**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра “Математическая кибернетика и информационные технологии”

**Отчет по курсовой работе**

по дисциплине «Структуры и алгоритмы обработки данных»

Вариант №9

Выполнил: студент группы БВТ2004

Гончаренко Анастасия Анатольевна

Проверил:

Мкртчян Грач Маратович

Москва

2022

# Задачи

## Задача №1

Перестановкой длины 𝑛 называется такой массив 𝑝=[𝑝1,𝑝2,…,𝑝𝑛], который содержит каждое число от 1 до 𝑛 (включительно) и притом ровно по одному разу. Например, 𝑝=[3,1,4,2,5] — перестановка длины 5.

Для заданного числа 𝑛 (𝑛≥2) найдите такую перестановку 𝑝, в которой разница (то есть модуль разности) любых двух соседних элементов находится в диапазоне от 2 до 4, включительно. Формально, для перестановки 𝑝 должно выполняться 2≤|𝑝𝑖−𝑝𝑖+1|≤4 для всех 𝑖 (1≤𝑖<𝑛).

Выведите любую такую перестановку для заданного значения 𝑛 или определите, что ее не существует.

Входные данные  
В первой строке записано целое число 𝑡 (1≤𝑡≤100) — количество наборов входных данных в тесте. Далее следуют описания 𝑡 наборов входных данных.

Каждый набор описывается единственной строкой, в которой содержится целое число 𝑛 (2≤𝑛≤1000).

Выходные данные  
Выведите 𝑡 строк. Очередная строка должна содержать -1, если для соответствующего набора входных данных искомой перестановки не существует. В противном случае выведите искомую перестановку. Если таких перестановок существует несколько, то выведите любую из них.

Пример:  
Входные данные  
6  
10  
2  
4  
6  
7  
13  
Выходные данные  
9 6 10 8 4 7 3 1 5 2  
-1  
3 1 4 2  
5 3 6 2 4 1\ 5 1 3 6 2 4 7\ 13 9 7 11 8 4 1 3 5 2 6 10 12

## Задача №2

Вам задана массив 𝑎 длины 𝑛, состоящий из нулей. Вы выполняете 𝑛 действий с этим массивом: в течение 𝑖-го действия происходит следующая последовательность операций:

Выбирается максимальный по длине подмассив (последовательный подотрезок), состоящий только из нулей, среди всех таких отрезков выбирается самый левый; Пусть этот отрезок равен [𝑙;𝑟]. Если 𝑟−𝑙+1 нечетно (не делится на 2), то присваивается 𝑎[(𝑙+𝑟)/2]:=𝑖 (где 𝑖 — номер текущего действия), иначе (если 𝑟−𝑙+1 четно) присваивается 𝑎[(𝑙+𝑟-1)/2]:=𝑖.  
Рассмотрим массив 𝑎 длины 5 (изачально 𝑎=[0,0,0,0,0]). Тогда он меняется следующим образом:

Сначала мы выбираем отрезок [1;5] и присваиваем 𝑎[3]:=1, таким образом 𝑎 становится равен [0,0,1,0,0];  
затем мы выбираем отрезок [1;2] и присваиваем 𝑎[1]:=2, таким образом 𝑎 становится равен [2,0,1,0,0];  
затем мы выбираем отрезок [4;5] и присваиваем 𝑎[4]:=3, таким образом 𝑎 становится равен [2,0,1,3,0];  
затем мы выбираем отрезок [2;2] и присваиваем 𝑎[2]:=4, таким образом 𝑎 становится равен [2,4,1,3,0];  
и наконец мы выбираем отрезок [5;5] и присваиваем 𝑎[5]:=5, таким образом 𝑎 становится равен [2,4,1,3,5].  
Ваша задача — найти массив 𝑎 длины 𝑛 после выполнения всех 𝑛 действий. Заметьте, что ответ существует и единственен.

Вам необходимо ответить на 𝑡 независимых наборов тестовых данных.

Входные данные  
Первая строка входных данных содержит одно целое число 𝑡 (1≤𝑡≤104) — количество наборов тестовых данных. Затем следуют 𝑡 наборов тестовых данных.

Единственная строка набора тестовых данных содержит одно целое число 𝑛 (1≤𝑛≤2⋅105) — длину 𝑎.

Гарантируется, что сумма 𝑛 по всем наборам тестовых данных не превосходит 2⋅105 (∑𝑛≤2⋅105).

Выходные данные  
Для каждого набора тестовых данных выведите ответ — массив 𝑎 длины 𝑛 после выполнения 𝑛 действий, описанных в условии задачи. Заметьте, что ответ существует и единственен.

Пример:  
Входные данные  
6  
1  
2  
3  
4  
5  
6  
Выходные данные  
1  
1 2  
2 1 3  
3 1 2 4  
2 4 1 3 5  
3 4 1 5 2 6 \

## Задача №3

Поликарп очень любит гамбургеры, особенно приготовленные собственноручно. Поликарп считает, что существует только три достойных ингредиента для приготовления гамбургера: хлеб, колбаса и сыр. Рецепт своего знаменитого «гамбургера от Поликарпа» он записывает в виде строки из букв 'B' (хлеб), 'S' (колбаса) и 'C' (сыр). В рецепте ингредиенты перечисляются снизу вверх, например, рецепт «ВSCBS» обозначает гамбургер в котором снизу вверх идут: хлеб, колбаса, сыр, хлеб и снова колбаса.

