполняется
    {
        next=back1+back2;
        printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
                i-1, back2,back1,next);
        back2=back1;//обмен содержимым переменных
        back1=next;//обмен содержимым переменных
    }
    printf("Шаг %d back2=%llu back1=%llu next=%llu\n", \
                i-1, back2,back1,back1+back2);
    return back1+back2;
}
```

```
Введите количество листьев кувшинки:
10
Шаг 1 back2=1 back1=2 next=3
Шаг 2 back2=2 back1=3 next=5
Шаг 3 back2=3 back1=5 next=8
Шаг 4 back2=5 back1=8 next=13
Шаг 5 back2=8 back1=13 next=21
Шаг 6 back2=13 back1=21 next=34
Шаг 7 back2=21 back1=34 next=55
Шаг 8 back2=34 back1=55 next=89
Ответ: необходимо прыгать с берега 89 раз.
Нажмите любую кнопку для завершения.
```

6. На карьере Марсианский автономный робот проводит взрывные работы. Для этого внутри границ карьера высверливаются отверстия под заряды. Заряды располагаются по углам прямоугольника со стороной 1 шаг робота и делаются по всей поверхности карьера. Робот может установить заряд только внутри карьера, так как по краям установке мешают стенки карьера. Карта карьера передается роботу со спутника в виде координат углов стенок в шагах робота. Спутник передал следующее сообщение: {5,2; 2,4; 4,8; 7,8; 10,10; 9,1}. Сколько отверстий под заряды высверлит робот в карьере? Напишите алгоритм и программу, позволяющую вычислить, сколько отверстий под заряды высверлит робот в карьере с произвольными целыми координатами углов стенок карьера. Алгоритм должен раскрывать решение на каждом шаге. Код программы должен состоять из простых и понятных команд. Не используйте специальные функции и библиотеки для реализации основных этапов решения. Считывание заранее рассчитанных значений не засчитывается за решение задачи. Допустимо задать готовые координаты в коде основной функции. Код должен содержать достаточное для понимания логики работы количество комментариев. Докажите правильность работы кода для координат углов стенок карьера в шагах робота {5,2; 2,4; 4,8; 7,8; 10,10; 9,1}, раскрыв содержимое переменных на каждом шаге. **(25 баллов)**

Решение:

Алгоритм решения задачи: задача из области вычислительной геометрии на вычисление количества точек с целочисленными координатами, лежащих внутри (но не на границе) многоугольника, заданного координатами своих вершин.

В задаче требуется рассчитать количество точек с целыми координатами, попадающими внутрь многоугольника, не считая точки, лежащие точно на ребрах многоугольника. Ответ можно проверить визуально. Ожидается увидеть алгоритм решения в виде рисунка/рисунков и формул Пика и НОД. **Ответ**: **39** точек.

Теория_1. Для любого многоугольника с целочисленными координатами вершин справедлива **формула Пика**: S = n + m/2 — 1, где S – площадь многоугольника, n – количество целых точек лежащих строго внутри многоугольника, m – количество целых точек, лежащих на границе многоугольника. Площадь многоугольника S вычисляется

методом трапеций (**ожидаемое решение**) или по формуле Гаусса (“шнуровки”, **допустимое равнозначное решение**). Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1, где катет_1,2 - катеты прямоугольного треугольника, образованного на каждом ребре многоугольника (гипотенуза). Искомое количество целых точек n, лежащих строго внутри многоугольника, вычисляется как n = S - m/2 + 1.

Теория_2. Площадь многоугольника. Для вычисления площади многоугольника необходимо “замкнуть” периметр, добавив последнюю_вершину = первая_вершина. Формулы площадей показаны ниже.

$$S_{Гаусс} = \frac{1}{2}\left|\sum_{i=1}^{n-1} x_i y_{i+1} + x_n y_1 - \sum_{i=1}^{n-1} x_{i+1} y_i - x_1 y_n\right|$$

$$S_{трап} = \frac{1}{2}\left|\sum_{i=1}^{n-1} (x_{i+1} - x_i)(y_{i+1} + y_i)\right|$$

Теория_3. Количество целых точек m, лежащих на границе многоугольника, вычисляется как НОД(катет_1, катет_2) + 1. Катеты прямоугольного треугольника для каждого ребра многоугольника вычисляются как разности координат вершин, образующих ребро. Для вертикального или горизонтального ребра один из катетов будет равен нулю. Необходимо последовательно перебрать все вершины за исключением “замыкающей”, вычислить НОД катетов и просуммировать их. Алгоритм вычисления показан ниже.

