



Fonte: luisdev.com.br

Estruturas de Dados

FILAS DE PRIORIDADES & HEAPS – Unidade 12

Prof. Kenia Kodel

Pontos Trabalhados na Disciplina

- (Plano da Disciplina) Ementa, Metodologia, Avaliação
- Introdução a Estruturas de Dados
- Linguagem C
- Variáveis Dinâmicas
- Arquivos Binários
- Eficiência de Algoritmos (Complexidade, Notação O)
- Listas Lineares: Sequenciais, Encadeadas, Dinâmicas
- Listas Lineares Restritas: Pilhas e Filas
- Hashing
- Árvores, Árvores Binárias e Árvores Binárias de Pesquisa
- Árvores Balanceadas AVL

Filas de Prioridades

CONTEXTO & DEFINIÇÃO

- Aplicações comumente requerem que haja comparação e/ou classificação de objetos de acordo com parâmetros ou propriedades chamadas **chaves**.
- Formalmente **chave** é definida como um elemento que é associado a um objeto, como um atributo específico deste (objeto), que pode ser usado para identificar, classificar ou estabelecer prioridade entre estes (objetos).

Filas de Prioridades

CONTEXTO & DEFINIÇÃO

- Aplicações comumente requerem que haja comparação e/ou classificação de objetos de acordo com parâmetros ou propriedades chamadas **chaves**.
- Formalmente **chave** é definida como um elemento que é associado a um objeto, como um atributo específico deste objeto, que pode ser usado para identificar, classificar ou estabelecer prioridades entre estes (objetos).




Definir situações problemas e respectivas chaves.

Identificam? Classificam?

Estabelece prioridade?

Filas de Prioridades

CONTEXTO & DEFINIÇÃO



Uma **fila de prioridade** é um contêiner de elementos, cada um tendo uma chave associada, já definida no instante em que o elemento é inserido na estrutura; a qual estabelece a *prioridade* usada para definir a ordem de remoção dos elementos da fila.

Filas de Prioridades

APLICAÇÃO



O atendimento Prioritário é lei. A principal lei federal é a Lei 10.048/00, que estabelece, no artigo 1º, os grupos de pessoas que têm direito ao atendimento prioritário: pessoas com deficiência; idosos com idade igual ou superior a 60 anos; idosos acima de 80 anos que possuem prioridade perante os outros idosos (Lei Federal 13.466/17); gestantes; lactantes; pessoas com crianças de colo e obesos.

Essa é uma lei federal e se aplica a repartições públicas, instituições bancárias e transportes. Porém, ela serviu de base para outras leis e normas que determinam o atendimento prioritário em outros estabelecimentos. Todos os estados possuem suas leis sobre atendimento prioritário.

Filas de Prioridades

CONTEXTO & DEFINIÇÃO

Uma **fila de prioridade** é um contêiner de elementos, cada um tendo uma chave associada, já definida no instante em que o elemento é inserido na estrutura; a qual fornece a *prioridade* usada para definir a ordem de remoção dos elementos da **fila**.

O que é fila?



Filas de Prioridades

CONTEXTO & DEFINIÇÃO

Segundo Szwarcfiter e Markenzon (2015), corresponde a “lista em que a seus itens de composição são associados prioridades definidas em geral por valor numérico e que estabelece a ordem de remoção destes na estrutura”.

Filas de Prioridades

CONTEXTO & DEFINIÇÃO

Segundo Szwarcfiter e Markenzon (2015), corresponde a “lista em que a seus itens de composição são associados prioridades definidas em geral por valor numérico e que define a ordem de remoção destes na estrutura”.

***Identificar situação
problema cuja
solução ideal implica
no uso de fila de
prioridade***



Filas de Prioridades

APLICAÇÃO

- As filas de prioridades são aplicadas em situações em que existe característica que distingue os dados e estabelece prioridade entre estes.

1

Necessitam de atendimento imediato.
CASOS DE EMERGÊNCIA

2

Necessitam de atendimento praticamente imediato.
CASOS MUITO URGENTES

3

Necessitam de atendimento rápido, mas podem aguardar.
CASOS DE URGÊNCIA

4

Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS POUCO URGENTES

5

Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS NÃO URGENTES

Fila cuja prioridade é estabelecida pela gravidade do quadro de saúde, pelo risco.

