# 9002 — Aula 18
# Algoritmos e Programação de Computadores

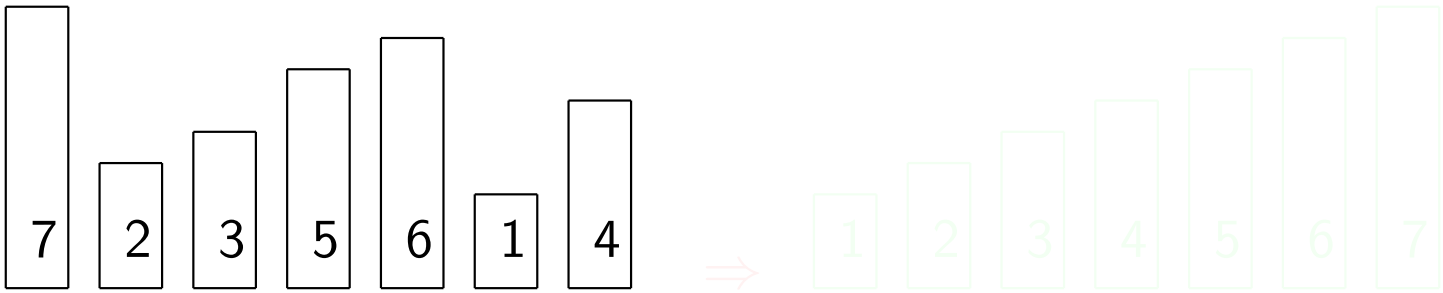
Instituto de Engenharia – UFMT

Segundo Semestre de 2014

25 de novembro de 2014

# Roteiro

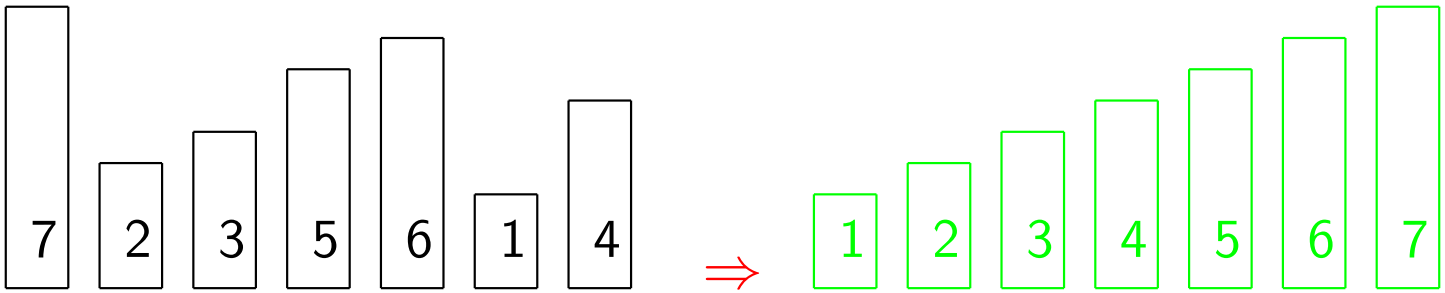
# Introdução




## Problema

Escreva um programa que recebe uma lista de números inteiros e imprima-os em ordem crescente.

# Introdução




## Problema

Escreva um programa que recebe uma lista de números inteiros e imprima-os em ordem crescente.

# Ordenação

## Ordenação

- Vamos estudar algoritmos para ordenar conjuntos de elementos
- Os elementos podem ser de qualquer tipo que possamos comparar:
  - números inteiros,
  - nomes de pessoas,
  - times de futebol... :)
- Os algoritmos podem ordenar crescente ou decrescentemente, dependendo da direção da comparação.

## Estratégias

Existem várias estratégias para ordenar:

- Selecionar o menor a cada vez e colocar na ponta
- Trocar itens fora de ordem
- Outras?

Estratégias diferentes levam a algoritmos diferentes.

# Ordenação

## Ordenação

- Vamos estudar algoritmos para ordenar conjuntos de elementos
- Os elementos podem ser de qualquer tipo que possamos comparar:
  - números inteiros,
  - nomes de pessoas,
  - times de futebol... :)
- Os algoritmos podem ordenar crescente ou decrescentemente, dependendo da direção da comparação.

## Estratégias

Existem várias estratégias para ordenar:

- Selecionar o menor a cada vez e colocar na ponta
- Trocar itens fora de ordem
- Outras?

Estratégias diferentes levam a algoritmos diferentes.

# Ordenação

## Ordenação

- Vamos estudar algoritmos para ordenar conjuntos de elementos
- Os elementos podem ser de qualquer tipo que possamos comparar:
  - números inteiros,
  - nomes de pessoas,
  - times de futebol... :)
- Os algoritmos podem ordenar crescente ou decrescentemente, dependendo da direção da comparação.

## Estratégias

Existem várias estratégias para ordenar:

- Selecionar o menor a cada vez e colocar na ponta
- Trocar itens fora de ordem
- Outras?

Estratégias diferentes levam a algoritmos diferentes.

# Ordenação

## Ordenação

- Vamos estudar algoritmos para ordenar conjuntos de elementos
- Os elementos podem ser de qualquer tipo que possamos comparar:
  - números inteiros,
  - nomes de pessoas,
  - times de futebol... :)
- Os algoritmos podem ordenar crescente ou decrescentemente, dependendo da direção da comparação.

## Estratégias

Existem várias estratégias para ordenar:

- Selecionar o menor a cada vez e colocar na ponta
- Trocar itens fora de ordem
- Outras?

Estratégias diferentes levam a algoritmos diferentes.

# Ordenação

## Ordenação

- Vamos estudar algoritmos para ordenar conjuntos de elementos
- Os elementos podem ser de qualquer tipo que possamos comparar:
  - números inteiros,
  - nomes de pessoas,
  - times de futebol... :)
- Os algoritmos podem ordenar crescente ou decrescentemente, dependendo da direção da comparação.

## Estratégias

Existem várias estratégias para ordenar:

- Selecionar o menor a cada vez e colocar na ponta
- Trocar itens fora de ordem
- Outras?

Estratégias diferentes levam a algoritmos diferentes.

# Ordenação

## Ordenação

- Vamos estudar algoritmos para ordenar conjuntos de elementos
- Os elementos podem ser de qualquer tipo que possamos comparar:
  - números inteiros,
  - nomes de pessoas,
  - times de futebol... :)
- Os algoritmos podem ordenar crescente ou decrescentemente, dependendo da direção da comparação.

## Estratégias

Existem várias estratégias para ordenar:

- Selecionar o menor a cada vez e colocar na ponta
- Trocar itens fora de ordem
- Outras?

Estratégias diferentes levam a algoritmos diferentes.

# Ordenação

## Ordenação

- Vamos estudar algoritmos para ordenar conjuntos de elementos
- Os elementos podem ser de qualquer tipo que possamos comparar:
  - números inteiros,
  - nomes de pessoas,
  - times de futebol... :)
- Os algoritmos podem ordenar crescente ou decrescentemente, dependendo da direção da comparação.

## Estratégias

Existem várias estratégias para ordenar:

- Selecionar o menor a cada vez e colocar na ponta
- Trocar itens fora de ordem
- Outras?

Estratégias diferentes levam a algoritmos diferentes.

# Ordenação

## Ordenação

- Vamos estudar algoritmos para ordenar conjuntos de elementos
- Os elementos podem ser de qualquer tipo que possamos comparar:
  - números inteiros,
  - nomes de pessoas,
  - times de futebol... :)
- Os algoritmos podem ordenar crescente ou decrescentemente, dependendo da direção da comparação.

## Estratégias

Existem várias estratégias para ordenar:

- Selecionar o menor a cada vez e colocar na ponta
- Trocar itens fora de ordem
- Outras?

Estratégias diferentes levam a algoritmos diferentes.

# Ordenação

## Ordenação

- Vamos estudar algoritmos para ordenar conjuntos de elementos
- Os elementos podem ser de qualquer tipo que possamos comparar:
  - números inteiros,
  - nomes de pessoas,
  - times de futebol... :)
- Os algoritmos podem ordenar crescente ou decrescentemente, dependendo da direção da comparação.

## Estratégias

Existem várias estratégias para ordenar:

- Selecionar o menor a cada vez e colocar na ponta
- Trocar itens fora de ordem
- Outras?

Estratégias diferentes levam a algoritmos diferentes.

# Ordenação

## Ordenação

- Vamos estudar algoritmos para ordenar conjuntos de elementos
- Os elementos podem ser de qualquer tipo que possamos comparar:
  - números inteiros,
  - nomes de pessoas,
  - times de futebol... :)
- Os algoritmos podem ordenar crescente ou decrescentemente, dependendo da direção da comparação.

## Estratégias

Existem várias estratégias para ordenar:

- Selecionar o menor a cada vez e colocar na ponta
- Trocar itens fora de ordem
- Outras?

Estratégias diferentes levam a algoritmos diferentes.

# Ordenação

## Ordenação

- Vamos estudar algoritmos para ordenar conjuntos de elementos
- Os elementos podem ser de qualquer tipo que possamos comparar:
  - números inteiros,
  - nomes de pessoas,
  - times de futebol... :)
- Os algoritmos podem ordenar crescente ou decrescentemente, dependendo da direção da comparação.

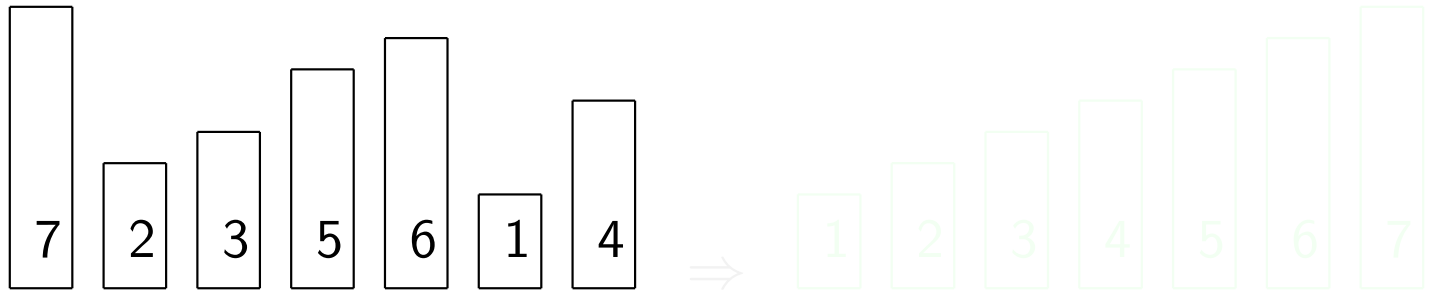
## Estratégias

Existem várias estratégias para ordenar:

- Selecionar o menor a cada vez e colocar na ponta
- Trocar itens fora de ordem
- Outras?

Estratégias diferentes levam a algoritmos diferentes.

