https://www.dcc.fc.up.pt/~nam/aulas/9900/pi/slides/slipi9916/

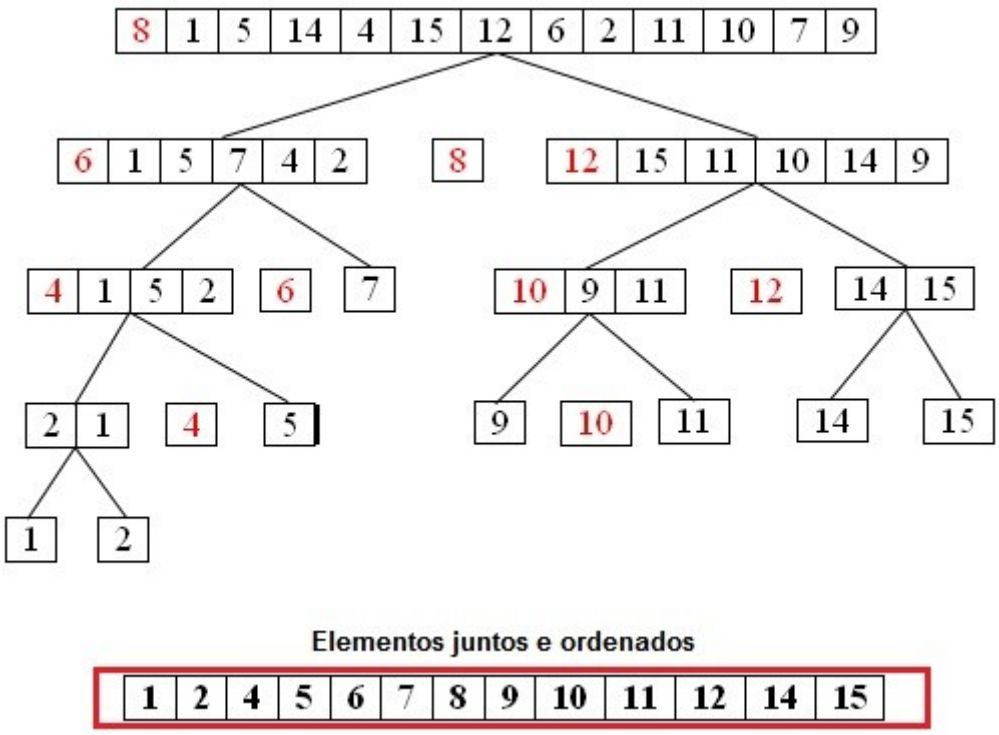
Método de ordenação Quicksort – Aplicação em Vetores

Método:

Dada uma sequência de valores, escolhe-se um elemento e os restantes são reagrupados em duas subsequências (partição): os que são *menores* que esse elemento e os que são *maiores*.

O mesmo processo é aplicado *recursivamente* a cada subsequência.

Quando um subconjunto tem menos que 2 elementos a recursão pára.



Melhor Caso: Os particionamentos produzem segmentos de mesmo tamanho.

Piores Casos: Ocorrem quando a sequência está ordenada ou em ordem inversa, pois a chave particionadora será o menor elemento (ou o maior).

```
void quicksort(int v[], int esq, int dir){
  int i;
  if(esq>=dir) return ;
  i=particao(v,esq,dir);
  quicksort(v,esq,i-1);
  quicksort(v,i+1,dir);
}

int particao(int v[],int esq,int dir){
  int i, fim;
  printf("\n Subvetor:\n");
  for(i=esq; i<=dir; i++) printf("%d ",v[i]); printf("\n");
  fim=esq;
  for(i=esq+1;i<=dir;i++)
    if(v[i]<v[esq]) {
       fim=fim+1;
       troca(v,fim,i);
    }
  troca(v,esq,fim);
  for(i=esq; i<=fim; i++) printf("%d ",v[i]);  printf("\n");
  for(i=fim+1; i<=dir; i++) printf("%d ",v[i]); printf("\n");
  return fim;
}

void troca(int v[], int i, int j){
  int temp;
  temp=v[i];
  v[i]=v[j];
  v[j]=temp;
}

#define N 16
main(){
  int i,a[N];
  for(i=0; i<N; i++) a[i]=rand()%100;
  for(i=0; i<N; i++) printf("%d ",a[i]);
  quicksort(a,0,N-1);
  printf("\n Ordenado:\n");
  for(i=0; i<N; i++) printf("%d ",a[i]);
}
```

