1. **ТАБЛИЦА** 10 очков 10 секунд

Дана таблица, состоящая из целых чисел. Написать программу, которая находит в таблице все такие места, где три идущие подряд элемента какой-либо строки, столбца или диагонали (параллельной главной диагонали) A, B, C удовлетворяют уравнению A+B=C. Элементы должны идти именно в таком порядке – в строках слева направо, в столбцах сверху вниз, в диагоналях слева-сверху вправо-вниз, сумма – именно третий элемент тройки.

Входные данные. В первой строке текстового файла находятся число строк N (1≤N≤100) и число столбцов M (1≤M≤100) таблицы. В каждой из следующих N строк файла находится M целых чисел, не превосходящих по модулю 1000 – элементы таблицы построчно.

Выходные данные. В текстовый файл TABEL.VAL вывести для каждой найденной тройки чисел в отдельную строку разделенные пробелом: номер ряда первого числа, номер столбца первого числа, и затем букву R (если тройка располагается в одной строке), V (если тройка располагается в одном столбце), или D (если тройка располагается на диагонали). Строки таблицы пронумерованы сверху вниз, стоблцы – слева направо, и обе нумерации начинаются с единицы. Если в таблице нет ни одной подходящей тройки чисел, вывести в первую строку файла слово POLE.

Пример.

```
TABEL.SIS          TABEL.VAL
3 4                1 1 R
4 5 9 7            1 1 V
3 2 7 7            1 1 D
7 7 6 7            1 2 V
```

2. **ОТРЕЗОК** 20 очков 10 секунд

Координатами своего левого верхнего и правого нижнего углов задано прямоугольное окно (стороны параллельны осям координат). Также заданы координаты концов нескольких отрезков. Написать программу, которая проверяет, какие части этих отрезков видны в окне и вычисляет координаты концов видимых частей отрезков.

Входные данные. В первой строке текстового файла LOIK.SIS находятся координаты левого верхнего и правого нижнего углов окна в формате x_1 y_1 x_2 y_2 ($x_1 \le x_2$, $y_1 \le y_2$). На второй строке файла находится количество отрезков N (1≤N≤100), и на следующих N строках – координаты концов отрезков также в формате x_1 y_1 x_2 y2. Все координаты – целые числа, не превосходящие по модулю 1000.

Выходные данные. В текстовый файл LOIK.VAL вывести для каждого заданного отрезка в отдельную строку: EI, если ни одна точка отрезка не находится внутри окна (считается, что стороны и углы прямоугольника также принадлежат ему), или в противном случае координаты той части отрезка, которая видна в окне (то есть принадлежит прямоугольнику), в том же формате. Выводимые координаты не должны отличаться от точных значений более чем на 0,01.

Пример.

```
LOIK.SIS           LOIK.VAL
10 20 100 200      EI
2                  50.00 60.00 70.00 80.00
5 6 7 8
50 60 70 80
```

3. **PL/S** 30 очков 10 секунд

PL/S – это простой язык программирования, в котором программа состоит из операций присвоения в виде

```
ПЕРЕМЕННАЯ=ЧИСЛО
```

или

```
ПЕРЕМЕННАЯ=ПЕРЕМЕННАЯ
```

где числа могут быть целыми числами, не превосходящими по модулю `10000`, а имена переменных могут состоять из `1-10` латинских букв, причем большие и маленькие буквы не различаются.

В операциях присвоения языка PL/S не допускается, чтобы справа от знака равенства встречалась переменная, значение которой к этому моменту не определено. Попытка использовать такую переменную считается ошибкой, и выполнение программы следует в этом месте прекратить с сообщением об ошибке.

Написать интерпретатор языка PL/S, то есть программу, которая исполняет команды языка PL/S.

Входные данные. В первой строке текстового файла `PLS.SIS` находится число команд программы `N` (`1≤N≤100`), в каждой из следующих `N` строк – одна команда присвоения.