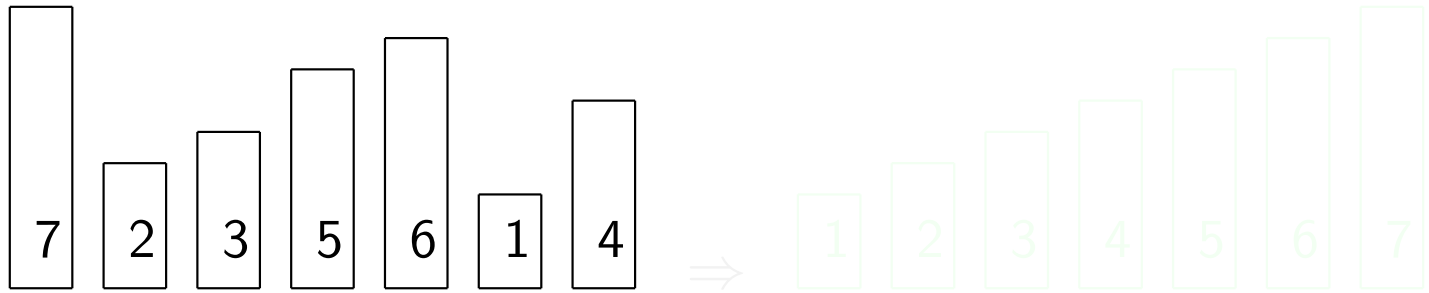
# Ordenação por seleção




**Ideia**

- Inicialmente temos uma lista de itens desordenados
- Selecionamos o menor elemento
- Movemos o item para uma nova lista
- Repetimos tudo com a lista restante (preta)

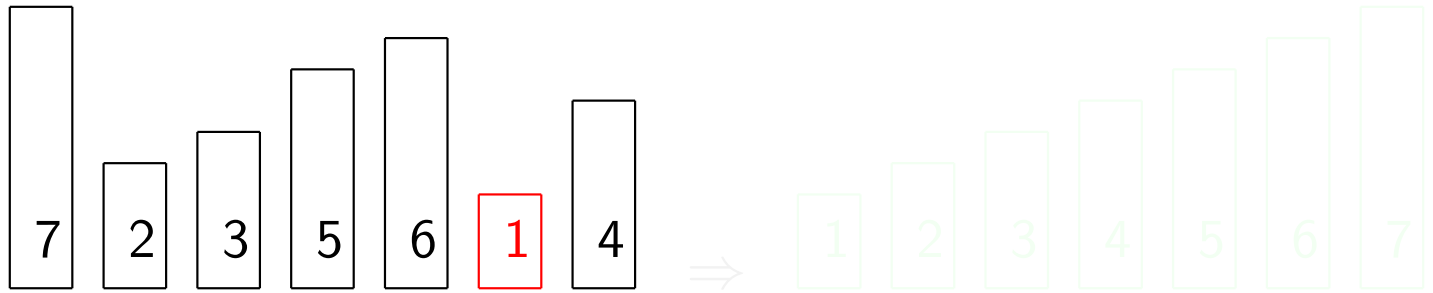
# Ordenação por seleção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Selecionamos o menor elemento
- Movemos o item para uma nova lista
- Repetimos tudo com a lista restante (preta)

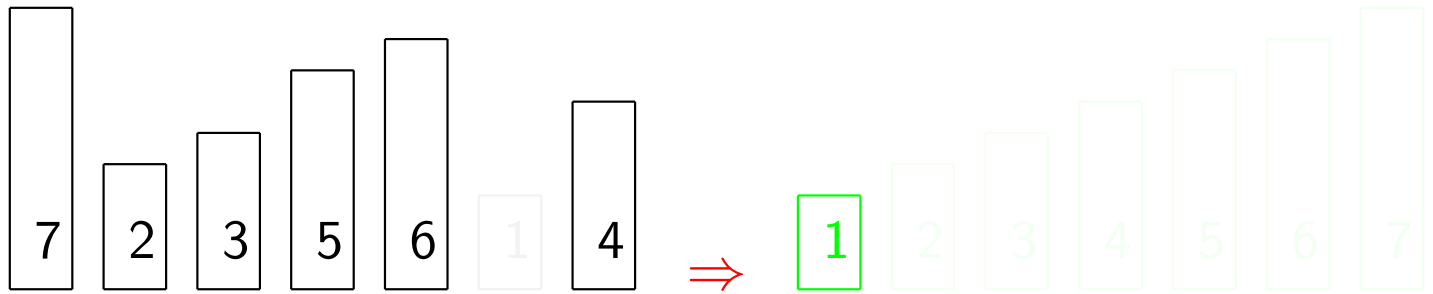
# Ordenação por seleção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Selecionamos o menor elemento
- Movemos o item para uma nova lista
- Repetimos tudo com a lista restante (preta)

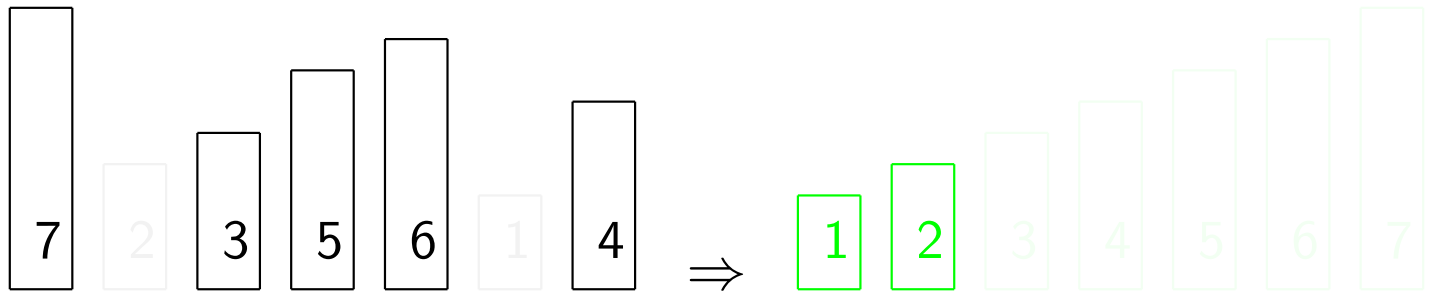
# Ordenação por seleção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Selecionamos o menor elemento
- Movemos o item para uma nova lista
- Repetimos tudo com a lista restante (preta)

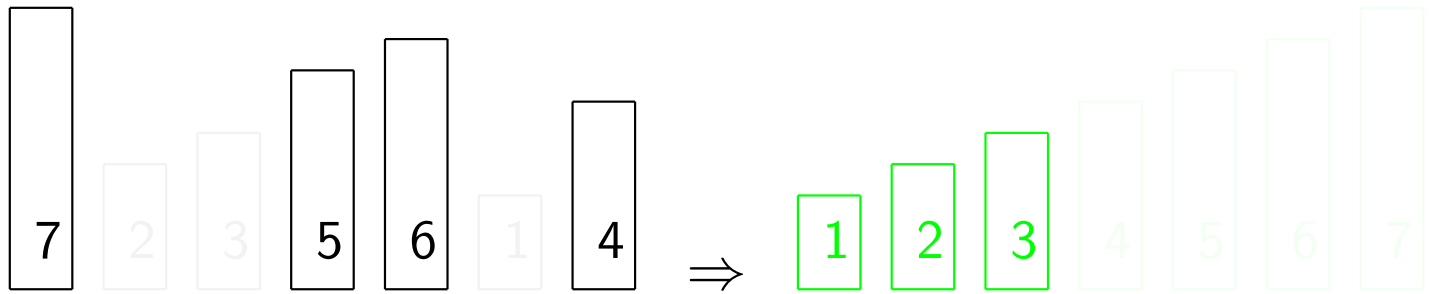
# Ordenação por seleção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Selecionamos o menor elemento
- Movemos o item para uma nova lista
- Repetimos tudo com a lista restante (preta)

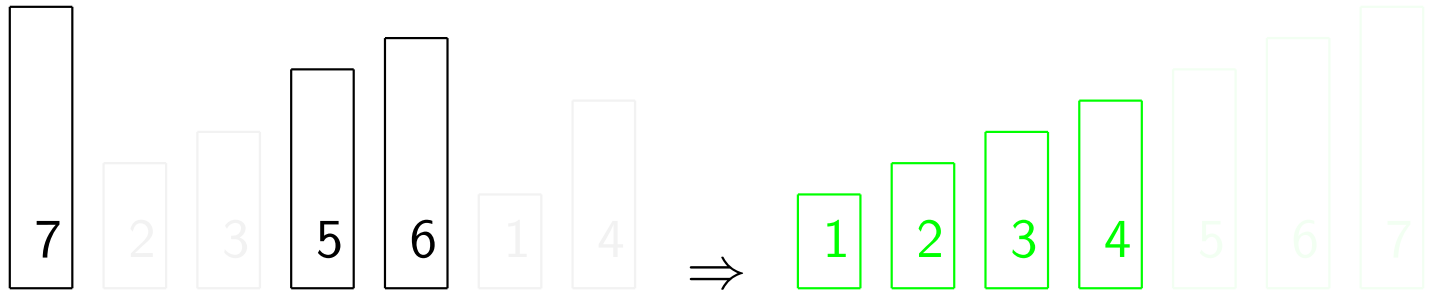
# Ordenação por seleção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Selecionamos o menor elemento
- Movemos o item para uma nova lista
- Repetimos tudo com a lista restante (preta)

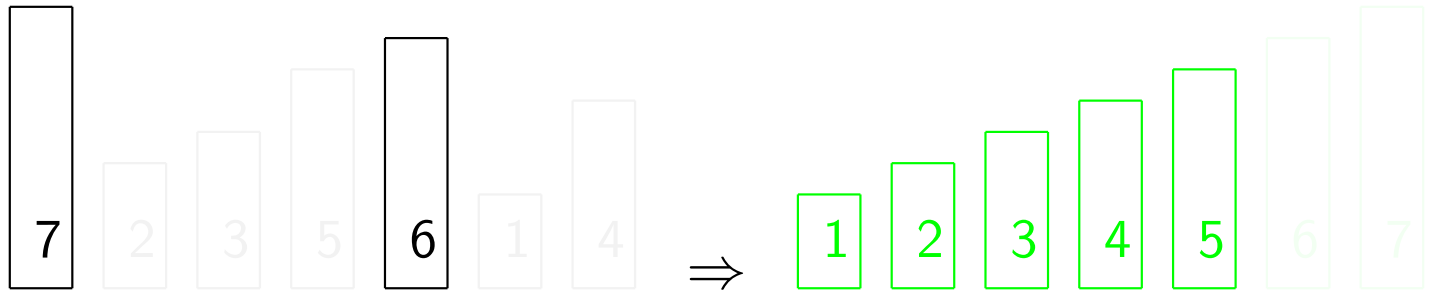
# Ordenação por seleção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Selecionamos o menor elemento
- Movemos o item para uma nova lista
- Repetimos tudo com a lista restante (preta)

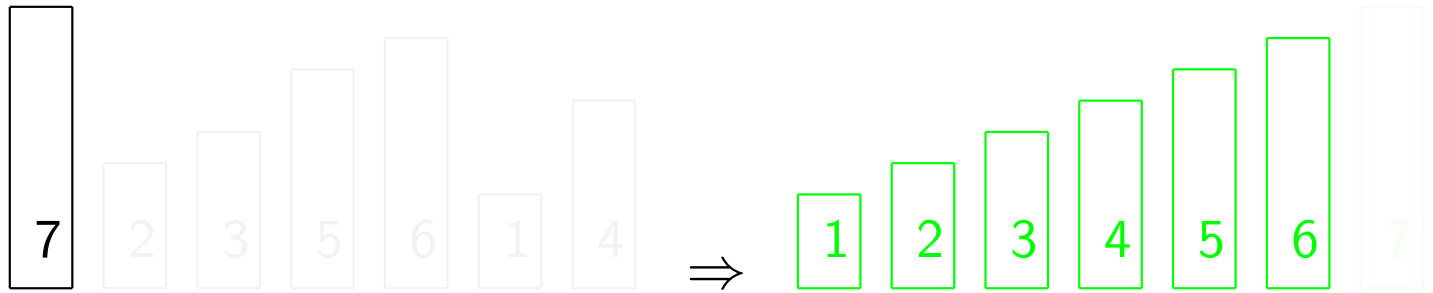
# Ordenação por seleção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Selecionamos o menor elemento
- Movemos o item para uma nova lista
- Repetimos tudo com a lista restante (preta)

