**Bolha**

**n- quantidade de elementos (total)**

for (i=n-1; i>=1; i--) //for para voltar

for (j=0; j<i; j++) //for para ir

if (v[j]>v[j+1]) //se for menor troca e maior que o proximo

{ /\* troca \*/

temp = v[j]; //guarda numa auxiliar

v[j] = v[j+1];

v[j+1] = temp;

}

Int vet[5]={2,5,1,3,10}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2 | 5 | 1 | 3 | 10 |
| 2 | 5 | 1 | 3 | 10 |
| 2 | 1 | 5 | 3 | 10 |
| 2 | 1 | 3 | 5 | 10 |
| 1 | 2 | 3 | 5 | 10 |
|  |  |  |  |  |

**Método extremamente lento: só faz comparações entre posições adjacentes**

• **É o método mais ineficiente entre os simples**

– **Melhor caso: vetor já ordenado**

– **Pior caso: vetor de entrada em ordem reversa**

• **Cada passo aproveita muito pouco do que foi “descoberto” em relação à ordem das chaves no passo anterior (exibe informações redundantes)**

**Inserção + rapido**

**Vai e volta já encaixando o menor numero, começa no índice 1 e compara o que esta antes dele**

**Se não é menor deixa a assim e volta a percorrer, se é menor troca.**

j=1; //começa no índice 1

while (j<tam) //controla ida

{

aux = vetor[j]; /aux armazena

i = j - 1;

while ( i >= 0 && vetor[i] > aux )

{ /jogando p frente os números maiores

vetor[i+1] = vetor[i]; //coloca a frente

i = i - 1;

}

vetor[i+1] = aux; //encaixa o menor no lugar dele, não precisa ficar indo ate o índice 0

j = j + 1;

}

Aux= 3

|  |  |  |  |
| --- | --- | --- | --- |
| 2 | 5 | 1 | 3 |
| 2 | 5 | 1 | 3 |
| 1 | 2 | 5 | 3 |
| 1 | 2 | 3 | 5 |
|  |  |  |  |
|  |  |  |  |

**Vantagens:**

– **O número mínimo de comparações e movimentos ocorre quando os itens já estão originalmente ordenados**

– **O número máximo ocorre quando os itens estão originalmente em ordem reversa, o que indica um comportamento natural para o algoritmo**

**Seleção**

**Seleciona o menor numero e**

coloca la no índice 0, o que estava no índice 0 vai para o índice que o menor numero estava

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

minimo = i;

// pega indice do menor

for (j=i+1;j<tam;j++){ //percorre

if ( vetor[j] < vetor[minimo] ) //fica procurando o menor

{

minimo = j;

} //chegou ao final troca

}

aux = vetor[minimo]; //utiliza aux

vetor[minimo] = vetor[i];

vetor[i] = aux;

}

COLOCA O MENOR NO PRIMEIRO INDICE SUBSTITUI O OUTRO

|  |  |  |  |
| --- | --- | --- | --- |
| 2 | 5 | 1 | 0 |
| *0* | 5 | 1 | 2 |
| 0 | 1 | 5 | 2 |
| 0 | 1 | 2 | 5 |

**Um dos algoritmos mais simples**

**• Mais recomendado para conjuntos pequenos**

• **Procedimento:**

**– Selecione o menor item do conjunto e troque-o com o item que está na posição i**

**– Repita essas operações com os demais itens até que reste apenas um elemento**