Universidade Federal de Ouro Preto – UFOP
Instituto de Ciências Exatas e Biológicas – ICEB
Departamento de Computação – DECOM
Disciplina: Estruturas de Dados I – BCC202
Professor: David Menotti (menottid@gmail.com)
Professora: Emiliana Mara Lopes Simões (simoes.eml@gmail.com)

Trabalho Prático 3 – Piramide [1]

Pilhas x Recursividade x ???

Valor: 0,5 pontos (5% da nota total) **Data de entrega: 11/05/2010**
Documentação não-Latex: -0,1 pontos
Interface gráfica: até 0,2 pontos extra
Impressão não frente-verso: -0,05 pontos

Objetivos

Consiste em utilizar conceitos de pilhas e recursividade para solução de um problema complexo.

Descrição

O Grande Museu do Egito está em construção há dois quilômetros da pirâmide de Giza e a abertura do museu está prevista para 2011. Algumas das peças históricas egípcias já foram movidas como, por exemplo, a estátua de Ramses II que foi movida em 2006. No Egito há mais de 100 pirâmides e para este problema estamos planejando mover a pirâmide de Giza para perto da entrada do museu.

É óbvio que a pirâmide é incrivelmente pesada, e não pode ser movida como um simples bloco. Então, ela será cuidadosamente fatiada em partes horizontais (blocos). Um enorme guindaste será usado para mover um bloco por vez para uma nova localização. É claro, que o guindaste pode segurar um único bloco ao mesmo tempo, e os blocos devem ser remontados em ordem apropriada, de forma que um espaço temporário é necessário para manter os blocos fora do caminho. Infelizmente, há um espaço muito limitado, e pode-se ter ao mesmo tempo três pilhas de blocos: a localização original, a nova localização, e um espaço temporário. O guindaste pode pegar somente o bloco no topo de uma pilha e colocá-lo no topo de uma pilha diferente.

Os engenheiros mediram e determinaram o peso e a capacidade de cada bloco. A capacidade de um bloco é o máximo peso que um bloco pode suportar sobre ele. Obviamente, cada bloco é forte o suficiente para agüentar os blocos que estavam originalmente sobre ele. Caso contrário a pirâmide teria desmoronado há muito tempo atrás.

Tarefa

Sua tarefa é planejar como mover todos os blocos da localização original até a nova localização, em uma ordem apropriada. Seu objetivo é completar o trabalho usando a menor quantidade de movimentos possíveis (um movimento consiste em pegar um bloco do topo de uma pilha e colocá-lo no topo de uma pilha diferente).

Esta é uma tarefa que produz uma saída. É dado a você (tp3-data.zip) 10 arquivos de entrada “pyrX.in”, onde X varia de 1 até 10 e você deve que produzir o arquivo de saída

correspondente "pyrX.out".

Entrada

Os arquivos de entrada são formatados da seguinte maneira:

- Primeira linha: um inteiro N ($2 \leq N \leq 20$) que representa o número de blocos horizontais;
- As próximas N linhas descrevem os N blocos horizontais em ordem, do topo até a base da pirâmide. Cada descrição de bloco consiste de 2 inteiros que são o peso e a capacidade do bloco. O peso e a capacidade variam de 1 a 100000000 (cem milhões), inclusive.

Saída

O arquivo de saída deve apresentar os movimentos realizados pelo guindaste. Cada movimento deve ser descrito em uma linha. Então, cada linha contém dois inteiros: a pilha fonte e destino, separadas por um simples espaço. Observe que a pilha 1, 2 e 3 representam a localização original, o espaço temporário e a nova localização da pirâmide, respectivamente. O número máximo de movimentos permitidos em qualquer um dos arquivos de saída é 3000000 (3 milhões) de movimentos.

Testando

É fornecido, junto com os dados de entrada, um verificador (PyramidChecker.exe) que pode ser usado para validar os seus arquivos de entrada e saída. Quando você executar o verificador, ele irá perguntar o nome dos arquivos de entrada e saída. Então, o verificador irá testar a validade dos seus arquivos e irá apresentar uma mensagem apropriada.