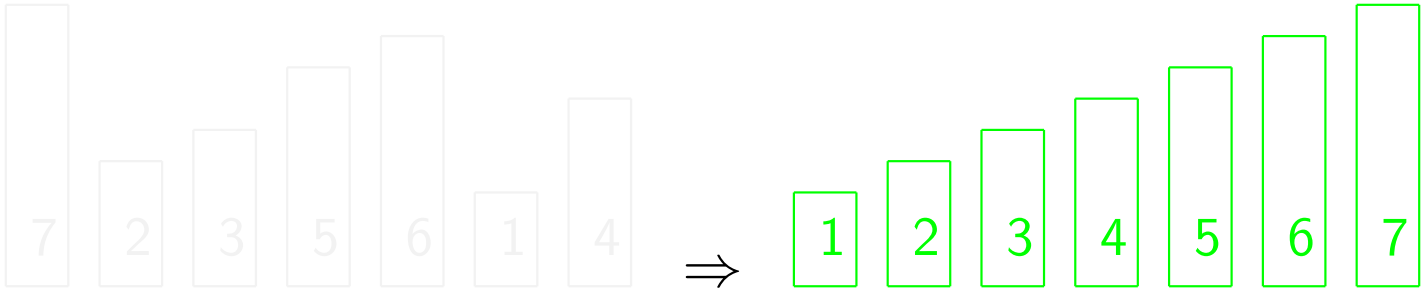
# Ordenação por seleção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Selecionamos o menor elemento
- Movemos o item para uma nova lista
- Repetimos tudo com a lista restante (preta)

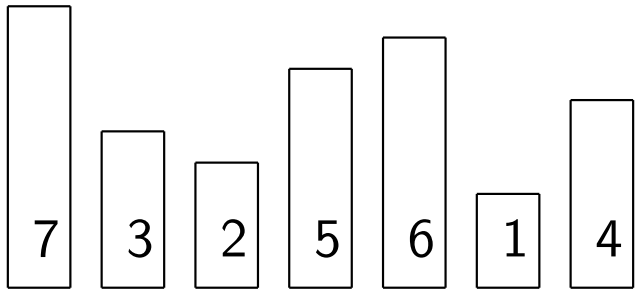
# Ordenação por seleção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Selecionamos o menor elemento
- Movemos o item para uma nova lista
- Repetimos tudo com a lista restante (preta)

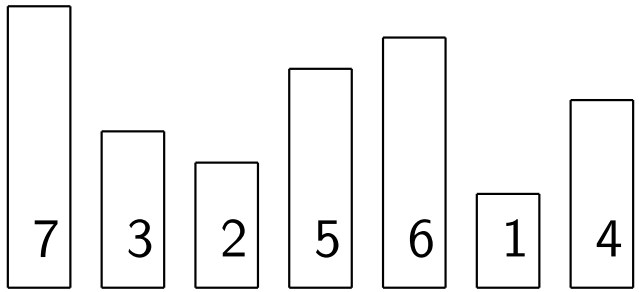
# Ordenação por seleção - Melhorando




**Ideia**

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

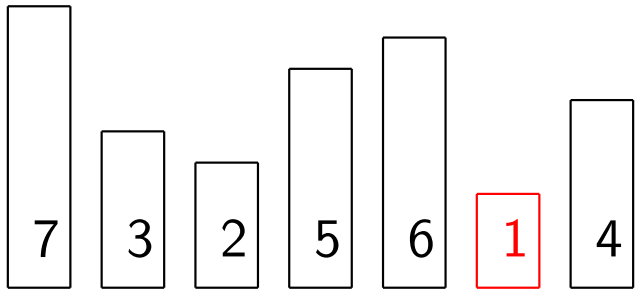
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

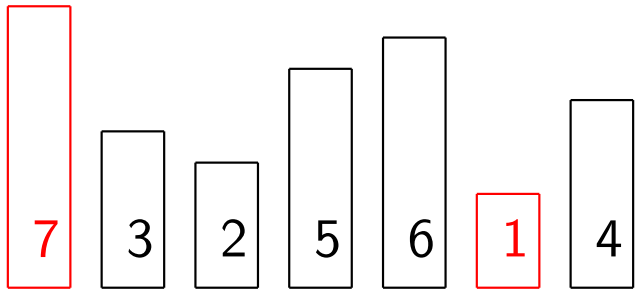
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

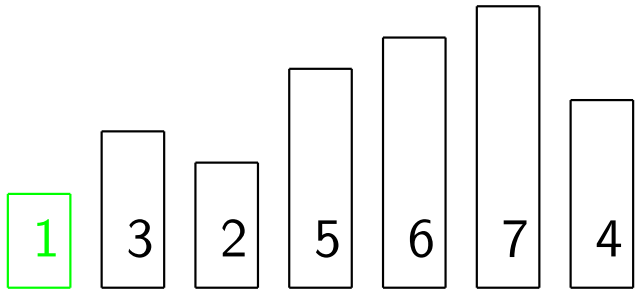
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

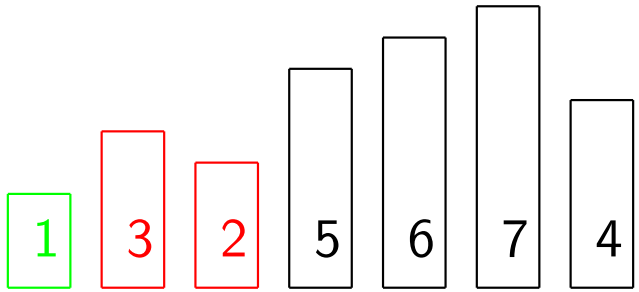
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

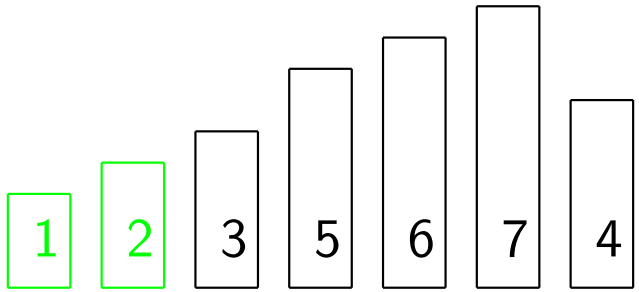
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

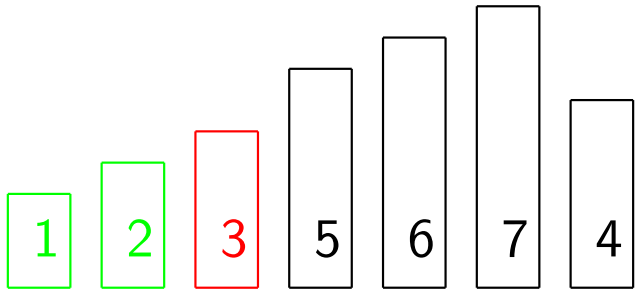
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

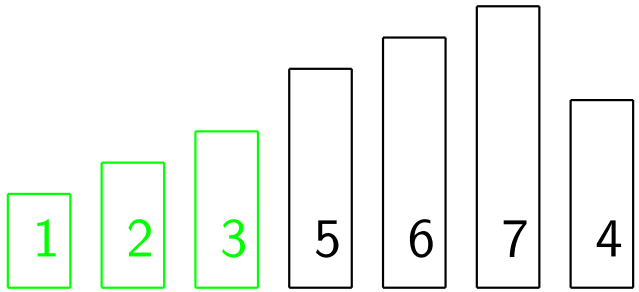
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

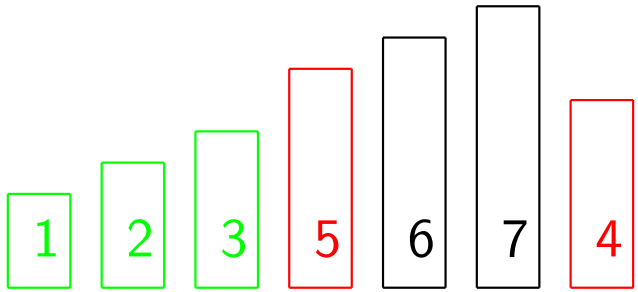
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

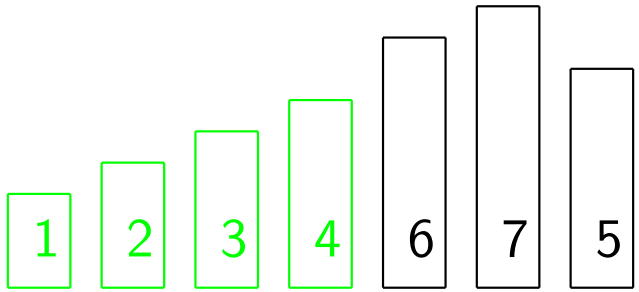
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

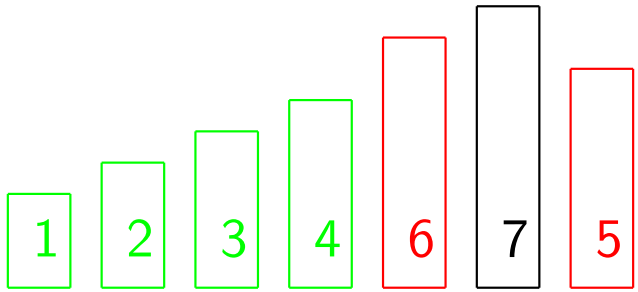
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

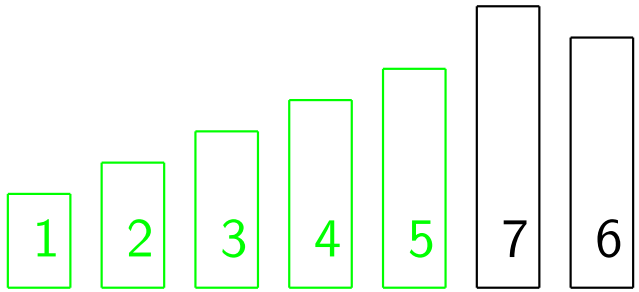
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

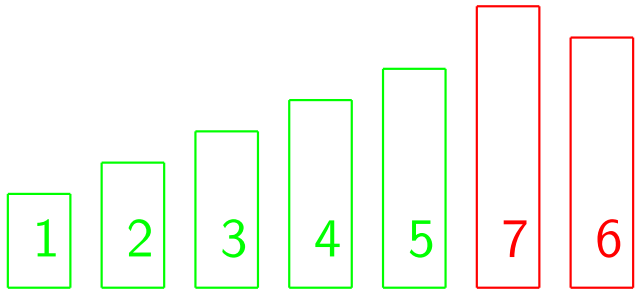
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

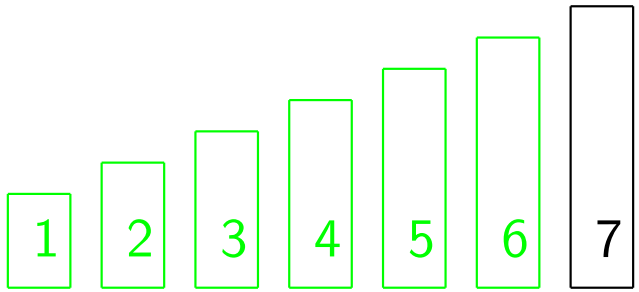
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

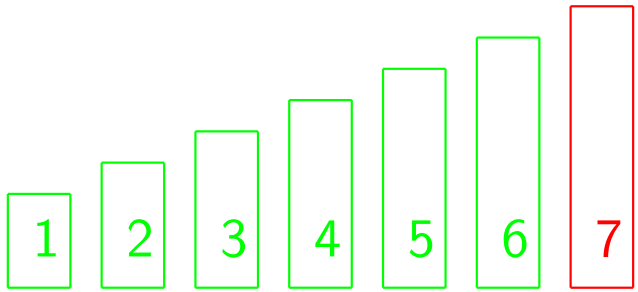
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

