

Unidade IV:

Ordenação Interna - Quicksort



PUC Minas

Instituto de Ciências Exatas e Informática
Curso de Ciência da Computação

Unidade IV:


Ordenação Interna - Algoritmo da Bolha



PUC Minas

Instituto de Ciências Exatas e Informática
Departamento de Ciência da Computação

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

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

Introdução

- Proposto por Hoare em 1960 e publicado em 1962
- Algoritmo de ordenação mais rápido para a maioria das situações
- Provavelmente, ele é o mais utilizado

Funcionamento Básico


- Divide o *array* em duas partes que serão independentemente ordenadas e a combinação de seus resultados produz a solução final
 - A parte da esquerda terá elementos menores ou iguais a um pivô
 - A parte da direita terá elementos maiores ou iguais a um pivô

Funcionamento Básico

- Particionamento:
 - Escolha arbitrariamente um pivô
 - Percorra o *array* a partir da esquerda enquanto $array[i] < \text{pivô}$
 - Percorra o *array* a partir da direita enquanto $array[j] > \text{pivô}$
 - Se $i \leq j$ então troque $array[i]$ com $array[j]$
 - Continue o processo enquanto $i \leq j$

Funcionamento Básico

- No final do particionamento, o *array* estará particionado de tal forma que:
 - Os elementos *array*[esq], *array*[esq+1], . . . , *array*[j] são \leq que *x*
 - Os elementos *array*[i], *array*[i+1], . . . , *array*[dir] são \geq que *x*

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

Algoritmo em C like

```

void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
        {   swap(i, j);   i++;   j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
    
```

| | | | | | | | | | | | | | | | |
|---|---|---|---|----|----|---|----|---|---|----|----|----|----|----|----|
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |

Algoritmo em C like

```

void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
        {   swap(i, j);   i++;   j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
    
```

| | | | | | | | | | | | | | | | |
|------------|---|---|---|----|----|---|----|---|---|----|----|----|----|----|------------|
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

```

void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
    
```

i

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

```

void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
        {   swap(i, j);   i++;   j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
    
```

| | | | | | | | | | | | | | | | | | |
|------------|----------|----------|----------|-----------|-----------|----------|-----------|----------|----------|-----------|----------|-----------|-----------|-----------|-----------|------------|--|
| | | | | | | | | | | | | | | | | | |
| i | | | | | | | | | | | | | | | | j | |
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | | |
| esq | | | | | | | | | | | | | | | | dir | |

Algoritmo em C like

pivô

10

$(0 + 15) / 2: 7$

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
        {
            swap(i, j);
            i++;
            j--;
        }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|-----|
| | | | | | | | | | | | | | | | j |
| i | | | | | | | | | | | | | | | |
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

0 <= 15: true

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
        {   swap(i, j);   i++;   j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

i

j

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

0 < 10: true

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

i

j

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

i

j

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô 10

1 < 10: true

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            swap(i, j);
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

i

j

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

i

j

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

5 < 10: true

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

i

j

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

i

j

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

3 < 10: true

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

i

j

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

i

j

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

15 < 10: false

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

i

j

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô 10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            swap(i, j);
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

82 > 10: true

| | | | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|-----|---|--|
| | | | | i | | | | | | | | | | | | j | |
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | | |
| esq | | | | | | | | | | | | | | | dir | | |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|----|-----|
| | | | | i | | | | | | | | | | | | j |
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | |
| esq | | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            swap(i, j);
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

71 > 10: true

| | | | | | | | | | | | | | | | | | |
|------------|----------|----------|----------|-----------|-----------|----------|-----------|----------|----------|-----------|----------|-----------|-----------|-----------|-----------|----------|------------|
| | | | | i | | | | | | | | | | | | j | |
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | | |
| esq | | | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | | | |
|-----|---|---|---|----------|----|---|----|---|---|----|----|----|----|----|----|----------|-----|
| | | | | <i>i</i> | | | | | | | | | | | | <i>j</i> | |
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | | |
| esq | | | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô 10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            swap(i, j);
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

48 > 10: true

| | | | | | | | | | | | | | | | | | |
|------------|----------|----------|----------|-----------|-----------|----------|-----------|----------|----------|-----------|----------|-----------|-----------|-----------|-----------|------------|--|
| | | | | i | | | | | | | | | | | | j | |
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | | |
| esq | | | | | | | | | | | | | | | | dir | |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | | | |
|------------|----------|----------|----------|-----------|-----------|----------|-----------|----------|----------|-----------|----------|-----------|-----------|-----------|-----------|------------|--|
| | | | | i | | | | | | | | | | | | j | |
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | | |
| esq | | | | | | | | | | | | | | | | dir | |

