ВАРИАНТ № 7

Оценка выполнения олимпиадной работы
(заполняется проверяющим)

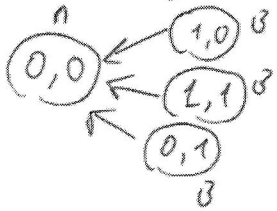
	№ задания	1	2	3	4	5	6	7	8	9	10	Σ
Первая проверка	Полученный балл	2	10	5	20	23	—					60
	Σ баллов прописью	шестьдесят (60)						Подпись проверяющего		шш		−98
								Подпись проверяющего				
Вторая проверка	№ задания	1	2	3	4	5	6	7	8	9	10	Σ
	Полученный балл	2	7	5	14	23						51
	Σ баллов прописью	пятьдесят один (51)						Подпись проверяющего				
								Подпись проверяющего				

2. Для решения задачи воспользуемся методом динамического программирования.

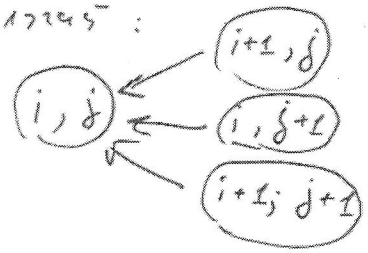
Будем обозначать позицию : (i, j), где i - кол-во камней в 1-й куче, а j - кол-во камней во 2-й куче.

Заметим, что в позицию $(i; j)$ можно попасть всего способами из позиций : $(i+1; j)$, $(i; j+1)$, $(i+1; j+1)$, ~~Это верно, если $i \neq 0$ или $j \neq 0$, иначе~~ это верно, если $i \neq 0$ или $j \neq 0$, иначе один способ

Позиция $(0,0)$ - проигрышная в ней 0 способов попасть:



Общий случай :



Будем хранить информацию о позиции в двумерном массиве $dp[i;j]$: $dp[i;j] = 1$, если позиция - выигрышная, и $dp[i;j] = 0$, если - проигрышная

Вариант № 7

Заполнение будет происходить с позиции (0, 0)

dp[0;0] := 0

Пусть S_1, S_2 - кол-во камней в 1-ой, 2-ой кучи (в начале)

Заполним массив при i = 0 :

j := 1

пока j <= S_2 :

Если dp[0; j-1] = 0 :

то dp[0; j] := 1 ;

иначе :

dp[0; j] := 0 ;

j := j+1 ;

Заполним при j = 0 :

i := 1 ;

Пока i <= S_1 :

Если dp[i-1; 0] = 0 :

то dp[i; 0] := 1 ;

иначе :

dp[i; 0] := 0 ;

j := i + 1 ;

Заполним оставшуюся часть массива :

i := 1 ;

Пока i <= S_1 :

j := 1 ;

Пока j <= S_2 :

Если (dp[i-1; j] = 0) или (dp[i; j-1] = 0) или (dp[i-1; j-1] = 0):

Вариант № 7

```
        то: dp[i;j]:=1;
    Иначе:
        dp[i;j]:=0;
    j:=j+1;
  i:=i+1;
```

Значение dp[s_1, s_2] будет содержать ответ.
Рассмотрим для случая $s_1 = 10$, $s_2 = 8$

	0	1	2	3	4	5	6	7	8
0	0	1	0	1	0	1	0	1	0
1	1	1	1	1	1	1	1	1	1
2	0	1	0	1	0	1	0	1	0
3	1	1	1	1	1	1	1	1	1
4	0	1	0	1	0	1	0	1	0
5	1	1	1	1	1	1	1	1	1
6	0	1	0	1	0	1	0	1	0
7	1	1	1	1	1	1	1	1	1
8	0	1	0	1	0	1	0	1	0
9	1	1	1	1	1	1	1	1	1
10	0	1	0	1	0	1	0	1	0

10б

-3б Угадал!

10б

Таким образом, мы видим, что ~~игр~~ игрок, делающий первый ход, находится в проигрышной позиции => при правильной игре выигрывает второй.

Шифр работы 436-01
(не заполнять)

Вариант № 7

3.