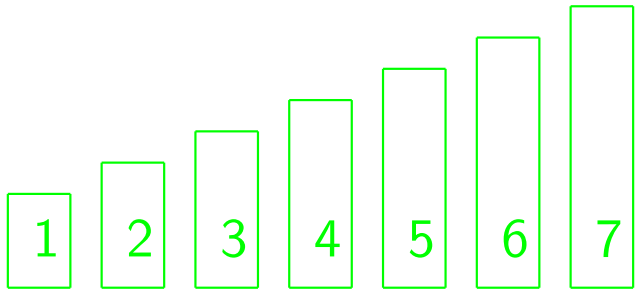
# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

# Ordenação por seleção - Melhorando




## Ideia

- Não precisamos de uma nova lista! Basta:
  1. Selecionar o menor elemento
  2. Trocar com o primeiro elemento da lista
  3. Continuar com a lista restante (preta)

# Convenções

## Algoritmo de ordenação

- Vamos implementar uma função para ordenar inteiros
- A função terá os seguintes parâmetros:
  - `int vetor[]`: vetor de inteiros onde os elementos estão
  - `int n`: o número de elemento do vetor
- A função deverá ordenar o vetor passado crescentemente

Para trocar dois valores, vamos sempre usar a seguinte função:

### Trocar dois valores inteiros

```
void trocar(int *a, int *b) {
    int aux = *a;
    *a = *b;
    *b = aux;
}
```

# Convenções

## Algoritmo de ordenação

- Vamos implementar uma função para ordenar inteiros
- A função terá os seguintes parâmetros:
  - `int vetor[]`: vetor de inteiros onde os elementos estão
  - `int n`: o número de elemento do vetor
- A função deverá ordenar o vetor passado crescentemente

Para trocar dois valores, vamos sempre usar a seguinte função:

### Trocar dois valores inteiros

```
void trocar(int *a, int *b) {
    int aux = *a;
    *a = *b;
    *b = aux;
}
```

# Convenções

## Algoritmo de ordenação

- Vamos implementar uma função para ordenar inteiros
- A função terá os seguintes parâmetros:
  - `int vetor[]`: vetor de inteiros onde os elementos estão
  - `int n`: o número de elemento do vetor
- A função deverá ordenar o vetor passado crescentemente

Para trocar dois valores, vamos sempre usar a seguinte função:

### Trocar dois valores inteiros

```
void trocar(int *a, int *b) {
    int aux = *a;
    *a = *b;
    *b = aux;
}
```

# Convenções

## Algoritmo de ordenação

- Vamos implementar uma função para ordenar inteiros
- A função terá os seguintes parâmetros:
  - `int vetor[]`: vetor de inteiros onde os elementos estão
  - `int n`: o número de elemento do vetor
- A função deverá ordenar o vetor passado crescentemente

Para trocar dois valores, vamos sempre usar a seguinte função:

Trocar dois valores inteiros

```
void trocar(int *a, int *b) {
    int aux = *a;
    *a = *b;
    *b = aux;
}
```

# Convenções

## Algoritmo de ordenação

- Vamos implementar uma função para ordenar inteiros
- A função terá os seguintes parâmetros:
  - `int vetor[]`: vetor de inteiros onde os elementos estão
  - `int n`: o número de elemento do vetor
- A função deverá ordenar o vetor passado crescentemente

Para trocar dois valores, vamos sempre usar a seguinte função:

### Trocar dois valores inteiros

```
void trocar(int *a, int *b) {
    int aux = *a;
    *a = *b;
    *b = aux;
}
```

# Convenções

## Algoritmo de ordenação

- Vamos implementar uma função para ordenar inteiros
- A função terá os seguintes parâmetros:
  - `int vetor[]`: vetor de inteiros onde os elementos estão
  - `int n`: o número de elemento do vetor
- A função deverá ordenar o vetor passado crescentemente

Para trocar dois valores, vamos sempre usar a seguinte função:

## Trocar dois valores inteiros

```
void trocar(int *a, int *b) {
    int aux = *a;
    *a = *b;
    *b = aux;
}
```

# Algoritmo de ordenação por seleção (*Selection-Sort*)

## Menor elemento não ordenado (na lista preta)

```
int menor_elemento(int vetor[], int n, int primeiro) {
    int i, menor = primeiro;
    for (i = primeiro + 1; i < n; i++) {
        if (vetor[i] < vetor[menor])
            menor = i;
    }
    return menor;
}
```

## Ordenação por seleção

```
int ordenar_selecao(int vetor[], int n) {
    int i, menor;
    for (i = 0; i < n; i++) {
        menor = menor_elemento(vetor, n, i);
        trocar(&vetor[i], &vetor[menor]);
    }
}
```

# Algoritmo de ordenação por seleção (*Selection-Sort*)

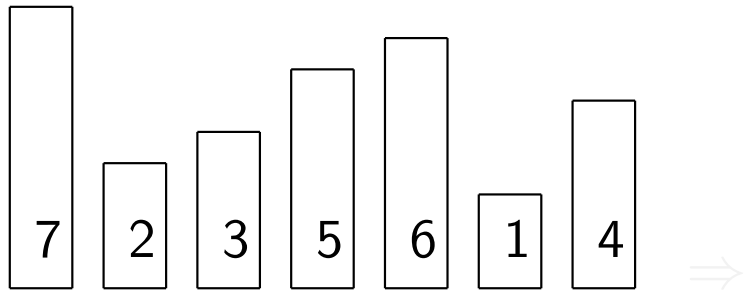
## Menor elemento não ordenado (na lista preta)

```
int menor_elemento(int vetor[], int n, int primeiro) {
    int i, menor = primeiro;
    for (i = primeiro + 1; i < n; i++) {
        if (vetor[i] < vetor[menor])
            menor = i;
    }
    return menor;
}
```

## Ordenação por seleção

```
int ordenar_selecao(int vetor[], int n) {
    int i, menor;
    for (i = 0; i < n; i++) {
        menor = menor_elemento(vetor, n, i);
        trocar(&vetor[i], &vetor[menor]);
    }
}
```

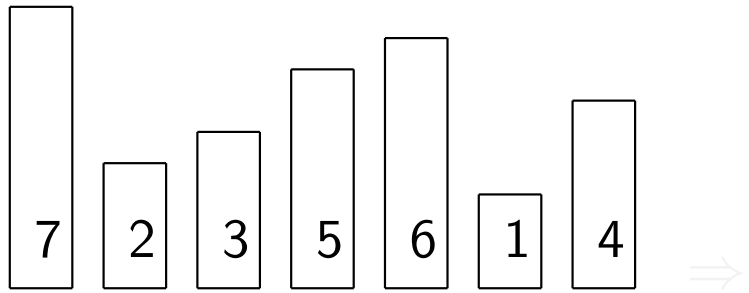
# Ordenação por inserção




**Ideia**

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

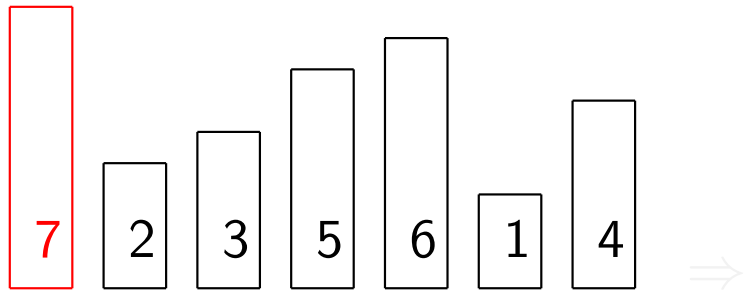
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

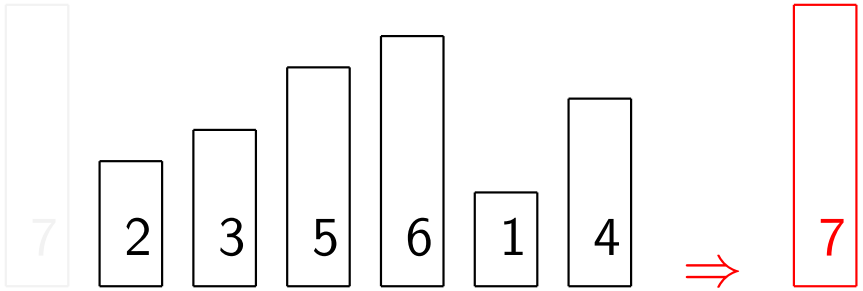
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

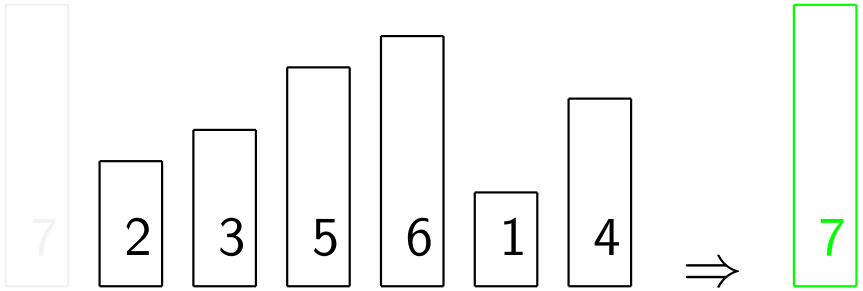
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

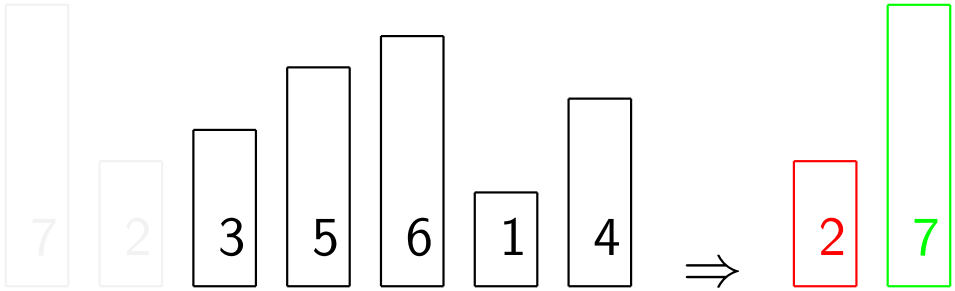
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

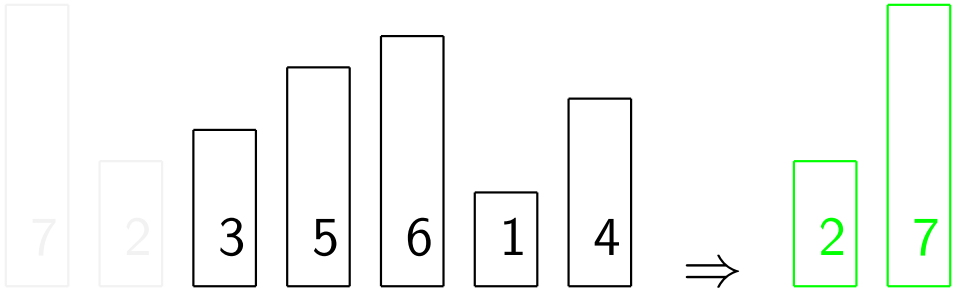
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

