

# Unidade V:


## Ordenação Interna - Shellsort



**PUC Minas**

Instituto de Ciências Exatas e Informática  
Departamento de Ciência da Computação

- Funcionamento básico
- Algoritmo em C#
- Análise dos número de movimentações e comparações

- **Funcionamento básico** 
- Algoritmo em C#
- Análise dos número de movimentações e comparações

# Introdução

- Proposto por Shell em 1959 como uma extensão da Inserção
- O problema da Inserção é efetuamos  $(n - 1)$  comparações e movimentações quando o menor elemento está à direita
- O método de Shell contorna esse problema permitindo trocas de registros distantes um do outro (passo  $h$ )

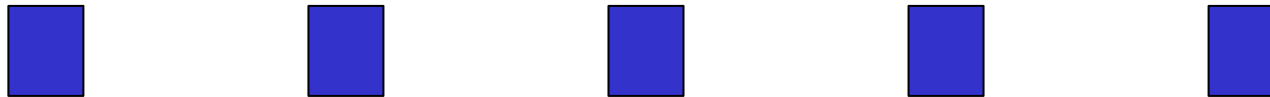
# Funcionamento Básico

- Os elementos separados de  $h$  posições são ordenados via Inserção
- Por exemplo, se  $h = 4$ , ordenamos com a Inserção cada um dos pseudo *array*



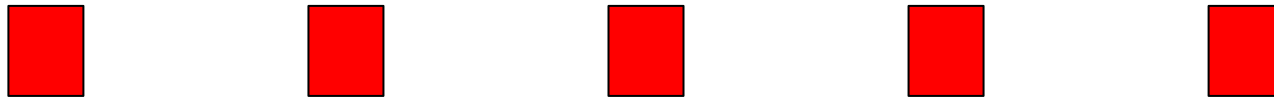
# Funcionamento Básico

- Os elementos separados de  $h$  posições são ordenados via Inserção
- Por exemplo, se  $h = 4$ , ordenamos com a Inserção cada um dos pseudo *array*



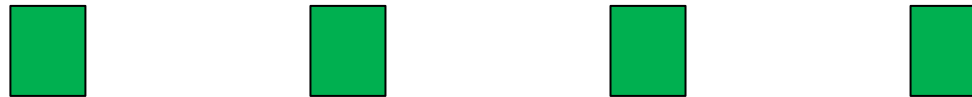
# Funcionamento Básico

- Os elementos separados de  $h$  posições são ordenados via Inserção
- Por exemplo, se  $h = 4$ , ordenamos com a Inserção cada um dos pseudo *array*



# Funcionamento Básico

- Os elementos separados de  $h$  posições são ordenados via Inserção
- Por exemplo, se  $h = 4$ , ordenamos com a Inserção cada um dos pseudo *array*





# Funcionamento Básico

- Os elementos separados de  $h$  posições são ordenados via Inserção
- Por exemplo, se  $h = 4$ , ordenamos com a Inserção cada um dos pseudo *array*



# Funcionamento Básico

- Os elementos separados de  $h$  posições são ordenados via Inserção
- Por exemplo, se  $h = 4$ , ordenamos com a Inserção cada um dos pseudo *array*



Pode parecer loucura, mas repetiremos a Inserção para cada pseudo *array*

# Funcionamento Básico

- Neste momento, nossa sequência é dita h-ordenada (para nosso exemplo 4-ordenada)
- Em seguida, reduzimos o valor de h e repetimos o processo. Isso até que o valor de h seja um e efetuamos a Inserção pela última vez

# Funcionamento Básico

- Neste momento, nossa sequência é dita h-ordenada (para nosso exemplo 4-ordenada)
- Em seguida, reduzimos o valor de h e repetimos o processo. Isso até que o valor de h seja um e efetuamos a Inserção pela última vez

Traduzindo: Nós já fizemos quatro vezes a Inserção e vamos fazer mais...

# Funcionamento Básico

- Neste momento, nossa sequência é dita h-ordenada (para nosso exemplo 4-ordenada)
- Em seguida, reduzimos o valor de h e repetimos o processo. Isso até que o valor de h seja um e efetuamos a Inserção pela última vez

... você acredita que isso  
ainda é bom!!! Fala sério!

