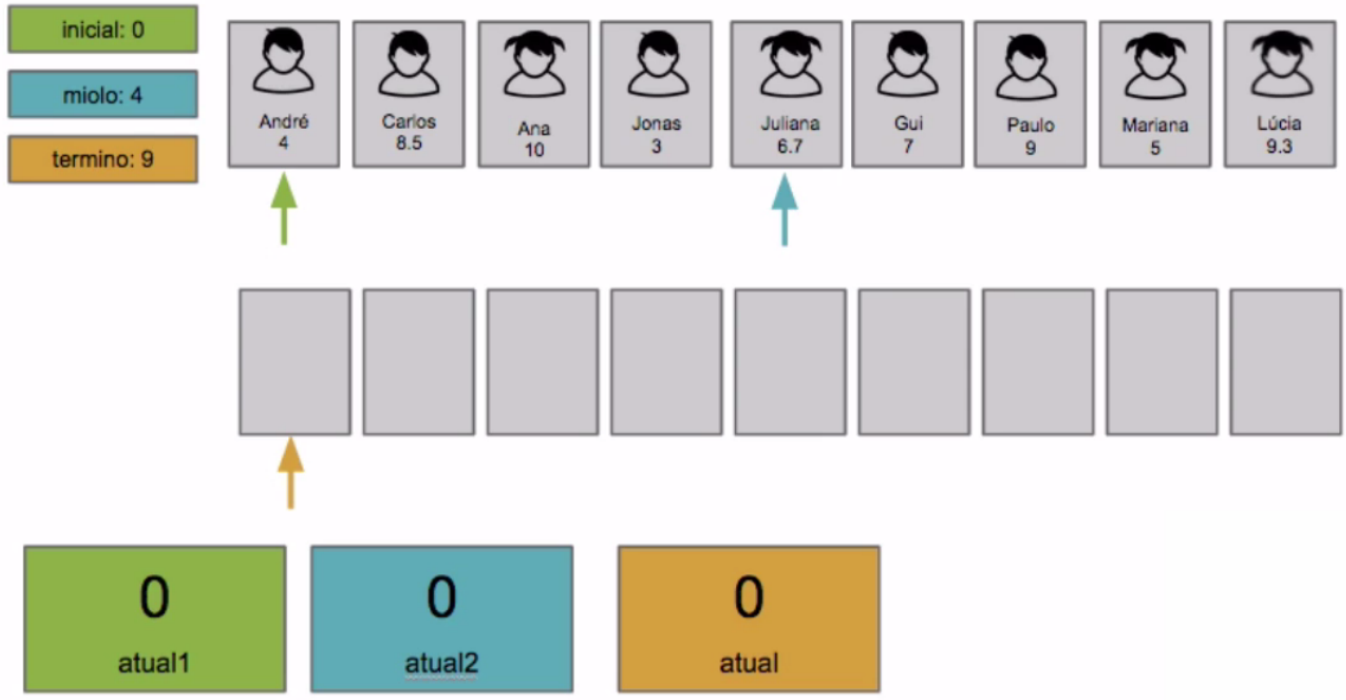
01

Intercalando diversas vezes

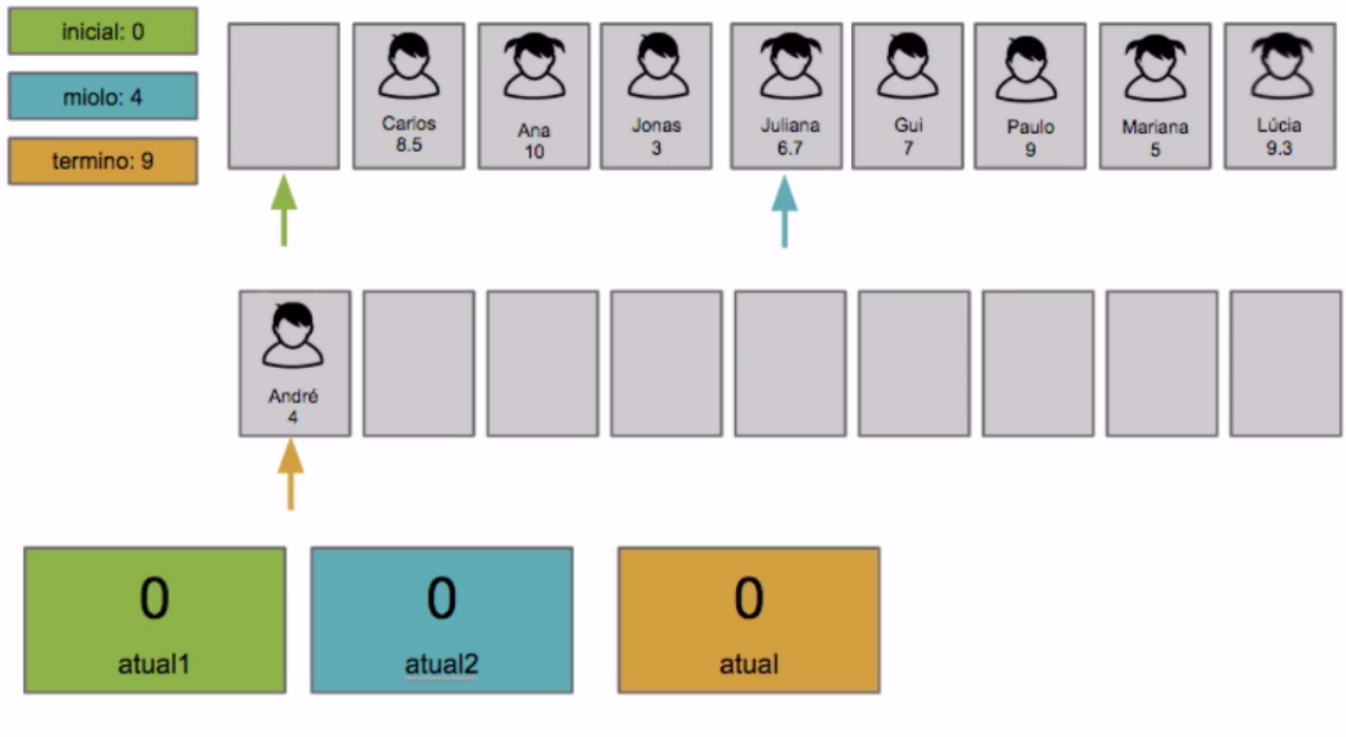
Transcrição

Nós vimos que se chamarmos uma série de intercalações, das menores para as maiores, conseguimos ordenar o nosso *array*. Então, como poderíamos implementar um algoritmo de ordenação? Caso me peçam para ordenar do 0 até 9, eu vou responder: É melhor ordenar primeiro do 0 até 4 e depois, do 5 ao 9. Assim poderemos intercalar os elementos.

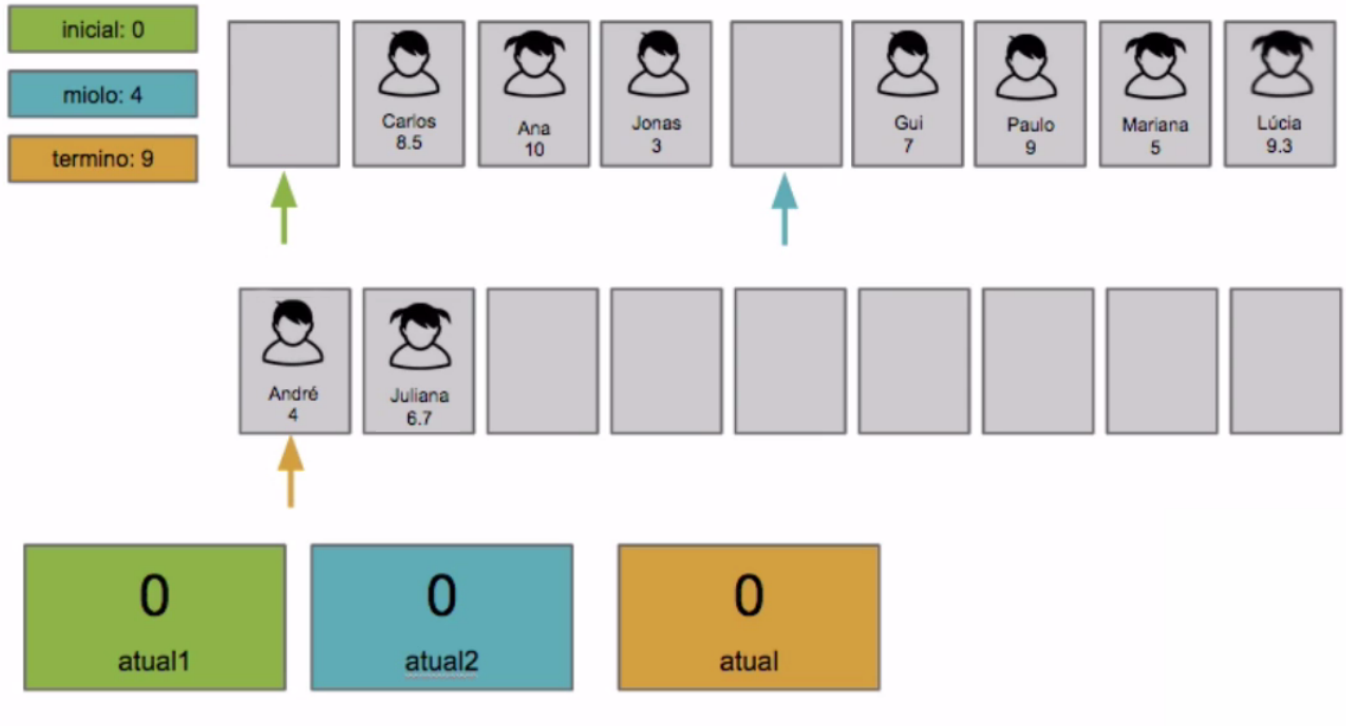
Mas quando tivermos que intercalar de 0 até 4, o que teremos que fazer? Podemos dividir o problema, ir do 0 até 2, de 2 até 4, e então iremos intercalar. Quando formos ordenar do 0 até 2, iremos ordenar um primeiro elemento, depois outro, e então, intercalamos os dois. Ordenar um ou dois itens é fácil. O algoritmo consegue intercalar os primeiros elementos. Após intercalarmos os dois primeiros, intercalaremos os dois seguintes. E continuaremos com o processo de ordenação.