Из схемы получим выражение: $F = x_2 \cdot x_3 + x_1 \oplus x_2$

x_1	x_2	x_3	$\bar{x}_3$	$x_2 \cdot x_3$	$x_1 \oplus x_2$	F	
0	0	0	1	0	0	0	✓
0	0	0	1	0	0	0	✓
0	0	1	0	0	0	0	✓
0	0	1	0	0	0	0	✓
0	1	0	1	0	1	1	
0	1	0	1	0	1	1	
0	1	1	0	1	1	1	
1	0	0	1	0	1	1	
1	0	0	1	0	1	1	
1	0	1	0	0	1	1	
1	0	1	0	0	1	1	
1	1	0	1	0	0	0	✓
1	1	0	1	0	0	0	✓
1	1	1	0	1	0	1	
1	1	1	0	1	0	1	
0	1	1	0	1	1	1	

x_1	x_2	x_3	$\bar{x}_3$	$\bar{x}_3 \oplus x_1$	$x_2 \to x_1$	F
0	0	0	1	1	1	1
0	0	1	0	0	1	0
1	1	0	1	0	1	0

58

50

Вариант № 7

Таким образом, выражение в «хеш» ложно при трёх начальных значениях (x_1, x_2, x_3)

После подстановки в логическое выражение получим столбец: 1, 0, 0

№ 1.

Восстановим исходную формулу:

=СЧЁТЕСЛИ($A$2:$J2;"<0")	4
=B2*E$2/(F$2-H2)	#DEL/0!
=F$2+G$2+H$2	-8
=A2-F2+A2	-6
=ЕСЛИ(B2+E2;1;2)	2
=$A2+B2	-4
=E2+D2+I2	4
=E2+D2*I2	5
=ОКРУГЛВВЕРХ(СРЗНАЧ($A$2:I2);0)-J2	-3

Таким образом, получаем:

1) Среди чисел 4 отрицательных 2) $F2 = H2$

3) $F2 + G2 + H2 = -8$ из 2: $2 \cdot F2 + G2 = -8$

4) $2 \cdot A2 - F2 = -6$ 5) $B2 + E2 = 0$ 6) $A2 + B2 = -4$

7) $E2 + D2 + I2 = 4$ 8) $E2 + D2 * I2 = 5$

9) $СРЗНАЧ. = J2 - 3$

Из диаграммы получаем, что ~~среди чисел есть одно равное нулю среди 1, 2, 8, 9, 10 это (остальные <0)~~

$D, E, F > 0$, $C, G = 0$

Вариант № 7

A	B	C	D	E	F	G	H	I	J
		0		-B	F	0	F		
1	2	3	4	5	6	7	8	9	10
	<0		>0	>0	>0		>0		

E = -B => B < 0

A, B, I, J < 0

20 20

Вариант № 7

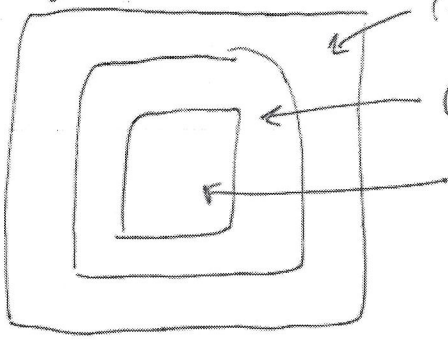
4.

Рассмотрим, как изменяются коды цветов после обработки:

(255, 255, 0) → (0, 0, 255)
(255, 0, 255) → (0, 255, 0)
(0, 0, 255) → (255, 255, 0)

Алгоритм обработки заключается в изменении, максимальной яркости на минимальную, а минимальной на максимальную.

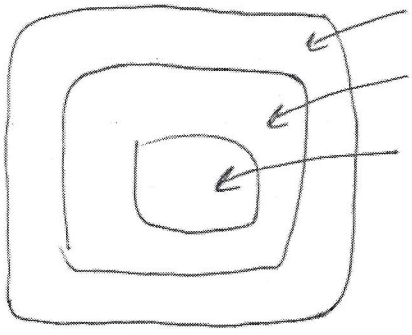
Рисунок 1:



RGB или HEX
(255, 255, 0) или (FF, FF, 00) +
(255, 0, 255) или (FF, 00, FF) +
(0, 0, 255) или (00, 00, FF) +

20 – 6б

Рисунок 2:



(0, 0, 255) или (00, 00, FF) + +
(0, 255, 0) или (00, FF, 00) + +
(255, 255, 0) или (FF, FF, 00) +

не верно
Т. О.

