

INSTITUTO FEDERAL
Norte de Minas Gerais
Campus Januária

HEAPSORT

GABRIELA VITÓRIA AQUINO PEREIRA
NATAN LOPES SILVA

BACHA. SISTEMAS DE INFORMAÇÃO

Heapsort

- **Desenvolvido em 1964 por Robert W. Floyd e J.W.J. Willians.**
- **Método de seleção em árvore binária do tipo Heap (vetor que simula uma árvore binária completa, com execução do último nível)**
- **Consiste em duas fases:**
 - ✓ Fase 1: Construção da heap(build-heap)
 - ✓ Fase 2: Seleção dos elementos na ordem desejada(heap sort)

Heapsort

Fase 1

- Dado um array, troca-se as chaves de lugares de forma que a árvore representada pela array passe a ser um heap.
- Para toda sub-árvore:
 $A[\text{Pai}(i)] \geq A[i]$ **max-heap**
 $A[\text{Pai}(i)] \leq A[i]$ **min-heap**
- Os testes de chaves se iniciam pela última sub-árvore prosseguindo, a partir daí, para as sub-árvores que antecedem essa, até testar a raiz da árvore.

Heapsort

Fase 1: EXEMPLO

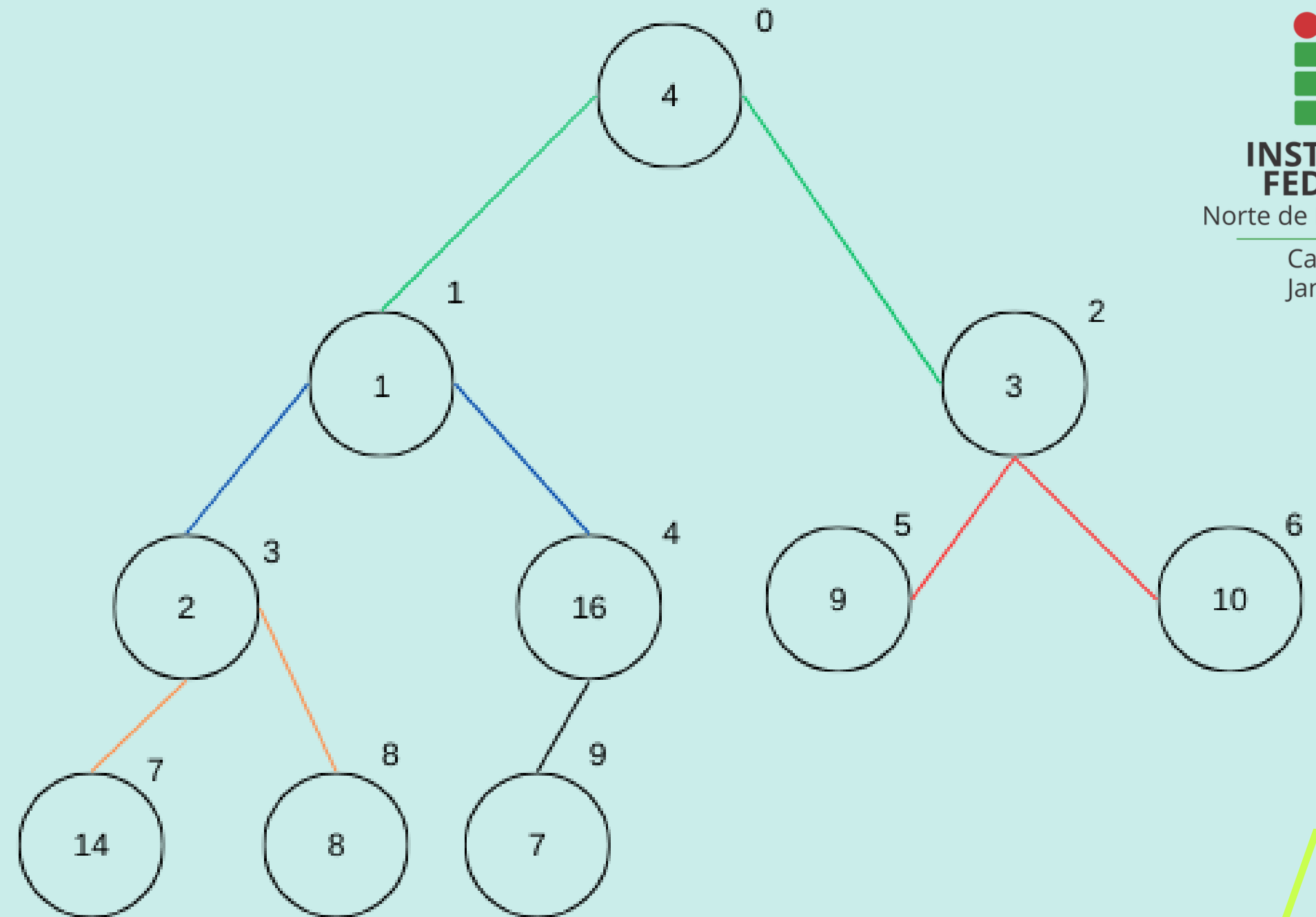
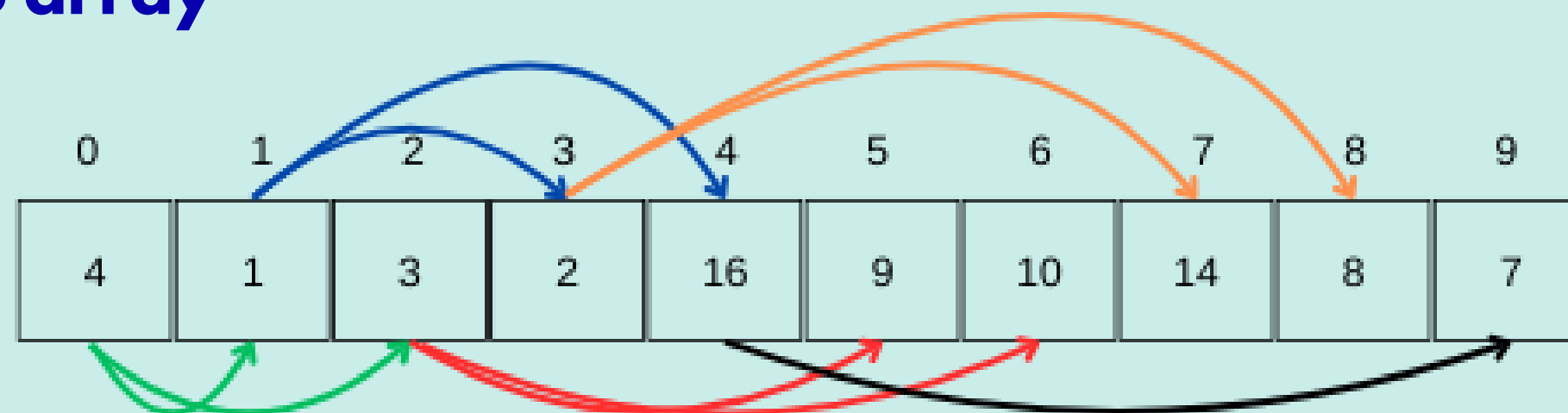
Raiz= 0