Filas de Prioridades

APLICAÇÃO

- As filas de prioridades são aplicadas em situações em que existe característica que distingue os dados e estabelece prioridade entre estes.

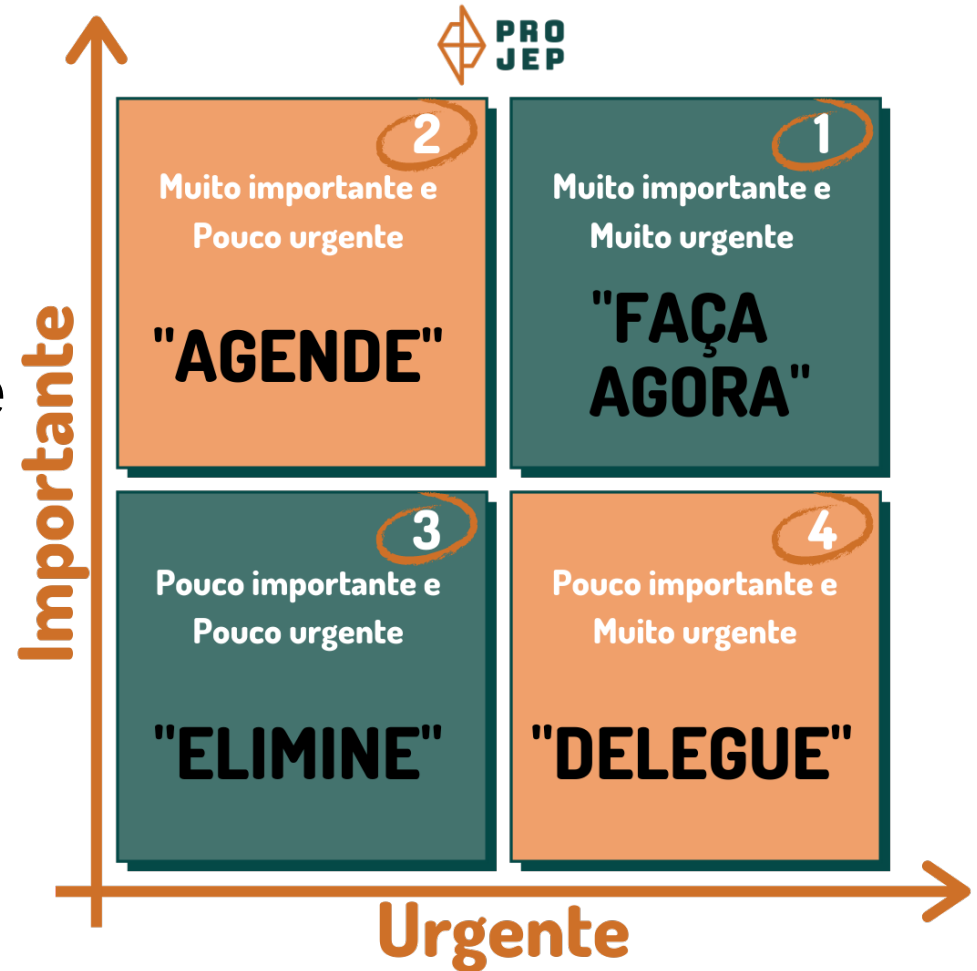


Fila de gestão do tempo cuja prioridade é estabelecida pelos critérios de importância e urgência.

