Разбор задач

Максимальное количество баллов — 500

Задача 1. Построение наибольшего

Чтобы трёхзначное число было как можно большим, на первое место нужно поставить цифру 9. Она нечётна и больше 6, поэтому для выполнения всех условий обе оставшиеся цифры должны быть чётными и одновременно меньшими 6. Сама цифра 6 не подходит (она не меньше 6), цифра 5 — нечётна. А вот цифра 4 чётна и при этом меньше шести, поэтому на вторую и третью позиции поставим её.

Ответ: 944.

Задача 2. Коты и собаки

В условии 4 написано, что Джульбарсу купили резиновый и деревянный мячи. Из условия 7 следует, что один из двух мячей Котангенса — пластиковый.

Из условий 5 и 6 получаем, что Мурсия, Котангенс и Сникерс – коты, а Джульбарс и Вук – собаки.

В условии 3 сказано, что одной из собак купили пластиковый и деревянный мячики, значит, это Вук.

В условии 8 сказано, что одному коту купили тряпичный и резиновый мячики, но это не могут быть Котангенс (у него один мяч пластиковый) и Мурсия (из условия 2 ей не покупали резиновый мячик). Значит, Сникерсу купили тряпичный и резиновый мячики.

Поскольку каждого мячика купили по два вида, то остались тряпичный и два меховых. Значит, Мурсии достались тряпичный и меховой, а Котангенсу — меховой и пластиковый (что мы установили ранее).

Джульбарс: резиновый и деревянный.

Вук: пластиковый и деревянный.

Сникерс: резиновый и тряпичный.

Мурсия: тряпичный и меховой.

Котангенс: пластиковый и меховой.

Задача 3. Баобаб

Чтобы строка после разрезания и перестановки оказалась наибольшей в лексикографическом порядке, необходимо на первое место поставить букву «О». Значит, буква «О» должна быть началом одного куска, то есть разрез необходимо сделать перед буквой «О»: «БА-ОБАБ».

Помимо буквы «О», остались только буквы «А» и «Б». Нам нужно после буквы «О» поставить как можно больше букв «Б». В слове «БАОБАБ» и так после буквы «О» идёт одна буква «Б», но за ней идёт буква «А», поэтому сделаем второй разрез между «Б» и «А»: «БА-ОБ-АБ».

Теперь переставим куски так, чтобы после «ОБ» оказалась буква «Б»: «ОБ-БА-АБ».

Ответ: ОББААБ.

Задача 4. Диалог нейросетей

Заметим, что в условии есть запрет на следование «poppush» — оно содержит «popp» и запрет на следование «inpush» — содержащее «npu». Отсюда следует, что окончание правильного диалога всегда будет иметь вид «offtoppush». Ещё запрещено повторение «poppop».

Остальные запреты касаются следования трёх слов подряд: запрещены «pushinpush», «pushinpop», «popinpop», «popinpush», «offtopinpush», «offtopinpop», «inpopofftop», «pushpopin».

Рассмотрим начало «pushpop». После этого нельзя поставить «in» из-за запрета «hpopi», остаётся добавить «offtop» и получить «pushpopofftop». Далее нужно добавить «in» и выйти на окончание «offtoppush», что даёт правильный диалог «pushpopofftopinofftoppush». Это один из самых коротких диалогов, он содержит минимальное число слов — 6 — и имеет длину 25 символов. Но из-за повторения длинного слова «offtop» — этот ответ не оптимален. Заметим, что «offtop» — единственное повторённое слово этом варианте диалога.

Начало «pushofftop» заведомо не может быть лучше, так как в дальнейшем мы снова должны будем использовать ещё одно вхождение «offtop» в окончании, а двойное вхождение «offtop» в ответ мы уже обсудили.

Теперь рассмотрим оптимальный вариант начала «pushin». Смысла добавлять далее «offtop» нет по причине того, что далее его придется добавлять ещё раз для выхода, поэтому желательно здесь поставить «pop». Напрямую этого делать нельзя из-за запрета «hinp». Но ничто не запрещает ещё раз повторить короткое слово «in» и избавиться от этого запрета: «pushininpop». Но теперь нельзя сразу добавить завершение «offtoppush» из-за запрета «npopo». Поэтому еще раз добавим слово «in» и только потом — «offtoppush». Получим самый короткий диалог «pushininpopinofftoppush». Он состоит из семи слов и имеет длину 23 символа.