Nós iremos dividir em duas partes o *array*. O fato de termos um número ímpar de elementos não é um problema.

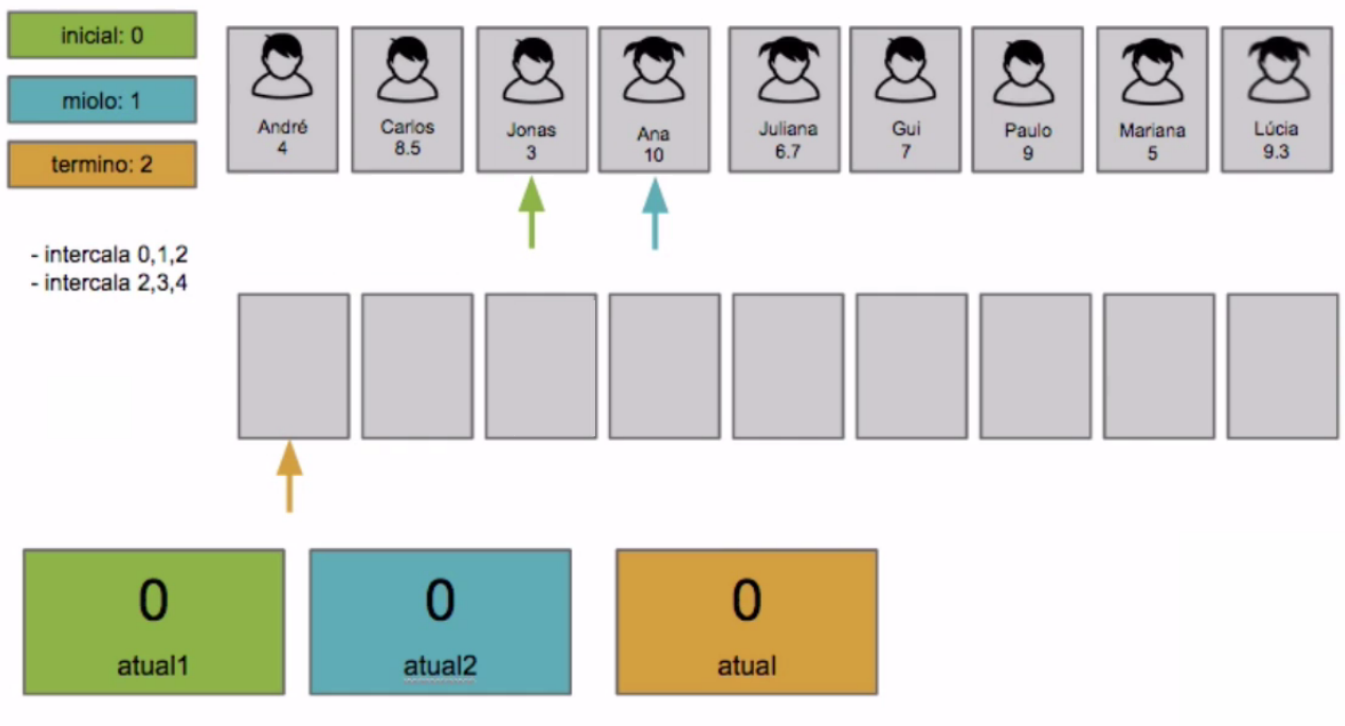


Vamos ordenar as partes da esquerda e da direita, para então intercalarmos todos os elementos. Como ordenar cada parte? Também iremos dividir o problema.