$\text{Pai}(i) = (i-1)/2$

$\text{Filho_esq}(i) = 2*i+1$

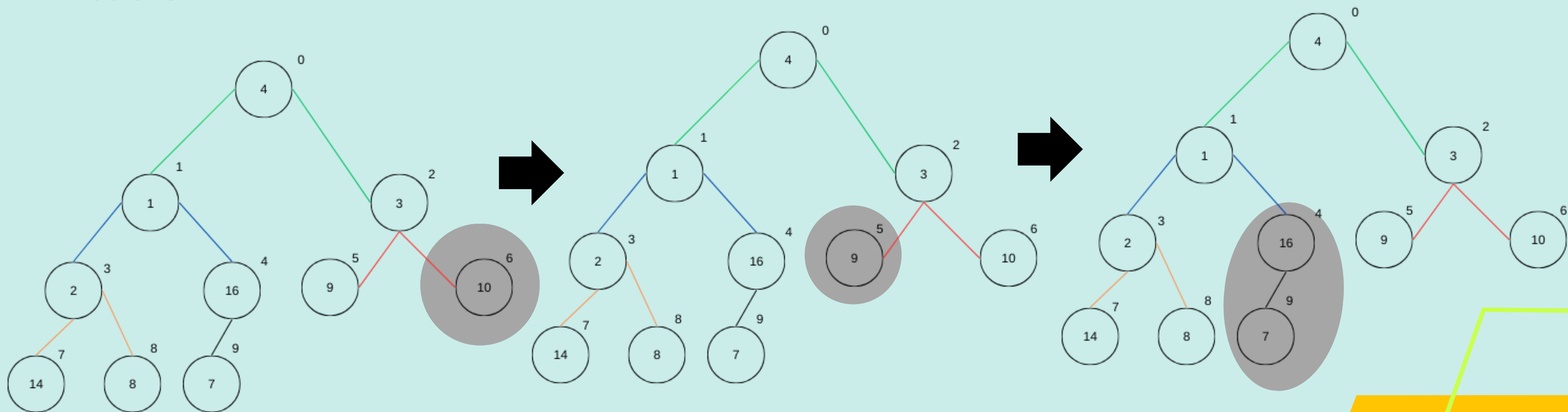
$\text{Filho_dir}(i) = 2*i+2$

**Onde i representa o índice
ocupado no array**



Heapsort


Fase 1: EXEMPLO