Filas de Prioridades

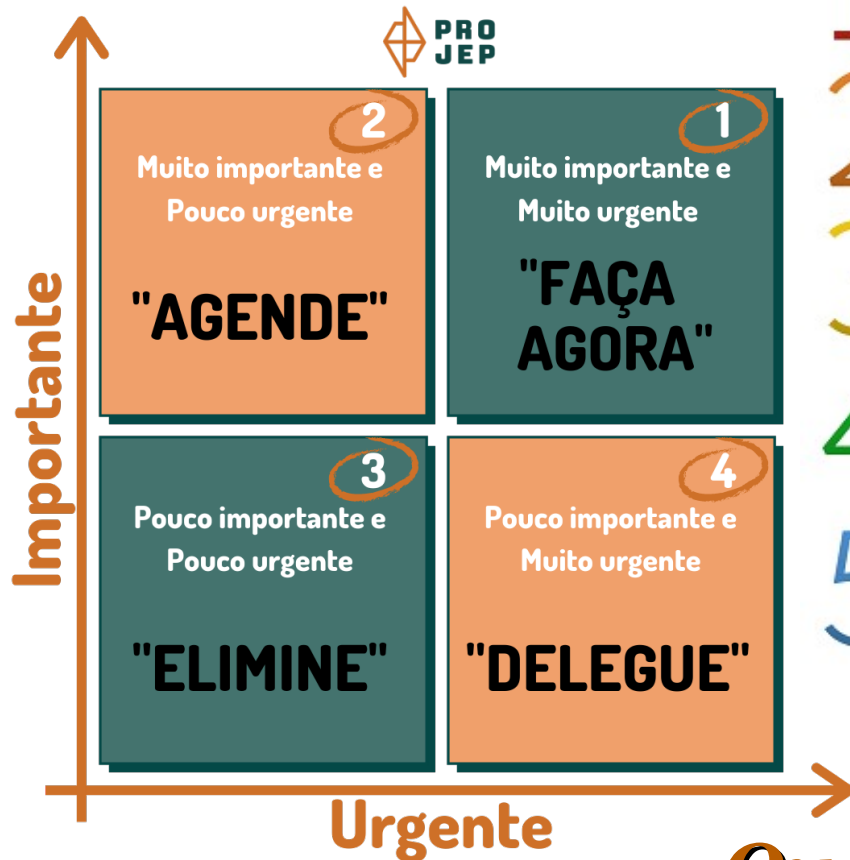
APLICAÇÃO

- Realizar tarefas conforme ordem de prioridades.
- Para gerir o tempo, é necessário escolher sucessivamente a tarefa de maior prioridade.
- Há variação da prioridade com o passar do tempo.
- Novas tarefas podem ser adicionadas na lista ao longo do tempo.



Filas de Prioridades

OPERAÇÕES



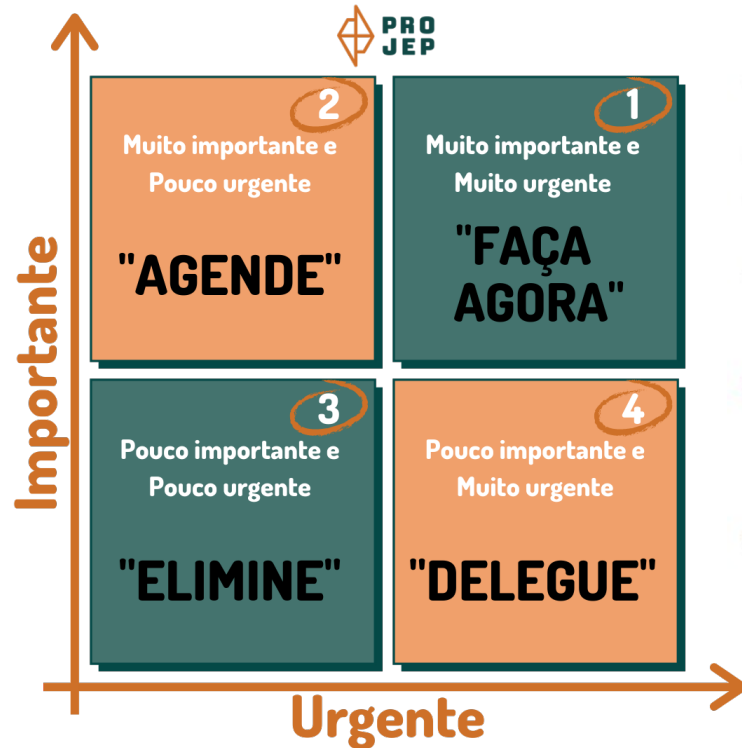
1. Necessitam de atendimento imediato. **CASOS DE EMERGÊNCIA**
2. Necessitam de atendimento praticamente imediato. **CASOS MUITO URGENTES**
3. Necessitam de atendimento rápido, mas podem aguardar. **CASOS DE URGÊNCIA**
4. Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde. **CASOS POUCO URGENTES**
5. Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde. **CASOS NÃO URGENTES**

Que operações são úteis à manipulação de dados mantidos em filas de prioridade?



Filas de Prioridades

OPERAÇÕES



- 1 Necessitam de atendimento imediato.
CASOS DE EMERGÊNCIA
- 2 Necessitam de atendimento praticamente imediato.
CASOS MUITO URGENTES
- 3 Necessitam de atendimento rápido, mas podem aguardar.
CASOS DE URGÊNCIA
- 4 Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS POUCO URGENTES
- 5 Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS NÃO URGENTES

1. **Seleção** de elemento de maior prioridade.
2. **Inserção** de novo elemento.
3. **Remoção** de elemento de maior prioridade.
4. **Alteração** de prioridades.

