Caça-palavras

Programação de Computadores I

Ciência da Computação

Prof. Daniel Saad Nogueira Nunes

1 Introdução

Os Caça-palavras são um passatempo extremamente comum em meios impressos, tais como: jorais, gibis e revistas.

O objetivo deste passatempo é encontrar determinadas palavras que estão escondidas em uma tabela com diversas letras dispostas de uma maneira que dificultam a visualização das palavras a serem buscadas.

2 Objetivos

O objetivo deste trabalho é a implementação de um sistema que acha as palavras escondidas no caça-palavras.

Este sistema deverá estar dividido em módulos e seguir as boas práticas de programação. Um `Makefile` deverá acompanhar o sistema na sua distribuição para facilitar a construção dele.

2.1 Especificação

O caça-palavras é representado por uma matriz quadrada de caracteres sobre o alfabeto Σ das letras maiúsculas de A a Z, isto é, $\Sigma = \{A, B, C, \ldots, Z\}$, conforme Figura 1.

```
FWSOQAPOAV
YEOYCCSEVW
TRMACARRAO
PLESAXMSNE
EROVRALTZL
NSRORPBVZW
JAJCOCCPDO
HAJSKQVIHU
PTTPFXTZBP
BEPQNZPIII
```

Figura 1: Exemplo de caça-palavras

As palavras a serem encontradas também só podem ser compostas pelos símbolos do alfabeto Σ. Estas palavras podem estar dispostas nas direções:

- Vertical;

- Horizontal;
- Diagonal.

Além disso, elas podem estar dispostas nas orientações esquerda-para-direita ou direita-para-esquerda.

Tomando a Figura 1, suponha que queremos achar as palavras `ARVORE`, `OVO`, `CARRO`, `MACARRAO`, `POSTE` e `ROEMOS`. Ao considerar todas as direções e orientações, temos a seguinte solução, apresentada pela Figura 2

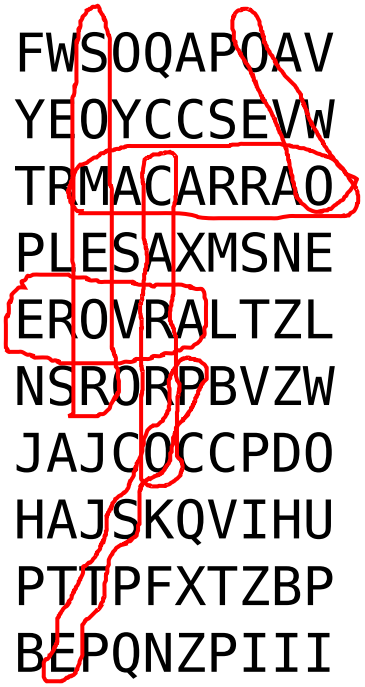

Figura 2: Solução para o caça-palavras da Figura 1

2.2 Modularização

O sistema deverá ser dividido em módulos, cada um para cumprir uma tarefa. Estes módulos podem ser organizados internamente através de várias funções e eles correspondem aos seguintes:

- Módulo de leitura: efetua a leitura dos dados de entrada.
- Módulo de saída: efetua a impressão das respostas para cada caso de teste.
- Módulo de processamento em strings: realiza o processamento da entrada para composição da saída.
- Módulo principal: contém a função `main` e as chamadas das funções exportadas pelos outros módulos. O ideal é que este módulo possua uma quantidade muito pequena de código, já que ele vai utilizar funções que estão presentes nos outros módulos.

Os módulos devem ser organizados em arquivos separados, com seus respectivos arquivos de cabeçalho e implementação.

2.3 Construção do sistema

Um `Makefile` deverá ser produzido para a compilação dos códigos-fontes no executável e deverá ser distribuído junto ao código.

2.4 Documentação

O código deve ser bem documentado, com presença de comentários explicando os trechos mais complexos do código. Além disso, um arquivo README deve ser providenciado com a devida identificação do autor descrevendo o projeto e instruindo como o código deve ser compilado através da ferramenta `make`.

2.5 Entrada

A primeira linha da entrada consiste de dois inteiros, N ($10 \leq N \leq 80$) e M ($1 \leq M \leq 10$), em que N representa a dimensão da matriz quadrada do caça-palavras e M a quantidade de palavras a serem encontradas.

As próximas M linhas correspondem cada uma a uma palavra a ser encontrada. Nenhuma das palavras a serem encontradas pode ter tamanho maior que N.

As próximas N linhas descrevem o caça-palavras.

2.6 Saída

Para cada palavra, na ordem da entrada, seu programa deverá imprimir uma linha contendo a palavra e os índices em que ela se encontra na matriz. Tais informações estão separadas por espaço e após o último índice não deve haver espaço. Os índices da matriz deverão ser apresentados da menor linha para a maior e em seguida da menor coluna para a maior e deverão estar entre parênteses e separados por vírgulas.

2.7 Exemplo

A Tabela 1 fornece um exemplo de entrada e saída esperada.

3 Critérios de correção

Deve ser utilizada a linguagem de programação `C` para a implementação do caça-palavras.

Para validação da correção do algoritmo, testes automatizados serão realizados, então é **crucial** que a saída esteja conforme o especificado.

Serão descontados pontos dos códigos que não possuírem indentação.

Tabela 1: Exemplo de entrada/saída.

Entrada	**Saída**
10 6 MACARRAO CARRO ARVORE ROEMOS POSTE OVO FWSOQAPOAV YEOYCCSEVW TRMACARRAO PLESAXMSNE EROVRALTZL NSRORPBVZW JAJCOCCPDO HAJSKQVIHU PTTPFXTZBP BEPQNZPIII	MACARRAO (2,2) (2,3) (2,4) (2,5) (2,6) (2,7) (2,8) (2,9) CARRO (2,4) (3,4) (4,4) (5,4) (6,4) ARVORE (4,0) (4,1) (4,2) (4,3) (4,4) (4,5) ROEMOS (0,2) (1,2) (2,2) (3,2) (4,2) (5,2) POSTE (5,5) (6,4) (7,3) (8,2) (9,1) OVO (0,7) (1,8) (2,9)

3.1 Ambiente de Correção

Para a correção dos projetos, será utilizada uma máquina de 64-bits com sistema operacional `GNU/LINUX` e compilador `GCC 10.2.0`, logo é imprescindível que o sistema seja capaz de ser compilado e executado nesta configuração.

4 Considerações

- o `GDB`, `Valgrind` e ferramentas gráficas associadas podem ajudar na depuração do código.
- Este trabalho deve ser feito **individualmente**.
- Não serão avaliados trabalhos que não compilem ou sem a presença de um `Makefile`.
- Como a correção é automatizada, deverá ser impresso apenas o que a especificação pede. Atentem-se para a formatação da saída.
- A incidência de plágio será avaliada automaticamente com nota 0 para os envolvidos. Medidas disciplinares também serão tomadas.
- O trabalho deve ser entregue dentro de uma pasta zipada com a devida identificação do aluno no prazo combinado pelo ambiente virtual de aprendizagem da disciplina.