

Instituto Federal Catarinense – Campus Videira.

Curso: Ciência da Computação

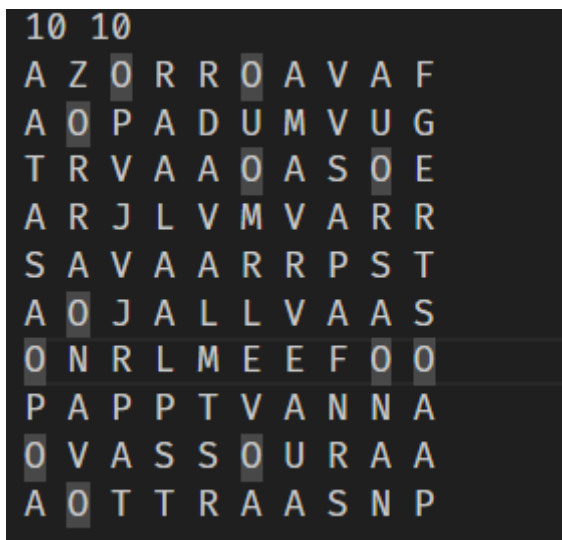
Disciplina: Algoritmos

Aluno: Ícaro Peretti Baseggio

Relatório Caça-palavras utilizando a linguagem de programação C

O algoritmo em questão possui o objetivo de determinar as coordenadas de uma palavra em uma matriz composta por caracteres. O programa faz a solicitação do nome do arquivo do arquivo txt que contém a matriz, lê esse arquivo e preenche a matriz com os caracteres contidos no arquivo. Em seguida é exibido um menu pedindo se o usuário quer buscar uma palavra ou finalizar a execução.

O arquivo txt deverá conter em sua primeira linha o tamanho da matriz e o restante dos caracteres em maiúsculo como mostra a imagem abaixo:



```
10 10
A Z O R R O A V A F
A O P A D U M V U G
T R V A A O A S O E
A R J L V M V A R R
S A V A A R R P S T
A O J A L L V A A S
O N R L M E E F O O
P A P P T V A N N A
O V A S S O U R A A
A O T T R A A S N P
```

Foram também importadas as bibliotecas padrão e também a ctype.h e string.h. A biblioteca ctype.h foi necessária para utilizar o recurso toupper na função que realiza a conversão da entrada do usuário para maiúsculo, para que seja comparada corretamente com os caracteres do arquivo.

```
void lower_to_upper(char *word)
{
    int i = 0;
    while (word[i] != '\0')
    {
        word[i] = toupper(word[i]);
        i++;
    }
}
```

Já a `string.h` foi utilizada para realizar algumas operações com string como copiar de uma variável ou constante para outra e para concatenar strings.

O algoritmo faz uma abordagem de força bruta, verificando todas as direções possíveis para encontrar a palavra dentro da matriz. As direções de busca incluem horizontal para frente e para trás, vertical para cima e para baixo e quatro direções diagonais (principal para frente, principal para trás, secundária para frente e secundária para trás).

No início da execução do programa, a primeira entrada que o usuário digita é o nome do arquivo que contém o caça-palavras (a matriz) e deve estar contido na pasta **files** do projeto. Após o usuário informar o nome do arquivo, o programa irá executar uma função para abrir e ler este arquivo e a função “`fill_matrix()`” que também irá chamar outra função para alocar dinamicamente a memória e prosseguirá com o preenchimento de uma matriz com os caracteres do arquivo.

Após isto um menu de escolha é executado em loop e só é encerrado caso o usuário escolha a opção sair. Caso ele escolha a opção buscar, será solicitado para que o usuário digite a palavra que deseja buscar. A palavra é convertida para maiúsculo, para garantir o funcionamento caso o usuário digite em minúsculo, pois como falado anteriormente o programa considerará que o arquivo estará com caracteres maiúsculos. Em seguida é chamada a função “`search_word()`”, essa função utiliza outras funções para realizar as buscas em diversas direções da matriz.