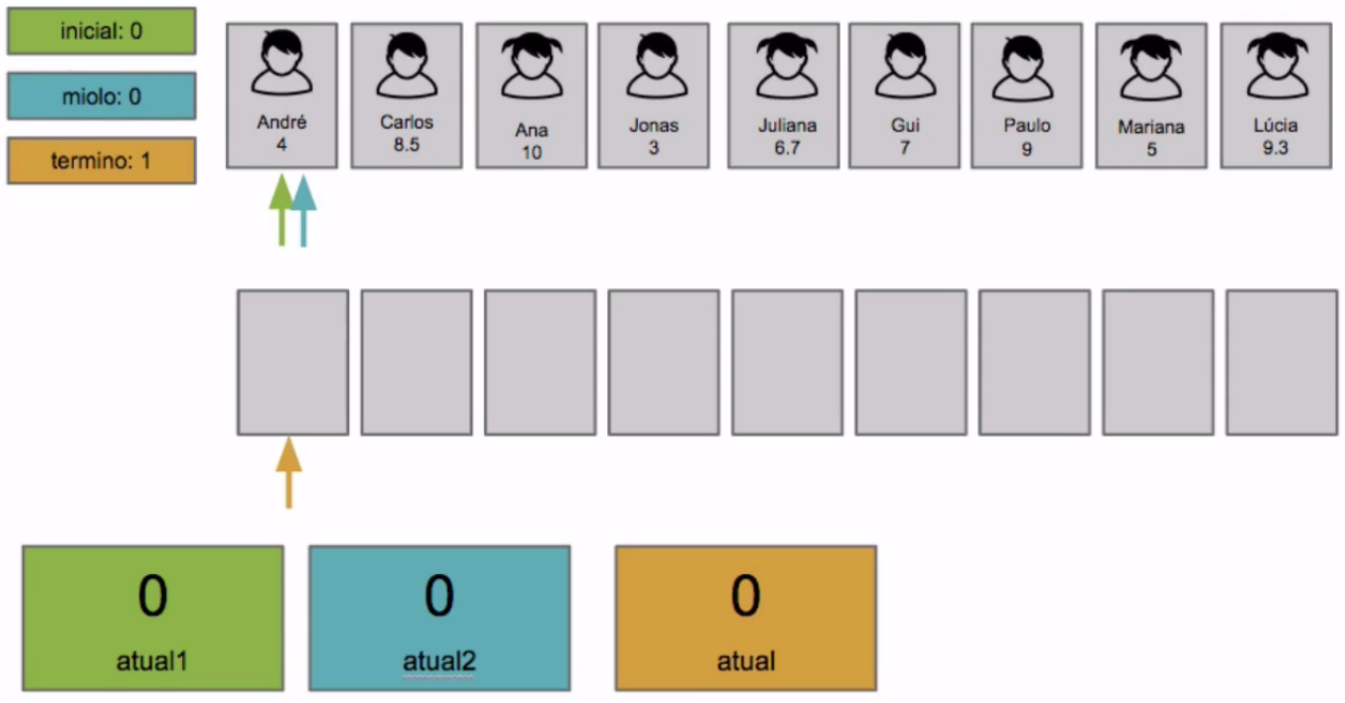


Depois vamos intercalar os elementos. Isto significar que o próprio **ordenar** está chamando o **ordenar**. Até ficar um ordenação bem pequena. Iremos ordenar o primeiro elemento, depois o segundo e então, intercalamos os dois - algo que já sabemos fazer.

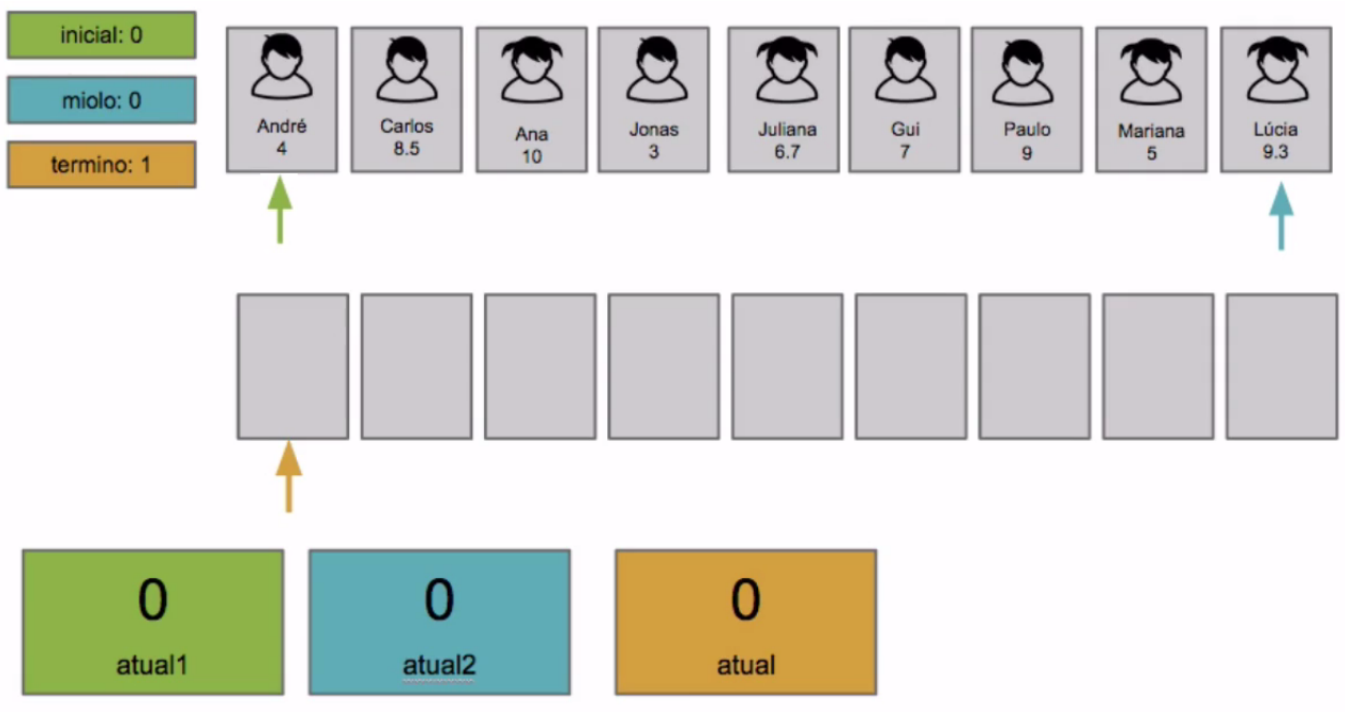
Precisamos seguir ordenando os outros elementos, vamos intercalar os dois seguintes.