На кухне у Поликарпа в наличии nb кусочков хлеба, ns кусочков колбасы и nc кусочков сыра. Кроме того, в магазине неподалеку есть в продаже все три ингредиента по цене: pb рублей за кусок хлеба, ps — за кусок колбасы и pc — за кусочек сыра.

У Поликарпа есть r рублей, которые он готов потратить в магазине. Какое наибольшее количество гамбургеров он сможет приготовить? Считается, что ломать или резать любой из кусочков (хлеба, колбасы, сыра) нельзя, а также что в магазине есть бесконечно много кусочков каждого из ингредиентов.

Входные данные  
В первой строке входных данных содержится непустая строка, описывающая рецепт «гамбургера от Поликарпа». Длина строки не превосходит 100, строка содержит только буквы 'B' (прописная латинская B), 'S' (прописная латинская S) и 'C' (прописная латинская C).

Вторая строка содержит три целых числа nb, ns, nc (1 ≤ nb, ns, nc ≤ 100) — количество кусочков хлеба, колбасы и сыра на кухне Поликарпа. Третья строка содержит три целых числа pb, ps, pc (1 ≤ pb, ps, pc ≤ 100) — цена одного кусочка хлеба, колбасы и сыра в магазине. Наконец, четвертая строка содержит целое число r (1 ≤ r ≤ 1012) — количество рублей у Поликарпа.

Пожалуйста, не используйте спецификатор %lld для чтения или записи 64-битных чисел на С++. Рекомендуется использовать потоки cin, cout или спецификатор %I64d.

Выходные данные  
Выведите наибольшее количество гамбургеров, которое Поликарп может приготовить. Если он не может приготовить ни одного гамбургера, выведите 0.

Примеры  
входные данные  
BBBSSC  
6 4 1  
1 2 3  
4  
выходные данные  
2  
входные данные  
BBC  
1 10 1  
1 10 1  
21  
выходные данные  
7

## Задача №4

Ayush и Ashish играют в игру на некорневом дереве, состоящем из 𝑛 вершин, пронумерованных от 1 до 𝑛. Игроки делают следующий ход по очереди:

Выберите любой лист в дереве и удалите его вместе со всеми ребрами, для которых этот лист является одним из концов. Лист — это вершина со степенью, не превосходящей 1. Дерево — это связный неориентированный граф без циклов.

Дана специальная вершина с номером 𝑥. Игрок, который удаляет эту вершину, выигрывает игру.

Ayush ходит первым. Определите победителя игры, если каждый игрок играет оптимально.

Входные данные  
В первой строке входных данных содержится одно целое число 𝑡 (1≤𝑡≤10) — количество наборов входных данных. Далее следуют описания наборов входных данных.

Первая строка каждого набора входных данных содержит два целых числа 𝑛 и 𝑥 (1≤𝑛≤1000,1≤𝑥≤𝑛) — количество вершин в дереве и специальную вершину, соответственно.

Каждая из следующих 𝑛−1 строк содержит два целых числа 𝑢, 𝑣 (1≤𝑢,𝑣≤𝑛, 𝑢≠𝑣), что означает, что между вершинами 𝑢 и 𝑣 есть ребро.

Выходные данные Для каждого набора входных данных, если побеждает Ayush, выведите "Ayush", иначе выведите "Ashish" (без кавычек).

Примеры  
входные данные  
1  
3 1  
2 1  
3 1  
выходные данные  
Ashish  
входные данные  
1  
3 2  
1 2  
1 3  
выходные данные  
Ayush  
Примечание  
В первом наборе входных данных Ayush может удалить только вершину 2 или 3, после чего вершина 1 становится листом, и Ashish может удалить ее в свою очередь. Во втором наборе входных данных Ayush может удалить вершину 2 на самом первом шаге.

## Задача №5

Вы играете в очередную компьютерную игру, и теперь вам предстоит убить 𝑛 монстров. Эти монстры стоят в круге, пронумерованном по часовой стрелке от 1 до 𝑛. Изначально 𝑖-й монстр имеет 𝑎𝑖 единиц здоровья.

Вы можете стрелять в монстров, чтобы убить их. Каждый выстрел требует ровно одной пули и уменьшает здоровье монстра на 1 (наносит ему 1 единицу урона). Кроме того, когда здоровье некоторого монстра 𝑖 становится 0 или меньше 0, он умирает и взрывается, нанося 𝑏𝑖 урон следующему монстру (монстру под номером 𝑖+1, если 𝑖<𝑛, или монстру под номером 1, если 𝑖=𝑛). Если следующий монстр уже мертв, то ничего не происходит. Если взрыв убивает следующего монстра, он тоже взрывается, повреждая монстра после него и, возможно, вызывая еще один взрыв, и так далее.

Вы должны посчитать минимальное количество пуль, которое нужно выстрелить, чтобы убить всех 𝑛 монстров в кругу.

Входные данные  
Первая строка содержит одно целое число 𝑇 (1≤𝑇≤150000) — количество наборов входных данных.