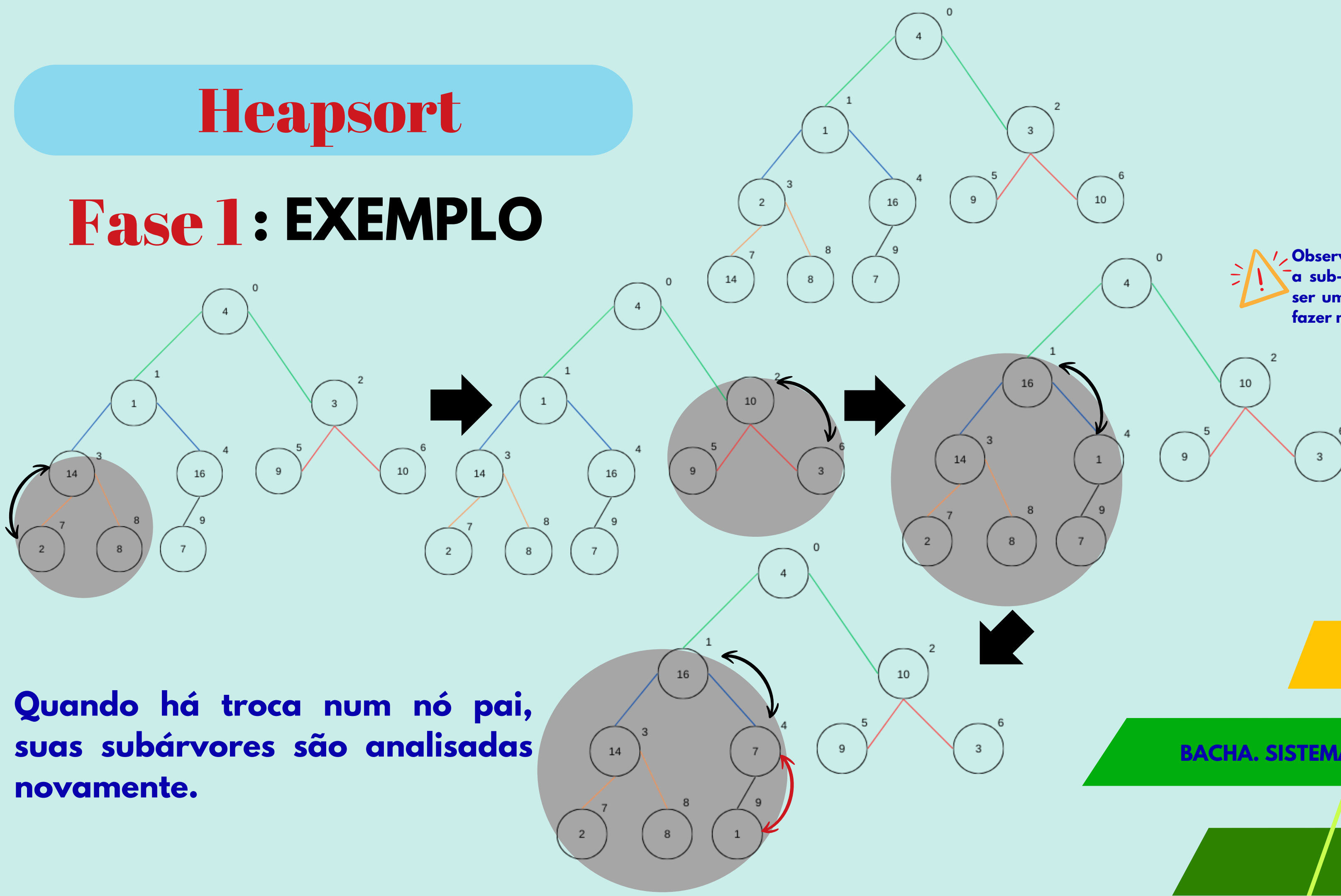


As subárvores são analisadas e o maior elemento da subárvore é colocado na raiz

Heapsort

Fase 1: EXEMPLO

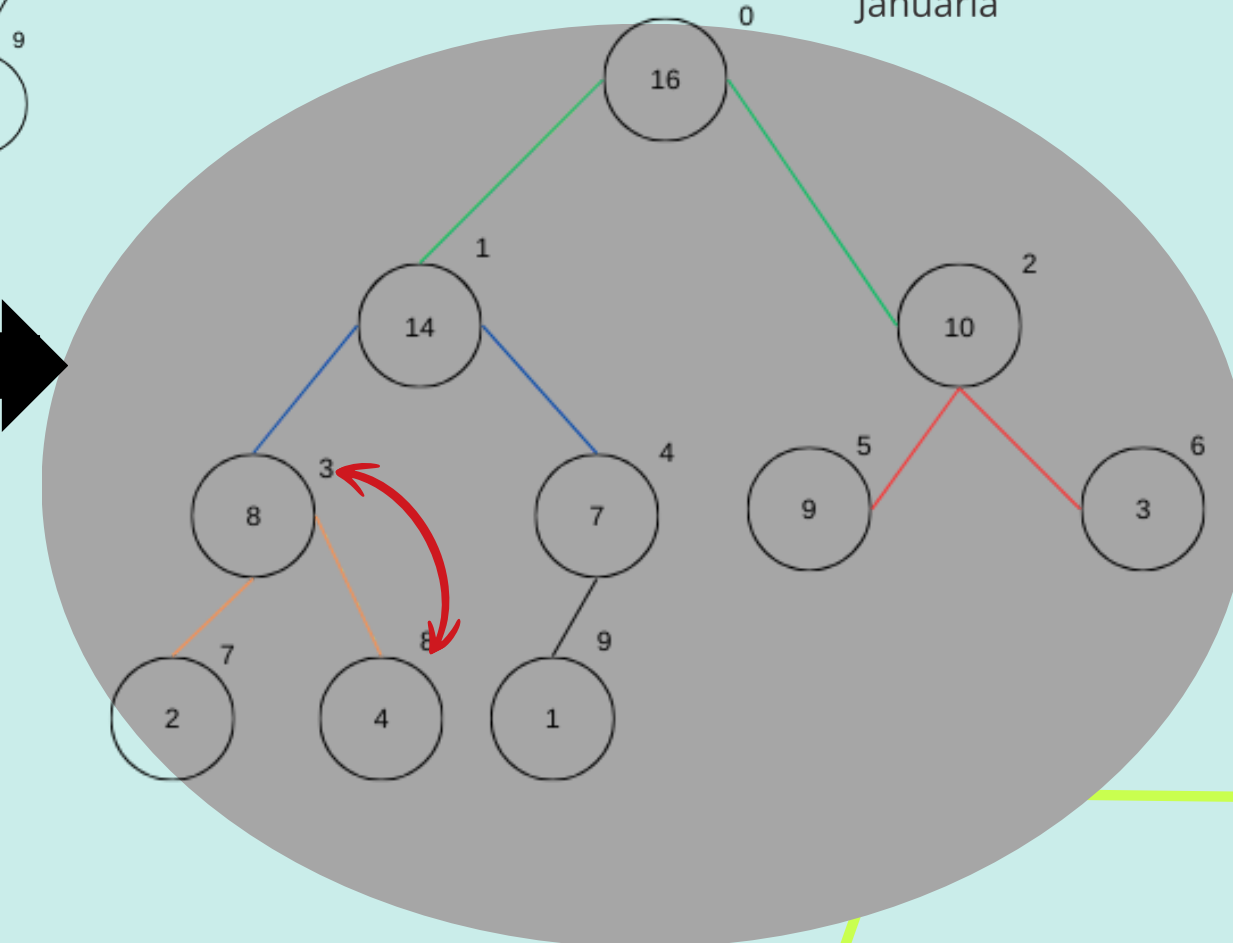
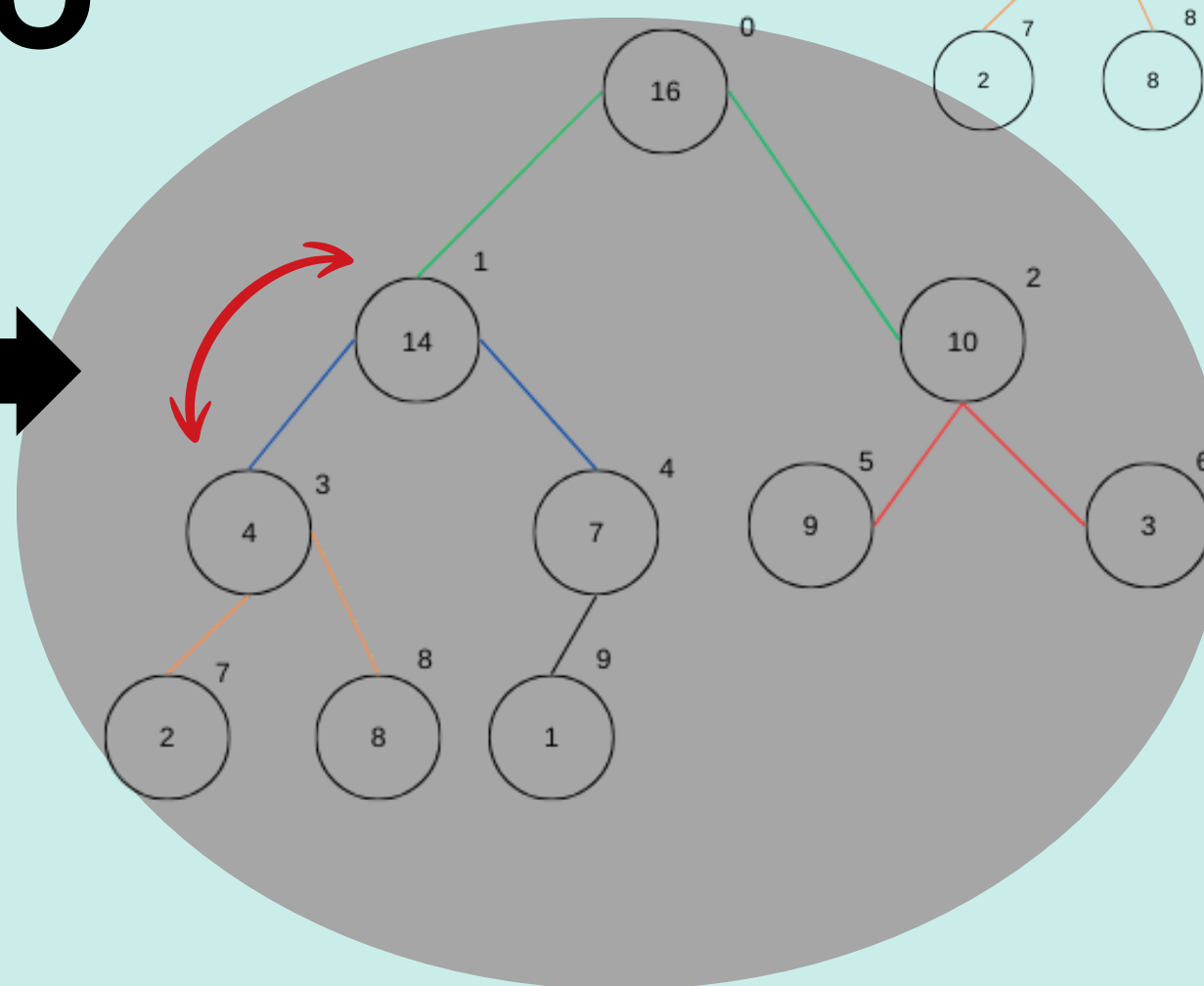