==========

Exemplo de execução:

```
Sequência: 57 82 36 13 84 37 97
      esq = 0
      dir = 6

 1)   Início do Primeiro particionamento: 57 82 36 13 84 37 97
      fim = 0
      i=1   →     57 82 36 13 84 37 97    (comparação entre 57 e 82)
      i=2   →     57 82 36 13 84 37 97    (comparação entre 57 e 36)
                  fim = 1
                  57 36 82 13 84 37 97    (troca posições: i=2,fim=1)
      i=3   →     57 36 82 13 84 37 97    (comparação entre 57 e 13)
                  fim = 2
                  57 36 13 82 84 37 97    (troca posições: i=3,fim=2)
      i=4   →     57 36 13 82 84 37 97    (comparação entre 57 e 84)
```

```
i=5   →     57 36 13 82 84 37 97    (comparação entre 57 e 37)
            fim = 3
            57 36 13 37 84 82 97    (troca posições: i=5,fim=3)
i=6   →     57 36 13 37 84 82 97    (comparação entre 57 e 97)

Ultima Troca: esq=0,fim=3     →     37 36 13 57 84 83 97
```

Partições Geradas:

```
      Pivô: 57                (posição fim)
      Partição 1: 37 36 13    (de esq até fim-1)
      Partição 2: 84 82 97    (de fim+1 até dir)
```

Análise: O elemento pivô foi posicionado corretamente, faltando ordenar as duas partições obtidas e suas próximas recursivas.

2)

```
Particionamento do subvetor Partição 1: 37 36 13
esq = 0, dir = 2
fim = 0
i=1   →     37 36 13    (comparação entre 37 e 36)
            fim = 1
            37 36 13    (troca posições: i=1,fim=1)
i=2   →     37 36 13
            fim = 2
            37 36 13    (troca posições: i=2,fim=2)

Última Troca: esq=0,fim=2     →     13 36 37
```

Partições Geradas:

```
      Pivô: 37                (posição fim)
      Partição 3: 13 36       (de esq até fim-1)
      Partição 4: vazia
```

3)

```
Particionamento do subvetor Partição 3: 13 36
esq = 0, dir=1
fim = 0
i=1   →     13 36       (comparação entre 13 e 36)

Última troca: esq=0, fim=0    →     13 36

Partições Geradas:
      Pivô: 13          (posição fim)
      Partição 5: vazia
      Partição 6: 36    (de fim+1 até dir)
```

4)

```
Particionamento do subvetor Partição 2: 84 82 97
esq = 4, dir=6
fim = 4
i=5   →     84 82 97    (comparação entre 84 e 82)
            fim = 5
            84 82 97    (troca posições: i=5, fim=5)
i=6   →     84 82 97    (comparação entre 84 e 97)

Última troca: esq=4, fim=5    →     82 84 97

Partições Geradas:
      Pivô: 84          (posição fim)
      Partição 7: 82    (de esq até fim-1)
      Partição 8: 97    (de fim+1 até dir)
```

5) As partições 4 e 5 ficaram vazias, e as partições 6, 7 e 8 foram criadas com apenas um elemento, estando então já ordenadas. Assim, todas as posições do vetor já se encontram ordenadas:

13 36 37 57 82 84 97

Resumo Gráfico do Processamento: