MAC–IME–USP

Carlos Eduardo Ferreira
Sala 108C Tel.: 3091 6079
e-mail cef@ime.usp.br

MAC 121 - Algoritmos e Estruturas de Dados I

Segundo semestre de 2018

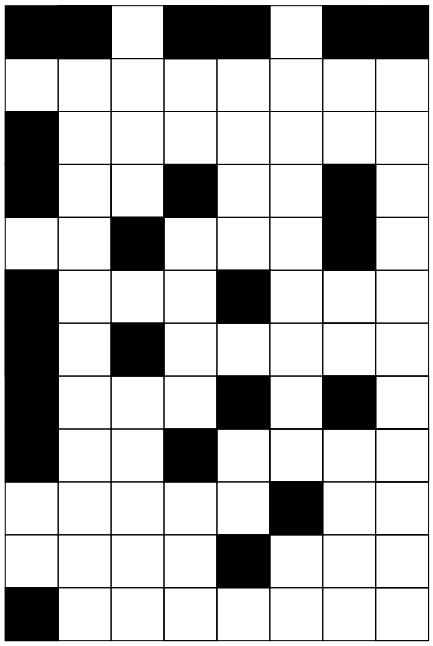
Segundo Exercício-Programa – Entrega: **23 de setembro**

Inventando palavras cruzadas

O objetivo deste primeiro exercício-programa é resolver um problema usando a técnica backtrack, que **deve ser implementada com pilhas** (não use recursão).

Interessado em fazer um dinheirinho extra, certo professor do Departamento de Computação do IME-USP resolveu passar aos alunos o seguinte exercício. Ele deseja montar desafios de palavras cruzadas (que revenderá aos jornais). Para isso, ele monta um tabuleiro e uma lista de palavras que ele deseja que esteja na matriz. A tarefa de vocês é encontrar um jeito de preencher o tabuleiro com as palavras, se existir. Neste caso o professor poderá vender o desafio aos jornais (e ficará rico).

Exemplo. Considere o tabuleiro abaixo e a lista de palavras:



ab, adido, ai, al, am, atice, banc, bo, boca, cao, dam, dm, enaltecedor, es, et, indo, it, itamaraca, mito, mo, moa, ms, mud, naus, neonato, permitir, remessa, res, robustecida, toast, trem, tri.

A tarefa deste exercício-programa será resolver o problema acima. Ou seja, dada a matriz de palavras cruzadas, e uma lista de palavras, seu programa deverá preencher a matriz, se for possível. Para isso, a estratégia usada deverá ser a de "força bruta" (backtrack), como feito em sala de aula para outros problemas. Note que no caso desse problema há vários jeitos de fazer um backtrack "mais inteligente". Não adianta tentar preencher uma linha com palavras que não têm o tamanho certo, ou que não possuem determinada letra na posição já preenchida. Seu programa poderá resolver mais palavras cruzadas em menos tempo se souber aproveitar isso e outras dicas para eliminar buscas desnecessárias.

Entrada

São dadas várias instâncias. Cada instância é composta pelo número m de linhas e de n colunas do tabuleiro de palavras cruzadas e, a seguir, por m linhas compostas cada uma por n 0's ou -1's (o -1 indica uma posição "preta" e o 0 indica uma posição branca. Depois do tabuleiro é dado o número p de palavras, que seguem nas p linhas seguintes. A entrada termina com uma matriz 0×0.

Exemplo:

```
5 4
 -1  0  0  0
  0  0  0  0
  0  0 -1  0
  0  0  0  0
  0 -1  0  0
10
ad
antas
carn
casa
do
lado
lua
os
soda
ur
3 3
 -1  0  0
  0  0  0
  0 -1  0
5
ab
au
bug
la
lua
0 0
```

Saída

Para cada instância seu programa deverá imprimir uma mensagem identificando a instância e a matriz modificada, imprimindo uma solução do jogo de palavras cruzadas com as palavras dadas, ou, dizendo que não há solução.

```
Instancia 1
* l u a
c a r n
a d * t
s o d a
a * o s

Instancia 2
nao ha solucao
```

Observações

- Vocês devem ler da entrada padrão (teclado) e imprimir na saída padrão (tela);
- Devem entregar apenas um arquivo com o código fonte (`<arquivo>.c`);
- As únicas bibliotecas permitidas são `stdio.h` e `stdlib.h` ;
- O formato de saída do exemplo deverá ser respeitado;
- Na correção o código será compilado no sistema linux usando

  `gcc -Wall <arquivo>.c -o ep2`