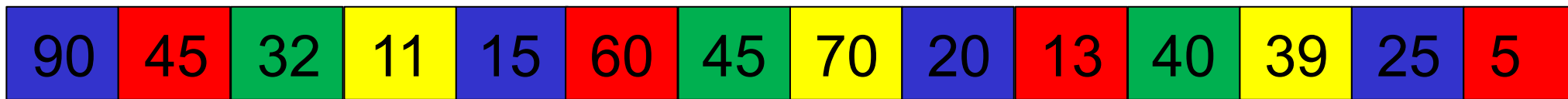
## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

90	45	32	11	15	60	45	70	20	13	40	39	25	10
----	----	----	----	----	----	----	----	----	----	----	----	----	----

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$



Como  $h = 4$ , define-se quatro pseudo *arrays* com as cores azul, vermelha, verde e amarela

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

90 45 32 11 15 60 45 70 20 13 40 39 25 10

Fazendo o algoritmo de inserção nos azuis



## Exemplo

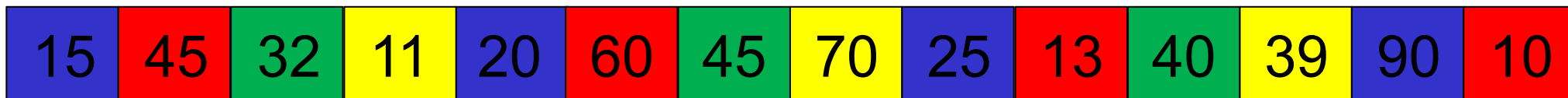
- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15 45 32 11 20 60 45 70 25 13 40 39 90 10

Fazendo o algoritmo de inserção nos azuis

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$



Fazendo o algoritmo de inserção nos vermelhos

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15 45 32 11 20 60 45 70 25 13 40 39 90 10

Fazendo o algoritmo de inserção nos vermelhos

## Exemplo

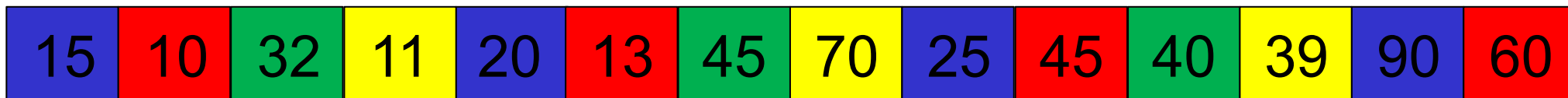
- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15 10 32 11 20 13 45 70 25 45 40 39 90 60

Fazendo o algoritmo de inserção nos vermelhos

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$



Fazendo o algoritmo de inserção nos verdes

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15   10   32   11   20   13   45   70   25   45   40   39   90   60

Fazendo o algoritmo de inserção nos verdes

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15 10 32 11 20 13 40 70 25 45 45 39 90 60

Fazendo o algoritmo de inserção nos verdes

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15 10 32 11 20 13 40 70 25 45 45 39 90 60

Fazendo o algoritmo de inserção nos amarelos



## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15 10 32 11 20 13 40 39 25 45 45 70 90 60

Fazendo o algoritmo de inserção nos amarelos

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15	10	32	11	20	13	40	39	25	45	45	70	90	60
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Com  $h = 4$ , o *array* não está ordenado, mas os números estão mais próximos de suas posições

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15	10	32	11	20	13	40	39	25	45	45	70	90	60
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Fazendo  $h = 2$ , define-se dois pseudo *arrays* com as cores azul e vermelha

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15	10	32	11	20	13	40	39	25	45	45	70	90	60
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Fazendo o algoritmo de inserção nos azuis

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15 10 32 11 20 13 40 39 25 45 45 70 90 60

Fazendo o algoritmo de inserção nos azuis

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15 10 20 11 25 13 32 39 40 45 45 70 90 60

Fazendo o algoritmo de inserção nos azuis

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15	10	32	11	20	13	40	39	25	45	45	70	90	60
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Com  $h = 4$ , o *array* não está ordenado, mas os números estão mais próximos de suas posições