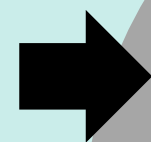
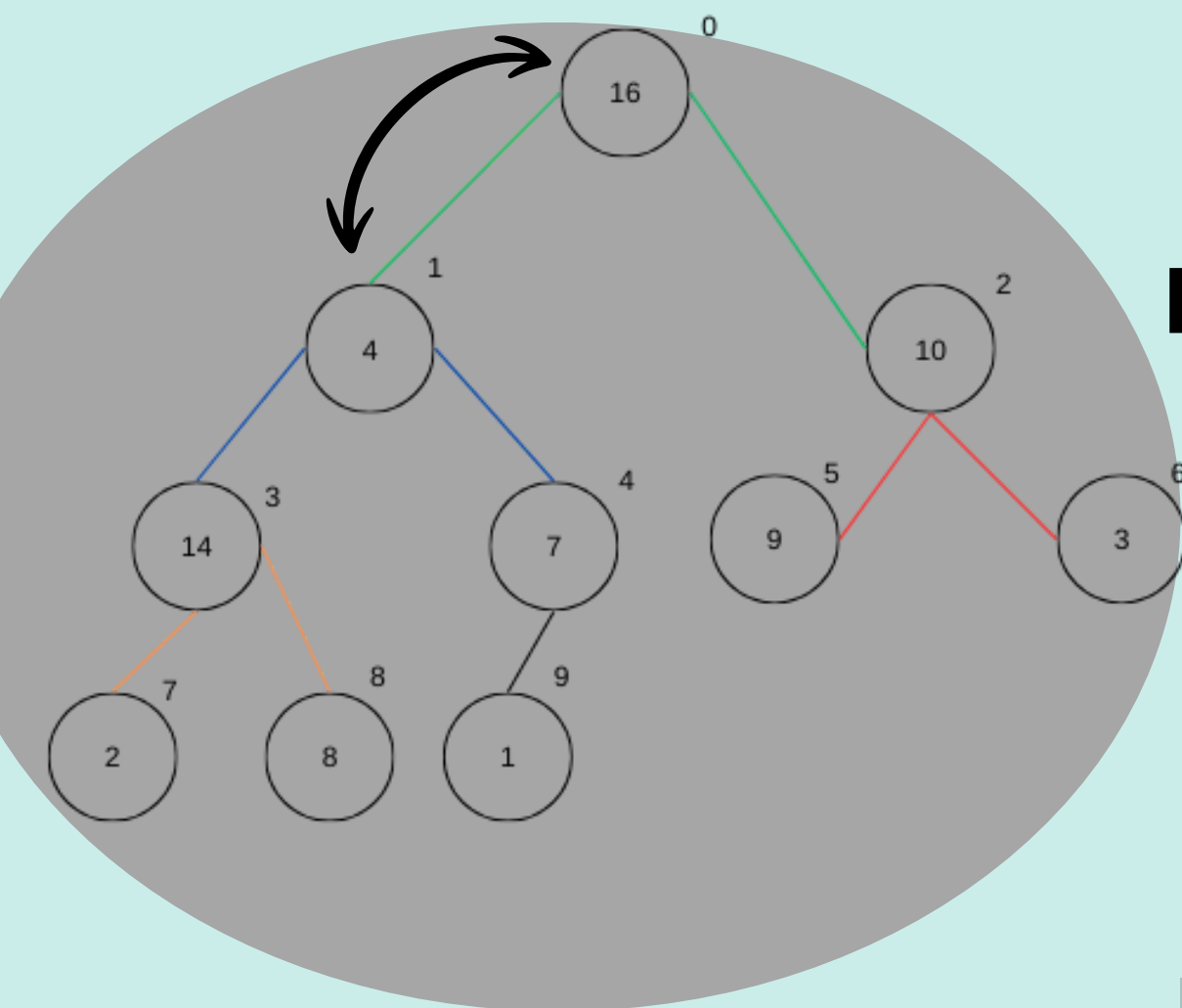
 **Observem que ao fazer essa troca, a sub-árvore de baixo deixou de ser uma heap, então é necessário fazer mais uma troca.**