Algoritmo em C like

pivô 10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            swap(i, j);
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

20 > 10: true

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----------|----|---|----|---|---|----|----|----------|----|----|-----|
| | | | | <i>i</i> | | | | | | | | <i>j</i> | | | |
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

i

j

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----|----|---|----|---|---|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô 10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            swap(i, j);
        if (esq < j)
            quicksort(esq, j);
        if (i < dir)
            quicksort(i, dir);
    }
}
```

5 > 10: false

| | | | | | | | | | | | | | | | |
|-----|---|---|---|----------|----|---|----|---|---|----|----|----------|----|----|-----|
| | | | | <i>i</i> | | | | | | | | <i>j</i> | | | |
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

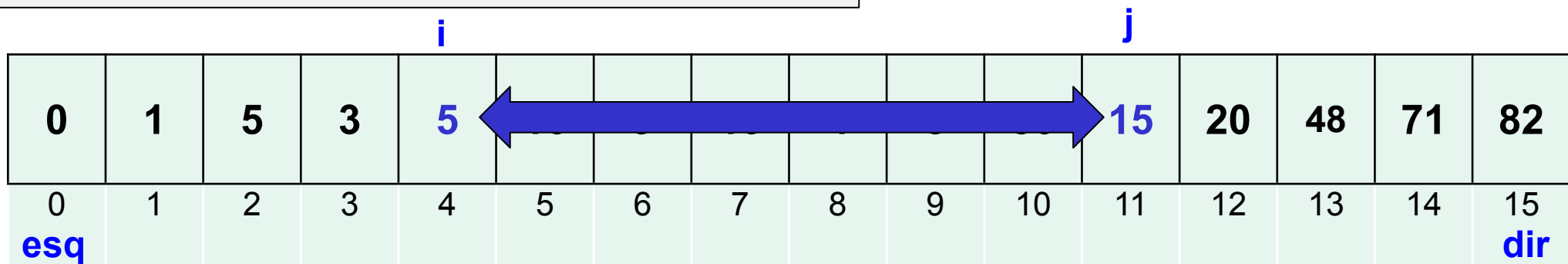
4 <= 11: true

| | | | | | | | | | | | | | | | | | |
|------------|---|---|---|----------|----|---|----|---|---|----|----|----|----|----|----|------------|--|
| | | | | i | | | | | | | | | | | | j | |
| 0 | 1 | 5 | 3 | 15 | 16 | 9 | 10 | 4 | 3 | 30 | 5 | 20 | 48 | 71 | 82 | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | | |
| esq | | | | | | | | | | | | | | | | dir | |

Algoritmo em C like

pivô 10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
        { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```



Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
        {
            swap(i, j);
            i++; j--;
        }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|----------|---|----|---|----------|----|----|----|----|----|-----|
| | | | | | <i>i</i> | | | | <i>j</i> | | | | | | |
| 0 | 1 | 5 | 3 | 5 | 16 | 9 | 10 | 4 | 3 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

5 <= 10: true

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
        {   swap(i, j);   i++;   j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

i

j

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|----|---|----|---|---|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 5 | 16 | 9 | 10 | 4 | 3 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô 10

16 < 10: false

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|----|---|----|---|---|----|----|----|----|----|-----|
| | | | | | i | | | | j | | | | | | |
| 0 | 1 | 5 | 3 | 5 | 16 | 9 | 10 | 4 | 3 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô 10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            swap(i, j);
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

30 > 10: true

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|----------|---|----|---|----------|----|----|----|----|----|-----|
| | | | | | <i>i</i> | | | | <i>j</i> | | | | | | |
| 0 | 1 | 5 | 3 | 5 | 16 | 9 | 10 | 4 | 3 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | | |
|-----|---|---|---|---|----|---|----|---|---|----|----|----|----|----|----|-----|
| | | | | | i | | | | j | | | | | | | |
| 0 | 1 | 5 | 3 | 5 | 16 | 9 | 10 | 4 | 3 | 30 | 15 | 20 | 48 | 71 | 82 | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | |
| esq | | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô 10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            swap(i, j);
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

3 > 10: false

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|----|---|----|---|---|----|----|----|----|----|-----|
| | | | | | i | | | | j | | | | | | |