Filas de Prioridades

OPERAÇÕES

1. **Seleção** de elemento de maior prioridade.
2. **Inserção** de novo elemento.
3. **Remoção** de elemento de maior prioridade.
4. **Alteração** de prioridades.

1

Necessitam de atendimento imediato.
CASOS DE EMERGÊNCIA

2

Necessitam de atendimento praticamente imediato.
CASOS MUITO URGENTES

3

Necessitam de atendimento rápido, mas podem aguardar.
CASOS DE URGÊNCIA

4

Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS POUCO URGENTES

5

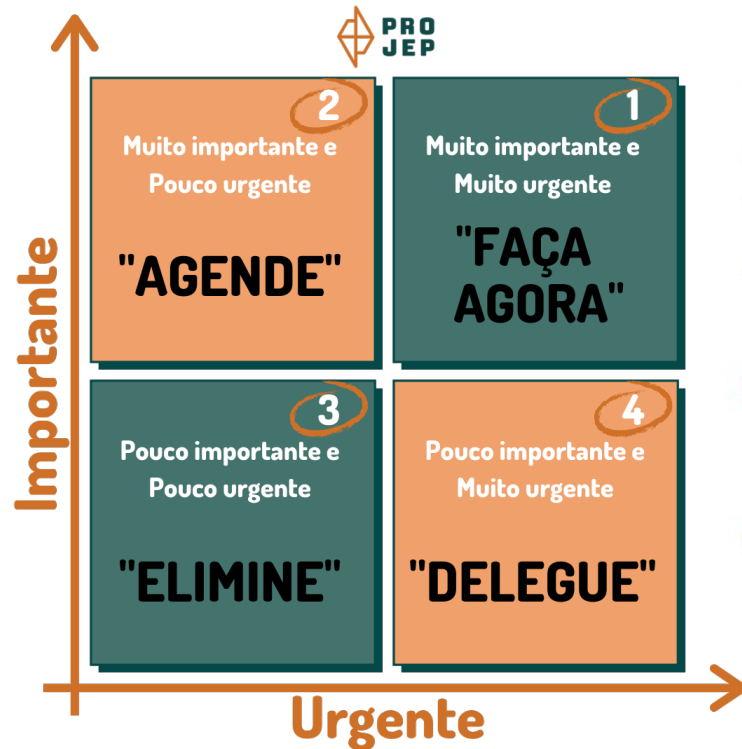
Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS NÃO URGENTES



**Operações úteis à
resolução desta
situação problema? Em
quais circunstâncias?**

Filas de Prioridades

IMPLEMENTAÇÃO



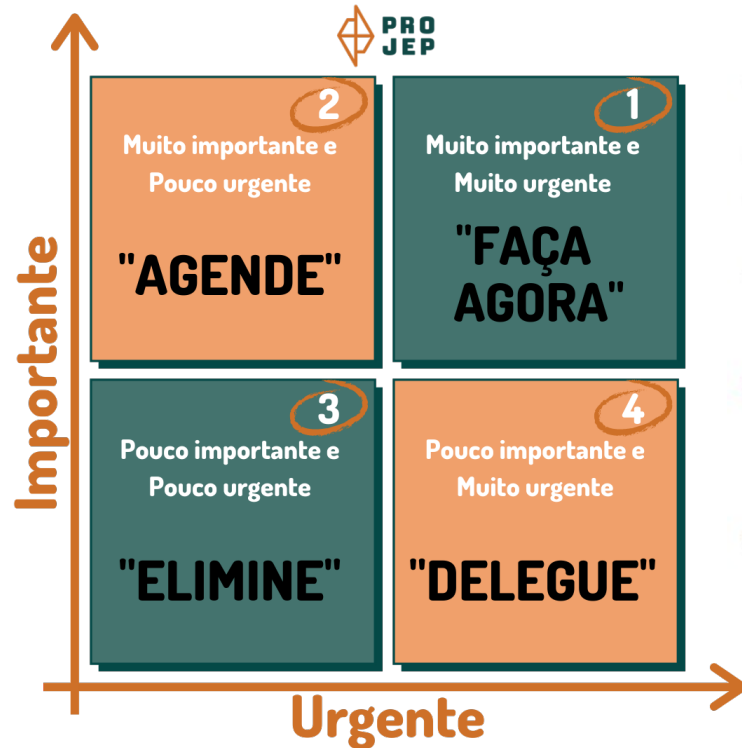
1. Necessitam de atendimento imediato.
CASOS DE EMERGÊNCIA
2. Necessitam de atendimento praticamente imediato.
CASOS MUITO URGENTES
3. Necessitam de atendimento rápido, mas podem aguardar.
CASOS DE URGÊNCIA
4. Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS POUCO URGENTES
5. Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS NÃO URGENTES