Quando há troca num nó pai, suas subárvores são analisadas novamente.

Heapsort

Fase 1: EXEMPLO



0	1	2	3	4	5	6	7	8	9
16	14	10	8	7	9	3	2	4	1

O processo reinicia: os nodos são reavaliados de forma a colocar o maior elemento de cada subárvore na respectiva raiz (a última posição fica de fora, agora).

Heapsort

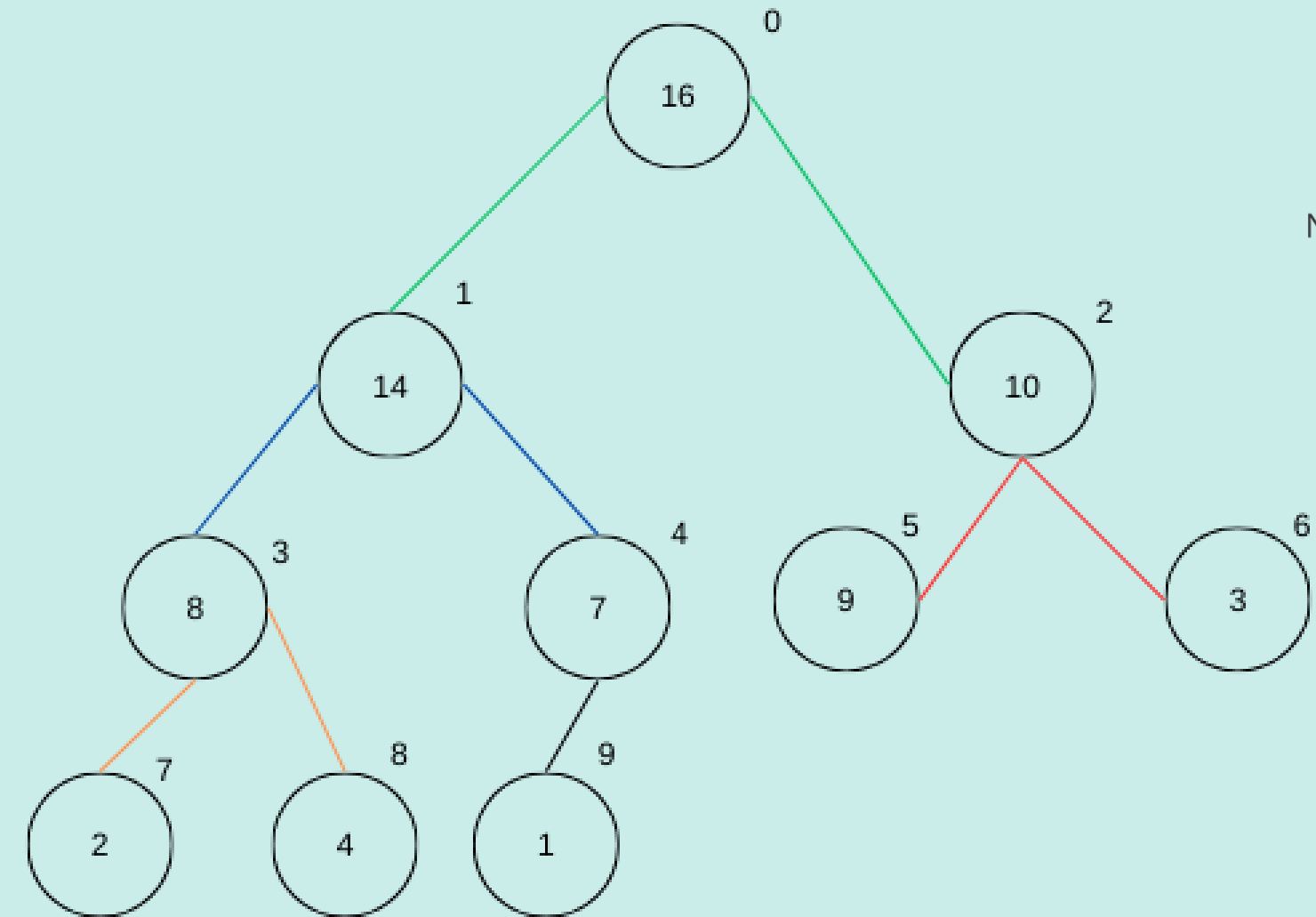
Fase 2

Uma vez que o heap máximo tenha sido construído, o maior elemento estará sempre na raiz da árvore (posição 0 do array). Esse elemento é trocado com o último elemento do array, que agora está na posição correta. Em seguida, o heap é reconstruído sem o último elemento, que já está na posição correta. Esse processo é repetido até que todos os elementos estejam na posição correta.