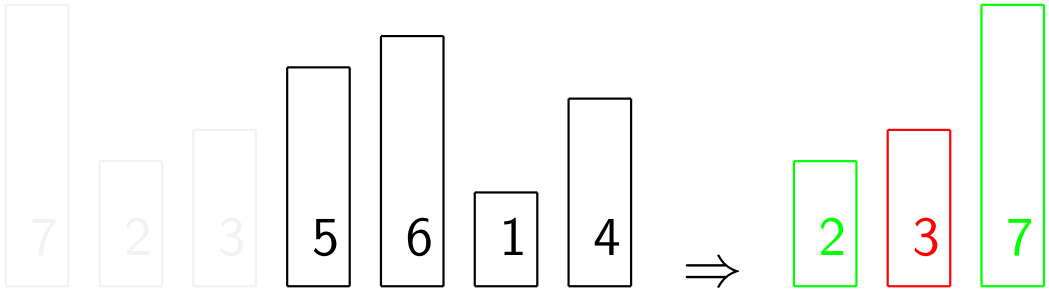
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

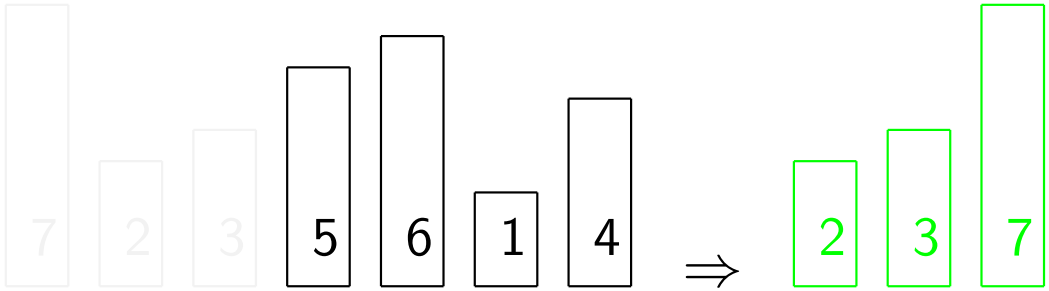
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

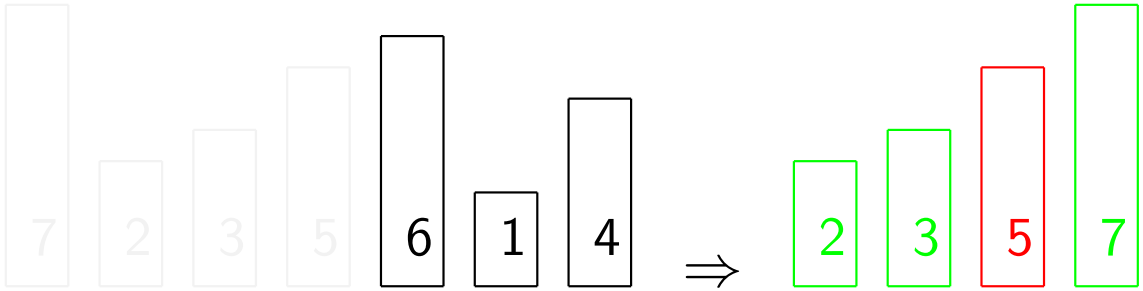
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

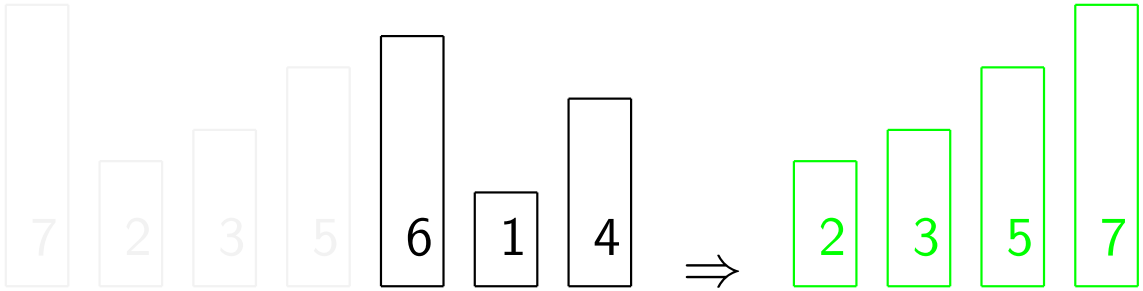
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

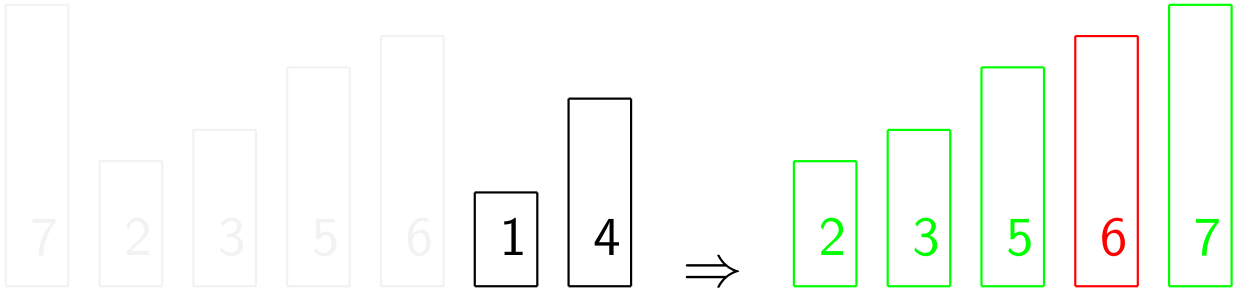
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

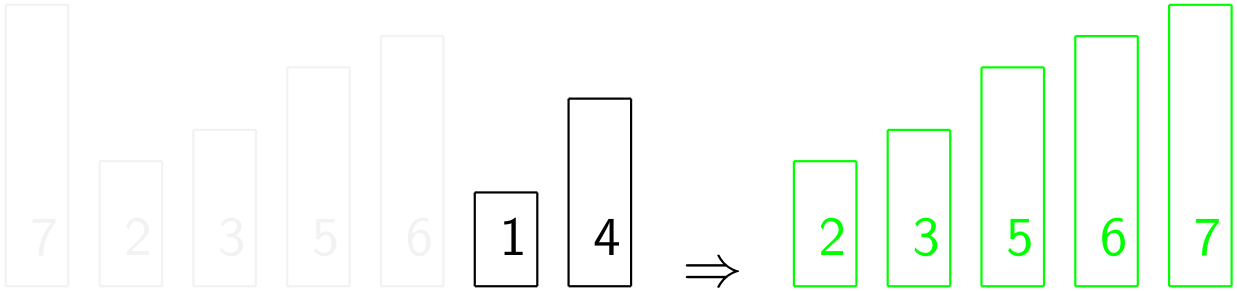
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

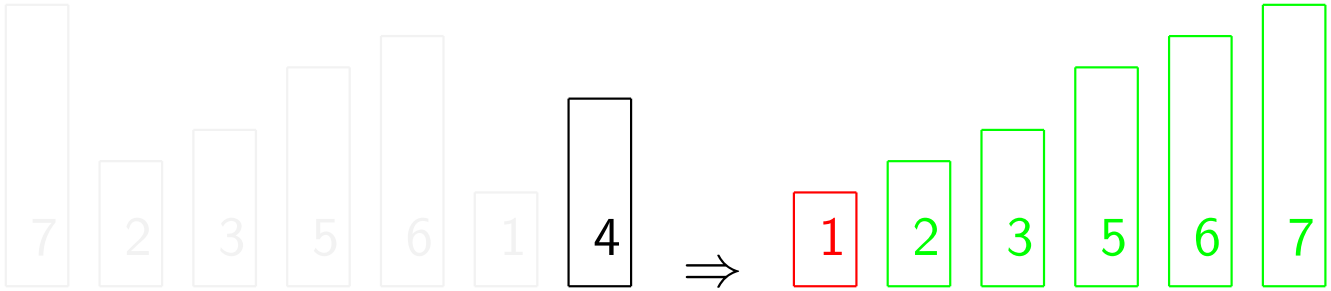
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

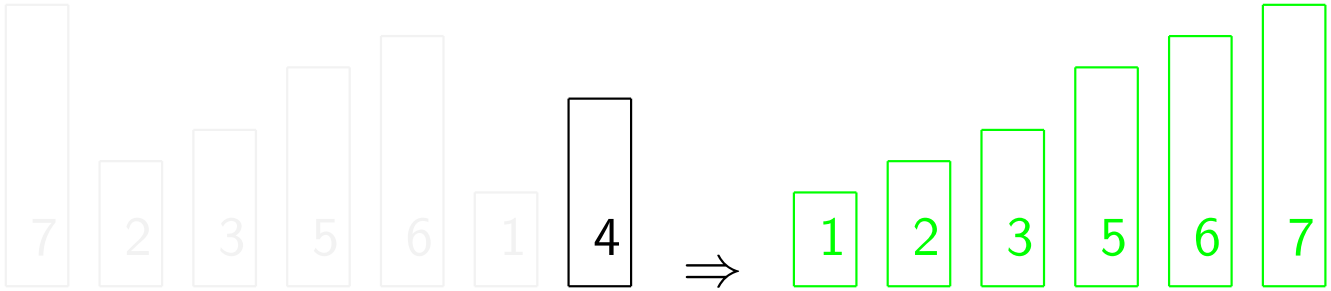
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

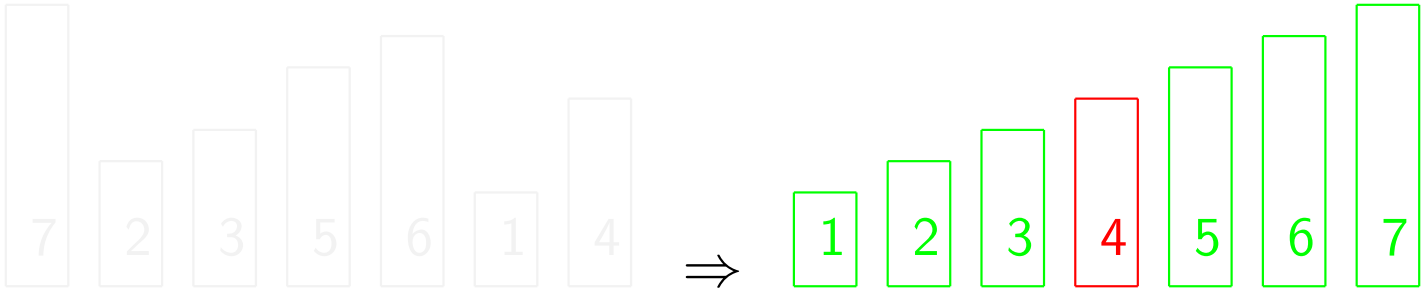
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

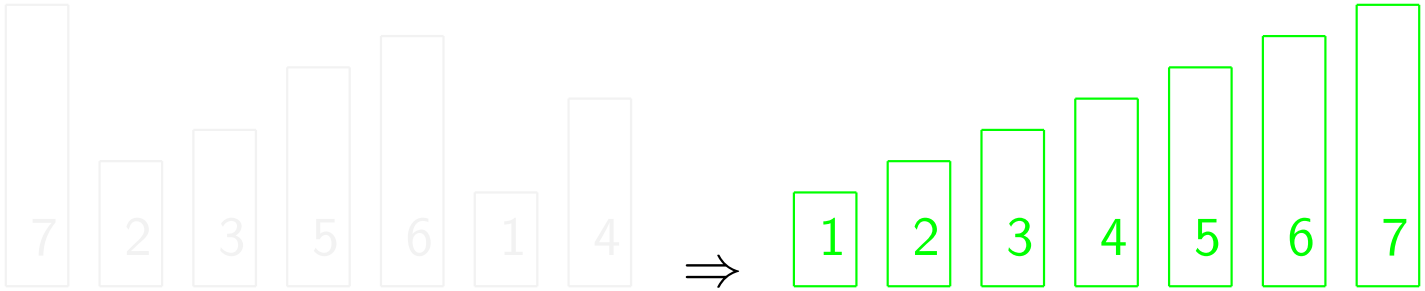
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