Затем следуют наборы входных данных, каждый из них начинается со строки, содержащей одно целое число 𝑛 (2≤𝑛≤300000) — количество монстров. Затем следуют 𝑛 строк, каждая из которых содержит два целых числа 𝑎𝑖 и 𝑏𝑖 (1≤𝑎𝑖,𝑏𝑖≤1012) — параметры 𝑖-го монстра в круге.

Гарантируется, что общее количество монстров во всех тестовых случаях не превышает 300000.

Выходные данные  
Для каждого набора входных данных выведите одно целое число — минимальное количество пуль, которые нужно выстрелить, чтобы убить всех монстров.

Пример  
входные данные  
1  
3  
7 15  
2 14  
5 3  
выходные данные  
6

## Задача №6

Отель Гильберта это очень необычный отель, потому что количество комнат в нем бесконечно! Для каждого целого числа существует ровно одна комната с таким номером, включая ноль и отрицательные числа. Не менее странно то, что сейчас отель полностью заполнен, что означает, что в каждой комнате находится ровно один гость. Менеджер отеля, сам Давид Гильберт, решил переместить гостей, потому что у него есть предположение, что за счет этого образуются свободные места.

Для любого целого числа 𝑘 и положительного целого числа 𝑛 обозначим за 𝑘mod𝑛 остаток при делении числа 𝑘 на число 𝑛. Более формально, 𝑟=𝑘mod𝑛 это наименьшее неотрицательное целое число такое, что 𝑘−𝑟 делится на 𝑛. Всегда выполнено, что 0≤𝑘mod𝑛≤𝑛−1. Например, 100mod12=4 и (−1337)mod3=1.

Процесс перемещения гостей выглядит следующим образом: есть массив, состоящий из 𝑛 целых чисел 𝑎0,𝑎1,…,𝑎𝑛−1. Тогда для всех целых чисел 𝑘 гость из комнаты с номером 𝑘 перемещается в комнату с номером 𝑘+𝑎𝑘mod𝑛.

Определите, верно ли, что после этого процесса перемещения в каждой комнате по-прежнему находится ровно один гость. Это означает, что нет пустых комнат и комнат, в которых больше одного гостя.

Входные данные  
Каждый тест состоит из нескольких тестовых случаев. Первая строка содержит единственное целое число 𝑡 (1≤𝑡≤104) — количество тестовых случаев. Следующие 2𝑡 строк содержат описания тестовых случаев.

В первой строке описания каждого тестового случая находится единственное целое число 𝑛 (1≤𝑛≤2⋅105) — длина массива.

Во второй строке описания каждого тестового случая находятся 𝑛 целых чисел 𝑎0,𝑎1,…,𝑎𝑛−1 (−109≤𝑎𝑖≤109).

Гарантируется, что сумма 𝑛 по всем тестовым случаям не превосходит 2⋅105.

Выходные данные  
Для каждого тестового случая выведите единственную строку, содержащую «YES», если в каждой комнате после перемещения находится ровно один гость, и «NO» иначе. Вы можете выводить каждый символ в любом регистре.

Пример  
входные данные  
6  
1  
14  
2  
1 -1  
4  
5 5 5 1  
3  
3 2 1  
2  
0 1  
5  
-239 -2 -100 -3 -11  
выходные данные  
YES  
YES  
YES  
NO  
NO  
YES  
Примечание  
В первом тестовом случае номер комнаты каждого гостя увеличился на 14, поэтому по-прежнему в каждой комнате находится ровно один гость.

1. Во втором тестовом случае гости в комнатах с четными номерами перемещаются в комнату с номером, на 1 большим исходного; гости в комнатах с нечетными номерами перемещаются в комнату с номером, на 1 меньшим исходного. Можно показать, что по-прежнему в каждой комнате находится ровно один гость.
2. В третьем тестовом случае каждый четвертый гость перемещается в комнату с номером, на 1 большим исходного, а остальные гости перемещаются в комнату с номером, на 5 большим. Можно показать, что по-прежнему в каждой комнате находится ровно один гость.
3. В четвертом тестовом случае гости, исходно находящиеся в комнатах 0 и 1, перемещаются в комнату с номером 3.
4. В пятом тестовом случае гости, находящиеся в комнатах 1 и 2, перемещаются в комнату с номером 2.

## Задача №7

Несколько дней назад я узнал, что существует такая штука как наименьшее общее кратное (НОК). Теперь я часто играю с этим понятием — хочу сделать большое число с помощью НОК.

Но я не хочу использовать слишком много чисел, поэтому я выберу три целых положительных числа (необязательно различных), каждое из которых не превышает n. Помогите мне найти максимально возможное наименьшее общее кратное этих трех целых чисел.

Входные данные  
В первой строке записано целое число n (1 ≤ n ≤ 106) — переменная n из условия.

Выходные данные  
Выведите единственное целое число — максимально возможное наименьшее общее кратное трех необязательно различных целых чисел, которые не превышают n.

Примеры  
входные данные  
9  
выходные данные  
504  
входные данные  
7  
выходные данные  
210  
Примечание  
Наименьшее общее кратное нескольких положительных целых чисел — это наименьшее положительное целое число, кратное им всем.

Результат может получиться достаточно большим. Возможно, 32-битного целого числа не будет достаточно для его хранения. Поэтому рекомендуется использовать 64-битные целые числа.

