UNIVERSIDADE ABERTA
www.uab.pt

Código 21046

Estruturas de Dados e Algoritmos Fundamentais

(ano letivo 2015-16)

e-fólio B

Este enunciado constitui o elemento de avaliação designado por "e-fólio B" no âmbito da avaliação contínua e tem a cotação total de 4 valores. A sua resolução deve ser entregue até às 23h55 do dia 9 de maio pelos alunos que escolheram a modalidade de avaliação contínua.

A resolução deve ser entregue através de um único ficheiro compactado .zip, que:

(i) contém os ficheiros .cpp que constituem o código dos programas, prontos a serem compilados;

(ii) contém um ficheiro pdf de formato livre, com um relatório sucinto com informações complementares de modo a permitir uma fácil compreensão do trabalho realizado. É desnecessário incluir uma listagem integral do código.

(iii) O nome do ficheiro .zip a entregar deve seguir a seguinte convenção para o seu nome,

"NumeroAluno-PrimeiroNome-Apelido-21046-efB.zip"

Por exemplo, um aluno com número 327555 e nome Paulo ... Costa, deverá dar o seguinte nome ao ficheiro, "327555-Paulo-Costa-21046-efB.zip"

O ficheiro deve ser única e exclusivamente entregue através do recurso "E-fólio B" disponibilizado na plataforma (Nota: apenas é visível para os alunos inscritos em avaliação contínua), não sendo aceites trabalhos enviados por outras vias, como por exemplo por e-mail.

Esta é uma prova de avaliação **individual** e não "um trabalho de grupo". A sua resolução deve provir unicamente do conhecimento adquirido e trabalho original desenvolvido pelo próprio aluno. Os alunos deverão saber distinguir claramente entre discutir os conteúdos abordados na unidade curricular (permitido) e discutir a resolução específica do e-fólio (não permitido).

I

1. Pretende-se desenvolver um programa em linguagem C++ padrão de nome "xbsti.cpp" que aceite comandos para a gestão de uma árvore de pesquisa binária (Binary search Tree) para armazenar elementos ou itens que são inteiros não negativos. Os comandos de um modo geral devem permitir inserir, remover, listar e ainda balancear a árvore aplicando operações de rotação na árvore (algoritmo DSW).

Os comandos podem ser dados um de cada vez terminados por <ENTER> ou colocados em série num ficheiro com um comando por linha e executados na totalidade utilizando o operador '<' de redirecionamento da entrada padrão na linha de comandos do sistema operativo. Do ponto de vista do programa não deve existir qualquer diferença entre os dois modos de funcionamento.

a)[0.5] Projete as estruturas de dados (classes) adequadas ao programa que se pretende desenvolver. Defina apenas atributos (variáveis membro). Os métodos (funções membro) serão solicitados noutras alíneas. Justifique a presença/necessidade de cada atributo que definir.

Para a execução de cada comando deve indicar e explicar no relatório os métodos (funções membro) que criou/utilizou.

Os métodos e os comandos devem ser implementados tendo em vista a sua eficiência.

b)[1] Projete e implemente os comandos a seguir descritos,

insert i1 i2 i3 …
"insert": comando que significa inserir vários itens na árvore pela ordem indicada.
i1 i2 i3 …: uma sequência de inteiros não negativos separados por espaços e terminada por uma mudança de linha.

exit
"exit": comando que termina o programa.

c)[0.5] Projete e implemente o comando "preorder" a seguir descrito,

preorder
"preorder": comando que significa efetuar a travessia da árvore pela ordem designada "preorder" (sequência VLR). Visitar um nó significa imprimir o valor do nó seguido entre parêntesis do valor do seu nó parente (exceto para o nó raiz). Exemplo 12 8(12) 5(8) …

c)[0.5] Projete e implemente o comando "delete" a seguir descrito

delete i1 i2 i3 …
"delete": comando que significa remover vários itens da árvore pela ordem indicada.
O algoritmo de remoção a implementar é o de remoção por cópia (Deletion by Copying).

e)[1.5] Projete e implemente o comando "dsw" a seguir descrito

dsw
"dsw": comando que significa efetuar o balanceamento da árvore pelo algoritmo DSW.

- O programa deve estar identificado com um cabeçalho similar ao seguinte,

```
/*
** UC: 21046-Estruturas de Dados e Algoritmos Fundamentais
** e-fólio B 2015-16 (xbsti.cpp)
**
** Aluno: 327555 - Paulo Costa
*/
```

- A inclusão de "print screens" da linha de comandos no relatório só deve ser feita com fundo branco.

- No desenvolvimento do programa em C++ não deve ser utilizada a STL no que respeita às estruturas de dados e algoritmos estudados, devendo o aluno escrever o próprio código. Restrições aplicam-se nomeadamente aos includes <array> <deque> <forward_list> <list> <map> <queue> <set> <stack> <unordered_map> <unordered_set> <vector> e em parte de <algorithm>. Em caso de dúvida questionar o seu uso. Não existem restrições para <string>.

Critérios de correção:
- Programa não compila ou produz avisos (warnnings) com `g++ -Wall` => 0 valores.
- Código do programa não está correta e uniformemente indentado de modo a permitir a sua leitura fácil => 0 valores
- Programa não está comentado => 0 valores. Os comentários no programa elucidam questões relevantes do código locais ao comentário.
- Funcionalidade do programa de acordo com o pedido, estrutura, nível de simplicidade e qualidade do código (até 65%)
- Relatório. Explique o como e porquê relativamente às opções e soluções técnicas que tomou para a estrutura e funcionamento do programa (até 35%)

Nota ética: Nunca é de mais referir que o código a apresentar como solução para este e-fólio deve ser 100% original do aluno. A probabilidade de duas pessoas que efetivamente não comunicaram entre si, apresentarem programas "quase iguais" é considerada nula. Isto é válido para qualquer par de alunos (cópia), assim como entre um aluno e qualquer outra pessoa, em particular através da Internet (cópia/plágio), onde existem inúmeras soluções e código para os mais variados problemas, em sites, fóruns, blogs, etc.
Cumpra estritamente as normas de realização individual, como se estivesse num exame com consulta, onde pode consultar a documentação mas não pode falar com ninguém.

FIM