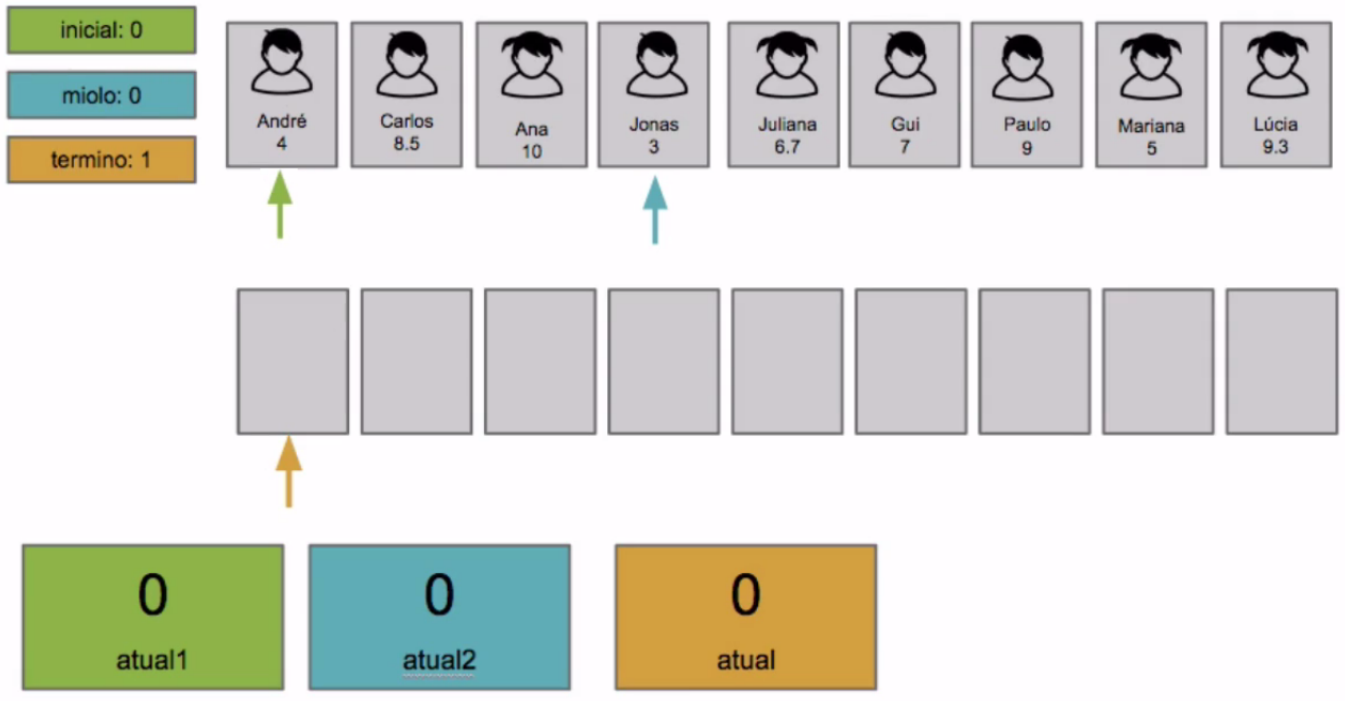
Agora que temos os dois primeiros e os dois últimos ordenados, vamos intercalar os quatro? Então, a minha função de ordenação pede que eu divida no meio a lista com os elementos. Depois chame a **ordenação** para os elementos da esquerda e depois para a direita. Agora que temos os dois trechos ordenados, iremos intercalar todos. Vamos ver o que acontece?