*Como implementar
filas de prioridade?
Hashing, AVL?*



Filas de Prioridades

IMPLEMENTAÇÃO



1. Necessitam de atendimento imediato.
CASOS DE EMERGÊNCIA
2. Necessitam de atendimento praticamente imediato.
CASOS MUITO URGENTES
3. Necessitam de atendimento rápido, mas podem aguardar.
CASOS DE URGÊNCIA
4. Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS POUCO URGENTES
5. Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS NÃO URGENTES

Organização da Estrutura

Ordenada Fisicamente
Ordenada por Link
Desordenada

Tipo de Estrutura

Dinâmica
Array
Arquivo

Filas de Prioridades **IMPLEMENTAÇÃO**

Como a prioridade é tratada num array desordenado, num array ordenado?

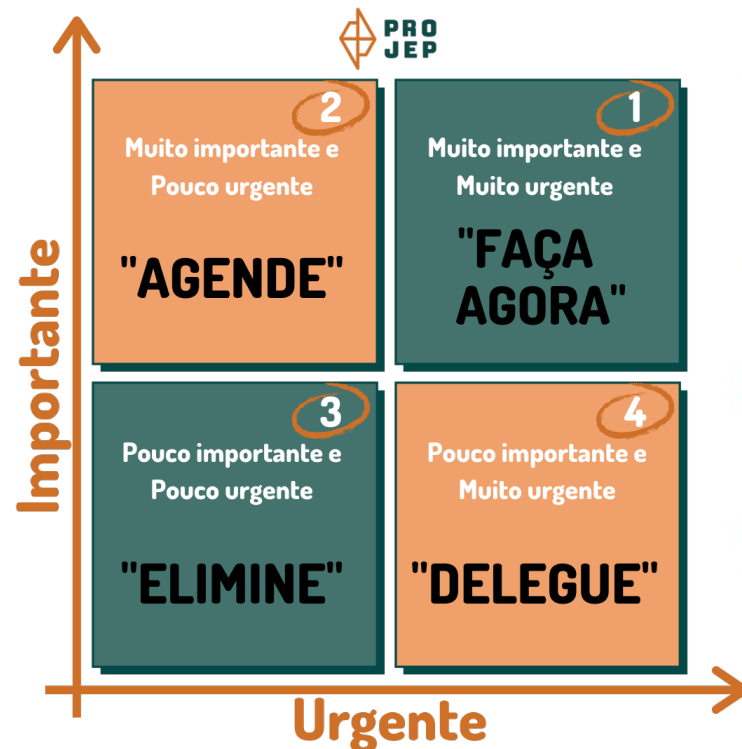


- 1 Necessitam de atendimento imediato.
CASOS DE EMERGÊNCIA
- 2 Necessitam de atendimento praticamente imediato.
CASOS MUITO URGENTES
- 3 Necessitam de atendimento rápido, mas podem aguardar.
CASOS DE URGÊNCIA
- 4 Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS POUCO URGENTES
- 5 Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS NÃO URGENTES

Filas de Prioridades **IMPLEMENTAÇÃO**



*Como a prioridade é tratada
numa lista dinâmica
desordenado, numa lista
dinâmica ordenado?*



1. Necessitam de atendimento imediato.
CASOS DE EMERGÊNCIA
2. Necessitam de atendimento praticamente imediato.
CASOS MUITO URGENTES
3. Necessitam de atendimento rápido, mas podem aguardar.
CASOS DE URGÊNCIA
4. Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS POUCO URGENTES
5. Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS NÃO URGENTES

Filas de Prioridades

IMPLEMENTAÇÃO

1. **Seleção** de elemento de maior prioridade.
2. **Inserção** de novo elemento.
3. **Remoção** de elemento de maior prioridade.
4. **Alteração** de prioridades.