В последнем примере мы можем выбрать числа 7, 6, 5, их НОК равен 7·6·5 = 210. Это — максимальный НОК, который мы можем получить.

## Задача №8

Для мультимножества натуральных чисел 𝑠={𝑠1,𝑠2,…,𝑠𝑘}, определим наименьшее общее кратное («LCM» по-английски) и наибольший общий делитель («GCD» по-английски) 𝑠 следующим образом:

gcd(𝑠) это максимальное натуральное число 𝑥, такое что все числа из 𝑠 делятся на 𝑥.  
lcm(𝑠) это минимальное натуральное число 𝑥, которое делится на все числа из 𝑠.  
Например, gcd({8,12})=4,gcd({12,18,6})=6 и lcm({4,6})=12. Обратите внимание, что для любого натурального числа 𝑥, gcd({𝑥})=lcm({𝑥})=𝑥.

У Орака есть последовательность 𝑎 длины 𝑛. Он придумал мультимножество 𝑡={lcm({𝑎𝑖,𝑎𝑗}) | 𝑖<𝑗} и попросил вас найти gcd(𝑡) для него. Иначе говоря, вам нужно найти НОД НОКов всех пар элементов в данной последовательности.

Входные данные  
В первой строке записано одно целое число 𝑛 (2≤𝑛≤100000).

Во второй строке записаны 𝑛 целых чисел, 𝑎1,𝑎2,…,𝑎𝑛 (1≤𝑎𝑖≤200000).

Выходные данные  
Выведите одно целое число: gcd({lcm({𝑎𝑖,𝑎𝑗}) | 𝑖<𝑗}).

Примеры  
входные данные  
2  
1 1  
выходные данные  
1  
входные данные  
4  
10 24 40 80  
выходные данные  
40  
входные данные  
10  
540 648 810 648 720 540 594 864 972 648  
выходные данные  
54  
Примечание  
В первом примере 𝑡={lcm({1,1})}={1}, и gcd(𝑡)=1.

Во втором примере 𝑡={120,40,80,120,240,80}. Нетрудно видеть, что gcd(𝑡)=40.

## Задача №9

Василий очень любит разные загадки. Сегодня он нашёл загадку, которую не смог решить сам, поэтому он просит вас помочь ему.

У Василия есть n строк, состоящих из строчных букв английского алфавита. Он хочет, чтобы строки располагались в лексикографическом порядке (как в словаре), но при этом не хочет менять их местами. Единственное, что Василий может делать, это разворачивать строки (первая буква становится последней, вторая предпоследней и так далее).

Чтобы развернуть i-ю строку Василию, надо потратить ci единиц энергии. Василия интересует минимальное количество энергии, которое необходимо потратить, чтобы строки шли в лексикографическом порядке.

Строка A лексикографически меньше строки B, если она короче B (|A| < |B|) и является её префиксом, либо ни одна из них не является префиксом другой, и в первой позиции, где они различаются, в строке A стоит символ с меньшим номером.

В данной задаче две одинаковые строки на соседних позициях не нарушают порядок лексикографической сортировки.

Входные данные  
В первой строке входных данных записано единственное целое число n (2 ≤ n ≤ 100 000) — количество строк.

Во второй строке записаны n целых чисел ci (0 ≤ ci ≤ 109), i-е из которых равняется количеству энергии, необходимому Василию для разворота i-й строки.

Далее следуют n строк, состоящих из строчных букв английского алфавита. Суммарная длина всех строк не превышает 100 000.

Выходные данные  
Если, разворачивая какие-либо из данных строк, невозможно добиться, чтобы они следовали в лексикографическом порядке, то выведите  - 1. В противном случае выведите минимальное количество энергии, которое придётся потратить Василию.

Примеры  
входные данные  
2  
1 2  
ba  
ac  
выходные данные  
1  
входные данные  
3  
1 3 1  
aa  
ba  
ac  
выходные данные  
1

Примечание  
Во втором примере можно развернуть строку 2 или строку 3. На разворот строки 3 тратится меньше энергии, поэтому правильным ответом будет развернуть её. В третьем примере обе строки не изменяются после разворота и расположены в неправильном порядке, поэтому ответом является  - 1. В четвёртом примере обе строки состоят из букв «a», но в отсортированном порядке строка «aa» должна располагаться раньше строки «aaa», поэтому ответ  - 1.

## Задача №10

У Феникса есть строка 𝑠, состоящая из строчных букв латинского алфавита. Он хочет распределить все буквы своей строки по 𝑘 непустым строкам 𝑎1,𝑎2,…,𝑎𝑘 так, что каждая буква из 𝑠 попадет ровно в одну из строк 𝑎𝑖. Строки 𝑎𝑖 не обязаны быть подстроками 𝑠. Феникс может распределить буквы 𝑠 и переупорядочить их внутри каждой строки 𝑎𝑖 так как захочет.

Например, если 𝑠= baba и 𝑘=2, Феникс может распределить буквы своей строки множеством способов, в том числе:

ba и ba a и abb ab и ab bb и aa Однако получить такие варианты он не может:

baa и ba b и ba baba и пустая строка (𝑎𝑖 должны быть непустыми) Феникс хочет разделить свою строку 𝑠 на 𝑘 строк 𝑎1,𝑎2,…,𝑎𝑘 так, чтобы минимизировать лексикографически максимальную строку среди них, т. е. минимизировать 𝑚𝑎𝑥(𝑎1,𝑎2,…,𝑎𝑘). Помогите ему найти оптимальное распределение и выведите минимально возможное значение 𝑚𝑎𝑥(𝑎1,𝑎2,…,𝑎𝑘).

