

**UNISUL – Universidade do Sul de Santa Catarina**

**Curso de Ciência da Computação**

**Disciplina de Estrutura de Dados**

**Professor Luciano Savio**

**Avaliação 1 – 2016/1**

ALUNO: Lucas dos Santos Gódeiro

1) De acordo com o código abaixo apresentado encontra-se parcialmente implementada uma TAD PILHA. Utilizando-se de alocação dinâmica de memória, implemente:

a) o(s) método(s) necessário(s) para que se consiga inserir e remover um novo elemento na pilha de carros. (Valor do ítem= 20% da nota)

b) faça uma adaptação no código apresentado, incluindo o código do ítem “a” desta questão, para que se consiga obter, a qualquer momento, o número de carros que existem na pilha.(Obs. Não é necessário implementar o método para remover elementos da pilha)

(Valor do ítem= 20% da nota)

```
///////////////  
// CLASSE Carro //  
///////////////  
public class Carro {  
  
    //atributos  
    public String placa;  
    public Carro proximo;  
  
    //construtor  
    public Carro(String p){  
        placa = p;  
        proximo = null;  
    }  
}
```

```
///////////////  
// CLASSE PilhaCarro ///  
///////////////  
public class PilhaCarro {  
  
    //atributos  
    private Carro topo = null; //o  
    topo da pilha  
  
    //construtor  
    public PilhaCarro(){  
    }  
}
```

2) Como se dá o funcionamento do TAD tipo Pilha? (Descrever)  
Valor da questão: 15% da nota)

- 3) Por que a análise experimental é uma métrica não muito utilizada para determinar a complexidade de um algoritmo?  
 (Valor da questão= 15% da nota)

- 4) Utilizando-se do método da análise teórica, determine a ordem de complexidade completa para o **MELHOR** e para o **PIOR** CASO do algoritmo abaixo apresentado:

Entrada: Tabela T : vetor[1...n]

Para i de 1 ate  $n - 1$  Faça

    minj  $\leftarrow$  i

    minx  $\leftarrow$  T(i)

    Para j de i + 1 ate  $n$  Faça

        Se T(j) < minx Entao

            minj  $\leftarrow$  j

            minx  $\leftarrow$  T(j)

    Fim Se

    Fim Para

    T(minj)  $\leftarrow$  T(i)

    T(i)  $\leftarrow$  minx

    Fim Para

    Fim.

(Valor da questão= 30% da nota)

PIOR	MELHOR
$1 + \overbrace{n+n}^{2n+1}$	$1 + n + n = 2n+1$
1	1
2	2
$2 + \overbrace{(n+1)+n}^{2n+3}$	$2 + (n+1) + n = 2n+3$
1	1
1	1
2	2
2	2
1	1
1	1

PIOR CASO

$$2n+1+3+2n+3+7+\dots = 4n+14$$

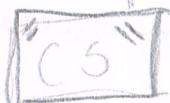
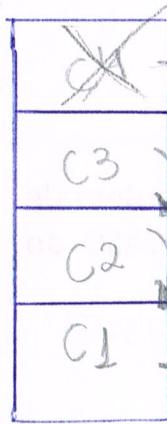
$$\boxed{4n+14}$$

Sucesso= oportunidade + preparo

MELHOR CASO

$$2n+1+3+2n+3+4+\dots$$

$$\boxed{4n+14}$$



para obj  
para topo

- ① retorna topo
- ② seta topo
- ③ retorna Retornar

$\text{Topo} = \text{C3}$

- ① seta topo
- ② seta prox
- ③ seta prox

```

public Cano remove() {
    if (topo == null)
        System.out.println("Pilha vazia");
    else {
        Cano cano = new Cano();
        topo = topo.prox;
        cano = topo;
        topo = topo.prox;
    }
}
  
```

Retornar C,

15/04 Lucas dos Santos Madeira  
4,9 + 3,0 7,9 20

② Na estrutura tipo pilha, o último elemento a ser inserido, é o primeiro a sair. Já o primeiro elemento que foi inserido será o último a sair.

③ Porque esta técnica constitui em realizar Testes de desempenho das soluções usando a mesma capacidade de hardware, ou seja, testa os algoritmos com o mesmo poder computacional. E mesmo dessa forma ainda pode-se obter variações de resultado, o que pode não ser tão eficaz.

#### IMPLEMENTAÇÃO

```
① public void inserir(String p){  
②     Carro c = new Carro();  
      if(topo == null){  
          c.prévia = p;  
          c.proximo = null;  
          topo = c; // topo = c; topo = null;  
      } else { // topo != null  
          c.prévia = topo.prévia; // topo.prévia = c; topo.prévia = null;  
          c.proximo = topo; // topo = c; topo = null;  
          topo.prévia = c; // topo.prévia = null;  
          topo = c; // topo = null;  
      } // topo = null;  
}
```

Def Prof aniversario sobre o que é

(a) public Carro remove(){  
    if (topo == null){  
        System.out.println("Pilha Vazia");  
    } else {  
        Carro c = new Carro();  
        c = topo;  
        topo = topo.proximo;  
    }  
    return c;  
}

(b) PilhaCarro.java

public class PilhaCarro{  
    int n = 0;  
    public void inserir(){  
        Carro c = new Carro();  
        if (topo == null){  
            c.placa = c;  
            c.proximo = null;  
            topo = c;  
            n = n + 1;  
        } else {  
            c.placa = c;  
            c.proximo = topo;  
            topo = c;  
            n = n + 1;  
        }  
    }

public Carro remove(){  
    if (topo == null){  
        System.out.println("Pilha Vazia");  
    } else {  
        Carro c = new Carro();  
        c = topo;  
        topo = topo.proximo;  
        n = n + 1; -1  
    }  
    return c;  
}