Lista de Exercícios

Prof. Charles Ferreira

Forma de Avaliação

1. A codificação será avaliada por:
   (a) Apresentação do código de forma limpa, ou seja, indentação do código;
   (b) Nome de variáveis e métodos com nomes significativos;
   (c) Uso do português correto;
   (d) Apresentação do resultado do problema.
2. Teste de mesa e passo a passo do algoritmo serão avaliados por:
   (a) Clareza do teste de mesa;
   (b) Apresentação do passo a passo;
   (c) Uso do português correto;
   (d) Apresentação do resultado do problema.
3. Análise de complexidade será avaliado por:
   (a) Apresentação do valor de cada linha do algoritmo;
   (b) uso do português correto;
   (c) Resultado final da análise de complexidade.

Entrega

- A atividade pode ser feita em **grupos de até 4 alunos**.
- Faça um arquivo com a extensão ".zip" com todos os arquivos da atividade.
- Basta que um integrante do grupo faça a submissão no Blackboard.
  - Nomeie o arquivo como: POTA_Lista_Nome_RA.zip.
  - Coloque um arquivo (.txt) contendo o nome e RA de todos os integrantes do grupo.
- A atividade deve ser entregue até o dia **03/10/2021**.

Exercícios

1. Escreva uma função recursiva que calcule o número de grupos distintos com k pessoas que podem ser formados a partir de um conjunto de n pessoas. A definição abaixo da função $Comb(n,k)$ define as regras: **(Valor 2,0)**

$$Comb(n,k) = \begin{cases} n & \text{se } k = 1 \\ 1 & \text{se } k = n \\ Comb(n-1,k-1) + Comb(n-1,k) & \text{se } 1 < k < n \end{cases}$$

2. Mostre, através do teste de mesa, o resultado das seguintes funções: **(Valor 1,5)**

```
1  public int alg5(int n, int k){
2    if (n == 0)
3      return 0;
4    else
5      return alg5(n-1, k) + k;
```

   (a) Considere as entradas:

   i. alg5(0,4);
   ii. alg5(4,6);
   iii. alg5(7,3):

3. Crie um método recursivo que procure um número dentro de um vetor, a partir de um índice. Se achar o mesmo, o método retorna o índice em que ele se encontra, senão, retorna -1 (busca linear recursiva). **(Valor 2,0)**

4. Analise o pior caso do método. Descreva o T(n) e mostre o O(n). **(Valor 3,0)**

(a)
```
1  float func1(int n, float A[], float x){
2    int k;
3    float y = 0.0;
4    for (k = n-1; k >= 0; k--){
5      y = A[k] + y * x;
6    }
7    return y;
8  }
```

(b)
```
1  int func2(int n){
2    int i, j, x, soma = 0;
3    for (i = 0; i < n; i++){
4      for(j = 0; j < n; j++){
5        for(x = 0; x < n; x++){
6           soma += n;
7        }
8      }
9    }
10   return soma;
11 }
```

5. Implemente os métodos de ordenação merge sort e insertion sort de tal modo que eles façam a ordenação de forma invertida. **(Valor 1,5)**