Вот ещё варианты правильных диалогов из 25 символов: «pushofftoppopinofftoppush» и «pushinofftoppopofftoppush» — они так же состоят из шести слов. Остальные правильные диалоги имеют длину не менее 27 символов.

Задача 5. Робот-пылесос

Решение основывается на переборе разных вариантов маршрутов, где мы стремимся набрать как можно больше пыли.

Маршруты легче искать в такой таблице, если закрасить сектора в разные цвета в зависимости от количества пыли. Это легко можно сделать в электронных таблицах: нужно переписать данные в таблицу, выделить её и применить «Условное форматирование» –> «Цветовые шкалы» –> «Цветовая шкала зеленый-жёлтый-красный». Теперь маленькие числа будут красными, а большие — зелёными. Такая таблица называется тепловой картой.

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

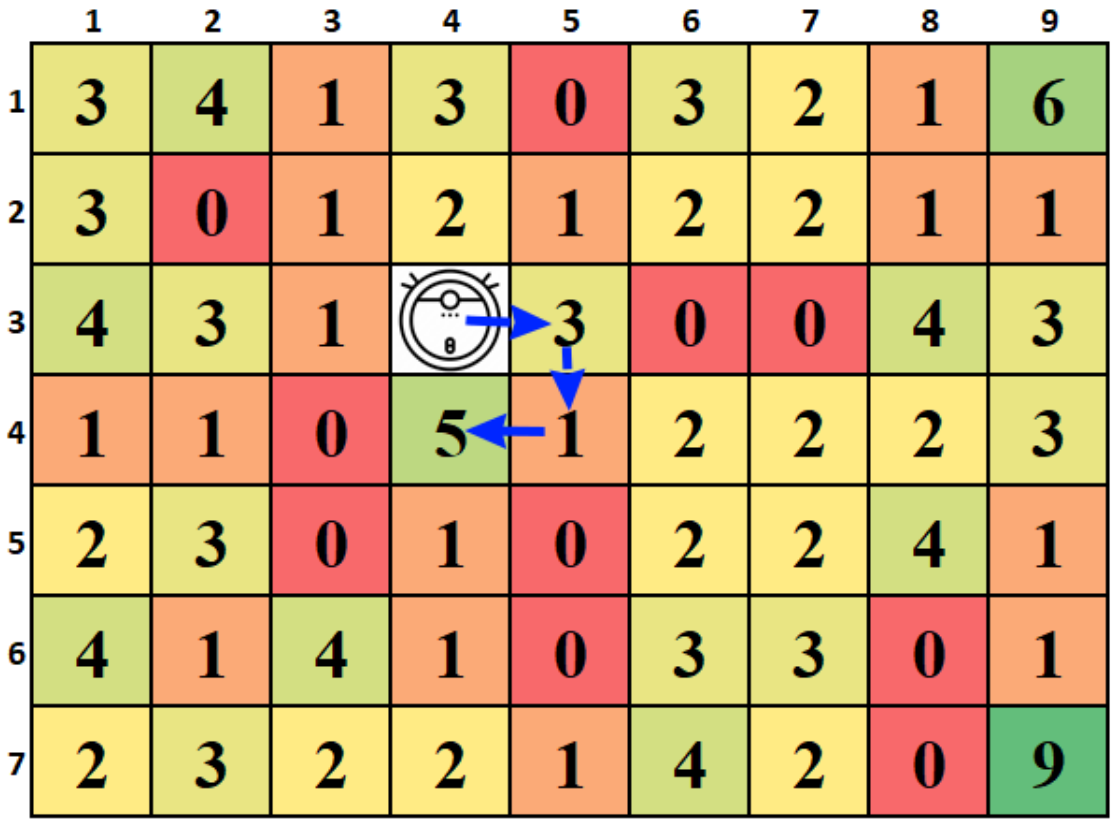
Тепловая карта помещения

Найдём решение для $X = 3$:

В радиусе трёх секторов от робота-пылесоса самые большие числа — это 5, 4 и несколько троек. Попытаемся их объединить и найти маршрут, который позволит роботу собрать наибольшее количество пыли.