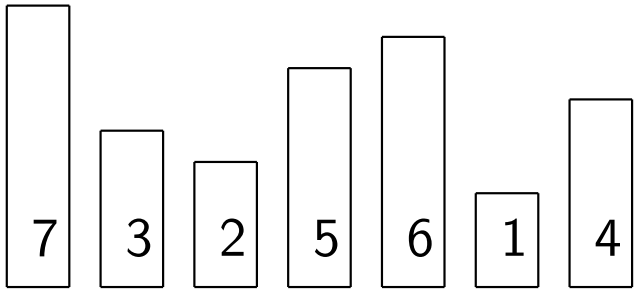
# Ordenação por inserção




## Ideia

- Inicialmente temos uma lista de itens desordenados
- Retiramos o primeiro elemento
- Inserimos este item em uma nova lista na ordem
- Repetimos tudo com a lista restante (preta)

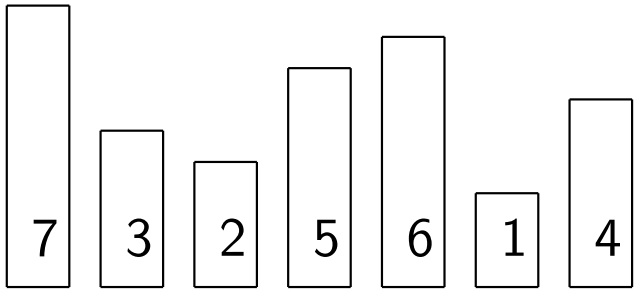
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

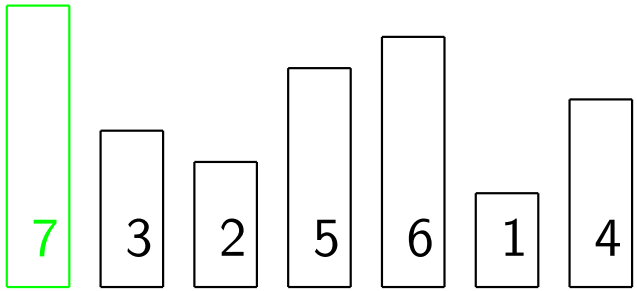
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?

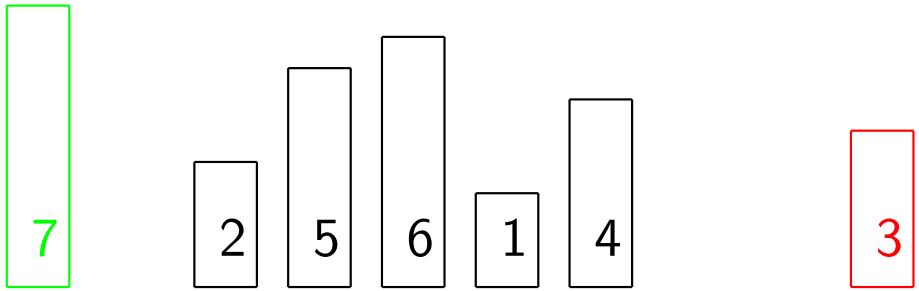
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

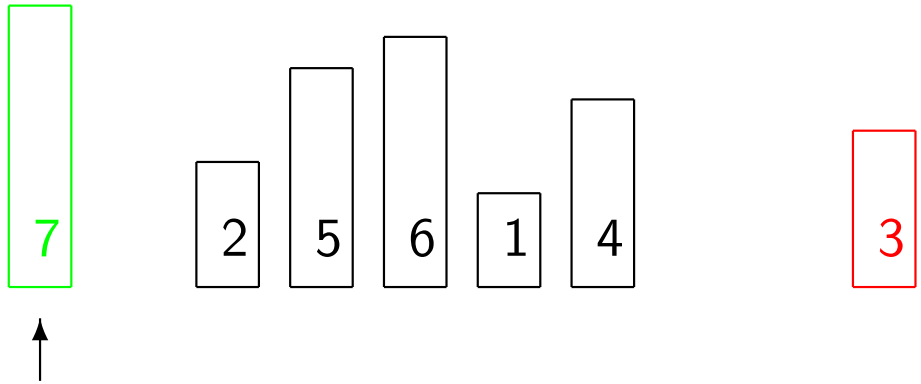
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

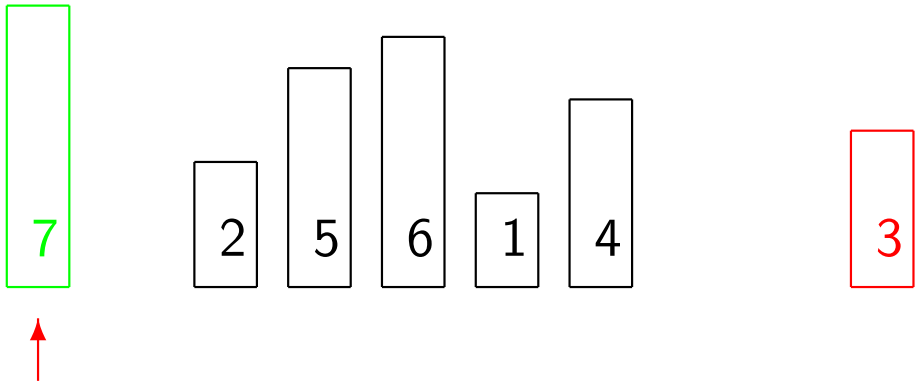
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

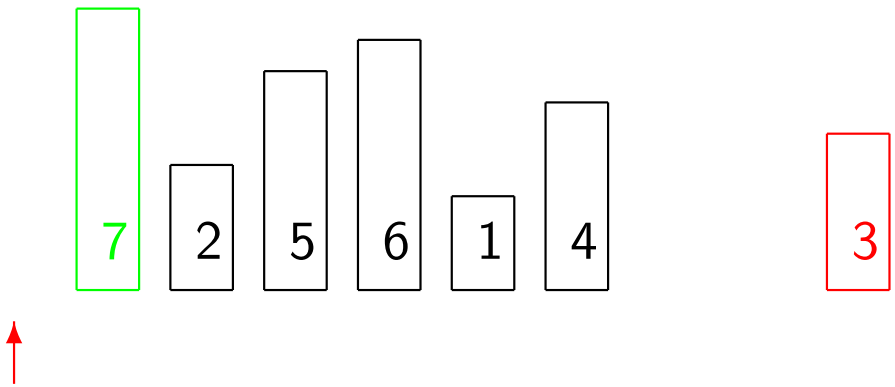
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

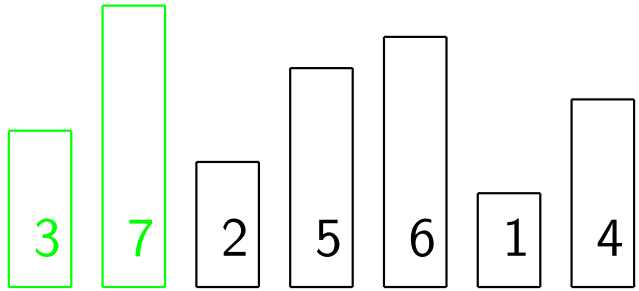
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

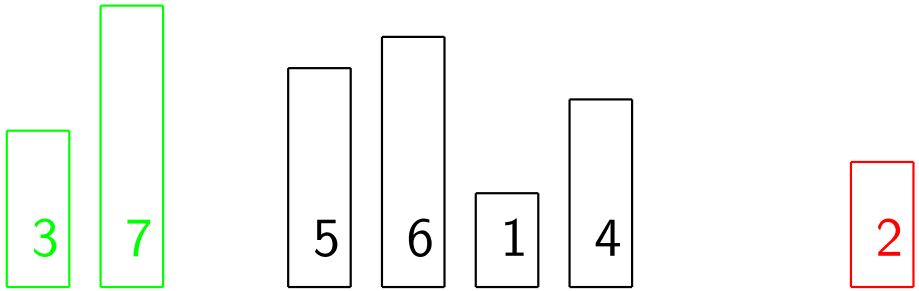
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

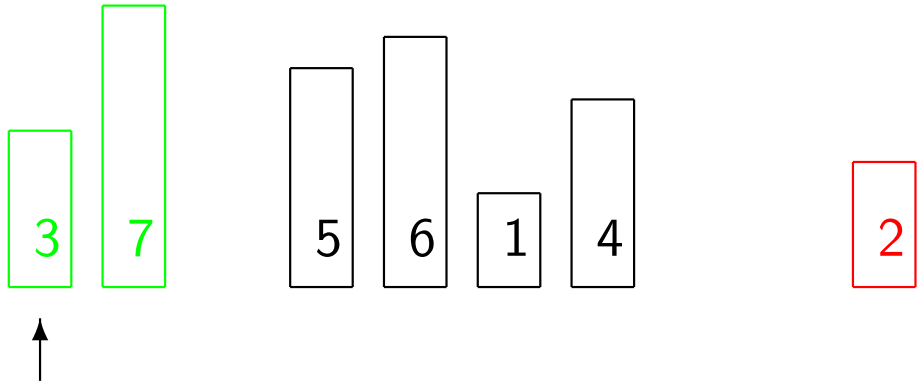
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

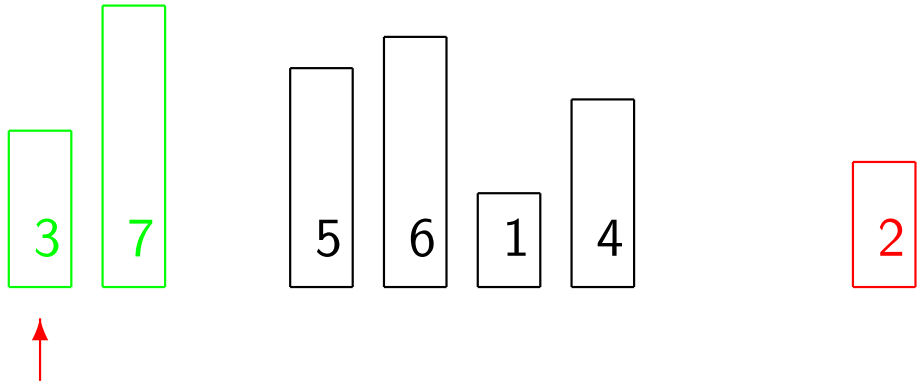
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

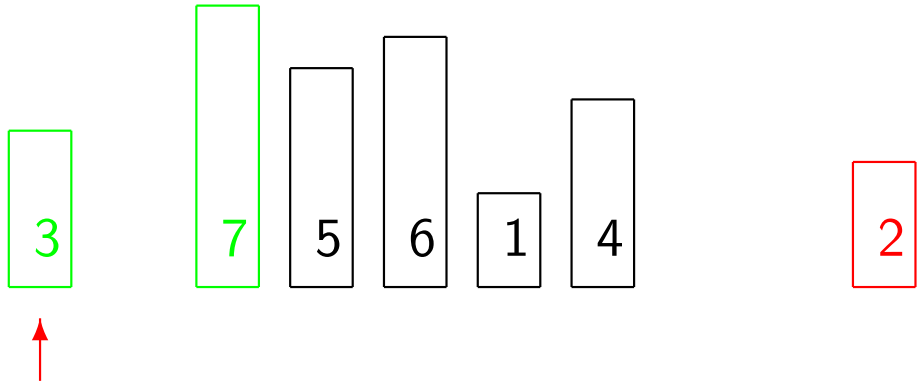
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