| 0 | 1 | 5 | 3 | 5 | 16 | 9 | 10 | 4 | 3 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

5 <= 9: true

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|----|---|----|---|---|----|----|----|----|----|-----|
| | | | | | i | | | | j | | | | | | |
| 0 | 1 | 5 | 3 | 5 | 16 | 9 | 10 | 4 | 3 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô 10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
        { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|-----|
| | | | | | i | | | | j | | | | | | | |
| 0 | 1 | 5 | 3 | 5 | 3 | | | | 16 | 30 | 15 | 20 | 48 | 71 | 82 | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | |
| esq | | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
        {
            swap(i, j);
            i++; j--;
        }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|----|---|----|----|----|----|----|----|----|-----|
| | | | | | | i | j | | | | | | | | | |
| 0 | 1 | 5 | 3 | 5 | 3 | 9 | 10 | 4 | 16 | 30 | 15 | 20 | 48 | 71 | 82 | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | |
| esq | | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

6 <= 8: true

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
        {   swap(i, j);   i++;   j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|----|---|----|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 5 | 3 | 9 | 10 | 4 | 16 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

9 < 10: true

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|----|---|----|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 5 | 3 | 9 | 10 | 4 | 16 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

i j

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|----|---|----|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 5 | 3 | 9 | 10 | 4 | 16 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

10 < 10: false

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

i j

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|----|---|----|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 5 | 3 | 9 | 10 | 4 | 16 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô 10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            swap(i, j);
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

4 > 10: false

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|----|---|----|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 5 | 3 | 9 | 10 | 4 | 16 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

7 <= 8: true

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|----|---|----|----|----|----|----|----|-----|
| | | | | | | | i | j | | | | | | | |
| 0 | 1 | 5 | 3 | 5 | 3 | 9 | 10 | 4 | 16 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô 10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
        { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|-----|
| | | | | | | | i | j | | | | | | | | |
| 0 | 1 | 5 | 3 | 5 | 3 | 9 | 4 | 10 | 16 | 30 | 15 | 20 | 48 | 71 | 82 | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | |
| esq | | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
        {
            swap(i, j);
            i++; j--;
        }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|----|----|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 5 | 3 | 9 | 4 | 10 | 16 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

8 <= 7: false

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
        {
            swap(i, j);
            i++;
            j--;
        }
        if (esq < j)
            quicksort(esq, j);
        if (i < dir)
            quicksort(i, dir);
    }
}
```