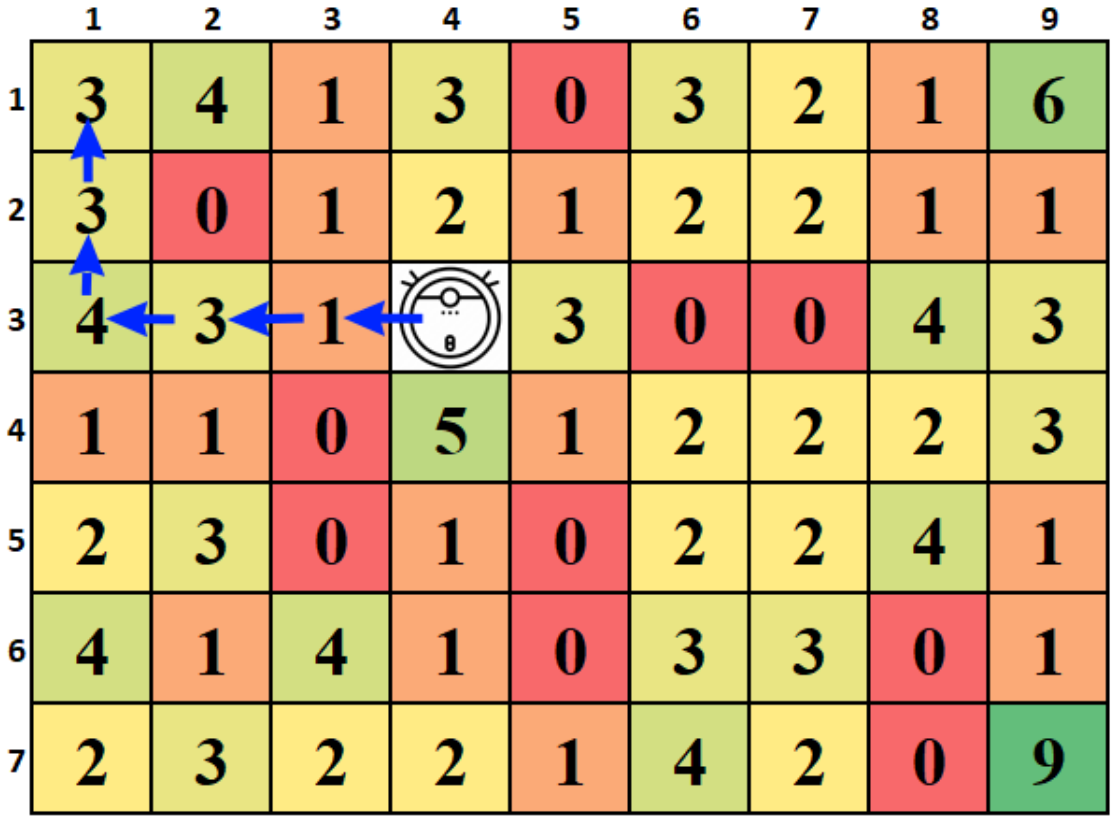
1. LLL 1+3+4=8
2. DRU 5+1+3=9
3. RDL 3+1+5=9

Остальные маршруты позволят собрать намного меньше пыли То есть наилучший маршрут позволяет собрать 9 единиц пыли и будет иметь вид «DRU» или «RDL». Пример маршрута «RDL»:

