**Aula 1 - Estrutura de dados - 07/08/2018**

**Ementa:**

* Algoritmos de busca e ordenação;
* Tipos abstratos de dados. Listas lineares e suas generalizações;
* Pilhas estáticas;
* Filas estáticas;
* Recursividade;
* Ponteiros;
* Lista dinâmica;
* Pilha dinâmica;
* Árvores Binárias;

**Metodologia de avaliação:**

Média = (P1 + P2) / 2, onde pode - se acrescentar até um ponto de nota de trabalho nas notas da P1 e P2.

**Bibliografia Básica:**

* TENEMBAUM et al.Estrutura de dados usando C. São Paulo. Ed. MC Graw Hill.1995;
* DROZDEL. Estrutura de dados e Algoritmos usando C. Ed Cengage Learning 2017;
* TAMASSIA, Robert. GOODRICH, Michael T. Estrutura de dados e Algoritmos em Java. Ed. Brookman 2013.

**Bibliografia Complementar:**

* ASCENCIO, Ana Fernanda Gomes. ARAUJO, Graziela Santos de. Estrutura de Dados. Ed. Prentice Hall 2010;
* Apostila Unisalesiano (Xerox).

**Ordenação de Vetores**

**Bubble Sort (troca)**

A ideia é fazer uma série de comparações entre os elementos do vetor. O primeiro elemento é comparado com o segundo, se uma inversão for encontrada a troca é feita. Em Seguida, independentemente de ter havido troca ou não, o segundo elemento é comparado com o terceiro e verifica -se se há inversão. As comparações ocorrem (aos pares) até que o vetor esteja ordenado.

Exemplo:

1° Varredura: 28 26 30 24 25 par (28,26) troca

26 28 30 24 25 par (28,30) não troca

26 28 30 24 25 par (30,24) troca

26 28 24 30 25 par (30,25) troca

26 28 24 25 30

2° Varredura: 26 28 24 25 30 par (26,28) não troca

26 28 24 25 30 par (28,24) troca

26 24 28 25 30 par (28,25) troca

26 24 25 28 30 par (28,30) não troca

3° Varredura: 26 24 25 28 30 par (26,24) troca

24 26 25 28 30 par (26,25) troca

24 25 26 28 30 par (26,28) não troca

24 25 26 28 30 par (28,30) não troca

**Exercício:**

Dado o seguinte vetor: [13,19,8,1,26]

1. Ordene pelo método da troca.

1° Varredura: 13 19 8 1 26 par (13,19) não troca

13 19 8 1 26 par (19,8) troca

13 8 19 1 26 par (19,1) troca

13 8 1 19 26 par (19,26) não troca

2° Varredura: 13 8 1 19 26 par (13,8) troca

8 13 1 19 26 par (13,1) troca

8 1 13 19 26 par (13,19) não troca

8 1 13 19 26 par (19,26) não troca

3° Varredura 8 1 13 19 26 par (8,1) troca

1 8 13 19 26 par (8,13) não troca

1 8 13 19 26 par (13,19) não troca

1 8 13 19 26 par (19,26) não troca

1. Quantas varreduras são necessárias para o elemento 1 atingir sua posição final?

**R:** 3 varreduras.

Exercício: ordenar o seguinte vetor através do método Buble Sort:

17 2 20 4 3 11 8 5

1° Varredura: **17 2** 20 4 3 11 8 5 par (17,2) troca

2 **17 20** 4 3 11 8 5 par (17,20) não troca

2 17 **20 4** 3 11 8 5 par (20,4) troca

2 17 4 **20 3** 11 8 5 par (20,3) troca

2 17 4 3 **20 11** 8 5 par (20,11) troca

2 17 4 3 11 **20 8** 5 par (20,8) troca

2 17 4 3 11 8 **20 5** par (20,5) troca

2° Varredura:  **2 17** 4 3 11 8 5 20 par (2,17) não troca

2 **17 4** 3 11 8 5 20 par (17,4) troca

2 4 **17 3** 11 8 5 20 par (17,3) troca

2 4 3 **17 11** 8 5 20 par (17,11) troca

2 4 3 11 **17 8** 5 20 par (17,8) troca

2 4 3 11 8 **17 5** 20 par (17,5) troca

2 4 3 11 8 5 **17 20** par (17,20) não troca

3°Varredura: **2 4** 3 11 8 5 17 20 par (2,4) não troca

2 **4 3** 11 8 5 17 20 par (4,3) troca

2 3 **4 11** 8 5 17 20 par (4,11) não troca

2 3 4 **11 8** 5 17 20 par (11,8) troca

2 3 4 8 **11 5** 17 20 par (11,5) troca

2 3 4 8 5 **11 17** 20 par (11,17) não troca

2 3 4 8 5 11 **17 20** par (17,20) não troca

4° Varredura: **2 3** 4 8 5 11 17 20 par (2,3) não troca

2 **3 4** 8 5 11 17 20 par (3,4) não troca

2 3 **4 8** 5 11 17 20 par (4,8) não troca

2 3 4 **8 5** 11 17 20 par (8,5) troca

2 3 4 5 **8 11** 17 20 par (8,11) não troca

2 3 4 5 8 **11 17** 20 par (11,17) não troca

2 3 4 5 8 11 **17 20** par (17,20) não troca

main(){

int i, j, achou, aux;

int vet[10];

for(i=0;i<=9;i++){

cout<<”Informe o elemento”<<i<<”do vetor”<<endl;

cin>>vet[ i ];

}

//Ordenação:

achou = 1;

while(achou==1){

achou=0;

for(j=0;j<=8;j++) {

if(vet[ j ] > vet [ j+1]){

achou=1;

aux=vet[ j ];

vet[ j ] = vet[ j+1];

vet[ j+1] = aux;

}

}

}