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|----|----|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 5 | 3 | 9 | 4 | 10 | 16 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
}
```

0 < 7: true

if (esq < j)

quicksort(esq, j);

if (i < dir)

quicksort(i, dir);

}

j i

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|----|----|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 5 | 3 | 9 | 4 | 10 | 16 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|----|----|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 5 | 3 | 9 | 4 | 10 | 16 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

8 < 15: true

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|----|----|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 5 | 3 | 9 | 4 | 10 | 16 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

Algoritmo em C like

pivô

10

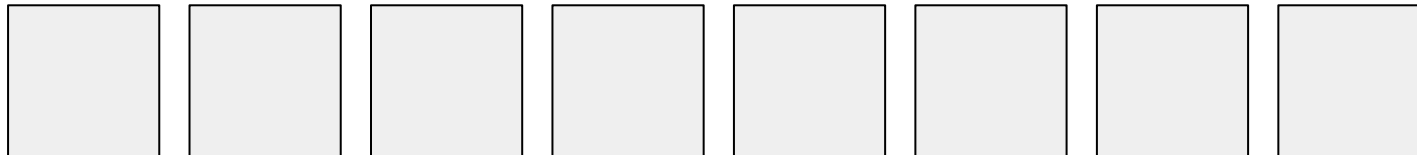
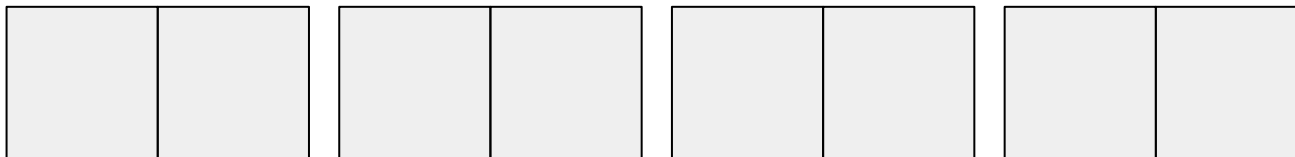
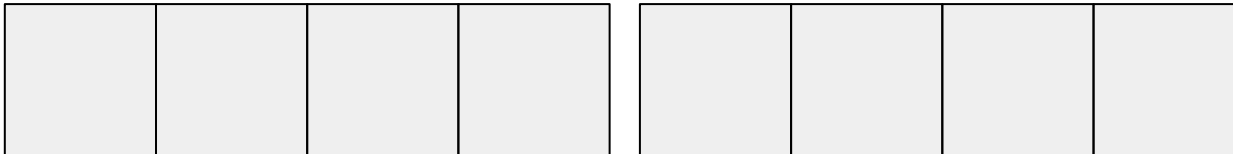
```
void quicksort(int esq, int dir) {
    int i = esq, j = dir, pivo = array[(dir+esq)/2];
    while (i <= j) {
        while (array[i] < pivo)
            i++;
        while (array[j] > pivo)
            j--;
        if (i <= j)
            { swap(i, j); i++; j--; }
    }
    if (esq < j)
        quicksort(esq, j);
    if (i < dir)
        quicksort(i, dir);
}
```

| | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|----|----|----|----|----|----|----|-----|
| 0 | 1 | 5 | 3 | 5 | 3 | 9 | 4 | 10 | 16 | 30 | 15 | 20 | 48 | 71 | 82 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| esq | | | | | | | | | | | | | | | dir |

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

Análise do Número de Comparações

- Melhor caso: Sistemáticamente, cada partição divide o arquivo em duas partes iguais



Número Comparações

8

4 + 4

2 + 2 + 2 + 2

n comparações
realizadas $\lg n$ vezes
 $O(n * \lg n)$

