Universidade Federal de Viçosa - Florestal
Ciências da Computação

Trabalho Prático 1 – Projeto e Análise de Algoritmos

Trabalho 1 (Fazenda)

João Victor Graciano Belfort de Andrade
Guilherme Augusto Schwann Wilke

Florestal

2022

1.Introdução

Esta documentação se refere ao projeto feito no Trabalho Prático 1, da disciplina ‘ “PROJETO E ANÁLISE DE ALGORITMOS” , de código CCF 330. No trabalho, foi sugerido a implementação de um algoritmo com backtracking para encontrar um caminho, seguindo a sequência de fibonacci em uma matriz, neste contexto, implementá-lo em C e documentá-lo, conforme as especificações dadas.

2.Desenvolvimento

2.1. Lógica do Algoritmo

O algoritmo proposto na descrição do trabalho foi adaptado para uma melhor execução e boa interação com o usuário, sem deixar de seguir as especificações passadas na proposta. Foi implementado um menu inicial como descrito nas especificações, com 3 opções: uma para sair do programa, uma para ler um arquivo e uma para ler um input manual. Todas as implementações referentes ao menu foram realizadas no arquivo principal do programa, para cada uma das duas opções pode ser ativado ou desativado o modo de análise, essa ativação é feita na chamada da função buscaCaminho que será explicada a seguir, foi necessária uma TAD Tmatriz para montar a matriz referente a fazenda.

2.2 Lógica do Algoritmo de Backtracking

O algoritmo de backtracking foi implementado em sua grande maioria em “buscaCaminho.c”, a função principal de backtracking é a buscaCaminho que recebe a estrutura de dados Tmatriz como input, essa será explicada adiante, primeiro é necessário criar e inicializar a matriz de solução para a matriz da fazenda. em busca caminho temos 2 vetores para os movimentos combinados possíveis de x e y nesta função chamaremos a função resolva para cada uma das possíveis posições ao redor da casa inicial, a função faz isso para todas as casas da primeira fileira, parando assim que achar uma solução viável, a função resolva é a função recursiva para o backtracking em, essa função chama a função casaSegura que checa se a casa a qual vai ser dado o passo é segura ou não, retornando 0 no caso de não ser segura e 1 no caso de ser segura, se a casa for segura é adicionada ao vetor de soluções, e a função da o próximo passo recursivo, e se esse retornar um ela retornará 1 tamém, caso contrário retornará 0 e retirará a posição do vetor de soluções.

Caso a casa não seja segura então alguma das outras posições ao redor da casa, com base no vetor das possíveis posições de x e y, e assim por diante chamando a recursividade para as 4 posições do próximo. Caso a opção de análise esteja ativada será imprimida na tela a quantidade de recursões ao final das chamadas recursivas.

2.2 TAD Tmatriz

A TAD matriz guarda diversas estruturas, para serem utilizadas para o cálculo de backtracking, sendo estas tanto a própria matriz como estruturas auxiliares, estas são

matriz(matriz de inteiros representando a fazenda), sol(matriz de solução representando a solução para o problema), largura da fazenda , e a altura da fazenda, sendo todos ints.

Para essa TAD foram criadas 3 funções, resetSol(para resetar a matriz sol), inicializaMatriz(para inicializar a matriz de acordo com a altura e a largura), coordValida(para checar se a coordenada está dentro dos limites da matriz).

2.3 Algoritmos de Fibonacci

Foram implementadas funções que sigam a ordem passada pelo trabalho prático, para que essas funções funcionem foram implementados loops para achar a devida ordem de funcionamento dessa ordenação específica, não sendo usada recursividade para a resolução do problema, sendo a ordem normal de fibonacci: 1, 1, 2, 3, 5, (…) e a ordem criada 1, 1, 1, 1, 1, 2, 1, 1, 2, 3, 1, 1, 2, 3, 5, (…).

3. Compilando o Projeto

Para compilar o projeto basta digitar ($ make all) e depois ($ make run) dentro do diretório principal do projeto, para isso é necessário o cmake e o gcc instalados em seu computador, já para limpar basta rodar ($make clean) e então o arquivo gerado será apagado, e poderá ser compilado novamente.