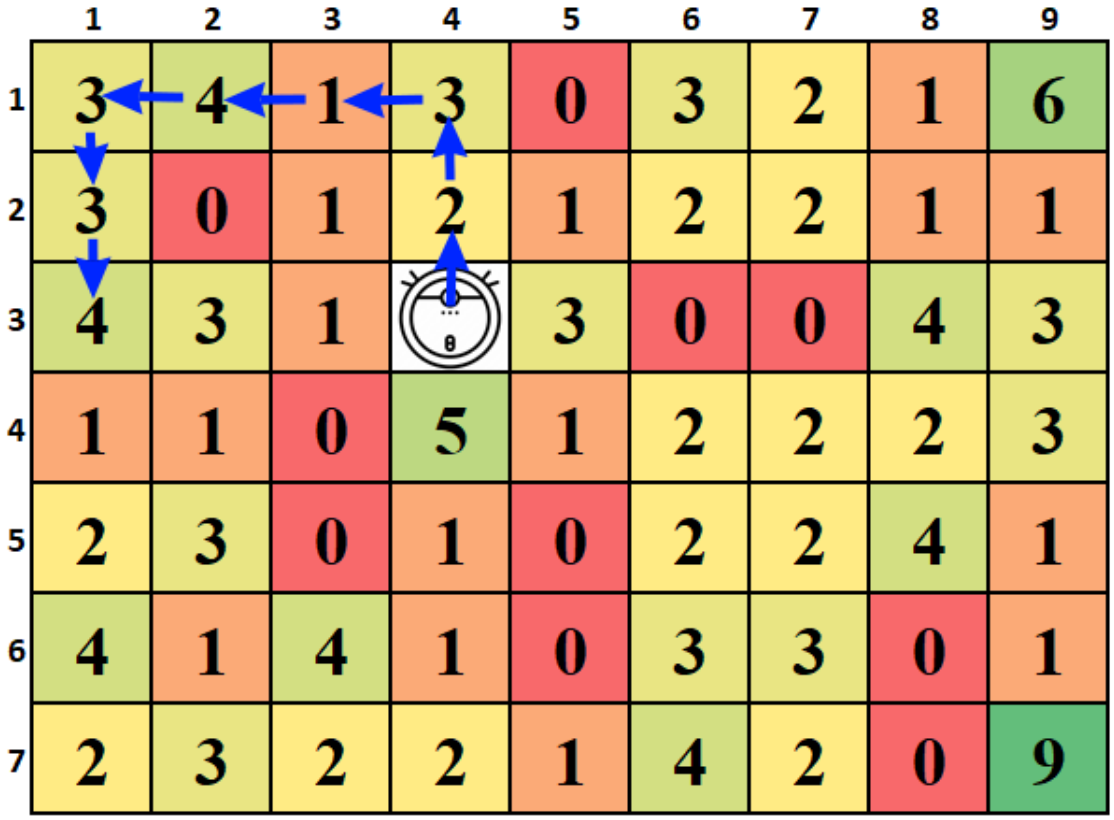


Для нахождения ответа при $X = 5, 7, 9$ используем аналогичную логику. Определяем области секторов, где мы можем набрать больше всего пыли, и строим маршрут туда через сектора с наибольшими числами.

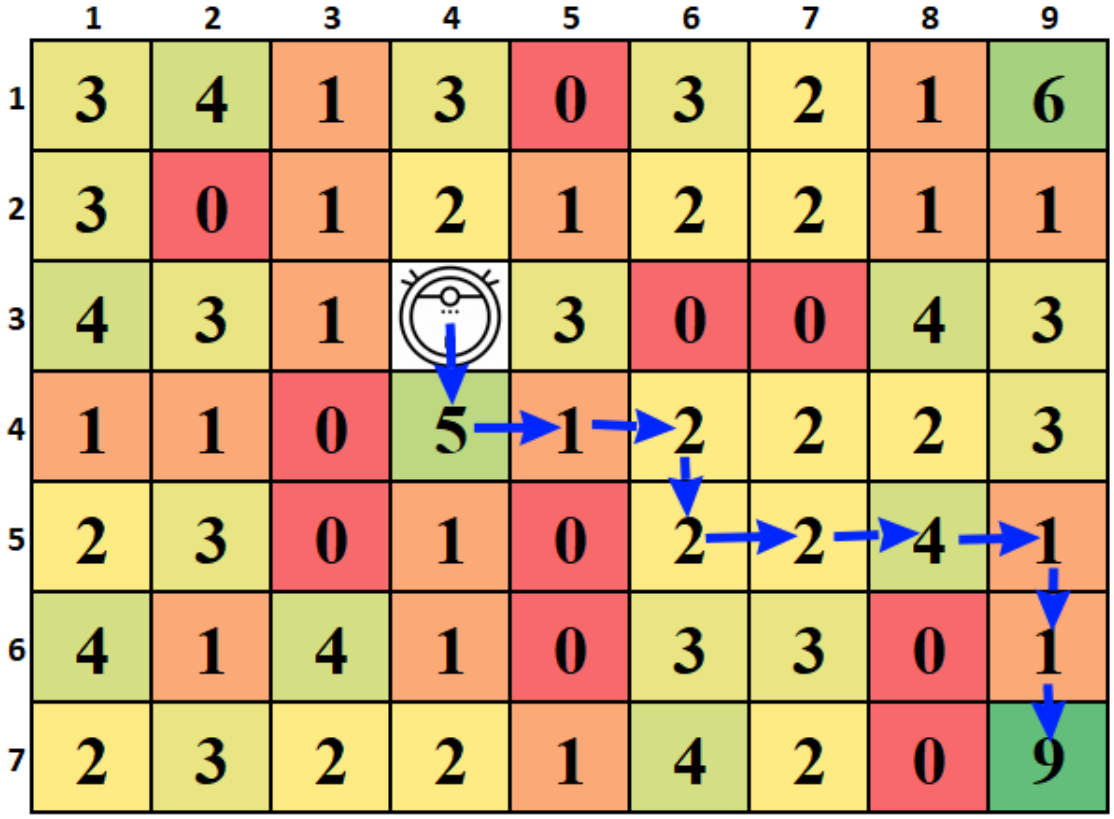
Для $X = 5$ маршрут «LLLUU» позволяет собрать 14 единиц пыли.



