Universidade Federal de Viçosa – Campus UFV-Florestal
Ciência da Computação – Projeto e Análise de Algoritmos
Professor: Daniel Mendes Barbosa

# Trabalho Prático 3

Este trabalho é **obrigatoriamente em dupla** e deverá ser entregue no PVANet de acordo com as instruções presentes no final da especificação. Cada tarefa é um programa separado a ser feito.

Tarefa A) Implementar o problema da pirâmide que vimos em sala, onde o objetivo é escolher a rota do topo à base da pirâmide com maior soma, sendo que só podemos caminhar para baixo, escolhendo direita ou esquerda para descer para a linha seguinte. Você deverá nessa tarefa:

- implementar a leitura da pirâmide por meio de arquivo texto, no formato definido a seguir.
- imprimir ao final o valor total da maior soma e a rota a ser seguida para obtê-la. A forma de imprimir a rota será escolhida, explicada e justificada por você. Seja criativo.
- implementar o algoritmo das três formas: recursivo, memoization e "de trás para frente".
- no **relatório** deverá constar uma explicação de como cada versão do algoritmo foi implementada, estruturas de dados criadas, funções, etc, sempre remetendo aos conceitos de programação dinâmica.
- no **relatório** deverá constar exemplos de entrada e saída para a execução do programa em cada uma de suas versões.
- a interface geral do programa deverá ser definida por você.
- importante: deverá ser feita ainda uma discussão a respeito do desempenho de cada versão implementada, comparando com as demais.

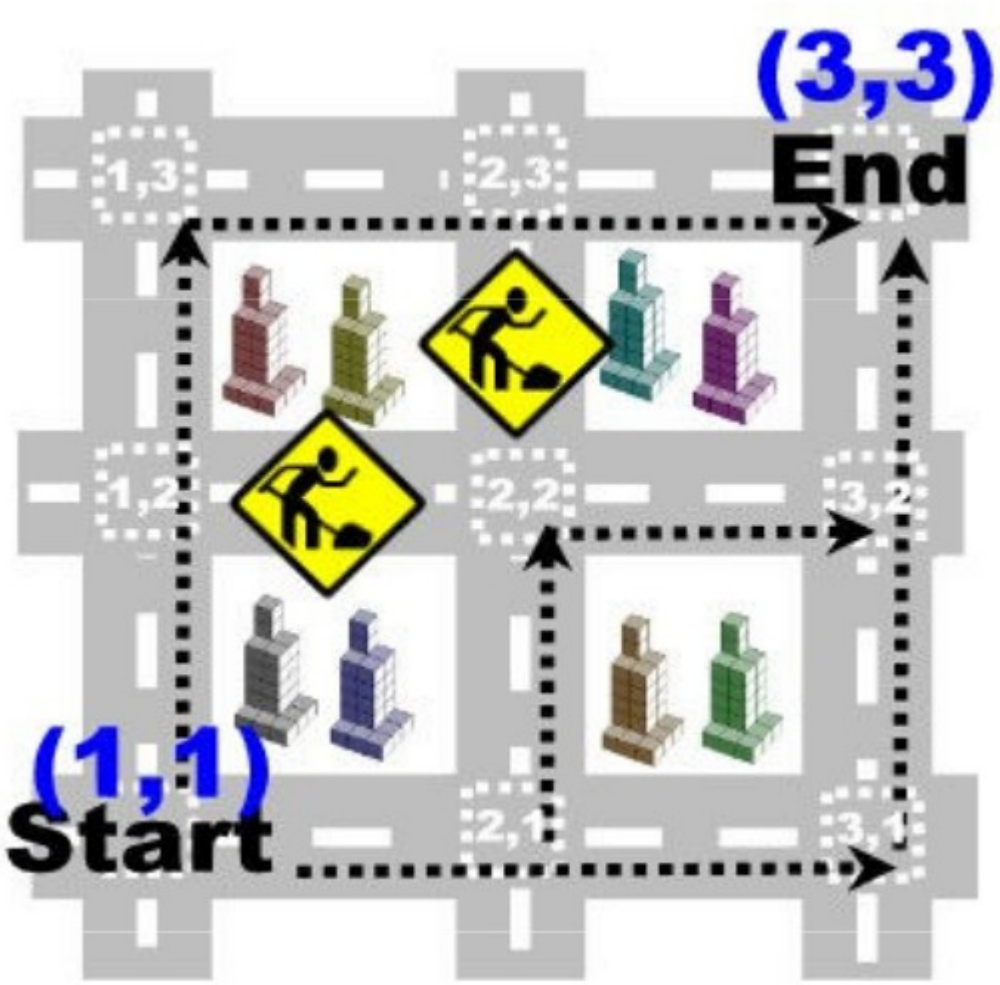
Formato do arquivo de entrada (para o primeiro exemplo dos slides):

```
7
3 8
8 1 0
2 7 4 4
4 5 2 6 5
```