Como implementar as operações dadas, em fila de prioridade usando as organizações e os tipos de estruturas propostas?



1. Necessitam de atendimento imediato. CASOS DE EMERGÊNCIA
2. Necessitam de atendimento praticamente imediato. CASOS MUITO URGENTES
3. Necessitam de atendimento rápido, mas podem aguardar. CASOS DE URGÊNCIA
4. Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde. CASOS POUCO URGENTES
5. Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde. CASOS NÃO URGENTES

Organização da Estrutura

Ordenada Fisicamente
Ordenada por Link
Desordenada

Tipo de Estrutura

Dinâmica
Array
Arquivo

Filas de Prioridades **IMPLEMENTAÇÃO**

Operações	Observações	Custos (Em Array)	
		Desordenada	Ordenada pela Prioridade
Seleção do elemento de maior prioridade	Dependem da ordenação.	$O(n)$, no pior caso quando se deve varrer toda a estrutura para identificar a maior prioridade.	$O(k)$, pois o elemento de maior prioridade é mantido no topo.
Inserção de novo elemento	Dependem da busca.	$O(k)$, já que novo item é inserido no final.	$O(n)$ no pior caso já que é possivelmente preciso localizar o ponto de inserção conforme a prioridade, e mover dados.
Remoção de elemento de maior prioridade		$O(n)$, no pior caso quando se deve varrer toda a estrutura para identificar a maior prioridade.	$O(k)$, pois o elemento de maior prioridade é mantido no topo.
Alteração de prioridade de dado elemento		$O(n)$ no pior caso quando se deve varrer toda a estrutura para identificar a elemento a ser alterado.	$O(n)$ no pior caso, pois o dado a ser alterado deve ser buscado, e em seguida a estrutura rearranjada com desclocamento.

Filas de Prioridades **IMPLEMENTAÇÃO**

Operações	Observações	Custos (Em Array)	
		Desordenada	Ordenada pela prioridade
Seleção do elemento de maior prioridade	Dependem da ordenação.	$O(n)$, no pior caso quando se deve varrer toda a estrutura para identificar a maior prioridade.	$O(k)$, pois o elemento de maior prioridade é mantido no topo.
Inserção de novo elemento	Dependem da busca.	$O(k)$, já que novo item é inserido no final.	$O(n)$ no pior caso já que é possivelmente preciso localizar o ponto de inserção conforme a prioridade, e mover dados.
Remoção de elemento de maior prioridade		$O(n)$, no pior caso quando se deve varrer toda a estrutura para identificar a maior prioridade.	$O(k)$, pois o elemento de maior prioridade é mantido no topo.
Alteração de prioridade de dado elemento		$O(n)$ no pior caso quando se deve varrer toda a estrutura para identificar a elemento a ser alterado.	$O(n)$ no pior caso, pois o dado a ser alterado deve ser buscado, e depois a estrutura rearranjada com deslocamento.

Filas de Prioridades **IMPLEMENTAÇÃO**

Operações	Observações	Custos (Implementado em Estrutura Dinâmica)	
		Desordenada	Ordenada pelas Prioridades
Seleção do elemento de maior prioridade	Dependem da ordenação.	$O(n)$, no pior caso quando se deve varrer toda a estrutura para identificar a maior prioridade.	$O(k)$, pois o elemento de maior prioridade é mantido no topo.
Inserção de novo elemento	Dependem da busca.	$O(k)$, já que novo item é inserido no final.	$O(n)$ no pior caso já que é preciso localizar o ponto de inserção conforme a prioridade.
Remoção de elemento de maior prioridade		$O(n)$, no pior caso quando se deve varrer toda a estrutura para identificar a maior prioridade.	$O(k)$, pois o elemento de maior prioridade é mantido no topo.
Alteração de prioridade de dado elemento		$O(n)$ no pior caso quando se deve varrer toda a estrutura para identificar a elemento a ser alterado.	$< O(n)$, pois é pesquisada região menor que a lista toda uma vez que se conhece a prioridade deste. No rearranjo também