Для $X = 7$ маршрут «UULLLDD» позволяет собрать 20 единиц пыли.



Для $X = 9$ маршрут «DRRDRRRDD» позволяет собрать 27 единиц пыли.



Ещё один способ решения — написать программу, которая переберёт все маршруты нужной длины и найдёт маршрут, позволяющий собрать больше всего пыли. Полный перебор можно реализовать через рекурсивный алгоритм поиска в глубину. Такое решение в данной задаче будет работать довольно быстро, потому что маршрут максимальной длины не очень длинный.

```
x, y = 2, 3  # начальная координата робота
n, m = 7, 9  # размеры помещения

# карта помещения. Препятствия заменены -99,
# чтобы роботу было явно невыгодно туда ходить
field = [[3, 4, 1, 3, -99, 3, 2, 1, 6],
         [3, -99, 1, 2, 1, 2, 2, 1, 1],
         [4, 3, 1, 0, 3, -99, -99, 4, 3],
         [1, 1, -99, 5, 1, 2, 2, 2, 3],
         [2, 3, -99, 1, -99, 2, 2, 4, 1],
         [4, 1, 4, 1, -99, 3, 3, -99, 1],
         [2, 3, 2, 2, 1, 4, 2, -99, 9]]

# двумерный список, где мы будем отмечать сектора, по которым
# проехал робот-пылесос. 0 - не проехал, 1 - проехал
used = [[0] * m for _ in range(n)]

# из любого сектора можно проехать в одну из четырёх сторон
(список направлений).
# 0. y-1, x+0 - движение вверх (U)
# 1. y+1, x+0 - движение вниз (D)
# 2. y, x-1 - движение влево (L)
# 3. y, x+1 - движение вправо (R)
d = [[-1, 0], [1, 0], [0, -1], [0, 1]]

# функция, которая проверяет, находится ли робот-пылесос внутри помещения
def coord_ok(i, j):
    return 0 <= i < n and 0 <= j < m

# Функция рекурсивного перебора.
# i, j - текущая координата робота
# curEnergy - количество потраченного заряда
# curPoints - объём собранной пыли
# bp - путь, пройденный до текущей координаты
def dfs(i, j, curEnergy, curPoints, bp):
    # Если заряд закончился, заканчиваем движение
    if curEnergy == energy:
        return curPoints, bp
    maxPoints = 0
    bestPath = ""
    # отправим робота на 4 разные стороны и посмотрим,
    # откуда он принесет больше всего пыли.
    for k in range(4):
        i1 = i + d[k][0]
        j1 = j + d[k][1]
        if coord_ok(i1, j1):
            x = field[i1][j1]
            field[i1][j1] = 0
            points, path = dfs(i1, j1, curEnergy+1, curPoints+x, bp+str(k))
```

```
                if points > maxPoints:
                    maxPoints = points
                    bestPath = path
                field[i1][j1] = x
    return maxPoints, bestPath

# переберём длины маршрутов и для каждого случая найдём маршрут,
# который позволит роботу собрать наибольшее количество пыли.
for i in 3, 5, 7, 9:
    energy = i
    a, b = dfs(x, y, 0, 0, "")
    # заменим номера направлений на буквы.
    print(b.replace("0","U").replace("1","D").replace("2","L").replace("3","R"))
```