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15	10	20	11	25	13	32	39	40	45	45	70	90	60
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Fazendo o algoritmo de inserção nos vermelhos



## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15 10 20 11 25 13 32 39 40 45 45 60 90 70

Fazendo o algoritmo de inserção nos vermelhos

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15	10	20	11	25	13	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

15	10	20	11	25	13	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

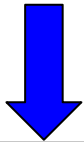
10	15	20	11	25	13	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----



Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$



10	15	20	11	25	13	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

10	15	11	20	25	13	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----



Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

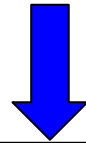
10	11	15	20	25	13	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----



Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$



10	11	15	20	25	13	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Fazendo  $h = 1$ , ou seja, a inserção original



## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

10	11	15	20	13	25	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----



Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

10	11	15	13	20	25	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----



Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

10	11	13	15	20	25	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----



Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

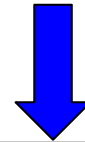


10	11	13	15	20	25	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$



10	11	13	15	20	25	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$



10	11	13	15	20	25	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$




10	11	13	15	20	25	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$




10	11	13	15	20	25	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Fazendo  $h = 1$ , ou seja, a inserção original



## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$




10	11	13	15	20	25	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$



10	11	13	15	20	25	32	39	40	45	45	60	90	70
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Fazendo  $h = 1$ , ou seja, a inserção original

## Exemplo

- Ordene o *array* abaixo com o Shellsort, fazendo  $h = 4, 2$  e  $1$

10	11	13	15	20	25	32	39	40	45	45	60	70	90
----	----	----	----	----	----	----	----	----	----	----	----	----	----




Fazendo  $h = 1$ , ou seja, a inserção original

# Sequência de Passos

- Segundo Knuth (1973, p. 95), de forma empírica, a sequência 1, 4, 13, 40, 121, ... é difícil de ser batida por mais de 20% em eficiência

$$\begin{cases} h(s) = 3h(s-1) + 1 \\ h(1) = 1 \end{cases}$$

- Funcionamento básico
- **Algoritmo em C#** 
- Análise dos número de movimentações e comparações

## Algoritmo em C#

```

void shellsort() {
    int h = 1;
    do { h = (h * 3) + 1; } while (h < n);
    do {
        h /= 3;
        for(int cor = 0; cor < h; cor++){
            insercaoPorCor(cor, h);
        }
    } while (h != 1);
}

void insercaoPorCor(int cor, int h){
    for (int i = (h + cor); i < n; i+=h) {
        int tmp = array[i];
        int j = i - h;
        while ((j >= 0) && (array[j] > tmp)) {
            array[j + h] = array[j];
            j -= h;
        }
        array[j + h] = tmp;
    }
}

```

```

void insercao(){
    for (int i = 1; i < n; i+= 1) {
        int tmp = array[i];
        int j = i - 1;
        while ((j >= 0) && (array[j] > tmp)) {
            array[j + 1] = array[j];
            j -= 1;
        }
        array[j + 1] = tmp;
    }
}

```

- Funcionamento básico
- Algoritmo em C#
- **Análise dos número de movimentações e comparações** 

# Análise do Número de Comparações

- A razão da eficiência do algoritmo ainda não é conhecida
- Sua análise contém alguns problemas matemáticos difíceis, a começar pela própria sequência de incrementos
- O que se sabe é que cada incremento não deve ser múltiplo do anterior
- Conjecturas para o número de comparações dado a seq. de Knuth:
  - Conjetura 1:  $C(n) = \Theta(n^{1,25})$
  - Conjetura 2:  $C(n) = \Theta(n(\ln n)^2)$



- Vantagens:
  - Shellsort é uma ótima opção para arquivos de tamanho moderado
  - Sua implementação é simples e requer pouco código
- Desvantagens:
  - Seu tempo de execução é sensível à ordem inicial do arquivo
  - Algoritmo não estável

## Exercício

- Mostre todas as comparações e movimentações do algoritmo anterior para o *array* abaixo:

12	4	8	2	14	17	6	18	10	16	15	5	13	9	1	11	7	3
----	---	---	---	----	----	---	----	----	----	----	---	----	---	---	----	---	---