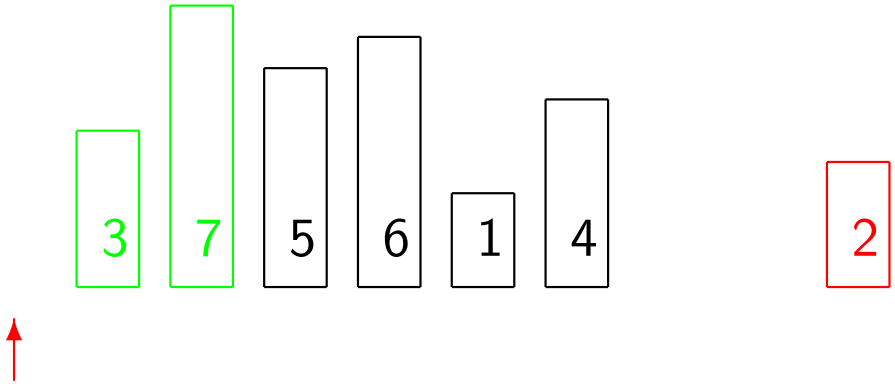
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

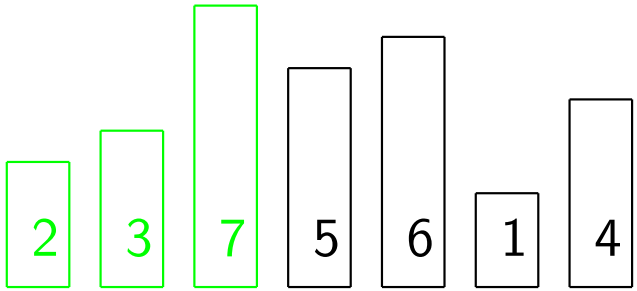
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

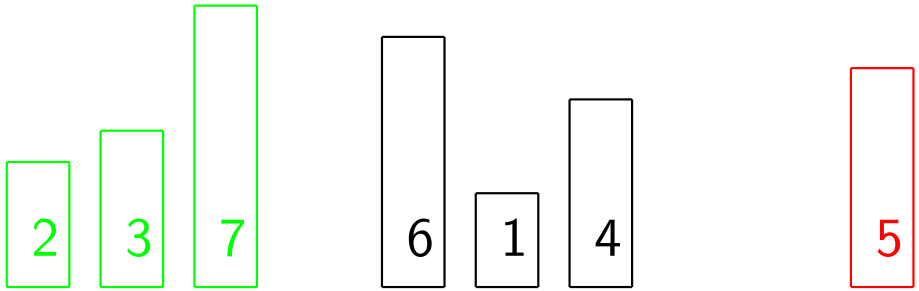
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

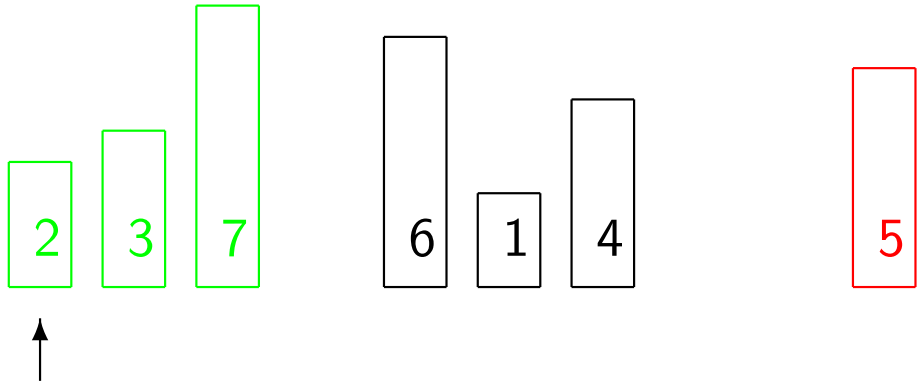
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

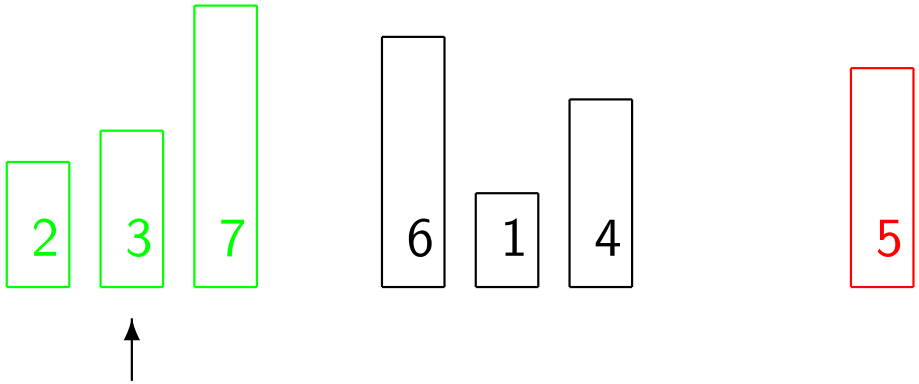
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

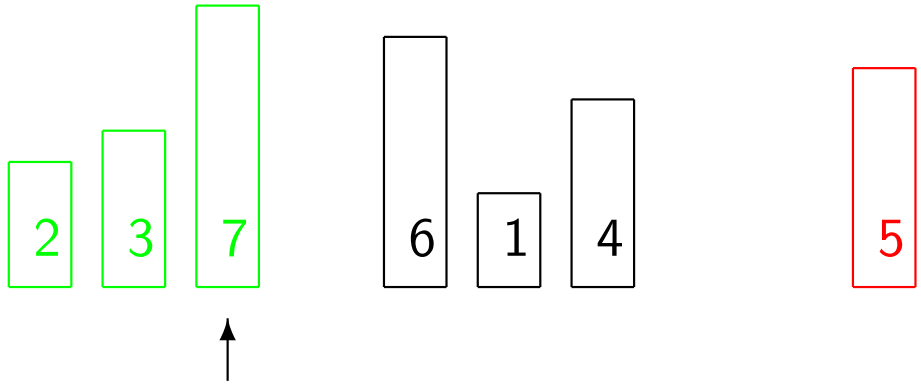
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

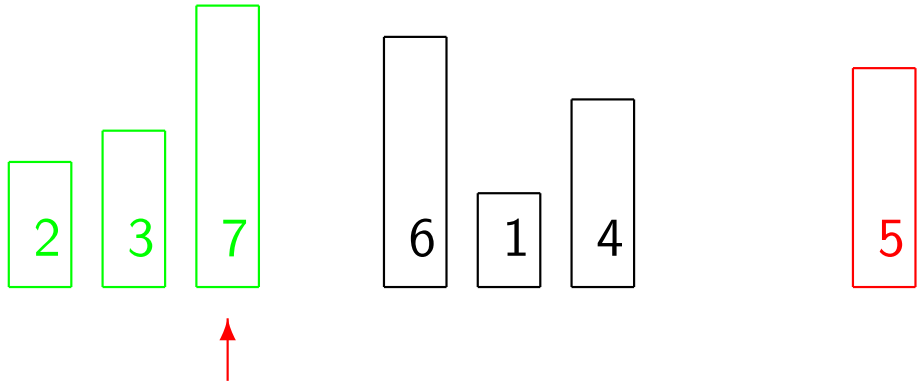
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

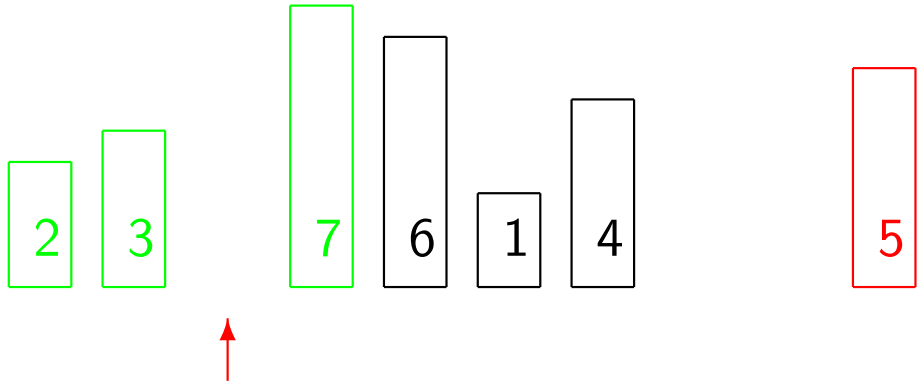
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

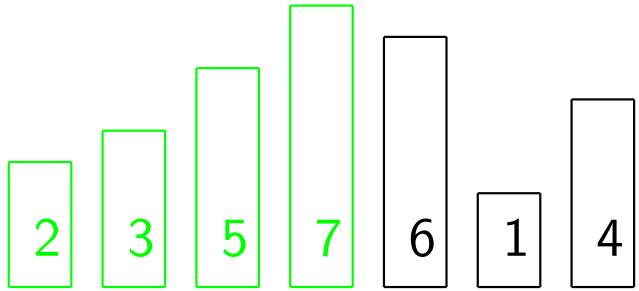
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

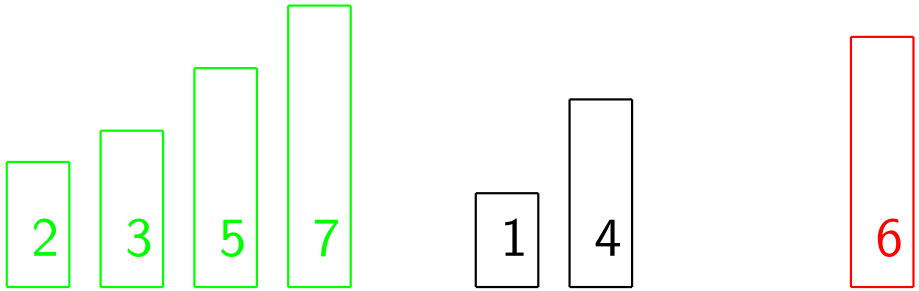
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

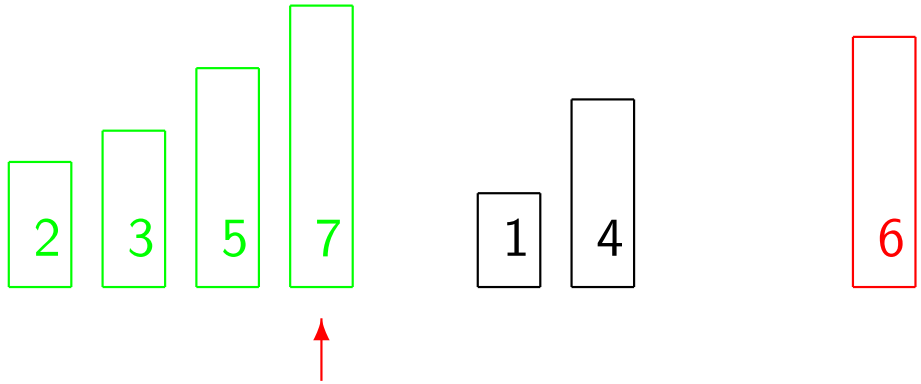
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

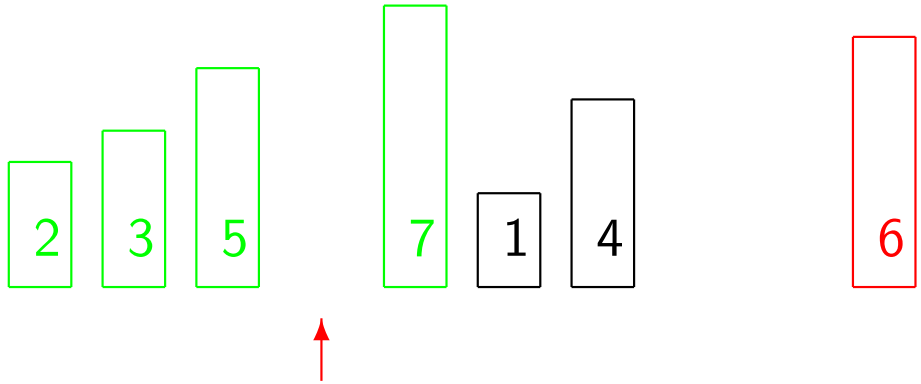
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