Heapsort

Fase 2: EXEMPLO

- Se a chave que está na raiz é a maior de todas, então sua posição definitiva correta, na ordem crescente, é na última posição do vetor;
- Então, esta maior chave é colocada na última posição do vetor, por trocar com a chave que ocupa aquela posição;

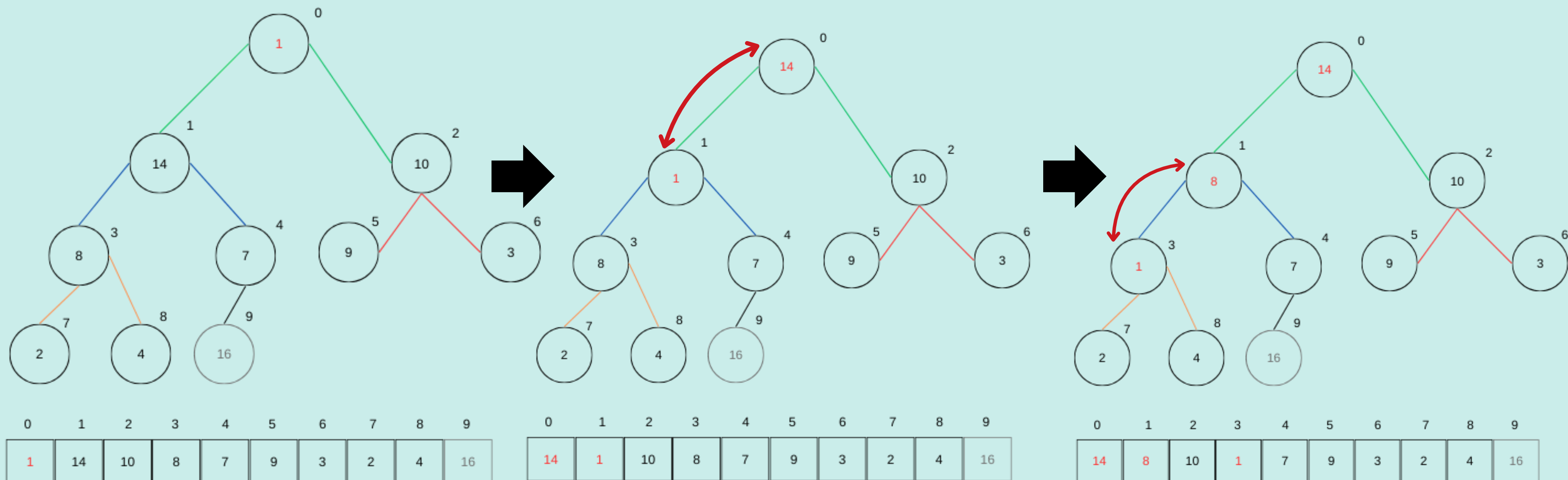


0	1	2	3	4	5	6	7	8	9
16	14	10	8	7	9	3	2	4	1

0	1	2	3	4	5	6	7	8	9
1	14	10	8	7	9	3	2	4	16

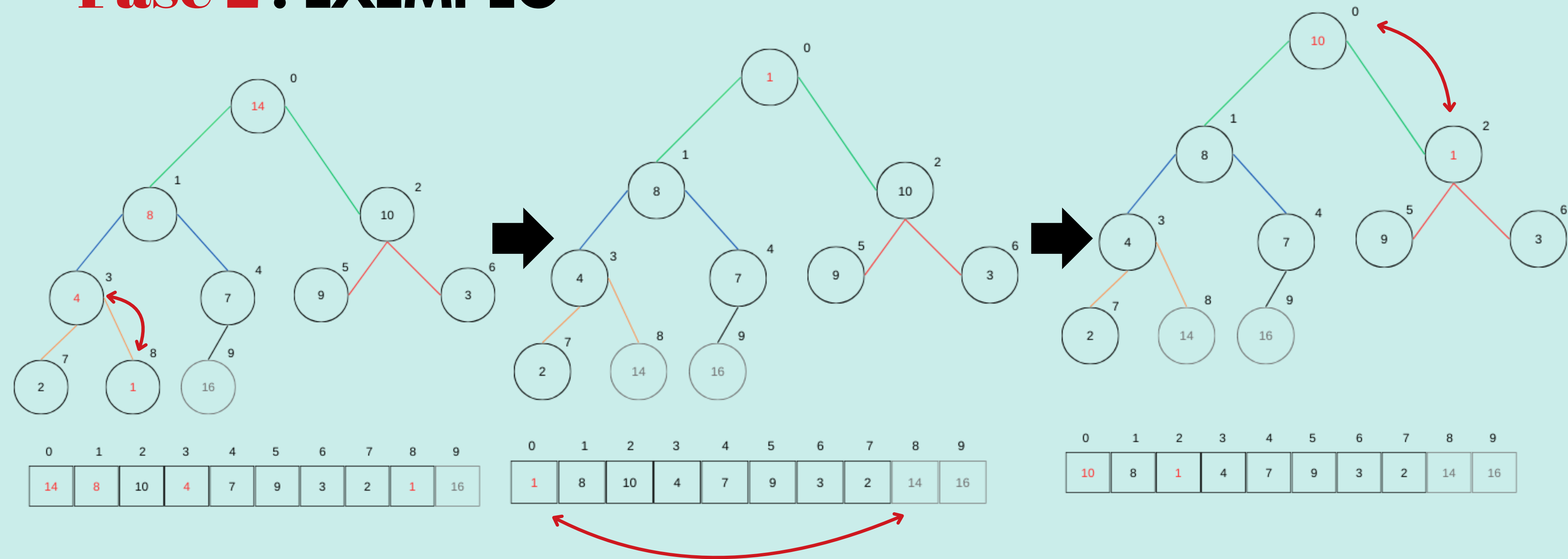