4. Testes

A seguir estão os testes de cada uma das opções do menu para mostrar o funcionamento e a capacidade do algoritmo:

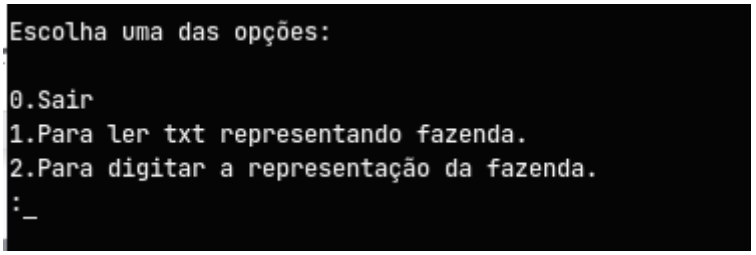

Imagem 1-menu básico do programa.

Imagem 2-pergunta se vai querer execução com análise ou não.

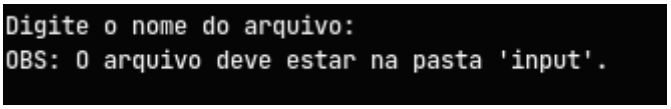

Imagem 3-pergunta o nome do arquivo dentro da pasta input, caso de leitura de texto.

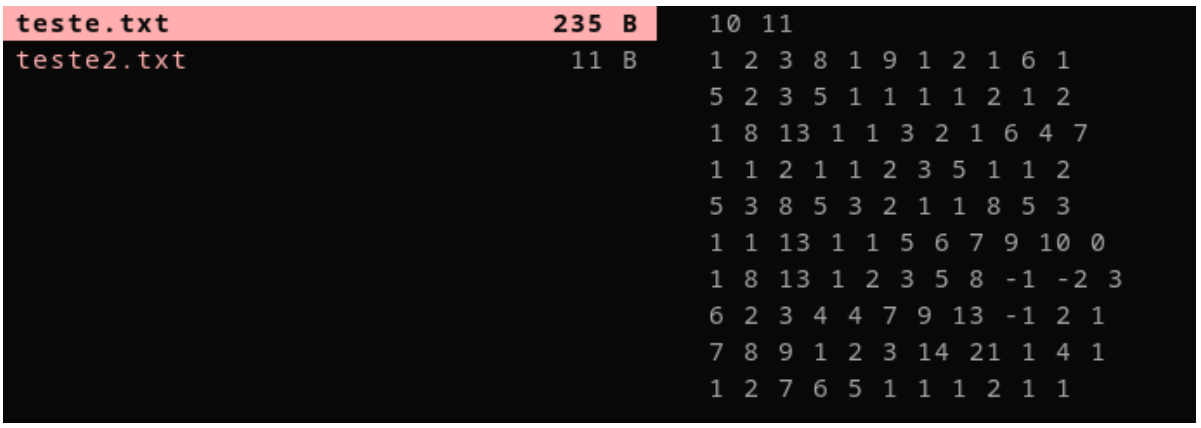

Imagem 4-Arquivo de teste.txt, dentro de inputs.

Imagem 5-Arquivo de teste2.txt, dentro de inputs.

Imagem 6-Arquivo de teste3.txt, dentro de inputs.