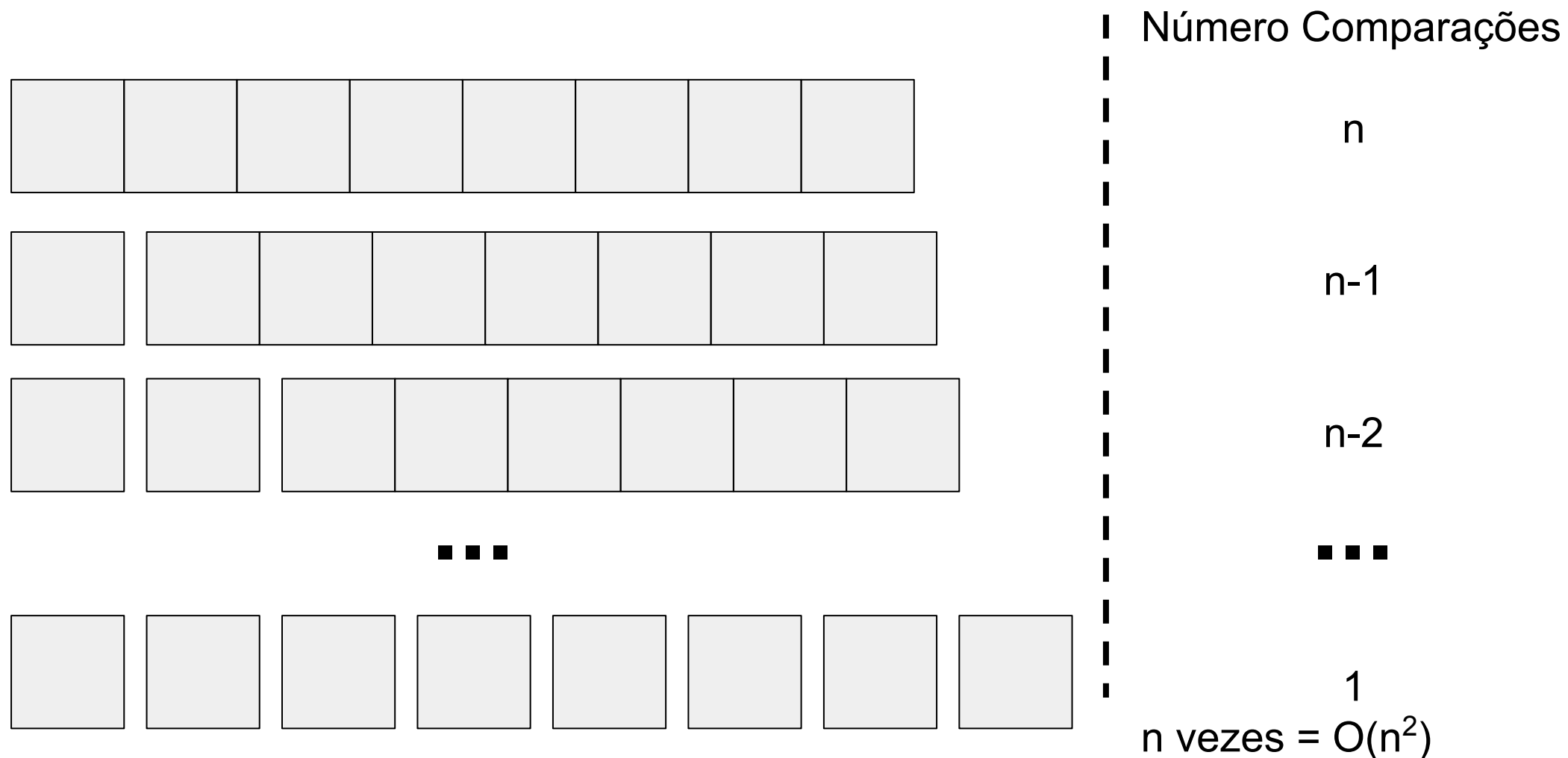
Análise do Número de Comparações

- Melhor caso: Sistemáticamente, cada partição divide o arquivo em duas partes iguais

$$C(n) = 2 * C\left(\frac{n}{2}\right) + n = n * \lg(n) - n + 1$$

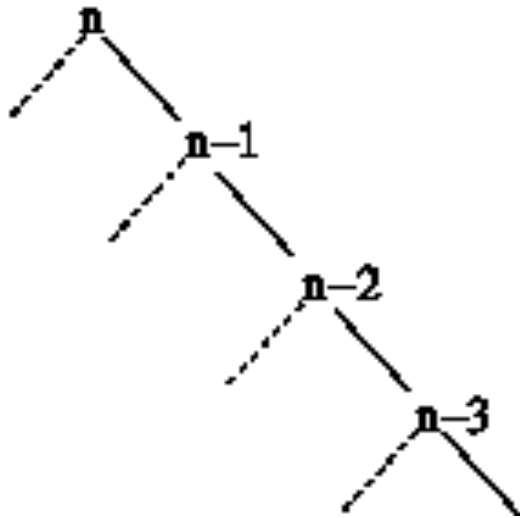
Análise do Número de Comparações

- Pior caso: Sistemáticamente, o pivô é o menor ou o maior elemento do *array*, eliminando um elemento em cada chamada do algoritmo



Análise do Número de Comparações

- Pior caso: Sistemáticamente, o pivô é o menor ou o maior elemento do *array*, eliminando um elemento em cada chamada do algoritmo



$$C(n) = \Theta(n^2)$$

- Existem diversas técnicas para evitar o pior caso como, por exemplo, fazer com que o pivô seja a mediana de três elementos do *array*

Análise do Número de Comparações

- Caso Médio: Sedgewick e Flajolet (1996, p. 17):

$$C(n) \approx 1,386 * n * \lg(n) - 0,846 * n$$

Análise do Número de Movimentações

- No pior caso, há $\lceil n/2 \rceil$ trocas ($3 * \lceil n/2 \rceil$ movimentações) em cada execução da função de partição
- Corresponde ao caso em que o pivô se coloca no meio do *array* e os elementos superiores estão sistematicamente no início da lista e os inferiores, no fim

Conclusões

- A razão de sua velocidade é a simplicidade do seu anel interno
- Vantagens:
 - Extremamente eficiente
 - Necessita de apenas uma pequena pilha como memória auxiliar
 - Faz em média $n \cdot \lg(n)$ comparações - $O(n \cdot \lg(n))$

Conclusões

- Desvantagens:
 - Seu pior caso para comparações é quadrático
 - Sua implementação é delicada e difícil
 - Método 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 |
|----|---|---|---|----|----|---|----|----|----|----|---|----|---|---|----|---|---|

Exercício

- Implemente o Quicksort paralelo. Em seguida, supondo a existência de um número infinito de processadores, faça a análise de complexidade do algoritmo proposto