Tarefa B) Considere o problema da cidade quadriculada também visto em aula. Você deverá implementá-lo, apenas da forma "de trás para frente", imprimindo ao final a quantidade total de maneiras para se chegar de um ponto inicial (x1, y1) até o ponto final (x2, y2).

Restrições importantes: x2 >= x1 e y2 >= y1, no entanto nenhum deles precisa estar na extremidade.

Figura 1:  exemplo de cidade quadriculada




Faça as estruturas de dados necessárias, justificando seus formatos, e preenchendo de trás para frente apenas o necessário.

Além de imprimir a quantidade total de maneiras, você deverá imprimir um dos caminhos possíveis do ponto inicial ao final. Você deverá imprimir na tela estas informações da melhor forma possível.

Formato do arquivo de texto de entrada:

- na primeira linha temos o número de linhas e colunas, separados por um espaço.
- na segunda linha temos as coordenadas (x, y) do ponto inicial, separadas por um espaço.

- na terceira linha temos as coordenadas (x, y) do ponto de destino, separadas por um espaço.

- a partir da quarta linha temos os locais onde temos obras, sempre colocando-se as coordenadas seguidas de N ou L, ou seja, norte ou leste.

Exemplo de arquivo de entrada para a Figura 1:

```
3 3
1 1
3 3
1 2 L
2 2 N
```

O **relatório** também deverá explicar as estruturas de dados utilizadas, os pontos principais da implementação, além de conter exemplos de entrada e saída para a execução do programa. A interface geral do programa deverá ser definida por você.

Observações importantes: para ambas as tarefas A e B, fazer várias execuções com tamanhos de entrada diferentes (tamanhos estes escolhidos por você), marcando o tempo total de execução pelo próprio programa (em um modo #debug, que também deverá ser documentado).
A ideia é traçar um gráfico para cada tarefa, de forma a mostrar o comportamento da execução do algoritmo em sua máquina. Para tanto, devem ser escolhidos tamanhos de entrada que façam sentido para a construção do gráfico.
Pode ser necessário criar programas auxiliares para gerar arquivos de entrada mais facilmente, com números aleatórios entre determinados intervalos definidos por você). Caso sejam utilizados, tais programas também devem ser entregues e constar no relatório, com suas devidas explicações.
Para entradas muito grandes, é possível que a saída de seus programas não funcionem muito bem (dependendo da forma que você fez), mas isso não é crítico, uma vez que nesses testes estamos mais interessados no tempo de execução. Lembrando também que o parâmetro n no problema da tarefa A é a quantidade de linhas da pirâmide enquanto na tarefa B é a quantidade de linhas e colunas do quadriculado.

**Faça exatamente o que está sendo pedido neste trabalho, ou seja, mesmo que você tenha uma ideia mais interessante para o programa, você deverá implementar exatamente o que está definido aqui, salvo nos pontos em que está definido que deverá ser definido por você. No entanto, novas opções que não entrem em conflito podem ser oferecidas.**

**Formato e data de entrega**:

Você deverá entregar todo o **código-fonte produzido (de preferência os dois projetos inteiros do Codeblocks)**, que será testado no sistema operacional **Linux**, bem como um **relatório** de documentação, que deverá conter os resultados de cada tarefa, conforme especificado anteriormente. Para a tarefa A deve ser feito apenas um projeto, com todas as versões do algoritmo disponíveis.

Importante: o arquivo a ser entregue no PVANet (até a data e horário limite lá estabelecidos) deverá ser um arquivo **.zip** contendo todo esse material produzido. O nome do arquivo deverá ter o nome e sobrenome dos membros da dupla. Exemplo: se os nomes dos alunos forem fulano jobs e ciclano jobs, o nome do arquivo deverá ser **fulanojobs-ciclanojobs.zip**.

Bom trabalho!