Строка 𝑥 лексикографически меньше, чем строка 𝑦, если либо 𝑥 является префиксом 𝑦 (и 𝑥≠𝑦), либо существует такой индекс 𝑖 (1≤𝑖≤𝑚𝑖𝑛(|𝑥|,|𝑦|)), что 𝑥𝑖 < 𝑦𝑖 и для всех 𝑗 (1≤𝑗<𝑖) 𝑥𝑗=𝑦𝑗. Здесь |𝑥| обозначает длину строки 𝑥.

Входные данные  
Входные данные состоят из нескольких наборов. В первой строке задано целое число 𝑡 (1≤𝑡≤1000) — количество наборов входных данных. Каждый набор состоит из двух строк.

В первой строке каждого набора задано два целых числа 𝑛 и 𝑘 (1≤𝑘≤𝑛≤105) — длина строки 𝑠 и количество не пустых строк, в которые Феникс хочет распределить буквы 𝑠, соответственно.

Во второй строке каждого набора задана строка 𝑠 длины 𝑛, состоящая из строчных латинских букв.

Гарантируется, что сумма 𝑛 по всем наборам входных данных ≤105.

Выходные данные  
Выведите 𝑡 ответов — по одному на набор входных данных; 𝑖-й ответ — минимально возможный 𝑚𝑎𝑥(𝑎1,𝑎2,…,𝑎𝑘) в 𝑖-м наборе.

Пример  
входные данные  
6  
4 2  
baba  
5 2  
baacb  
5 3  
baacb  
5 3  
aaaaa  
6 4  
aaxxzz  
7 1  
phoenix  
выходные данные  
ab  
abbc  
b  
aa  
x  
ehinopx  
Примечание  
В первом наборе входных данных, одно из оптимальных решений — разбить baba на ab и ab.

Во втором наборе входных данных, одно из оптимальных решений — разбить baacb на abbc и a.

В третьем наборе, одно из оптимальных решений — разбить baacb на ac, ab и b.

В четвертом наборе, одно из оптимальных решений — разбить aaaaa на aa, aa и a.

В пятом наборе, одно из оптимальных решений — разбить aaxxzz на az, az, x и x.

В шестом наборе, одно из оптимальных решений — разбить phoenix на ehinopx.

# Выполнение

## Листинг кода задачи №1

1. **def** algorithm(n):
2. lis **=** []
4. **if** (n **<** 4): *#т.к. в таком случае невозможно соблюсти разрыв между числами в диапозоне от 2 до 4, а при n = 4 и больше - можно*
5. **return** **-**1
7. **if** n**%2** != 0:
8. **for** i **in** range(n,0,**-**2): *#вывожу все нечётные числа в порядке убывания*
9. lis**.**append(i)
11. lis**.**append(4) *#"критический" момент, т.к. иначе идёт 1, 2 - не соблюдается условие*
12. lis**.**append(2)
13. **for** i **in** range(6,n**-**1,2): *#вывожу чётные числа в порядке возрастания, начинаю с 6, т.к. 2 и 4 уже напчетананы*
14. lis**.**append(i)
15. **return** lis
16. **else**:
17. **for** i **in** range(n**-**1,0,**-**2): *#вывожу все нечётные числа в порядке убывания*
18. lis**.**append(i)
20. lis**.**append(4)
21. lis**.**append(2)
22. **for** i **in** range(6,n**-**1,2): *#вывожу чётные числа в порядке возрастания*
23. lis**.**append(i)
24. lis**.**append(n) *#поскольку начала с нечетного, то изначальное чётное число "висит", нужно его добавить, оно будет максимальным*
25. *#и чётным, поэтому в конец записываю*
26. **return** lis
28. t **=** int(input('Введите кол-во наборов входных данных: '))
29. **if** t **<** 1 **or** t **>** 100: *#защита от дурака*
30. **raise** Exception('t должно быть >= 1 и t <= 100')
32. arr **=** []
33. **for** i **in** range(0,t):
34. n **=** int(input('Введите описание набора входных данных (n): '))
36. **if** n **<** 2 **or** n **>** 1000: *#защита от дурака*
37. **raise** Exception('n должно быть >= 2 и n <= 1000')
39. arr**.**append(n)
40. print("----------------------------------------------")
41. **for** i **in** arr:
42. result **=** algorithm(i)
43. print(**\***result) *#выводит красиво массив*

