UNIVERSIDADE ABERTA
www.uab.pt

Código 21046

Estruturas de Dados e Algoritmos Fundamentais

(ano letivo 2017-18)

e-fólio A

Este enunciado constitui o elemento de avaliação designado por "e-fólio A" no âmbito da avaliação contínua e tem a cotação total de 4 valores. A sua resolução deve ser entregue até às 23h55 do dia 13 de abril pelos alunos que escolheram a modalidade de avaliação contínua.

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

(i) contém o ficheiro que constitui o código do programa;

(ii) contém um ficheiro "relatorio.pdf" de formato livre, com as respostas às questões.

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

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

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

O ficheiro deve ser única e exclusivamente entregue através do recurso "E-fólio A" 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 que aceite comandos para a gestão de uma lista simplesmente ligada (single linked list) para armazenar itens que são números inteiros (positivos ou negativos). Neste caso os itens representam ambos os papéis de chave e de informação. Os comandos de um modo geral devem permitir inserir, remover, alterar, procurar itens na lista além de outros comandos mais específicos.

A lista é inicializado por defeito como vazia e é alterada com comandos especificados num ficheiro de entrada fornecido ao programa pela entrada padrão (stdin em C e cin em C++), um comando por linha, podendo um comando ter vários argumentos, com o seguinte formato,

cmd arg0 arg1 ...

onde "cmd" indica o nome do comando a executar, "arg" é um tipo de argumento do comando e "..." a seguir a um argumento indica que este pode ser repetido várias vezes indefinidamente. Na descrição dos comandos, um argumento pode ser representado por: "pos" indicando uma posição na lista (inteiro >= 0); "item" o item a armazenar (inteiro).

A lista dos comandos a implementar é a indicada a seguir. No caso de mensagens de saída de dados, o seu formato é indicado em estilo da linguagem C.

insert_0 item ...
Comando que insere itens no início da lista pela ordem apresentada.

insert_end item ...
Comando que insere itens no fim da lista pela ordem apresentada.

print_0
Comando que imprime item do início da lista. Formato "Lista(0)= %d\n".

print_end
Comando que imprime item do fim da lista. Formato "Lista(end)= %d\n".

print
Comando que imprime toda a lista. Formato "Lista= %d %d ...\n".

delete_0
Comando que remove um nó do início da lista.

delete_end
Comando que remove um nó do fim da lista.

dim
Comando que imprime o número total de itens na lista. Formato "Lista tem %d itens\n".

clear
Comando que remove todos os nós da lista.

find item
Comando que procura a primeira ocorrência do item na lista e imprime a sua posição. Formato "Item %d na posicao %d\n" ou "Item %d nao encontrado!\n".

find_max
Comando que procura a primeira ocorrência do maior item na lista e imprime a sua posição. Formato "Max Item %d na posicao %d\n".

delete_pos pos
Comando que remove um nó da posição pos da lista.

invert_range pos1 pos2
Comando que inverte a ordem dos itens a partir da posição pos1 até à posição pos2 (inclusive) da lista. A estratégia a seguir é criar uma lista auxiliar, remover e reinserir os itens em causa.

Todos os comandos que não possam ser executados por a lista estar vazia devem imprimir uma mensagem com o formato "Comando %s: Lista vazia!\n".

Todos os comandos que não possam ser executados devido a uma posição indicada ser inválida devem imprimir uma mensagem com o formato "Comando %s: Posição invalida!\n".

É da responsabilidade do programa assegurar que nenhuma operação que comprometa a estabilidade do programa é executada, por exemplo remover um nó de uma lista vazia.

a)[0.75] Projete as estruturas de dados (classes) adequadas ao programa que se pretende desenvolver no que apenas respeita aos atributos (variáveis membro). A lista deve possuir apontadores para os nós inicial e final, assim como um contador que a cada momento indica quantos nós tem a lista. Justifique a presença/necessidade de cada atributo que definir.

Considerando uma lista inicialmente vazia, apresente diagramas do estado da lista (seguindo um estilo por exemplo inspirado no livro recomendado) com apontadores, nós e contador, após a execução de cada um dos seguintes comandos,

```
insert_0     2
insert_end  -1  5
delete_pos   2
```

Nota: Podem ser apresentados diagramas desenhados à mão e digitalizados.

b)[2.5] Projete e teste uma versão do programa que implemente as especificações e comandos pedidos. Utilize a plataforma HackerRank a partir do link disponibilizado na página da unidade curricular.

Os métodos (funções membro) são livres e uma consequência dos comandos a implementar. Deverá na elaboração do programa definir e criar os métodos e construtores que considerar mais

adequados. Os métodos e os comandos devem ser implementados tendo em conta também a sua eficiência.

O ficheiro de entrada pode ter linhas em branco e o nº de espaços que separa o comando e os argumentos entre si pode ser qualquer. Também podem existir linhas de comentário, caso em que começam obrigatoriamente pelo caracter '#' na 1ª posição.

No desenvolvimento do programa em C++ não deve ser utilizada a STL, nomeadamente no que respeita às estruturas de dados e algoritmos estudados, devendo o aluno escrever o próprio código. Restrições aplicam-se 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>.

A primeira posição da lista é a posição 0.

c)[0.75] Explique em detalhe o funcionamento e estratégia do código do método (ou métodos) que utilizou para implementar o comando "delete_pos pos", identificando todos os casos especiais (por exemplo lista com um elemento) e diferenciando-os do caso genérico.

Indique a complexidade do comando na notação Big-O. Justifique.

Critérios de correção:
- 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.
- A componente de funcionalidade é avaliada tendo como ponto de partida a fração de casos de teste com resultado positivo relativamente ao número total de casos de teste. O nível de simplicidade e qualidade do código também é avaliado. Programas considerados mal estruturados, demasiado complexos, confusos ou ineficientes podem ser penalizados até 30%.
- O relatório deve dar respostas às alíneas a) e c).

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