Heapsort

Fase 2: EXEMPLO



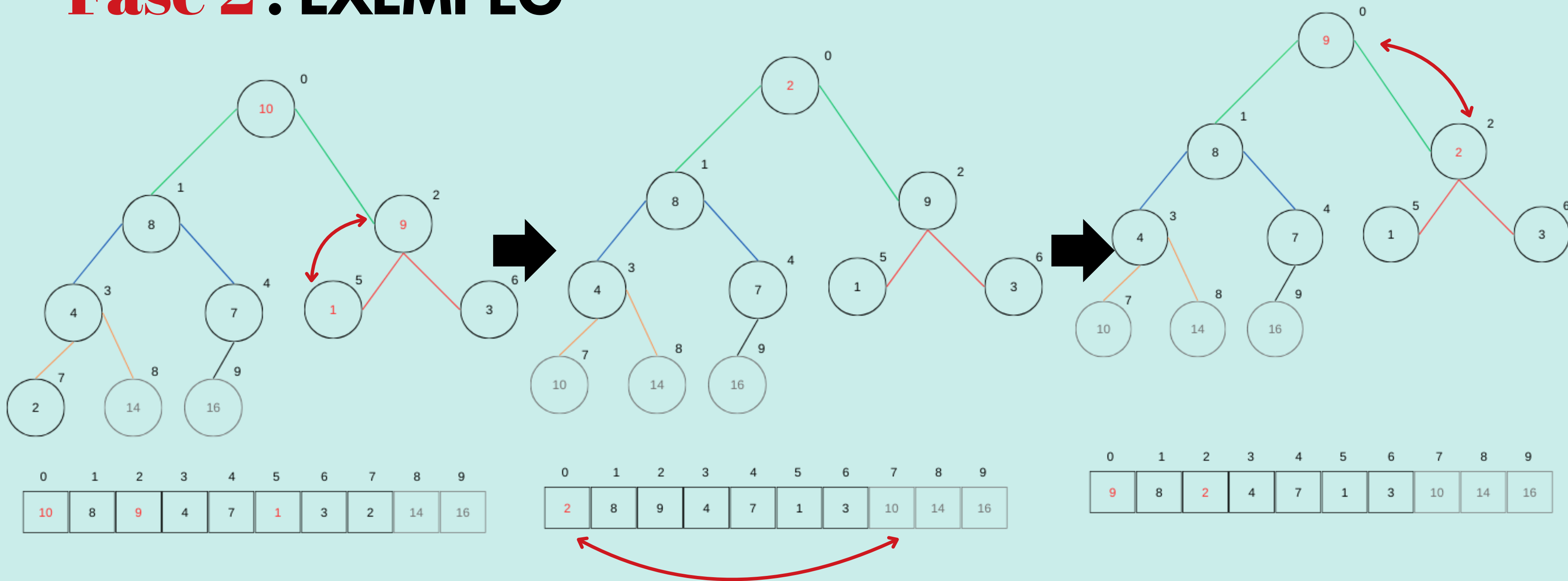
Heapsort

Fase 2: EXEMPLO



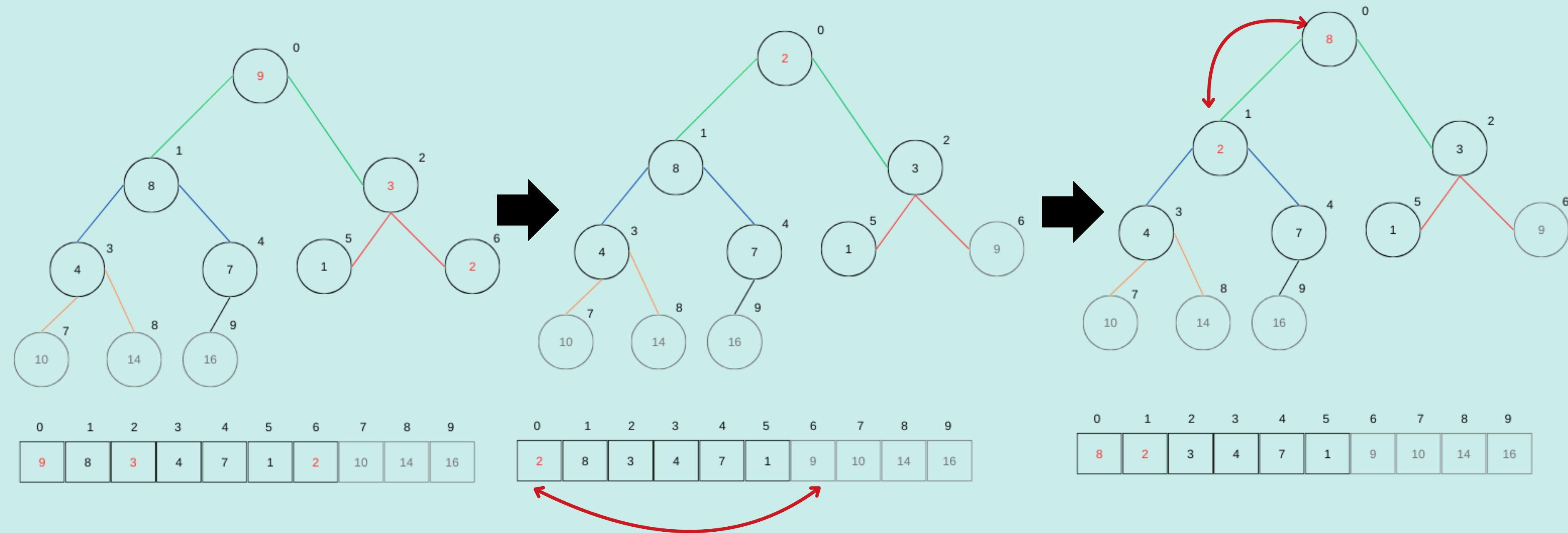
Heapsort

Fase 2: EXEMPLO



Heapsort

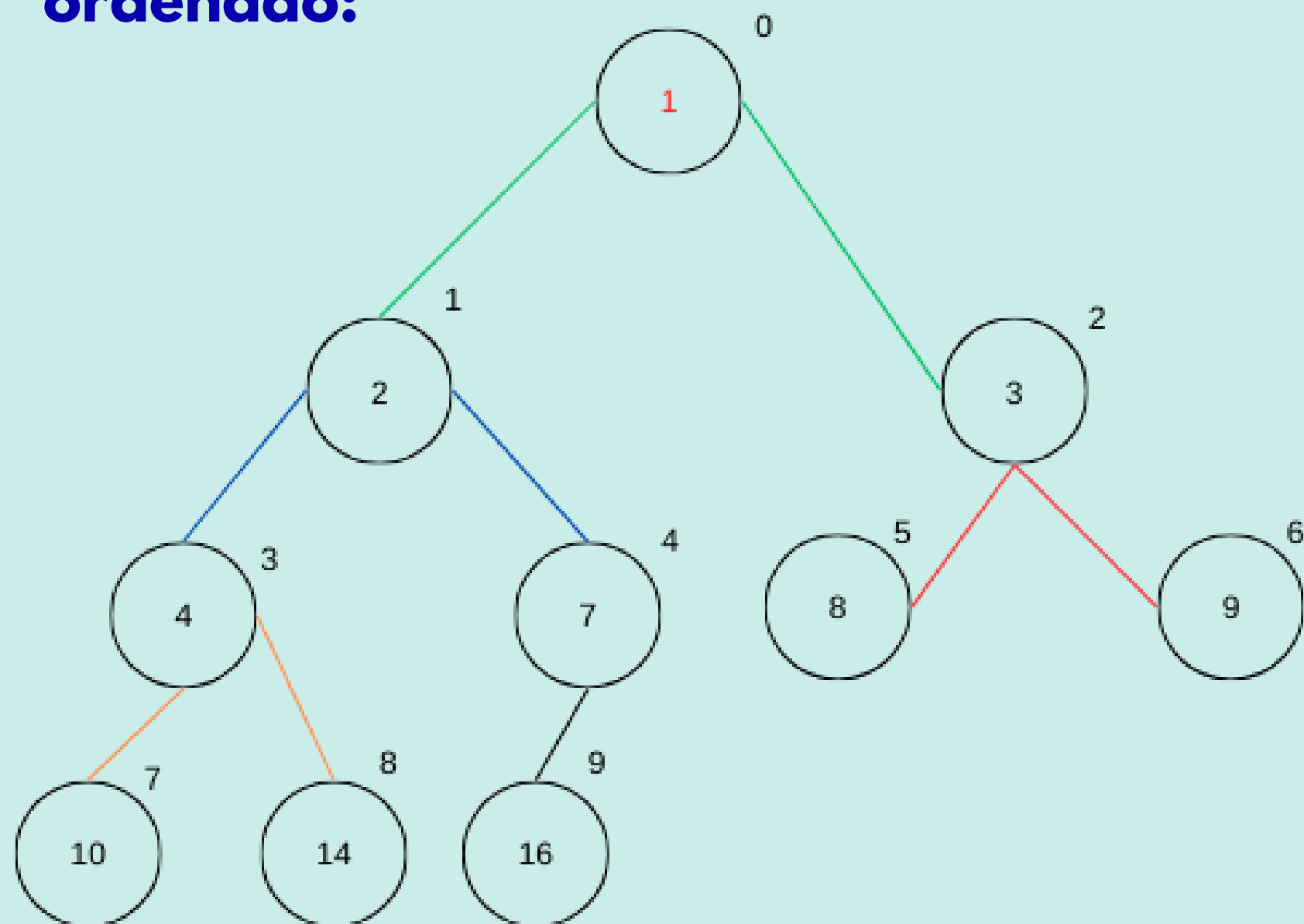
Fase 2: EXEMPLO



Heapsort

Fase 2 : EXEMPLO

E o processo continua repetindo, até que o vetor esteja completamente ordenado:



0	1	2	3	4	5	6	7	8	9
1	2	3	4	7	8	9	10	14	16

Heapsort

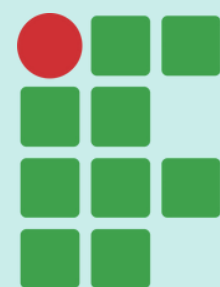
- **Complexidade de Tempo**

A complexidade de tempo do heapsort é $O(n * \log(n))$, onde “n” é o número de elementos a serem ordenados.

- **Complexidade de Espaço**

A complexidade de espaço do heapsort é $O(1)$, o que significa que o algoritmo consome uma quantidade de memória constante, independente do tamanho do array a ser ordenado.

OBRIGADO!



INSTITUTO FEDERAL
Norte de Minas Gerais
Campus Januária

BACHA. SISTEMAS DE INFORMAÇÃO