Выходные данные. Результат выполнения программы вывести в текстовый файл `PLS.VAL`. Если выполнение программы завершилось успешно, вывести конечные значения всех используемых в программе переменных в формате `ПЕРЕМЕННАЯ=ЧИСЛО`, для каждой переменной в отдельной строке, имена переменных отсортировать в алфавитном порядке. Если во время выполнения программы возникла ошибка, вывести одну строку в формате `VIGA REAL X`, где `X` – номер команды, в которой произошла ошибка (команды программы пронумерованы от `1` до `N`).

Пример.

```
PLS.SIS
2
A=1
B=a
```

```
PLS.VAL
A=1
B=1
```

Пример.

```
PLS.SIS
2
A=1
B=C
```

```
PLS.VAL
VIGA REAL 2
```

4. **ПОСЛЕДОВАТЕЛЬНОСТЬ** 30 очков 10 секунд

Дано слово (строчка) `S`. Рассмотрим все слова (строчки) S_i, которые получаюся из `S` перестановкой его букв. Рассмотрим последовательность всех S_i, расположенные в алфавитном порядке. Написать программу, которая находит в этой последовательности слова предыдущее и следующее слове `S`.

Например, если S=’`bac`’, то соответствующая последовательность – ’`abc`’, ’`acb`’, ’`bac`’, ’`bca`’, ’`cab`’, ’`cba`’, из которой видно, что слово, предыдущее `S` – это ’`acb`’, а слово, следующее `S` – это ’`bca`’.

Входные данные. В единственной строке текстового файла JADA.SIS находится состоящее из маленьких букв латинского алфавита слово S длиной не более 80 символов.

Выходные данные. На первую строку текстового файла JADA.VAL вывести слово, предыдущее S в искомой последовательности, на вторую строку – слово, следующее за S. Если одно из них не существует (потому что S – первое или последнее слово в последовательности), вывести в соответствующую строку файла единственный символ '–' (минус).

Пример.

```
JADA.SIS        JADA.VAL
bac             acb
                bca
```

5. КОНТРОЛЬНАЯ СУММА 40 очков 10 секунд

Для проверки целостности доставленного сообщения в системах связи используются раз ные алгоритмы вычисления контрольных сумм. Одним из самых распространенных является алгоритм CRC (англ. *cyclic redundancy check*). При вычислении контрольной суммы CRC передаваемые данные воспринимаются как одно большое неотрицательное двоичное число (биты первого байта сообщения считаются самыми старшими битами этого двоичного числа, а биты последнего байта сообщения – самыми младшими). Также выбирается некоторое положительное число g (англ. *generator*), и в конец передаваемого сообщения добавляется в качестве контрольной суммы несколько байтов так, что полученное сообщение (если воспринимать его как одно большое двоичное число) делилилось бы на выбранное значение генератора g, причем при использовании n-байтового генератора добавляется ровно n байтов.

Входные данные. В первой строке текстового файла CRC.SIS находится генератор g. ($0<g<2^{16}$). Во второй строке находится число передаваемых сообщений N ($1\le N\le 5000$), и в каждой из следующих N строк находится ровно одно сообщение в виде слова, состоящего только из больших и маленьких букв латинского алфавита. Длина каждого сообщения не превышает 80 символов.

Выходные данные. В текстовый файл CRC.VAL для каждого заданного сообщения в отдельной строке вывести три целых числа r r_1 r_2, где r ($0\le r<g$) – остаток от деления сообщения (как большого двоичного числа) на g, а r_1 r_2 ($0\le r_1, r_2<256$) – байты, добавляемые в конец сообщения в качестве контрольной суммы. Причем 2-байтовое число, получаемое выписыванием подряд байтов ч r_1 и r_2, должно быть минимальное возможное. Все числа вывести в десятичной системе.

Пример.

```
CRC.SIS         CRC.VAL
34943           27560 123 22
3               0 0 0
ABCD            8670 42 251

abcdABCD
```

Оценивание. В этой задаче отдельно оценивают нахождение остатка от деления сообщения на генератор и нахождение сообстбенно контрольной суммы. Если ваша программа не вычисляет какое-либо из искомых чисел, то в его место бо быходной файл вывести знак '–' (минус). Нахождение остатка составляет 25%, а нахождение контрольной суммы 75% из стоимостьи задачи.