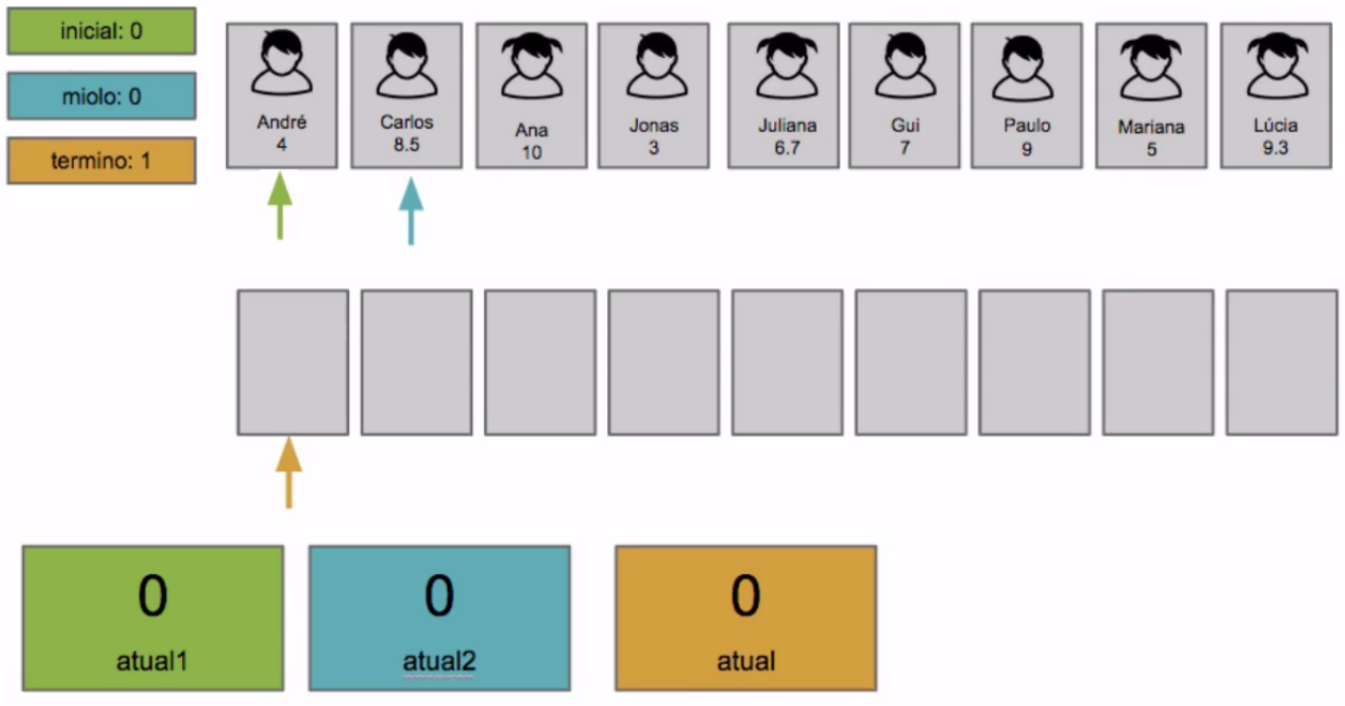
Temos o nosso código. No começo iremos chamar a minha função de ordenação do 0 até 9.