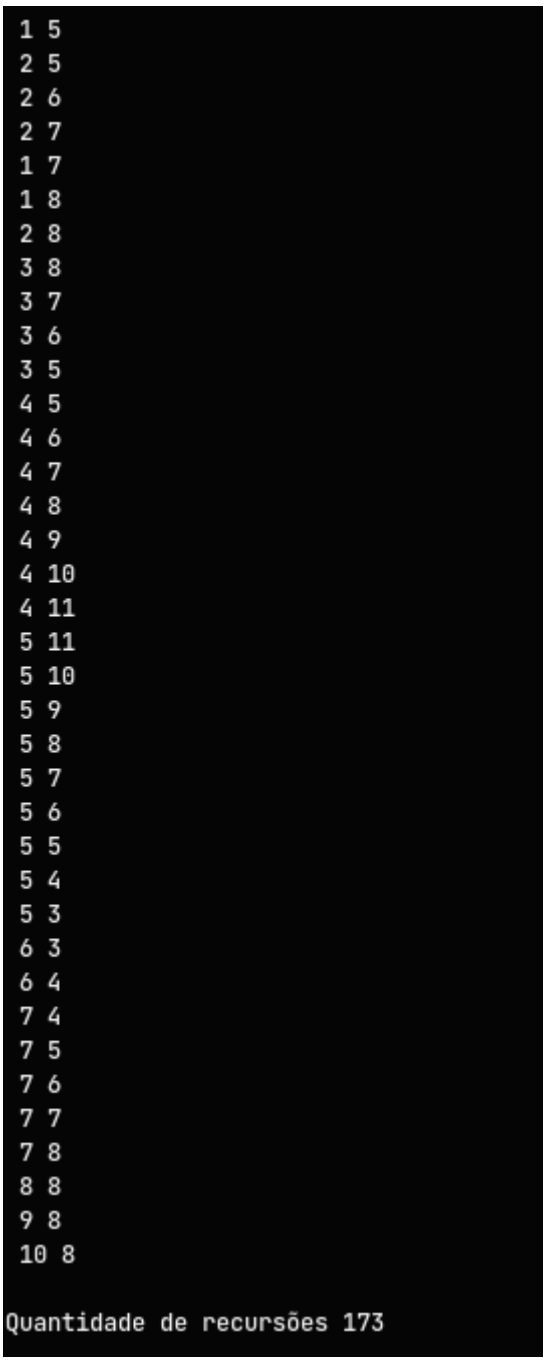

Imagem 5-Teste do arquivo teste.txt, com analise.

```
Formatação:

Primeira linha: Y X.
Outras: representação das linhas da matriz da fazenda.
entrada(Entrada incorreta retornara [IMPOSSÍVEL]):
1 1
1
 1 1

Quantidade de recursões 1
```

Imagem 6-Teste feito a mão com input digitado no terminal, em modo de analise.

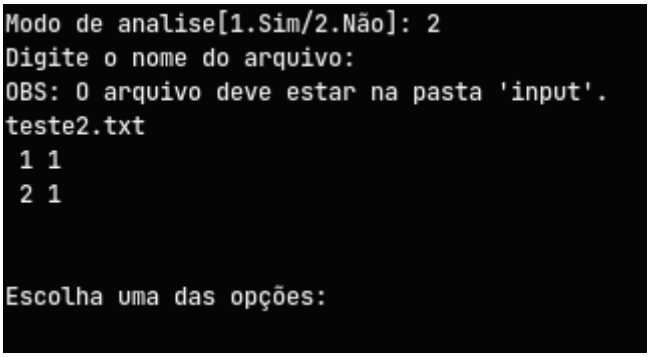

Imagem 7-Teste feito com teste2.txt, sem modo análise.

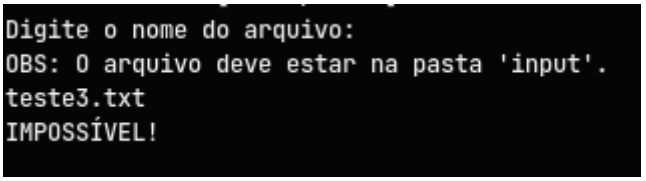

Imagem 8-Teste feito com teste3.txt, caso de impossibilidade.

5.Decisões tomadas no Projeto

A principal decisão feita no projeto foi a de como realizar o algoritmo de backtracking, a escolha feita foi baseada no algoritmo do cavalo do tabuleiro de xadrez, passado em aula pelo professor, testando diversas opções, utilizando ponteiros, memória dinamicamente alocada, e estruturas de uma forma que a memória não fosse muito utilizada para achar a solução.

6.Conclusão

De forma geral as diretrizes passadas pelo professor de CCF-330 foram seguidas para a realização do trabalho, tanto para projeção e execução do código quanto para confecção deste documento. Os resultados foram os esperados de acordo com as diretrizes, tentando conduzir uma abordagem interativa e mais prática possível com o usuário. Deste modo o trabalho foi conduzido com criatividade e seriedade para poder-se ter uma boa ideia de funcionamento, aplicação e execução de um algoritmo de backtracking.

7.Referências

Instruções para realização do trabalho [1].
Instruções para realização da documentação [2].
Introduction to Backtracking – Data Structure and Algorithm Tutorials[3]
Knight's Tour (backtracking) in C[4]

[1] Diretrizes básicas para as documentações da disciplina. Fornecida pelo professor Daniel através do Moodle.
[2] Trabalho 1. Fornecida pelo professor Daniel através do
Moodle.

[3]www.geeksforgeeks.org/introduction-to-backtracking-data-structure-and-algorithm-tutorials/.
[4]https://www.youtube.com/watch?v=1YspHboTIWg.