```
lineNodes(polygon) := "Количество узлов, лежащих на ребрах многоугольника."
                      "Сумма НОД катетов прямоугольных трегольников"
                      "для каждого ребра-гипотенузы"
                      "Замкнутый контур. Последняя_вершина=первая_вершина"
                      s ← 0
                      for i ∈ ORIGIN.. rows(polygon) − 1
                        s ← s + gcd(polygon_{i+1,1} − polygon_{i,1}, polygon_{i+1,2} − polygon_{i,2})
                      s
```

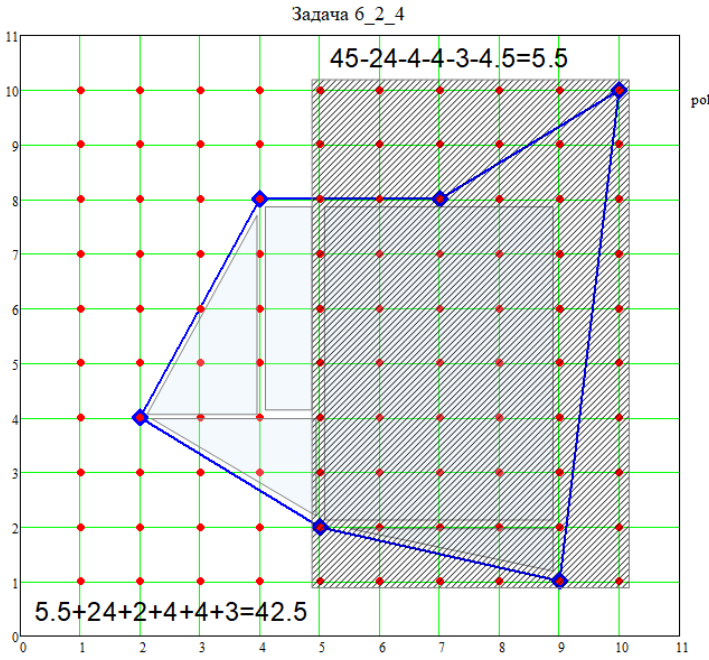
Алгоритм вычисления площади многоугольника **методом трапеций**:

```
trapSquare(polygon) := "Площадь многоугольника методом трапеций"
                       "Замкнутый контур. Последняя_вершина=первая_вершина"
                       s ← 0
                       for i ∈ ORIGIN.. rows(polygon) − 1
                         a ← polygon_{i+1,1} − polygon_{i,1}
                         b ← polygon_{i+1,2} + polygon_{i,2}
                         s ← s + a·b
                       s ← 0.5 |s|
                       s
```

Алгоритм вычисления площади многоугольника **методом Гаусса**:

```
gaussSquare(polygon) :=
    "Площадь многоугольника методом Гаусса ("шнуровки")"
    "Замкнутый контур. Последняя_вершина=первая_вершина"
    s ← 0
    for i ∈ ORIGIN .. rows(polygon) − 1
        s ← s + polygon_{i+1,1} · polygon_{i,2}
        s ← s − polygon_{i,1} · polygon_{i+1,2}
    s ← s + polygon_{rows(polygon),1} · polygon_{1,2}
    s ← s − polygon_{1,1} · polygon_{rows(polygon),2}
    s ← 0.5 |s|
    s
```

Графическое решение и ответ показаны ниже.



$$polygon\_1 \equiv \begin{pmatrix} 5 & 2 \\ 2 & 4 \\ 4 & 8 \\ 7 & 8 \\ 10 & 10 \\ 9 & 1 \end{pmatrix}$$

Границы карьера

$$polygon\_1 := stack\left[polygon\_1, \left(polygon\_1^{T}\right)^{\langle 1 \rangle^{T}}\right]$$

$$trapSquare(polygon\_1) = 42.5$$

$$squarePoly\_1 := gaussSquare(polygon\_1) = 42.5$$

$$linenodesPoly\_1 := lineNodes(polygon\_1) = 9$$

$$inlineNodes := squarePoly\_1 - \frac{linenodesPoly\_1}{2} + 1 = 39$$

Ответ

Ниже приведен пример решения на Python 3.82 и C.

