Харламов Алексей, группа 153

№1
В качестве структуры данных необходимо выбрать дерево отрезков с групповыми операциями методом проталкивания. В каждой из вершин дерева будем хранить флаг необходимости инвертирования этого отрезка, текущей суммы, флаг присвоения.

№2
Рассмотрим задачу со стороны комбинаторики. Заметим, что чисел длиннее 10 знаков со всеми различными цифрами быть не может, так как различных цифр всего 10. Тогда, на 1 разряд мы можем выбрать любой вариант из 9 (т.к на первом месте 0 быть не может, а все остальные цифры возможны), на 2 разряд можно выбрать 9 цифр(т.к всего 10 цифр, 1 какую-то выбрали на предыдущем шаге), на 3 разряд можно выбрать 8 цифр(10 - 2 возможности) и так далее, таким образом мы за $O(1)$ можем отвечать на поставленный вопрос для любого n.

```
def solver(n):
    if (x == 0) or (x > 10):
        return 0
    if (x == 1):
        return 9
    if (x == 2):
        return 9 * 9
    if (x == 3):
        return 9 * 9 * 8
    if (x == 4):
        return 9 * 9 * 8 * 7
    if (x == 5):
        return 9 * 9 * 8 * 7 * 6
    if (x == 6):
        return 9 * 9 * 8 * 7 * 6 * 5
    if (x == 7):
        return 9 * 9 * 8 * 7 * 6 * 5 * 4
    if (x == 8):
        return 9 * 9 * 8 * 7 * 6 * 5 * 4 * 3
    if (x == 9):
        return 9 * 9 * 8 * 7 * 6 * 5 * 4 * 3 * 2
    if (x == 10):
        return 9 * 9 * 8 * 7 * 6 * 5 * 4 * 3 * 2 * 1
```

№3

Для нахождения самого надежного пути, перейдем от вероятностей быть ограбленным на дороге, к вероятности проехать по этой дороге спокойно (то есть не быть ограбленным) путем замены значений на дороге на обратные вероятности, то есть $1 - p_e$, а так же возьмем от этого логарифм со знаком минус, т.е $-\log(1 - p_e)$. После

этого найдем кратчайшее расстояние с помощью алгоритма Дейкстры. Рассмотрим, что здесь произошло:
Вероятность не быть ограбленным на первом участке равна $1 - a$, на втором $- 1 - b$.
Вероятность не быть ограбленным на обоих участках одновременно равна $c = (1 - a)(1 - b)$. Отсюда следует, что вероятность быть ограбленным при прохождении пути равна $1 - c = 1 - (1 - a)(1 - b)$. Учитывая то, что $\log(xy) = \log(x) + \log(y)$, минимизируя расстояние с помощью алгоритма Дейкстры как раз будет найдена требуемая вероятность самого надежного пути.

№6а

Переберем каждую точку с каждой и составим потенциальные кандидаты на искомую прямую (будем соединять точки друг с другом и находить серединный перпендикуляр для полученного отрезка, таким образом получим $O(n^2)$ прямых). Заметим, что искомых разделяющих прямых у нас в полученном списке будет не менее n (т.к в предположении о том, что множество делимо на симметричные подмножества, у каждой точки будет хоть одна симметричная ей (относительно прямой), тогда имеем для каждой точки хотя бы 1 нужную прямую, следовательно на всех парах будем иметь не менее n таких прямых), тогда одинаково параметризовав все прямые (например находя для каждой прямой пересечение в 2 точках для bounting box, которым мы заранее обрамили все множество), сложим полученные пары точек в сет и посчитаем, сколько раз встречалась каждая из них, тогда пара, которая была встречена больше либо равно чем n раз, будет искомой парой, которая задает нашу разделяющую прямую.

№7

```
def solver(begin):
    min_num = inf;
    Node to_return = begin;
    Node temp = begin;
    while (temp.next != NULL):
        temp = temp.next;
        temp_num = random();
        if (min_num > temp_num):
            min_num = temp_num;
            to_return = temp;
    return to_return
```