Teste 21 Dezembro 2019

Algoritmos e Complexidade

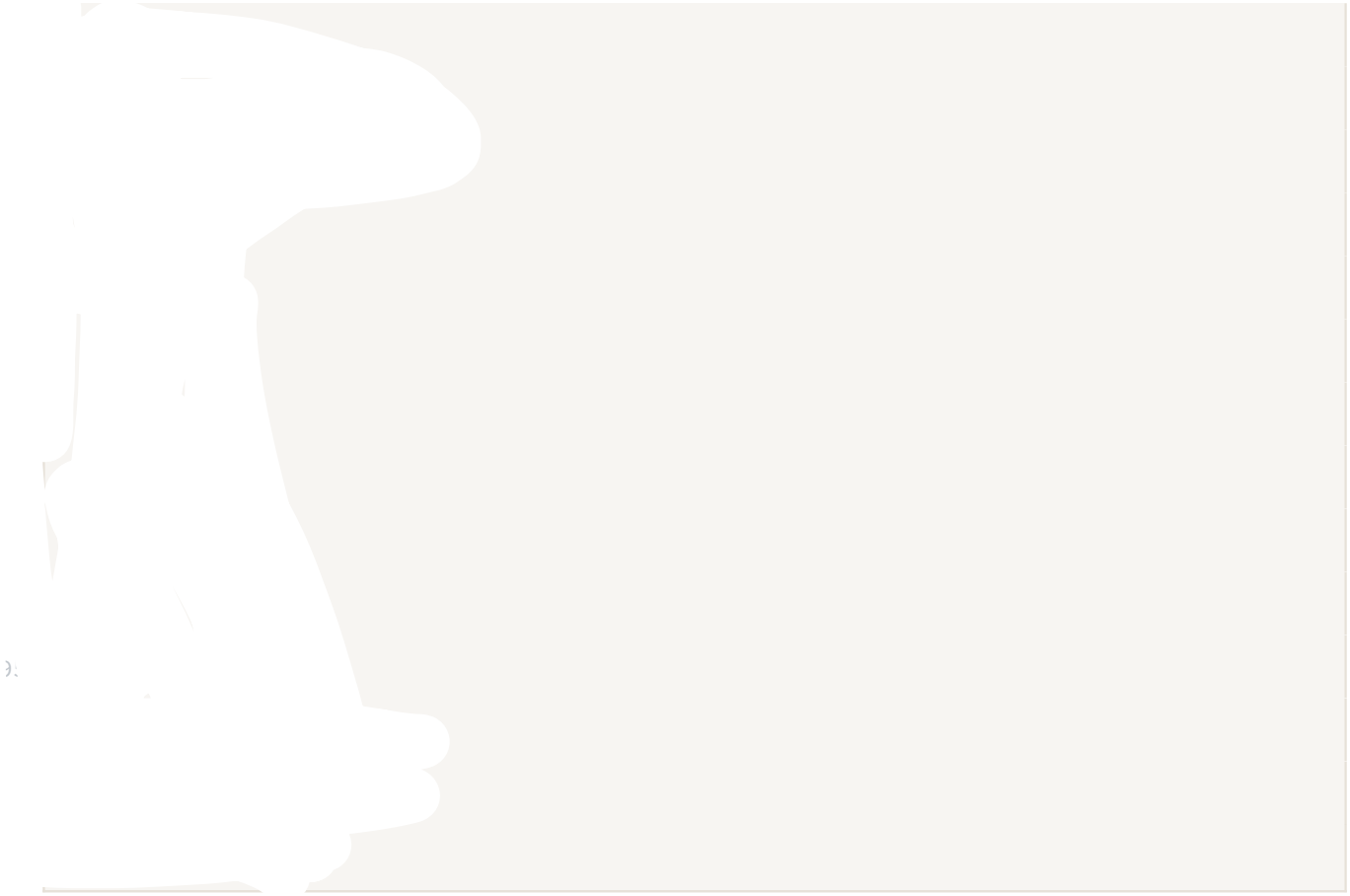
Universidade do Minho

Questão 1 [3 valores]
Considere uma estrutura de dados *min-heap* implementada sobre um *array* dinâmico, com comprimento inicial igual a 1. Quando completamente preenchido, o array é realocado com o dobro do tamanho. Apresente todos os estados do array, incluindo o comprimento alocado em cada estado, para a sequência de operações seguinte:

```
Insert 30; Insert 20; Insert 10; Insert 100; Insert 90; Insert 80; ExtractMin; ExtractMin; Insert 40;Insert 50; Insert 60; Insert 20; Insert 10; ExtractMin; ExtractMin
```

Questão 2 [3 valores]
Simule a evolução de uma árvore AVL, inicialmente vazia, ao longo da seguinte sequência de inserções. Identifique claramente todos os pontos em que o invariante é violado e a forma como é reposto.

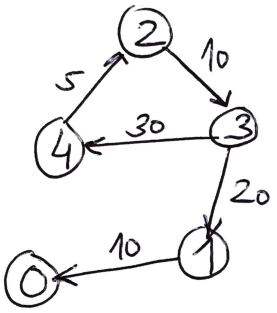
```
Insert 10; Insert 20; Insert 30; Insert 100; Insert 90; Insert 80; Insert 40; Insert 50;
```



Questão 3 [4 valores]
Pretende-se desenhar uma estrutura de dados para a implementação de *conjuntos de números naturais*, suportando as seguintes operações:

- Inserção
- Teste (dado um inteiro, testar se pertence / não pertence ao conjunto)
- *Rank* (dado um inteiro, contar o número de elementos do conjunto de valor inferior ou igual a ele).

Proponha uma implementação eficiente desta estrutura de dados e (sem escrever código) analise o tempo de execução de cada uma das operações, justificando e referindo assunções adicionais da sua análise.



Questão 4 [4 valores]

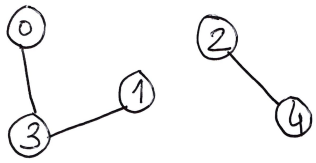
Defina em C a função `int pesoC (GraphM g, int V[], int k)` que calcula o custo do caminho do grafo g constituído por k vértices armazenados no *array* V. A função deverá devolver -1 caso a sequência V não corresponda a um caminho no grafo.

Por exemplo no grafo ao lado, o array V = {2, 3, 1} com k=3, corresponde ao caminho constituído pelas arestas (2, 3) e (3, 1), e o peso calculado deverá ser 10+20 = 30.

Note que o grafo é representado por uma **matriz de adjacências!**

Questão 5 [6 valores]

(i) Pretende-se etiquetar os vértices de um **grafo não-orientado** com um número que identifique o **componente ligado** a que pertence. Escreva a função `void componentes(GraphL g, int n, int comp[])` que coloca esta informação no *array* comp. Por exemplo se num grafo com 5 vértices tivermos os vértices 0, 1, 3 num componente e 2, 4 num outro, no final da execução teremos comp[0] = comp[1] = comp[3] = 0, e comp[2] = comp[4] = 1.



(ii) Analise o tempo de execução da função.