```python
#Формула Пика
#НОД
def NOD(a,b):
    while b != 0:
        a, b = b, a % b
    return a
#Количество узлов, лежащих
#на ребрах многоугольника
def lineNodes(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        dX=polygon[0][i+1]-polygon[0][i]
        dY=polygon[1][i+1]-polygon[1][i]
        s=s+NOD(abs(dX),abs(dY))
    return s

#Площадь многоугольника методом трапеций
def trapSquare(polygon):
    s=0
    for i in range(len(polygon[0])-1):
        a=polygon[0][i+1]-polygon[0][i]
        b=polygon[1][i+1]+polygon[1][i]
        s=s+a*b
    s=abs(s)/2
    return s

#Площадь многоугольника методом Гаусса
def gaussSquare(polygon):
    s=0; last=len(polygon[0])-1
    for i in range(last):
        s=s+polygon[0][i+1]*polygon[1][i]
        s=s-polygon[0][i]*polygon[1][i+1]
    s=s+polygon[0][last]*polygon[1][0]
    s=s-polygon[0][0]*polygon[1][last]
    s=0.5*abs(s)
    return s
#Вычисление по формуле Пика
#S = n + m/2 – 1
def inlineNodes(polygon):
    S=trapSquare(polygon)
    print("Площадь многоугольника",S)
    m=lineNodes(polygon)
    print("Количество точек на ребрах",m,"шт.")
    n=S-m/2+1
    return int(n)


#Координаты углов многоугольника
#двумерный список (x,y)
Plgn=[[5,2,4,7,10,9],[2,4,8,8,10,1]]
#"Замыкание" многоугольника
Plgn[0].append(Plgn[0][0])
Plgn[1].append(Plgn[1][0])

print("Количество точек внутри многоугольника" \
      ,inlineNodes(Plgn),"шт.")
```

```
Площадь многоугольника 42.5
Количество точек на ребрах 9 шт.
Количество точек внутри многоугольника 39 шт.
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>//Чтобы консоль сама не закрывалась

int NOD(int a, int b);//НОД
int lineNodes(int *X, int *Y, int len);//Количество точек на ребрах мнгоуголника
int inlineNodes(int *X, int *Y, int len);//Вычисление по формуле Пика
float trapSquare(int *X, int *Y, int len);//S многогольника методом трапеций
float gaussSquare(int *X, int *Y, int len);//S многогольника методом Гаусса


int main()
{
    system("chcp 1251 > nul");// Текущая кодовая страница 1251
    system("cls");              // Очистка консоли
    //Сразу задаем "замкнутый" массив координат углов
    int pointsX[]={5,2,4,7,10,9,5};
    int pointsY[]={2,4,8,8,10,1,2};

    //Количество углов в массиве координат
    int n=sizeof(pointsX)/sizeof(pointsX[0]);
    printf("Количество точек внутри многоугольника %i шт.\n", \
            inlineNodes(pointsX,pointsY,n));
    getchar();//Задержка для отображения вывода
    printf("Нажмите любую кнопку для завершения.\n");
    int ch = getch();
    return (EXIT_SUCCESS);
}

```

```
30    //Вычисление НОД
31    int NOD(int a, int b){
32        int t;
33        while (b != 0){
34            t = b;
35            b = a % b;
36            a = t;
37        }
38        return a;
39    }
40    //Вычисление площади методом трапеций
41    float trapSquare(int *X, int *Y, int len){
42        float s=0;
43        for (int i=0; i<len-1; i++){
44            int a= X[i+1]- X[i];
45            int b= Y[i+1]+ Y[i];
46            s=s+a*b;
47        }
48        s=(float)abs(s)/2;
49        return s;
50    }
51    //Вычисление площади методом Гаусса
52    float gaussSquare(int *X, int *Y, int len){
53        float s=0; int last=len-1;
54        for (int i=0; i<last; i++){
55            s=s+X[i+1]*Y[i];
56            s=s-X[i]*Y[i+1];
57        }
58        s=s+X[last]*Y[0];
59        s=s-X[0]*Y[last];
60        s=(float)abs(s)/2;
61        return s;
62    }
```

```c
//Количество точек на ребрах многоугольника
int lineNodes(int *X, int *Y, int len){
    int s=0;
    for (int i=0; i<len-1; i++){
        int dX=X[i+1]-X[i];
        int dY=Y[i+1]-Y[i];
        s=s+NOD(abs(dX),abs(dY));
    }
    return s;
}
//Вычисление по формуле Пика
//S = n + m/2 — 1
int inlineNodes(int *X, int *Y, int len){
    float S=trapSquare(X,Y,len);
    printf("Площадь многоугольника %.2f\n",S);
    int m=lineNodes(X,Y,len);
    printf("Количество точек на ребрах %i шт.\n",m);
    int n=S-m/2+1;//Точек внутри многоугольника
    return n;
}
```

c_pik

```
Площадь многоугольника 42.50
Количество точек на ребрах 9 шт.
Количество точек внутри многоугольника 39 шт.
```