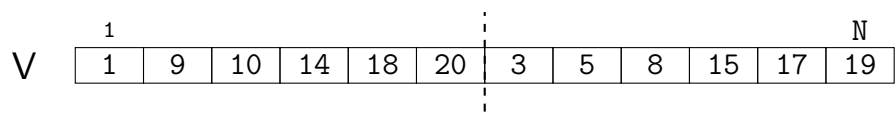


Agora imagine que nós temos uma lista onde a primeira metade e a segunda metade já estão ordenadas.

Por exemplo,



A tarefa é

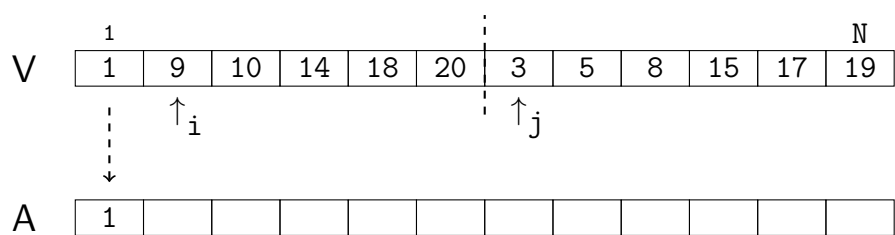
→ Ordenar a lista completamente

Bom, aqui a gente pode usar uma ideia semelhante à do exemplo anterior

Quer dizer, a gente coloca um dedo em cada metade.

E verifica qual elemento é o menor.

Daí, o menor deles é copiado para uma lista auxiliar



A coisa continua assim até que todos os elementos tenham sido copiados para a lista A.

E daí, é só copiar os elementos de volta para a lista V.

Abaixo nós temos o algoritmo (já anotado)

```

O(1) | i = 1;    j = N/2 + 1           // um dedo em cada metade
      | k = 1           // um dedo na lista auxiliar
      | Enquanto ( i ≤ N/2  OU  j ≤ N )
      |   Se ( i > N/2 )           // a 1a metade já acabou
      |       A[k] = V[j];    k++;    j++
      |   Sse ( j > N )           // a 2a metade já acabou
      |       A[k] = V[i];    k++;    i++
      |   Sse ( U[i] < V[j] )       // quem é o menor?
      |       A[k] = V[i];    k++;    i++
      |   Senão
      |       A[k] = V[j];    k++;    j++
      |
O(n) | O(1) | i = 1           //
      |       | Enquanto ( i ≤ N )           // copia tudo de volta
      |       |     V[i] = A[i];    i++           //

```

Agora basta examinar as anotações para obter o tempo de execução do algoritmo

$O(1) + O(n) \cdot O(1) + O(1) + O(n) \cdot O(1) = O(n)$