## Листинг кода задачи №2

1. **def** null\_algorithm(l,r):
3. **if** l **>** r: *#если индекс начала больше индекса конца, то это не имеет смысла*
4. **return**
6. e **=** (l**+**r)**//**2 *#формула для нахождения инедкса для нечетного результата r-l+1*
7. a[e] **=** (l**-**r,e) *#в массив а кладется сет из двух значений*
8. null\_algorithm(l,e**-**1) *#рекурсия для левого подмассива; всегда будет деление на два подмассива с каждой стороны*
9. null\_algorithm(e**+**1,r) *#рекурсия для правого подмассива*
10. t **=** int(input('Введите кол-во наборов данных: '))
11. **if** t **<** 1 **or** t **>** 10**\*\***4: *#защита от дурака*
12. **raise** Exception('t должно быть >= 1 и t <= 10\*\*4')
14. arr **=** [] *#массив для значений n*
15. sum **=** 0
16. **for** i **in** range(t):
17. n **=** int(input('Введите длинну а:')) **+** 1 *#т.к. отрезки считаются с 1, а нумерация массива с 0 (n - длина массива из 0)*
19. **if** n **<** 1 **or** n **>** 210: *#защита от дурака*
20. **raise** Exception('n должно быть >= 1 и n <= 2\*10\*\*5')
22. arr**.**append(n)
23. sum **+=** n
25. **if** sum **>** 210: *#защита от дурака*
26. **raise** Exception('Сумма наборов тестовых данных не должна превышать 2\*10\*\*5!')
27. print("----------------------------------------------")
28. a **=** b **=** [0]**\***n *#заполняю два массива нулями*
29. **for** z **in** arr:
31. null\_algorithm(1,z**-**1)
32. **for** i,j **in** enumerate(sorted(a[1:z])): *#сортируются сеты по возрастанию (в таком порядке заполняются действтия i+1)*
33. *#нулевое число l-r, первое индекс, по которому заполняется, тогда j[1] это индекс*
34. b[j[1]] **=** i**+**1 *#b - массив с 0, заменяю в нём по индексам на нужные значения*
35. print(**\***b[1:z]) *#красиво вывожу массив*

## Листинг кода задачи №3

1. **def** reciept(l, r, n, p, rub): *#r - число большее максимума рублей Поликарпа, n - кол-во кусочков, p - цена кусочков,*
2. *#rub - деньги Поликарпа*
4. **while** l **+** 1 **<** r:
6. mid **=** (l**+**r)**//**2 *#берётся середина*
7. c **=** 0
8. **for** i **in** range(3):
9. c **+=** p[i]**\***max(a[i]**\***mid **-** n[i],0) *#складывает для каждой буквы сумму, начинаем*
10. *#с максимумуа денег у Поликарпа, функция максимума работает для случаев получения отриц.*
11. *#величин; в массиве а содержится количество каждых букв*
12. **if** c **>** rub:
13. r **=** mid
14. **else**:
15. l **=** mid
17. **return** l
18. s **=** list(input('Введите рецепт: '))
19. **if** len(s) **<** 1 **or** len(s) **>** 100: *#защита от дурака*
20. **raise** Exception('Длина строки не превосходит 100 и не может быть меньше 1 символа')
21. **for** i **in** s: *#защита от дурака*
22. **if** i **!=** 'B' **and** i **!=** 'C' **and** i **!=** 'S':
23. **raise** Exception('Введённая строка содержит недопустимые значения!')
25. a **=** [s**.**count(c) **for** c **in** 'BSC'] *#считаю, сколько кусочков хлеба, сыра и колбасы*
26. n **=** list(map(int,input('Введите кол-во хлеба, колбасы и сыра на кухне: ')**.**split()))
27. **if** len(n) **!=** 3: *#защита от дурака*
28. **raise** Exception('Нужно ввести 3 числа!')
29. **for** i **in** n:
30. **if** i **>** 100 **or** i **<** 1:
31. **raise** Exception('Кол-во кусочков не может быть больше 100 или 0!')
33. p **=** list(map(int,input('Введите цену кусочка хлеба, колбасы и сыра в магазине: ')**.**split()))
35. **if** len(p) **!=** 3: *#защита от дурака*
36. **raise** Exception('Нужно ввести 3 значения цены!')
37. **for** i **in** p:
38. **if** i **>** 100 **or** i **<** 1:
39. **raise** Exception('Цена не может быть больше 100 или равна 0!')
41. rub **=** int(input('Введите кол-во рублей у Поликарпа: '))
42. **if** rub **>** 10**\*\***12 **or** rub **<** 1:
43. **raise** Exception('Поликарп работает, у него не может быть 0 рублей, но и он не богач, у него не может быть больше 10\*\*12 рублей!')
44. print('Наибольшее кол-во гамбургеров = ', reciept(0, 10**\*\***13, n, p, rub))

## Листинг кода задачи №4

1. **def** derevo(t, n, vershu, vershv):
3. **while** t **>** 0:
5. degree **=** [0]**\***(n**+**1) *#будет n элементов, просто нумерация с 0, а я использую отрезок от 1 до n+1*
6. *#кладу в таблицу под номером вершины (индекс) единицу, т.е. если между 2 и 3 вершиной есть путь*
7. *#то degree[2] = 1 и degree[3] = 1; если в одной вершине несколько путей, то больше единицы будет*
8. *#так получается условие достижения специальной вершины <= 1*
10. **for** i **in** range(n**-**1): *#между вершинами u, v есть ребро*
12. degree[vershu[i]] **+=** 1
13. degree[vershv[i]] **+=** 1
15. **if** degree[x] **<=** 1:
16. **return** 'Ayush'
17. **elif** n**%2**:
18. **return** 'Ashish'
19. **else**:
20. **return** 'Ayush'
22. t **-=** 1
23. t **=** int(input('Введите кол-во наборов входных данных: '))
24. **if** t **<** 1 **or** t **>** 10: *#защита от дурака*
25. **raise** Exception('t должно быть >= 1 и t <= 10')
26. n, x **=** map(int, input()**.**split()) *#кол-во вершин в дереве и специальная вершина*
27. vershu **=** [0]**\***n
28. vershv **=** [0]**\***n
29. **if** n **<** 1 **or** n **>** 1000: *#защита от дурака*
30. **raise** Exception('n должно быть >= 1 и n <= 1000')
31. **if** x **<** 1 **or** x **>** n: *#защита от дурака*
32. **raise** Exception('x должно быть >= 1 и x <= n')
34. **for** i **in** range(n**-**1): *#между вершинами u, v есть ребро*
35. u, v **=** map(int, input()**.**split())
37. **if** u **<** 1 **or** u **==** v: *#защита от дурака*
38. **raise** Exception('u должно быть >= 1 и u != v')
39. **if** v **>** n: *#защита от дурака*
40. **raise** Exception('v должно быть <= n')
42. vershu[i] **=** u
43. vershv[i] **=** v
45. print(derevo(t,n,vershu,vershv))

## Листинг кода задачи №5

1. **def** shooter(t, p1, p2):
3. res **=** []
4. **for** i **in** range(t):
5. a0, prevb **=** p1[i] *#кладутся данные первого монстра, его здоровье и какой он наносит урон от взрыва*
6. ans, mn **=** 0, a0
7. **for** p **in** p2[i]:
8. a, b **=** p
9. ans **+=** max(0, a **-** prevb) *#убьет ли взрывом следующего монстра, если да, то 0, если нет, то остаток здоровья*
10. mn **=** min(mn, a, prevb) *#выбирается минимум между здоровьем первого монстра, здоровьем второго и уроном от взрыва первого (далее предыдущего)*
11. prevb **=** b *#предыдущим значением урона от взрыва теперь считается следующий монстр*
12. ans **+=** max(0, a0 **-** prevb) *#умрет ли монстр задетый взрывом, от этого зависит нужна ли ещё пуля и сколько*
13. res**.**append(ans **+** min(mn, prevb))
15. **return** res
16. t **=** int(input('Кол-во наборов: '))
17. p1 **=** []
18. p2 **=** []
19. **for** j **in** range(t):
20. n **=** int(input('Кол-во монстров: '))
21. p1**.**append(list(map(int, input()**.**split()))) *#кладем сюда a0 и prevb*
22. p2**.**append([list(map(int, input()**.**split())) **for** i **in** range(n**-**1)]) *#кладем сюда a и b*
24. res **=** shooter(t, p1, p2)
25. print(**\***res)

## Листинг кода задачи №6

1. **def** chet(A, n, t):
3. **for** i **in** range(t):
5. count **=** [0]**\***n
7. **for** k **in** range(n): *#формула из условия задачи*
9. **if** k **%** n **<** 0 **or** k **%** n **>** n**-**1: *#защита от дурака*
10. **raise** Exception('Всегда должно выполняться условие 0 ≤ 𝑘mod𝑛 ≤ 𝑛−1')
12. count[(A[k] **+** k) **%** n] **+=** 1
14. **if** **not** all(count): *#проверяет каждый элемент на истинность, где 1 - истина, 0 - ложь*
15. **return** 'NO'
16. **else**:
17. **return** 'YES'
19. t **=** int(input('Кол-во тестовых случаев: '))
20. sum **=** 0
21. result **=** []
22. **if** t **<** 1 **or** t **>** 10**\*\***4: *#защита от дурака*
23. **raise** Exception('t должно быть >= 1 и t <= 10\*\*4')
25. **for** i **in** range(t):
26. n **=** int(input('Длина массива: '))
27. sum **+=** n
29. **if** sum **>** 2**\***10**\*\***5: *#защита от дурака*
30. **raise** Exception('Сумма n по всем тестовым случаям не превосходит 2\*10^5!')
31. **if** n **<** 1 **or** n **>** 10**\*\***4: *#защита от дурака*
32. **raise** Exception('n должно быть >= 1 и n <= 10\*\*5')
34. A **=** [int(x) **for** x **in** input('Массив из целых чисел: ')**.**split()] *#массив из целых чисел для переносов гостей*
36. **for** i **in** A:
37. **if** i **<** **-**10**\*\***9 **or** i **>** 10**\*\***9: *#защита от дурака*
38. **raise** Exception('a должно быть >= -10\*\*9 и a <= 10\*\*9')
40. result**.**append(chet(A, n, t))
42. print(**\***result)

## Листинг кода задачи №7

1. **def** nok(n): *#вышла закономерность, что наименьшее максимальное число, кратное трем числам меньшим n - это*
2. **if** n **<=** 2:
3. **return**(n)
4. **elif** n **%** 2**!=** 0:
5. **return** n**\***(n **-** 1)**\***(n **-** 2)
6. **elif** n **%** 3**!=** 0:
7. **return** n**\***(n **-** 1)**\***(n **-** 3)
8. **elif** n **%** 2 **==** 0:
9. **return** (n **-** 1)**\***(n **-** 2)**\***(n **-** 3)
11. n**=**int(input('Введите n: '))
12. **if** n **<** 1 **or** n **>** 10**\*\***6: *#защита от дурака*
13. **raise** Exception('n должно быть >= 1 и t <= 10\*\*6')
15. print(nok(n))

