Preparação Prova 3

1. (`alfabetica.c`) Uma matriz de caracteres A, com m linhas e n colunas, é dita **linha-alfabética** se os caracteres de cada linha estão dispostos em ordem crescente. Isto é, se

   - $a_{i,j} < a_{i,j+1}$ para todo par i, j, com $1 \leq i \leq m$ e $1 \leq j < n$.

   Do mesmo modo, dizemos que a matriz A é **coluna-alfabética** se os caracteres de cada coluna estão dispostos em ordem crescente. Ou seja, se

   - $a_{i,j} < a_{i+1,j}$ para todo par i, j, com $1 \leq i < m$ e $1 \leq j \leq n$.

   Dizemos ainda que a matriz A é **toda-alfabética** se os caracteres de cada linha estão dispostos em ordem crescente e de cada coluna também estão dispostos em ordem crescente. Formalmente, a matriz $A_{m \times n}$ é toda-alfabética se:

   - $a_{i,j} < a_{i,j+1}$ para todo par i, j, com $1 \leq i \leq m$ e $1 \leq j < n$, e
   - $a_{i,j} < a_{i+1,j}$ para todo par i, j, com $1 \leq i < m$ e $1 \leq j \leq n$.

   (a) Escreva uma função com a seguinte interface:

   ```
   int linha(char u[MAX+1])
   ```

   que receba uma cadeia de caracteres u e devolva 1 se os caracteres que ocorrem nas células de u estão dispostos em ordem crescente. Caso contrário, devolva 0.

   (b) Escreva uma função com a seguinte interface:

   ```
   int coluna(int m, char A[MAX][MAX+1], int j)
   ```

   que receba um número inteiro $m > 0$, uma matriz A com m linhas onde cada linha é uma cadeia de caracteres, e um índice j, e devolva 1 se os caracteres que ocorrem nas células da coluna j de A estão dispostos em ordem crescente. Caso contrário, devolva 0.

   (c) Escreva um programa que receba um número inteiro $k > 0$ que representa a quantidade de casos de teste. Para cada caso de teste, receba um par de números inteiros m, n, com $1 \leq m, n \leq 100$, e uma matriz A de dimensão $m \times n$, onde cada linha é uma cadeia de caracteres, e verifique se a matriz é linha-alfabética, coluna-alfabética, toda-alfabética ou não-alfabética. Considere que os caracteres fornecidos na entrada são todos minúsculos. Na saída, imprima `L`, `C`, `T` ou `N` para uma matriz linha-alfabética, coluna-alfabética, toda-alfabética ou não-alfabética, respectivamente.

   Exemplo de entrada:

```
4

3 3
ahl
dep
bcq

3 4
acbe
fdhk
gijl

3 4
ajux
hkvy
ilwz

3 3
ahl
dim
fen
```

Exemplo de saída:

```
L
C
T
N
```

2. (`biblioteca.c`) Após seu emprego como ajudante de telefonista da UFMS e sua implementação impecável do sistema telefônico, o LEDES tomou conhecimento do seu incrível software feito para o sistema telefônico da UFMS e convidou-o para um estágio. Sua primeira tarefa é o de desenvolver um sistema para a Biblioteca Central da UFMS. Agora, sua tarefa é, dado uma lista de n livros, onde cada livro tem como informação o nome de seu autor e a prateleira onde ele se encontra e outra lista, com m nomes de livros, dizer qual é o nome do autor do livro e em qual prateleira os livros que estão nessa segunda lista se encontram. Caso o livro que esteja na segunda lista, não esteja na primeira lista, ou seja, não exista na biblioteca, imprimir apenas -1.

   Entrada: A primeira linha contém um inteiro n indicando o número de livros (máximo de 100), seguido dos n nomes de livros (nomes formados por uma única palavra de no máximo 49 caracteres)/ nomes dos autores dos livros (nomes formados por uma única palavra de no máximo 49 caracteres)/ prateleiras onde os livros de encontram, após isso, outro número inteiro m, indicando o número de livros da outra lista, onde para os m livros dados, você deverá imprimir o autor do livro, seguido da prateleira onde o livro se encontra.

   ```
   2
   Algoritmos Cormen 3
   Quimica Feltre 1
   3
   Quimica
   Historia
   Biologia
   ```

   Saída: A saída corresponde a impressão do nome dos autores do livro/ prateleiras onde os livros se encontram, ou -1, caso o livro não exista na biblioteca.

   ```
   Feltre 1
   -1
   -1
   ```

3. (`compare.c`) Dado n pares de strings, imprimir 1 caso elas sejam iguais e 0 caso sejam diferentes, sem diferenciar caracteres maiúsculos e minúsculos. Obs; O código ASCII dos caracteres maiúsculos varia de 65 a 90. Para transformá-los em minúsculos, basta adicionar 32 (ASCCI de A = 65, ASCII de a = 97). Você pode usar também a função `int tolower(int c)` (`tolower('A') = 'a'`).

   Entrada: A primeira linha contém um inteiro n que indica a quantidade de pares de strings, seguido de n linhas com 2 strings (de no máximo 49 caracteres cada) em cada linha.

   ```
   4
   pera pera
   uva Uva
   laranja laranja
   salada mista
   ```

   Saída: o tamanho de cada número

   ```
   1
   1
   1
   0
   ```

4. (`3n+1.c`) Considere o seguinte algoritmo para gerar uma sequência de números. Comece com um inteiro n: se n for par, divida-o por 2; se n, ímpar, multiplique-o por 3 e some 1 ao resultado. Repita esse processo para cada novo valor de n, terminando quando $n = 1$. Por exemplo, para $n = 22$ será gerada a seguinte sequencia de números:

   22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1

   Conjectura-se (mas ainda não foi provado) que este algoritmo termina em $n = 1$ para qualquer inteiro n. Sabe-se que essa conjectura se cumpre, pelo menos, para qualquer inteiro até 1.000.000. Para uma entrada n, o comprimento do ciclo de n é a quantidade de números gerados até, e incluindo, o 1. No exemplo anterior, o comprimento do ciclo de 22 é 16. Dados dois números quaisquer, i e j, deve-se determinar o máximo comprimento de ciclo dos números compreendidos entre i e j, incluindo ambos os extremos.

   Entrada: A entrada contém um número inteiro n, seguido de n pares de números inteiros, i e j.

   ```
   1 10
   100 200
   ```

   Saída: Escrever o máximo comprimento de ciclo dos inteiros compreendidos entre i e j para os n pares dados.

   ```
   20
   125
   ```