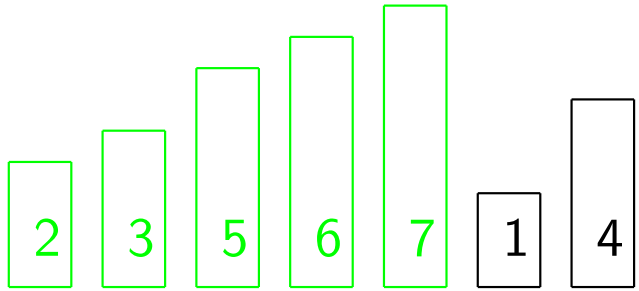
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

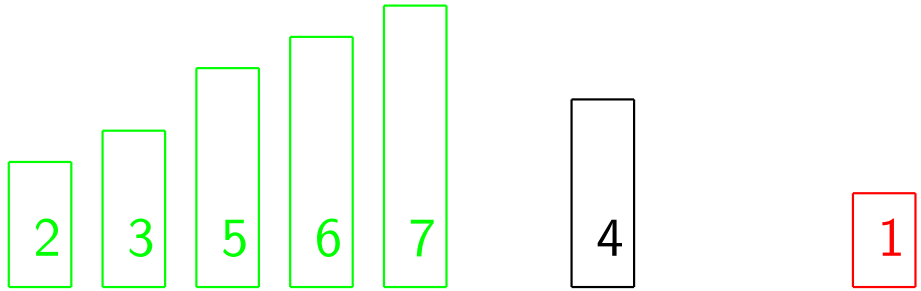
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

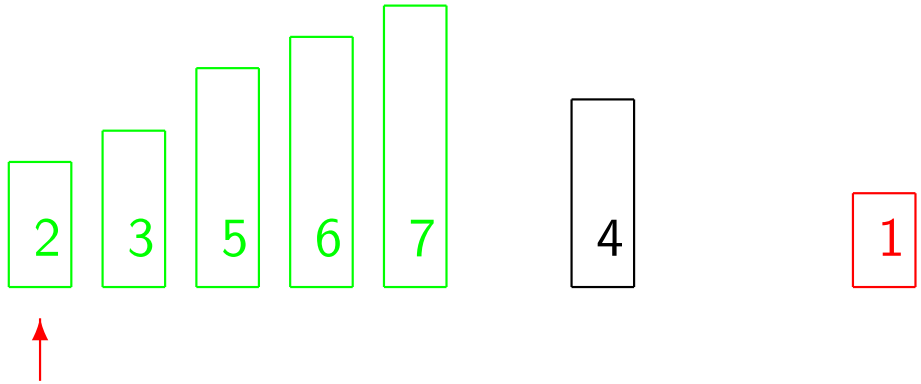
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

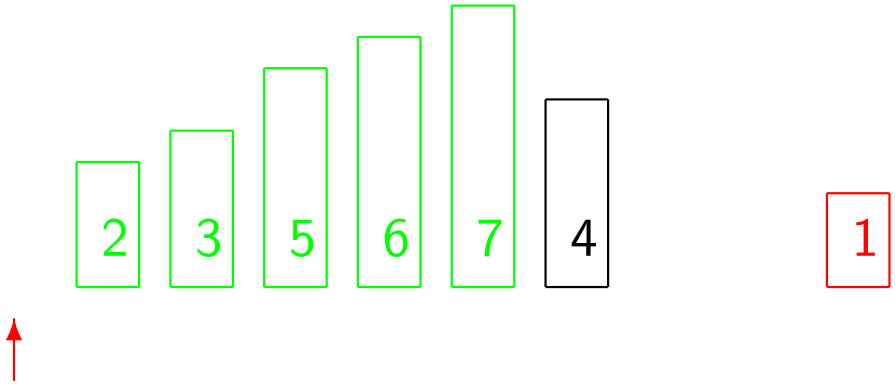
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

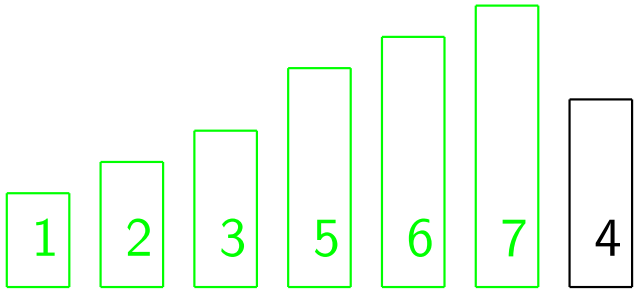
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

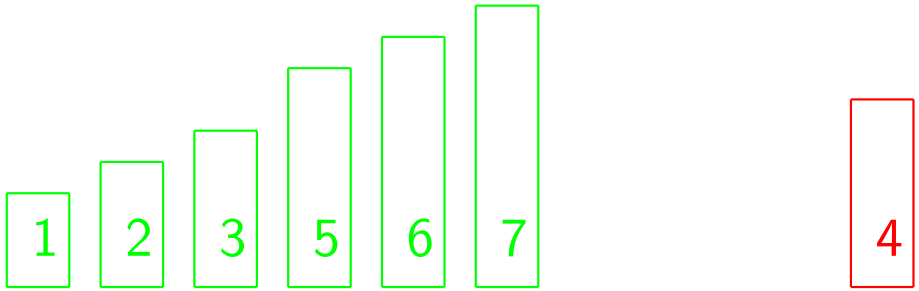
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

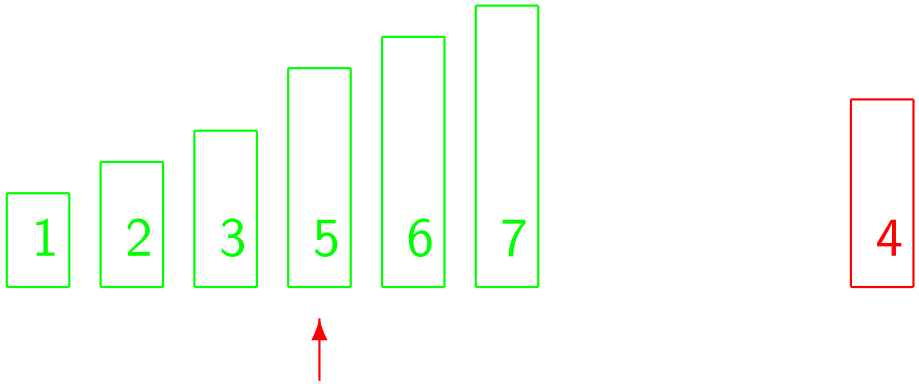
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

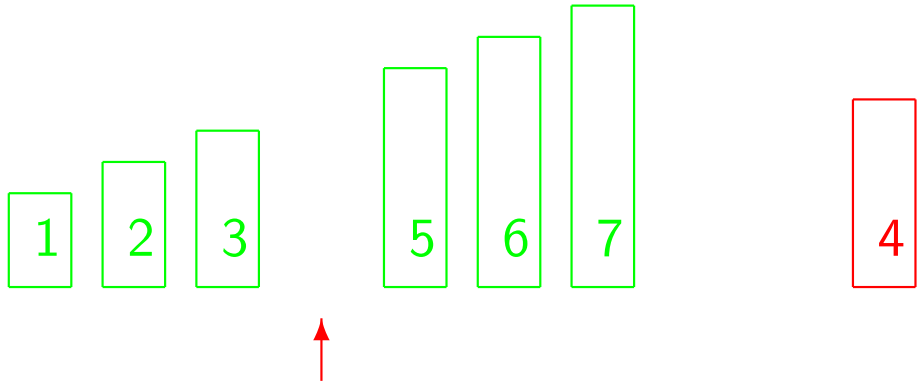
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

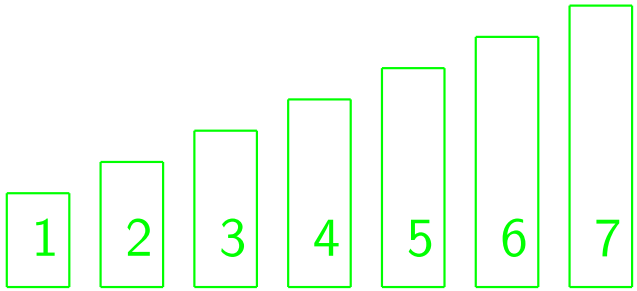
# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

# Ordenação por inserção - No mesmo vetor




## Ideia

- Como usar apenas um vetor?
  1. O primeiro elemento já está ordenado
  2. Retirar o primeiro elemento desordenado
  3. Procurar a posição em que deve ser inserido
  4. Deslocar os elementos ordenados seguintes
  5. Inserir o elemento retirado na ordem correta
  6. Continuar com a lista restante (preta)

# Algoritmo de ordenação por inserção (*Insertion-Sort*)

## Posição de inserção (na lista verde)

```
int posicao_elemento(int vetor[], int ultimo, int elemento) {
    int i;
    for (i = 0; i <= ultimo && vetor[i] <= elemento; i++);
    return i;
}
```

## Deslocar parte do vetor

```
void deslocar_subvetor(int vetor[], int primeiro, int ultimo) {
    int i;
    for (i = ultimo; i >= primeiro; i--) {
        vetor[i+1] = vetor[i];
    }
}
```

# Algoritmo de ordenação por inserção (*Insertion-Sort*)

## Posição de inserção (na lista verde)

```
int posicao_elemento(int vetor[], int ultimo, int elemento) {
    int i;
    for (i = 0; i <= ultimo && vetor[i] <= elemento; i++);
    return i;
}
```

## Deslocar parte do vetor

```
void deslocar_subvetor(int vetor[], int primeiro, int ultimo) {
    int i;
    for (i = ultimo; i >= primeiro; i--) {
        vetor[i+1] = vetor[i];
    }
}
```

# Algoritmo de ordenação por inserção (*Insertion-Sort*)

## Ordenação por inserção

```
int ordenar_insercao(int vetor[], int n) {
    int i, posicao;
    int elemento;

    for (i = 1; i < n; i++) {
        elemento = vetor[i];
        posicao = posicao_elemento(vetor, i-1, elemento);
        deslocar_subvetor(vetor, posicao, i-1);
        vetor[posicao] = elemento;
    }
}
```

# Exercício 1

1. Reescreva a função `ordenar_selecao` para que ela não utilize as funções auxiliares (`menor_elemento` e `trocar`). Faça o mesmo para a função `ordenar_insercao`.
2. Na função `ordenar_selecao`, é realmente necessária a última iteração do laço de repetição? Por quê? E para a função `ordenar_insercao`?

# Exercício 2

## Ordenação da bolha (*Bubble-Sort*)

```
void ordenar_bolha(int vetor[], int n) {
    int i, j;
    for (i = n - 1; i > 0; i--) {
        for (j = 0; j < i; j++) {
            if (vet[j] < vet[j+1])
                trocar(&vetor[j], &vetor[j+1]);
        }
    }
}
```

1. Explique o que faz e qual é a ideia do algoritmo.
2. Faça um teste de mesa para um vetor com elementos $(5, 4, 3, 2, 1)$ e para um vetor com elementos $(1, 4, 3, 2, 5)$. Conte as trocas.
3. Você consegue dizer por que o algoritmo tem esse nome? Por quê?