## Листинг кода задачи №8

1. **from** math **import** **\***
2. **def** lcm(x,y): *#наименьшее общее кратное*
4. **return** x**\***y**//**gcd(x,y) *#gsd - наибольший общий делитель (из math)*
5. n **=** int(input('Введите целое число:'))
6. **if** n **<** 2 **or** n **>** 10**\*\***5: *#защита от дурака*
7. **raise** Exception('n должно быть >= 2 и n <= 10\*\*5')
9. l **=** list(map(int,input('Введите n чисел a: ')**.**split()))
10. e **=** 0 *#защита от дурака*
11. **for** i **in** l:
12. e **+=** 1
13. **if** i **<** 1 **or** n **>** 2**\***10**\*\***5: *#защита от дурака*
14. **raise** Exception('a должно быть >= 2 и n <= 2\*10\*\*5')
15. **if** e **!=** n:
16. **raise** Exception('Кол-во элементов a должно быть равным n!')
17. *#---------------------------------------------------------------------------*
18. t **=** [0]**\***(n**+**1) *#создаю массив из нулей*
19. t[**-**1] **=** l[**-**1] *#последний элемент l становится элементом последним для t*
20. **for** i **in** range(len(l)**-**1,0,**-**1):
22. t[i] **=** gcd(l[i],t[i**+**1]) *#нахожу наибольший общий делитель между элементом a и элементом t[i+1], всё кладется в предыдущий индекс t*
23. *#теперь в t лежат наиб общ делители*
24. b**=**[0]**\***n
25. **for** i **in** range(1,n**-**1): *#кладу в b (а \* наиб общ делителя из t)//предыдущее число в t, получается, что в b лежат кратные числа*
27. b[i]**=**(l[i]**\***t[i**+**1])**//**t[i]
29. x**=**b[0]
30. **for** i **in** range(1,n**-**1):
32. x**=**gcd(x,b[i]) *#наибольший общий делитель среди массива с кратными*
34. **if** len(l) **==** 1:
36. print(l[0])
38. **elif** len(l) **==** 2:
40. print(lcm(l[0],l[1]))
42. **else**:
44. print(x)

## Листинг кода задачи №9

1. **def** riddle(n, C):
3. prev **=** ''
4. c **=** cs **=** 0
5. **for** i **in** range(n):
6. x **=** y **=** 10**\*\***15 *#присваиваю максимально большое число, чтобы искать минимум*
7. **if** arrs[i] **>=** prev: x **=** c *#сравинваю вес двух строк*
8. **if** arrs[i] **>=** prev[::**-**1]: x **=** min(x,cs) *#сравниваю вес первой и перевернутой второй*
9. **if** arrs[i][::**-**1] **>=** prev: y **=** c **+** C[i] *#сравниваю вес перевернутой первой и второй*
10. **if** arrs[i][::**-**1] **>=** prev[::**-**1]: y **=** min(y,cs**+**C[i])
11. c **=** x
12. cs **=** y
13. prev **=** arrs[i]
14. ans **=** min(c,cs)
15. **if** ans **==** 10**\*\***15: ans **=** **-**1
17. **return** ans
18. n **=** int(input('Кол-во строк: '))
19. **if** n **<** 2 **or** n **>** 10**\*\***5: *#защита от дурака*
20. **raise** Exception('n должно быть >= 2 и n <= 10\*\*5')
22. C **=** list(map(int, input('Энергозатратность действий: ')**.**split()))
23. **if** n **!=** len(C):
24. **raise** Exception('Неверное количество чисел!')
25. **for** i **in** C:
26. **if** C **<** 0 **or** C **>** 10**\*\***9: *#защита от дурака*
27. **raise** Exception('ci должно быть >= 0 и ci <= 10\*\*9')
28. **for** i **in** range(n):
29. arrs**.**append(str(input()))
31. print('Минимальное количество необходимой энергии: ', riddle(n, C))

## Листинг кода задачи №10

1. **def** solver(n, k, s):
3. **if** s[0] **!=** s[k **-** 1] **or** k **==** n: *#длина строки равна количеству пустых строк*
4. **return** s[k **-** 1]
6. **elif** s[k] **==** s[**-**1]: *#элемент строки k равен последнему элементу строки*
7. **return** s[0] **+** s[k] **\*** ((n **-** 1) **//** k)
8. **else**:
9. **return** s[k **-** 1:]
10. e **=** int(input('Введите t: '))
11. **if** e **<** 1 **or** e **>** 10**\*\***3: *#защита от дурака*
12. **raise** Exception('t должно быть >= 1 и t <= 10\*\*3')
13. **for** i **in** range(e):
15. N, K **=** [int(i) **for** i **in** input('Введите длину строки и кол-во непустых строк: ')**.**split()]
17. **if** N **<** 1 **or** N **>** 10**\*\***5 **or** K **<** 1 **or** K **>** 10**\*\***5: *#защита от дурака*
18. **raise** Exception('N.K должны быть >= 1 и <= 10\*\*5')
20. t **=** ""**.**join(sorted(input()))
21. print(solver(N, K, t))

# Вывод

В ходе проделанной курсовой работы, я выполнил 10 алгоритмических задач на языке Python.