Universidade Federal de Ouro Preto – UFOP
Instituto de Ciências Exatas e Biológicas – ICEB
Departamento de Computação – DECOM
Disciplina: Algoritmos e Estrutura de Dados I – CIC102 / 20091
Professor: David Menotti (menottid@gmail.com)

Trabalho Prático 3

Pilhas – Robô × Labirinto

Valor: 0,5 pts (5% da nota total) **Data de entrega:**
Documentação não-Latex: -0,1 pts **05/05/2009**
Interface gráfica com editor de labirinto: +0,1 pts
Impressão Frente-Verso: +0,05 pts (ecologicamente correto)

O objetivo deste trabalho é aplicar o conceito de pilhas para que um fictício robô ande para todos os espaços possíveis de um labirinto (especificado através de um arquivo) e retorne a sua posição inicial.

Durante o trajeto, o robô deverá coletar prêmios e desviar de possíveis buracos. Ao final, ele deve reportar quantos prêmios ele encontrou no labirinto e estes dados devem ser comparados com os fisicamente existentes no labirinto.

O robô pode andar somente em quatro direções: cima; baixo; direita e esquerda.

O labirinto deve ser armazenado em um arquivo do tipo texto com a extensão".dat".

- A primeira linha deste arquivo deve conter a palavra "**TP3ROBO**";
- A segunda linha do arquivo é composta por dois números que indicam o número de **linhas** e de **colunas** do labirinto, respectivamente. Obviamente que o tamanho do labirinto deve ser limitado tão e exclusivamente pela memória livre no computador. Em outras palavras, é proibido o uso de alocação estática, devendo o espaço utilizado para armazenar o labirinto em memória principal ser alocado em tempo de execução (ou seja deve-se usar alocação dinâmica de memória para matrizes).
- Após as duas primeiras linhas do arquivo, o labirinto começa a ser definido.
    - O labirinto é basicamente constituído de **vazios** (indicados pelo caractere espaço ' '), **paredes** (indicados pelo caractere 'x'), **buracos** (indicados pelo caractere 'b') e **prêmios** (indicados pelo caractere 'p');
    - Para evitar o tratamento de possíveis saídas do robô do labirinto, todas as "bordas" do labirinto devem ser preenchidas com paredes (x). Lembre-se que o após percorrer todo o labirinto, o robô deve retornar a sua posição inicial (indicada pelo caractere 'o').
    - Para simplificação, todas as letras usadas para definir o labirinto devem estar em caixa baixa - **minúscula**.

Deve-se implementar uma função para verificar a **consistência** do labirinto segundo todas as regras descritas acima.

Veja abaixo um exemplo de um arquivo de labirinto, chamado `lab1.dat`.

```
TP3ROBO
10 30
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
x   x     x     b   x xp     x px
x x x xxxx xxxxxx x  x xx x xx
x x   x px x        x x x  x  x
xpx x xxxx   xxx p x   xxxxxpx
x xxx x  xxxxx x   xpx x   x x
x x x x xx   x x       x xxx x
x xbx p  x x x   xxxx xb   x x
x   x xxox x   x x  p   xx   x
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

Cada posição (**`TPosicao`**) do labirinto deve ser implementada como um TAD, contendo dois campos: o tipo do campo; e um campo lógico para visita (vistado ou não). O labirinto (**`TLabirinto`**) deve ser implementado com um TAD composto por uma matriz de posições.

Procure testar o programa com vários labirintos. Crie-os usando sua imaginação.

Universidade Federal de Ouro Preto – UFOP
Instituto de Ciências Exatas e Biológicas – ICEB
Departamento de Computação – DECOM
Disciplina: Algoritmos e Estrutura de Dados I – CIC102 / 20091
Professor: David Menotti (menottid@gmail.com)

O que deve ser entregue

- Código fonte do programa em C ou C++ (bem identada e comentada).
- 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.
  3. Estudo de Complexidade: estudo da complexidade do tempo de execução dos procedimentos implementados e do programa como um todo (notação O).
  4. Listagem de testes executados: os testes executados devem ser simplesmente apresentados.
  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
  7. Em Latex: Caso o trabalho não seja elaborado/escrito em latex, perde-se 0,1 pontos.
  8. Impressão: Caso o trabalho seja impresso usando os dois lados da folha (impressão frente-verso), ganha-se 0,05 pontos extra – um incentivo à "preservar o planeta".
  9. Formato: mandatoriamente em PDF (http://www.pdf995.com/).

  Obs1: Consulte as dicas do Prof. Nívio Ziviani de como deve ser feita uma boa implementação e documentação de um trabalho prático: http://www.dcc.ufmg.br/~nivio/cursos/aed2/roteiro/

  Obs2: Veja modelo de como fazer o trabalho em latex:
  http://www.decom.ufop.br/prof/menotti/aedI091/tps/modelo.zip
  (caso alguém desenvolva um modelo similar em word, favor me enviar)

Como deve ser feita a entrega:

A entrega DEVE ser feita pelo 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 ou prática) após a data de entrega do trabalho.

Universidade Federal de Ouro Preto – UFOP
Instituto de Ciências Exatas e Biológicas – ICEB
Departamento de Computação – DECOM
Disciplina: Algoritmos e Estrutura de Dados I – CIC102 / 20091
Professor: David Menotti (menottid@gmail.com)

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;
- Trabalhos entregue em atraso serão aceitos, todavia a nota atribuída ao trabalho será zero;
- Evite discussões inócuas com o professor em tentar postergar a data de entrega do referido trabalho.