Depois de realizar a busca, o programa verifica se o tipo abstrato de dados nomeado ROI (Region Of Interest, ou Região de interesse) possui resultados. O ROI é um tipo abstrato de dados composto por outro tipo abstrato de dados que tem

a função de armazenar as coordenadas onde a palavra foi encontrada dentro da matriz, como exibe a imagem abaixo:

```
Icaro Peretti, 24 hours ago | 1 author (Icaro Peretti)
typedef struct
{
    int x, y;
} CORD;

Icaro Peretti, 24 hours ago | 1 author (Icaro Peretti)
typedef struct
{
    CORD A, B;
} ROI;
```

Caso o ROI possua valor, a função “show_roi()” é chamada para que exiba as coordenadas, sendo a linha inicial, linha final, coluna inicial e coluna final de onde a palavra foi encontrada na matriz, caso a palavra não tenha sido encontrada o programa exibirá uma mensagem “A palavra não foi encontrada”.

Como o programa foi desenvolvido utilizando alocação dinâmica de memória, ao final da execução toda memória alocada é liberada e o programa finalizado.

```
void destroy_matrix(char **matrix, int *rows)
{
    for (int i = 0; i < *rows; i++)
    {
        free(matrix[i]);
    }

    free(matrix);
}
```

A função na imagem acima demonstra como a memória da matriz é liberada. Como as outras variáveis que realizaram a alocação de memória são variáveis normais como por exemplo do tipo char, apenas a função free() da linguagem C foi utilizada para realizar a tarefa de liberar memória.

Testes

Durante a testagem com a [planilha](#) e os arquivos de texto disponibilizados para testes, foi possível identificar de maneira correta todas as palavras propostas, sendo da esquerda para direita, da direita para esquerda ou de baixo para cima e de cima para baixo. No entanto, novos arquivos com outras possibilidades não foram gerados para realizar o teste.

Os testes seguiram de acordo com a planilha disponibilizada, abaixo foram exibidos alguns com o primeiro arquivo disponibilizado.

Algumas das palavras que deveriam ser encontradas no arquivo 1 eram: sapato, amora, laranja, uva e cada uma estava disposta de uma maneira dentro do arquivo, como por exemplo a laranja que estava de baixo para cima. As imagens abaixo apresentam a execução e resultados do programa desenvolvido.

```
Digite o nome do arquivo (deve estar na pasta files): grade1.txt
0 S   A   P   A   A   C   A   V   A   A
1 A   A   P   A   D   A   V   U   U   J
2 T   A   P   A   A   P   A   S   O   N
3 A   S   J   A   V   M   V   A   R   A
4 J   P   O   A   T   R   R   A   S   R
5 A   O   J   A   L   O   R   A   A   A
6 O   P   A   O   M   O   U   P   S   L
7 P   A   P   P   M   V   A   U   A   A
8 O   A   T   A   L   S   V   A   P   A
9 A   O   T   T   R   A   A   S   O   P

Sair - 1
Buscar - 2
Escolha uma opcao: 2
Digite a palavra que deseja buscar: sapato

Buscando a palavra: SAPATO
Palavra encontrada.
linha inicial: 0, coluna inicial: 0
linha final: 5, coluna final: 5

Sair - 1
Buscar - 2
Escolha uma opcao:
```

```
Buscando a palavra: LARANJA
Palavra encontrada.
linha inicial: 6, coluna inicial: 9
linha final: 0, coluna final: 9

Sair - 1
Buscar - 2
Escolha uma opcao:
```

```
Buscando a palavra: UVA
Palavra encontrada.
linha inicial: 1, coluna inicial: 7
linha final: 1, coluna final: 5

Sair - 1
Buscar - 2
Escolha uma opcao:
```

```
Buscando a palavra: AMORA
Palavra encontrada.
linha inicial: 8, coluna inicial: 3
linha final: 4, coluna final: 7

Sair - 1
Buscar - 2
Escolha uma opcao:
```

```
Buscando a palavra: LARANJA
Palavra encontrada.
linha inicial: 6, coluna inicial: 9
linha final: 0, coluna final: 9

Sair - 1
Buscar - 2
Escolha uma opcao:
```

Caso comparados com a planilha, é possível afirmar que o algoritmo encontrou corretamente as coordenadas onde encontram-se as palavras. Os testes realizados com os outros dois arquivos também apresentaram resultados positivos, encontrando todas as palavras necessárias.