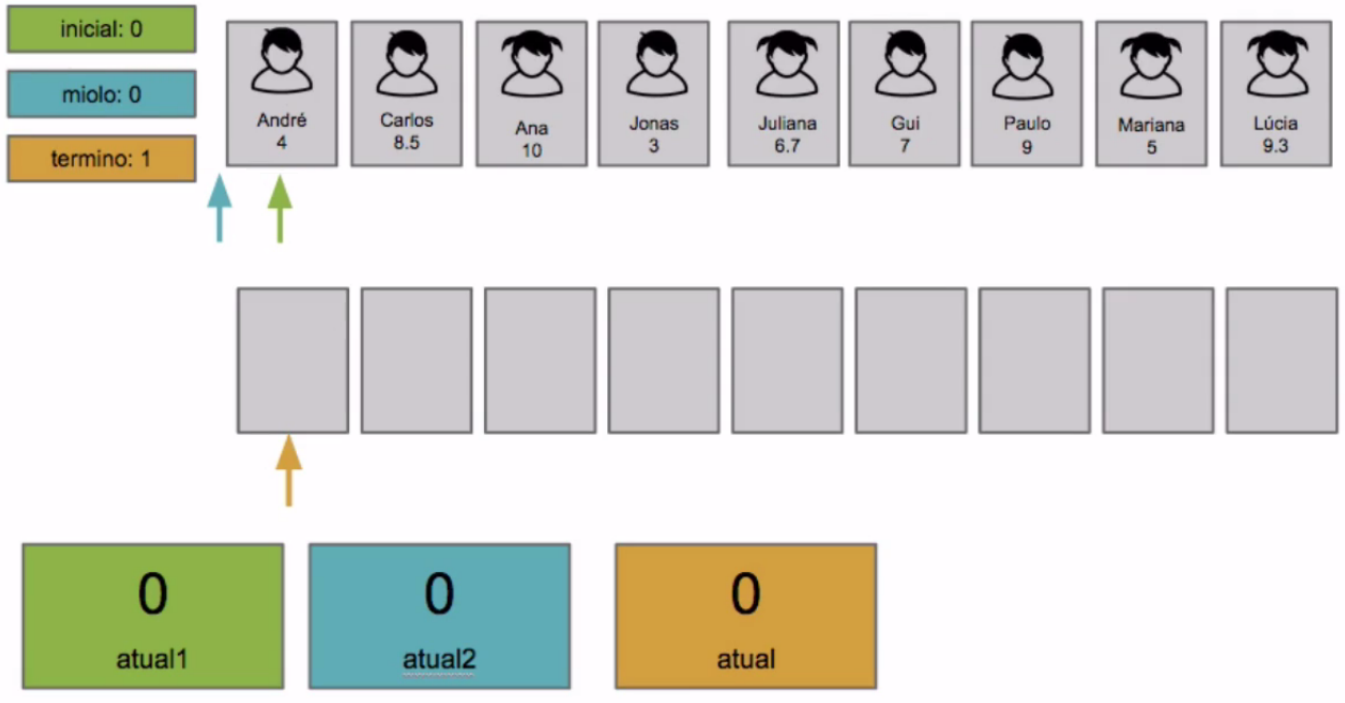
Se dividir nove, a quantidade total de elementos, precisaremos de um resultado inteiro. No caso, será o quatro. Então, iremos ordenar do 0 até 4.



Depois, o código irá pedir para dividirmos novamente os elementos. Se temos quatro elementos, iremos dividir do 0 até 2.

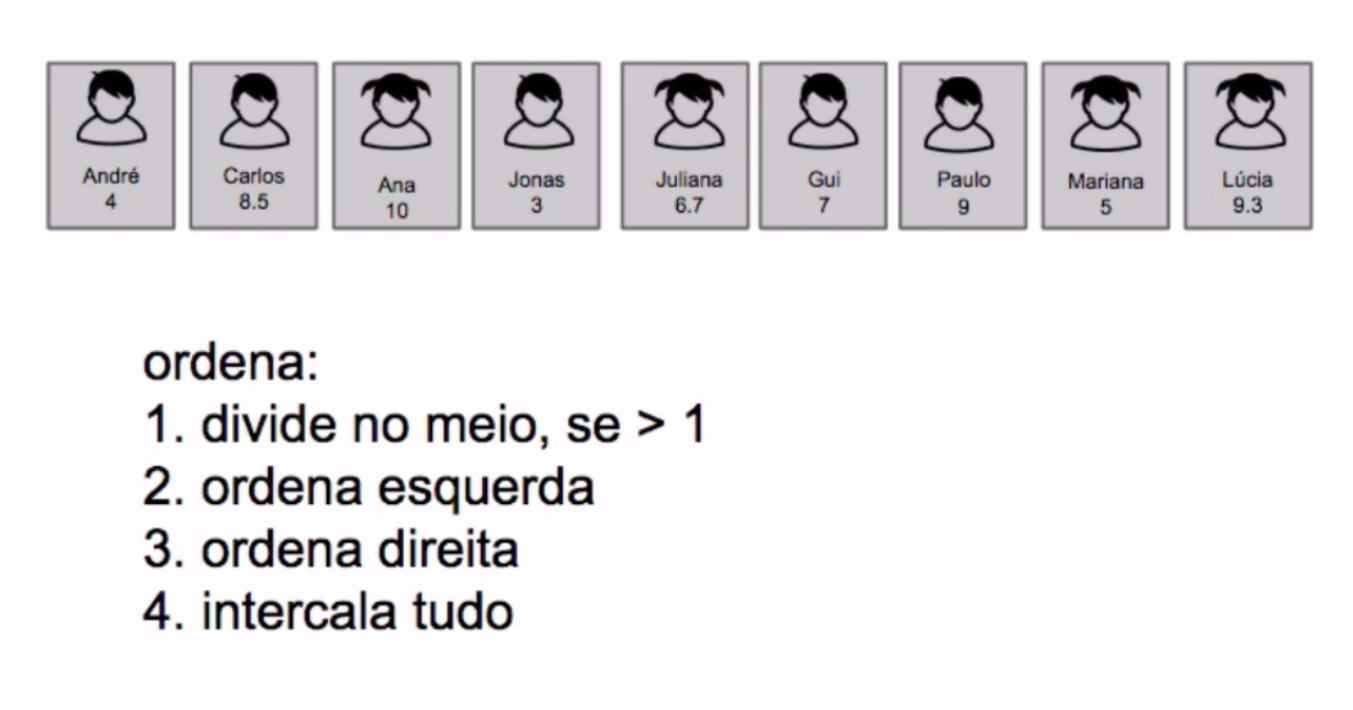


Quando tivermos dois elementos, teremos que dividir por dois.



