Олимпиада школьников «Гранит науки»

БЛАНК ОЛИМПИАДНОЙ РАБОТЫ

Шифр работы Ц23-57 **(не заполнять)**

Вариант № 2

Оценка выполнения олимпиадной работы
(заполняется проверяющим)

№ задания		1	2	3	4	5	6	7	8	9	10	Σ
Полученный балл		–	–	–	–	25	20					45
I проверка	Подпись проверяющего				Σ баллов прописью	сорок пять						
	Подпись проверяющего				Σ баллов прописью	сорок пять						
II проверка	Подпись проверяющего				Σ баллов прописью	сорок пять						
	Подпись проверяющего				Σ баллов прописью							

N6 (python)

```
flat = [ ]
for x in range(17):
    m = [ ]
    if(x==0 or x == 16):
        for y in range(32):
            m.append("#")
    else:
        line = input()
        m.append("#")
        for y in line:
            m.append(y)
        m.append("#")
    flat.append(m)
```

заполняем массив комнаты с дополнительной рамкой в 1 значение

20

Частный случай 5б - ?

```
maxSquare = 0
for row in range(len(flat)):
    for col in range(len(flat[0])):
        if(flat[row][col] == "~"):
            for k in range(4):
                newflat = [ ]
                for m in flat:
                    newflat.append(m)
                square = 1
                x = col
                y = row
                newflat[y][x] = square
                stop = false
                !
                !
                !
```

k = 0 – вверх
k = 1 – вправо
k = 2 – вниз
k = 3 – влево

для каждой свободной клетки проверяем площадь территории, на которой был робот.

```
while stop == False:
    if(k==0):
        y -= 1
    elif(k==1):
        x += 1
    elif(k==2):
        y += 1
    elif(k==3):
        x -= 1
    if(newflat[y][x] == "~"):
        square += 1
        newflat[y][x] = square
    elif(newflat[y][x] == "#"):
        if(k==0):
            y += 1
        elif(k==1):
            x -= 1
        elif(k==2):
            y -= 1
        elif(k==3):
            x += 1
        k += 1
        if(k==4):
            k = 0
    else:
        stop = True

maxSquare = max(maxSquare, square)
print(maxSquare)
```

Вариант № 2

№5

```
def findIsland1(map, posX, posY):
    global island1
    if(map[posY][posX+1] == "#"):
        island1.append((posX+1, posY))
        findIsland1(map, posX+1, posY)
    if(map[posY][posX-1] == "#"):
        island1.append((posX+1, posY))
```

№5

```
def findIsland1(map, posX, posY):
    global island1
    if(map[posY][posX+1] == "#"):
        if((posX+1, posY) not in island1):
            island1.append((posX+1, posY))
            findIsland1(map, posX+1, posY)
    if(map[posY][posX-1] == "#"):
        if((posX-1, posY) not in island1):
            island1.append((posX-1, posY))
            findIsland1(map, posX-1, posY)
    if(map[posY+1][posX] == "#"):
        if((posX, posY+1) not in island1):
            island1.append((posX, posY+1))
            findIsland1(map, posX, posY+1)
    if(map[posY-1][posX] == "#"):
        if((posX, posY-1) not in island1):
            island1.append((posX, posY-1))
            findIsland1(map, posX, posY-1)
```

заполняем 1 остров

```
island1 = []
island2 = []
map = []
for y in range(17):
    mapy = []
    if(y==0 or y==16):
        for col in range(32):
            mapy.append("/")
    else:
```

```
        line = input()
        mapy.append("/")
        for col in range(len(line)):
            mapy.append(line[col])
            if(line[col]=="#" and len(island1)==0):
                island1.append((col+1,y))
        mapy.append("/")
    map.append(mapy)
findIsland1(map, island1[0][0], island1[0][1])
for row in range(len(map)):
    for col in range(len(map[0])):
        if(map[row][col]=="#"):
            if((col,row) not in island1):
                island2.append((col,row))
def algorithm(row, col, map, length):
    if(map[row][col]=="~"):
        global d
        if(length < d[row][col]):
            d[row][col] = length
        else: return
        for k in range(4):
            if(k==0):
                row -= 1
            elif(k==1):
                col += 1
            elif(k==2):
                row += 1
            elif(k==3):
                col -= 1
            algorithm(row, col, map, length+1)
d = dict.fromkeys([x for x in range(30*15)], 1000)
for colI, rowI in island1:
    algorithm(rowI, colI, map, 0)
minDist = 1000
for colI, rowI in island2:
    if(map[rowI][colI-1]=="~"):
        minDist = min(minDist, d[rowI][colI-1])
    if(map[rowI][colI+1]=="~"):
        minDist = min(minDist, d[rowI][colI+1])
```

заполняем массив всей карты и находим кусок 1 острова

1 заполняем 1 остров

заполняем 2 остров

Ищем минимальную длину к каждой клетке воды

Вариант № 2

```
if(map[rowI-1][colI] == "~"):
    minDist = min(minDist, d[(rowI-1)·colI])
if(map[rowI+1][colI] == "~"):
    minDist = min(minDist, d[(rowI+1)colI])
print(minDist)
```

ищем минимальную длину от 1 острова к 2.

25 5δ-?

Вариант № _____