Avaliação

A nota de avaliação da implementação do seu trabalho prático será composta pela média aritmética da avaliação dos arquivos de saída da seguinte forma:

- 0 pontos se o arquivo de saída viola ao menos uma das regras especificadas acima;
- 10 pontos se o número de movimentos é o menor entre seus companheiros
- 2 + (6*A)/B pontos onde A é o menor número de movimentos entre seus companheiros e B é o seu número de movimentos. O valor será arredondado para o inteiro mais próximo.

Exemplo

Arquivo de Entrada 1	Arquivo de Saída 1A	Arquivo de Saída 1B
4 3 4 2 3 3 6 2 10	1 3 1 3 1 2 3 2 3 2 1 3 2 1 2 1 2 3 1 3 1 3	1 2 1 2 1 3 1 2 3 1 2 3 1 3 2 3 2 3

Para este caso, **Saída 1A** resolve o problema com 11 movimentos enquanto que **Saída 1B** resolve o problema com 9 movimentos. Assumindo que 9 movimentos é a melhor solução para este caso, então **Saída 1B** recebe 10 pontos e **Saída 1A** recebe 2 + (6*9)/11 arredondado para o menor inteiro que é 7 pontos.

O que deve ser entregue

- Código fonte do programa em C ou C++ (bem identado e comentado).
- Documentação do trabalho. Entre outras coisas, a documentação deve conter:
  1. Introdução: descrição do problema a ser resolvido e visão geral sobre o funcionamento do programa.
  2. Implementação: descrição sobre a implementação do programa. Deve ser detalhada a estrutura de dados utilizada (de preferência com diagramas ilustrativos), o funcionamento das principais funções e procedimentos utilizados, o formato de entrada e saída de dados, bem como decisões tomadas relativas aos casos e detalhes de especificação que porventura estejam omissos no enunciado. **Muito importante**: os códigos utilizados nas implementações devem ser inseridos na documentação.
  3. Análise de Complexidade: estudo da complexidade de tempo e espaço das funções implementadas e do programa como um todo (notação O).
  4. Listagem de testes executados: os testes executados devem ser apresentados e analisados e discutidos, quando convier.
  5. Conclusão: comentários gerais sobre o trabalho e as principais dificuldades encontradas em sua implementação.
  6. Bibliografia: bibliografia utilizada para o desenvolvimento do trabalho, incluindo sites da Internet se for o caso. Uma referência bibliográfica deve ser citada no texto quando da sua utilização
  7. Em Latex: Caso o trabalho não seja elaborado/escrito em latex, perde-se 0,1 pontos.

8. Formato: mandatoriamente em PDF (http://www.pdf995.com/).

Obs: Veja modelo de como fazer o trabalho em latex:
http://www.decom.ufop.br/menotti/edI101/tps/modelo.zip

Como deve ser feita a entrega:

A entrega DEVE ser feita via Moodle (www.decom.ufop.br/moodle) na forma de um **único** arquivo zipado, contendo o código, os arquivos e a documentação. Também deve ser entregue a documentação impressa na próxima aula teórica após a data de entrega do trabalho.

Comentários Gerais:

- Comece a fazer este trabalho logo, enquanto o problema está fresco na memória e o prazo para terminá-lo está tão longe quanto jamais poderá estar;
- Clareza, identação e comentários no programa também serão avaliados;
- O trabalho é individual (grupo de UM aluno);
- Trabalhos copiados (e FONTE) terão nota zero. Devido a recorrentes problemas com cópias de trabalhos (plágios), os autores de trabalhos copiados também terão a maior nota nos testes teóricos levada a zero, como forma de punição e coação ao plágio acadêmico;
- Trabalhos entregues em atraso serão aceitos, todavia será descontado 0,1 pontos por teto(hora) de atraso;
- Evite discussões inócuas com o professor em tentar postergar a data de entrega do referido trabalho.

Referências

[1] *International Olympiad in Informatics*, http://www.ioinformatics.org/ (sítio de internet), visitado em 30/04/2010.