20б

Олимпиада школьников «Гранит науки» Шифр работы И36-01 (не заполнять)

БЛАНК ОЛИМПИАДНОЙ РАБОТЫ

Вариант № 7

5.

Найдём, сколько красных клеток будет в произвольный день D.

После первого дня – клеток R

После второго дня каждая из R клеток создаст (R − 1) клетку //т.к. в нулевой день – 1, а в первый R, то одна создаёт R − 1 // Всего: $R(R-1)+R=R^2$

Аналогично после третьего дня: $R^2(R-1)+R^2=R^3$

После дня d: $R^{d-1}(R-1)+R^{d-1}=R^d$

Таким образом, получаем закономерность:

$1 \to R \to R^2 \to R^3 \to \ldots \to R^D$ или $R^0 \to R^1 \to R^2 \to \ldots \to R^D$

Кол-во красных клеток в день D: R^D

Первая клетка создала N клеток, из которых R – красные ⇒ зелёных (N−R) ⇒ в единичном дереве (под единичным деревом я имею ввиду – дерево, вырастающее из одной красной клетки за одну ночь) на R красных клеток приходится (N−R) зелёных ⇒ если в день D R^D красных клеток, то зелёных: $\frac{R^D-1}{R}(N-R)$ // пропорция всегда сохраняется; −1 – убираем 1-ую клетку //

Таким образом, необходим алгоритм для возведения R в степень D по модулю $m = 1000000007$

Рассмотрим умножение по модулю m:

Вариант № 7

Пусть $a = K_1 \cdot m + r_1$, $b = K_2 \cdot m + r_2$

$a \cdot b = (K_1 m + r_1)(K_2 m + r_2) = K_1 K_2 m^2 + K_1 r_2 m + K_2 r_1 m + r_1 r_2 =$

$= m(K_1 K_2 m + K_1 r_2 + K_2 r_1) + r_1 r_2$

$\Rightarrow (a \cdot b) \bmod m = ((a \bmod m) \cdot (b \bmod m)) \bmod m$

Функция для умножения по модулю на языке python:

```
def proi_m(a, b, m):
    return (a % m) * (b % m)
```

Чтобы ускорить возведение в степень заметим, что $N^{2n} = N^n \cdot N^n$

Реализуем функцию на python:

```
def stepen(a, n, m):
    if n == 1:
        return a
    if n == 0:
        return 1
    if n % 2 == 0:
        b = stepen(a, n//2, m)
        return proi_m(b, b, m)
    else:
        return proi_m(stepen(a, n-1, m), a, m)
```

Вариант № 7

На полоске кол-во зелёных клеток $B_k = K(R_k - 1)$,
где $K = \frac{N-R}{R}$, R_k - конечное кол-во красных
$B_k = b_k \cdot m + r_b$ $R_k = a_k \cdot m + r_a$

$B_k = K \cdot (a_k \cdot m + r_a - 1) = m(K \cdot a_k) + r_a \cdot K - K$

$\Rightarrow B_k \bmod m = ((r_a - 1)K) \bmod m$

Таким образом, получаем следующий код:

```
def proi_m(a, b, m)
    return ((a%m)*(b%m))%m
def stepen(a,n,m)
    if n==1:
        return a
    if n==0:
        return 1
    if n%2==0:
        ~~bob = stepen(a,n//2,m)~~ stepen(a, n//2, m)
        return proi_m(b, b, m)
    else:
        return ~~stepen~~ proi_m(a, stepen(a,n-1,m), m)
```

235

230

```
N, R, D = map(int, input().split()) # считываем входные данные
m = 1000000007
r_a = stepen(R,D,m) # кол-во красных по модулю m
r_b = (((r_a-1)*(N-R))//R)%m # кол-во зелёных по мод. m
print(r_a, r_b)
```