É possível repetir o processo várias vezes. Mas se os valores forem muito pequenos, em algum momentos, teremos que parar.

Este algoritmo de ordenação só irá dividir no meio, se tiver mais de um elemento.



Se pedirmos para ordenar **um** elemento, o resultado será imediato. E ordenar **dois**? Mandaremos ordenar um e depois o outro, então intercalaremos. Para ordenar **três** elementos, mandaremos ordenar os dois, depois o item restante, logo intercalaremos os três. Para organizar os dois primeiros, precisaremos chamar o `ordena` para dois elementos - sabemos que isto funciona. Como faremos para ordenar **quatro** elementos? Chamaremos o `ordena` para os dois primeiros e depois para os dois últimos, então, intercalaremos. Como faremos com **cinco** elementos? Chamaremos o `ordena` para dois, depois para três. Logo intercalaremos as duas partes. Como ordenaremos **seis** elementos? Chamaremos o `ordena` para os três primeiros, depois para os três últimos. No fim, intercalaremos todos. E para ordenar **sete** elementos? Chamaremos o `ordena` para três, em seguida para quatro - sabemos que funciona - e intercalaremos. Basta sabermos que o `intercala` funciona para um e dois elementos, que ele funcionará para três, quatro, cinco ou para qualquer número. E então, teremos nossa ordenação.

O que a função de ordenação faz? Se trabalhamos apenas com um número, já está ordenado. Se são dois, ordenaremos o da esquerda e o da direita, em seguida intercalamos. Na verdade, só iremos intercalar os dois. Se forem três elementos, iremos quebrar em uma parte com dois números e outra com apenas um. Fazendo as divisões, tudo passa a funcionar. Esta é nossa função de **ordenação** e o código que iremos implementar.

Intercalando valores inválidos em Java

Intercalar é um processo interessante, nós trabalhamos com dois *arrays* já ordenados e os intercalamos na ordem correta. Porém, para intercalarmos todos os resultados de provas do Enem, primeiro eles precisam estar ordenados em partes menores. Isto significa que eles precisam ter sido distribuídos entre diversas pessoas que ordenaram partes pequenas. Enquanto as partes menores não estiverem ordenadas, não temos o que intercalar. Se temos um *array* com 1 milhão de provas, iremos dividi-las em duas partes com 500 mil e duas pessoas irão ordená-las. Só após esta ação, poderemos intercalar. Temos um problema grande.

Se quiséssemos ordenar as provas sem ter nada ordenado, isto é, se quisermos ordenar um *array* e não termos nada ordenado, apenas chamar o `intercala` não irá resolver. Vamos testar?

Iremos copiar o código do `TestaIntercalaEmUmArray` e daremos o nome de `TestaOrdenacaoAoIntercalar`. Teremos um *array* que era igual ao que já usamos, mas misturaremos os elementos.

```
public static void main(String[] args) {
    Nota[] notas = {
            new Nota("andre", 4).
            new Nota("carlos", 8.5),
            new Nota("ana", 10),
            new Nota("jonas", 3),
            new Nota("juliana", 6.7),
            new Nota("guilherme", 7),
            new Nota("paulo", 9),
            new Nota("mariana", 5),
            new Nota("lucia", 9.3)
    };

    Nota[] rank = intercala(notas, 0, 4, notas.length);
    for(Nota nota : rank) {
        System.out.println(nota.getAluno() +" " + nota.getValor());

    }
    }
```

Misturamos todos os elementos, se chamarmos o `intercala` o resultado provavelmente não será o correto.

```
andre 4.0
juliana 6.7
guilherme 7.0
carlos 8.5
paulo 9.0
mariana 5.0
lucia 9.3
ana 10.0
jonas 3.0
```

A ordem não ficou totalmente correta. Para intercalar precisávamos ter dois *array*s já intercalados. Eles poderiam ser de tamanho 1, ou um ser de tamanho 0 e o outro 1. Ambos poderiam ser de de tamanho 3. Mas era preciso que cada pedaço do *array* já estivesse ordenado, para podermos intercalar. Como faremos para intercalar já ordenando? O que poderemos fazer?

Usando o próprio *array*

O que queremos fazer agora é tentar ordenar um *array* inteiro, chamando o `intercala` . Já vimos que se o *array* é grande, e os dois trechos não estão ordenados, o intercala não irá funcionar. Observe o resultado do nosso *array* com os elementos misturados.

```
andre 4.0
juliana 6.7
guilherme 7.0
carlos 8.5
paulo 9.0
mariana 5.0
lucia 9.3
ana 10.0
jonas 3.0
```