Filas de Prioridades **IMPLEMENTAÇÃO**

Operações	Observações	Custos (Implementado em Estrutura Dinâmica)	
		Desordenada	Ordenada pelas Prioridades
Seleção do elemento de maior prioridade	Dependem da ordenação.	$O(n)$, no pior caso quando se deve varrer toda a estrutura para identificar a maior prioridade.	$O(k)$, pois o elemento de maior prioridade é mantido no topo.
Inserção de novo elemento	Dependem da busca.	$O(k)$, já que novo item é inserido no final.	$O(n)$ no pior caso já que é preciso localizar o ponto de inserção conforme a prioridade.
Remoção de elemento de maior prioridade		$O(n)$, no pior caso quando se deve varrer toda a estrutura para identificar a maior prioridade.	$O(k)$, pois o elemento de maior prioridade é mantido no topo.
Alteração de prioridade de dado elemento		$O(n)$ no pior caso quando se deve varrer toda a estrutura para identificar a elemento a ser alterado.	$< O(n)$, pois é pesquisada região menor que a lista toda uma vez que se conhece a prioridade deste. No rearranjo não há deslocamento.

Filas de Prioridades

IMPLEMENTAÇÃO

Operações	Observações	Custos (Implementado em Array)	
		Desordenada	Ordenada
Seleção do elemento de maior prioridade	Dependem da ordenação.	$O(n)$, no pior caso quando se deve varrer toda a estrutura para identificar a maior prioridade.	$O(k)$, pois o elemento é mantido no topo.
Inserção de novo elemento	Dependem da busca.	$O(k)$, já que novo item é inserido no final.	$O(n)$ no pior caso já que é preciso localizar o ponto de inserção conforme a prioridade.
Remoção de elemento de maior prioridade		$O(n)$, no pior caso quando se deve varrer toda a estrutura para identificar a maior prioridade.	$O(k)$, pois o elemento de maior prioridade é mantido no topo.
Alteração de prioridade de dado elemento		$O(n)$ no pior caso quando se deve varrer toda a estrutura para identificar a região menor que a lista toda uma vez que se conhece a prioridade deste. No rearranjo não há deslocamento.	

Operações	Observações	Custos (Implementado em Estrutura Dinâmica)	
		Desordenada	Ordenada
Seleção do elemento de maior prioridade	Dependem da ordenação.	$O(n)$, no pior caso quando se deve varrer toda a estrutura para identificar a maior prioridade.	$O(k)$, pois o elemento de maior prioridade é mantido no topo.
Inserção de novo elemento	Dependem da busca.	$O(k)$, já que novo item é inserido no final.	$O(n)$ no pior caso já que é preciso localizar o ponto de inserção conforme a prioridade.
Remoção de elemento de maior prioridade		$O(n)$, no pior caso quando se deve varrer toda a estrutura para identificar a maior prioridade.	$O(k)$, pois o elemento de maior prioridade é mantido no topo.
Alteração de prioridade de dado elemento		$O(n)$ no pior caso quando se deve varrer toda a estrutura para identificar a região menor que a lista toda uma vez que se conhece a prioridade deste. No rearranjo não há deslocamento.	



E quando implementado por arquivo?

Filas de Prioridades

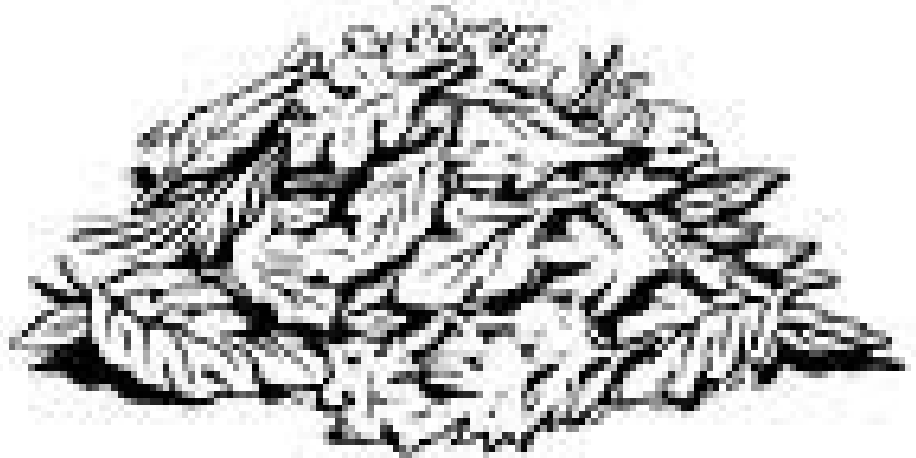
REFLEXÃO

Quando mantém-se filas de prioridades ordenadas, o gargalo da manutenção destas reside na inserção. Já quando estas são mantidas não ordenadas, o gargalo passa a ser as remoções e consultas. Assim, diante de uma situação problema a ser resolvida com aplicação de filas de prioridade, que critério usar para decidir se manter esta ordenada ou desordenada? Justifique sua resposta:



Heap

ESTRUTURA DE DADOS



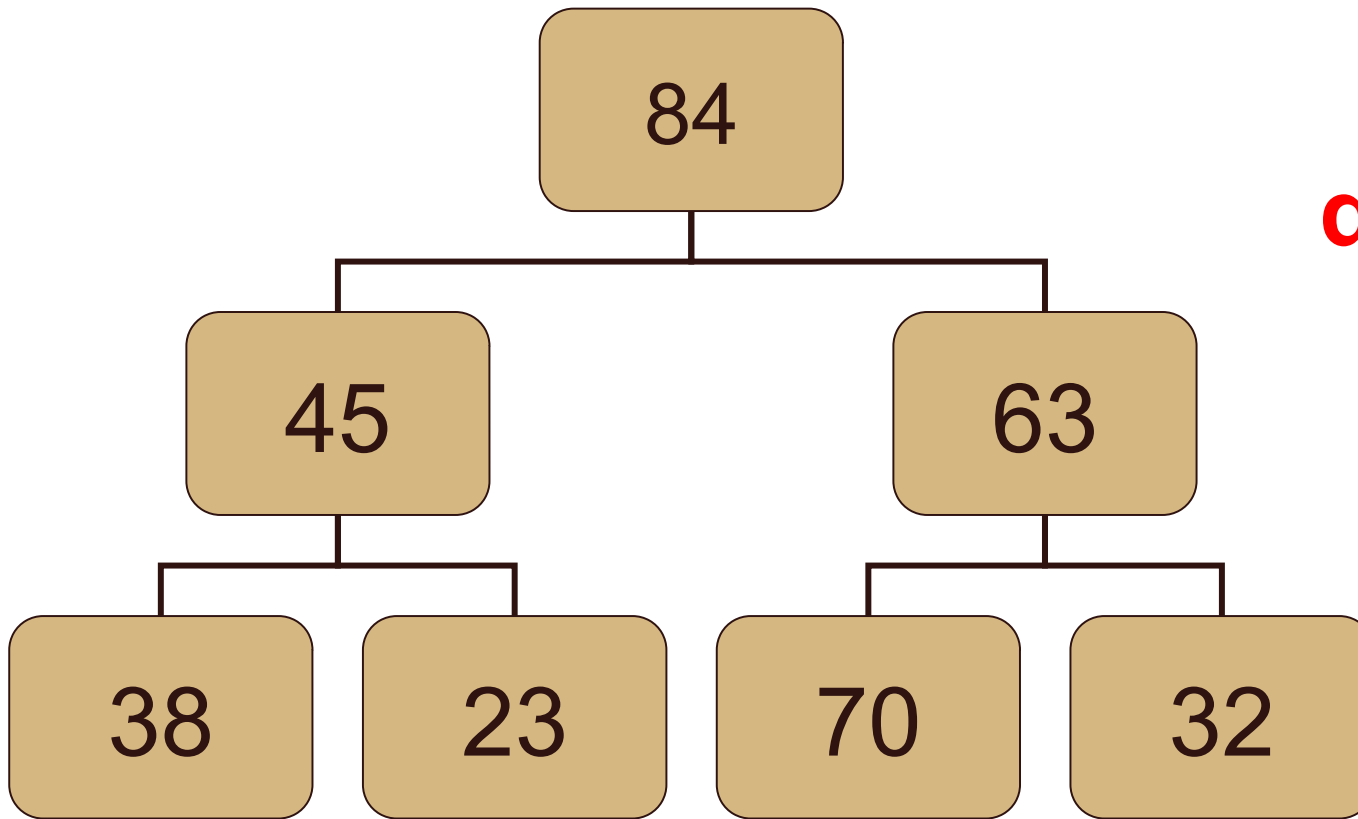
HEAP

definição

- Denomina-se **heap** a árvore binária completa, onde o valor contido em cada nó é maior que os valores de seus sucessores (ou descendentes).
- Uma árvore **completa** A é aquela em que, se n é um nó de A com alguma subárvore vazia, então n se localiza no penúltimo ou no último nível.

HEAP

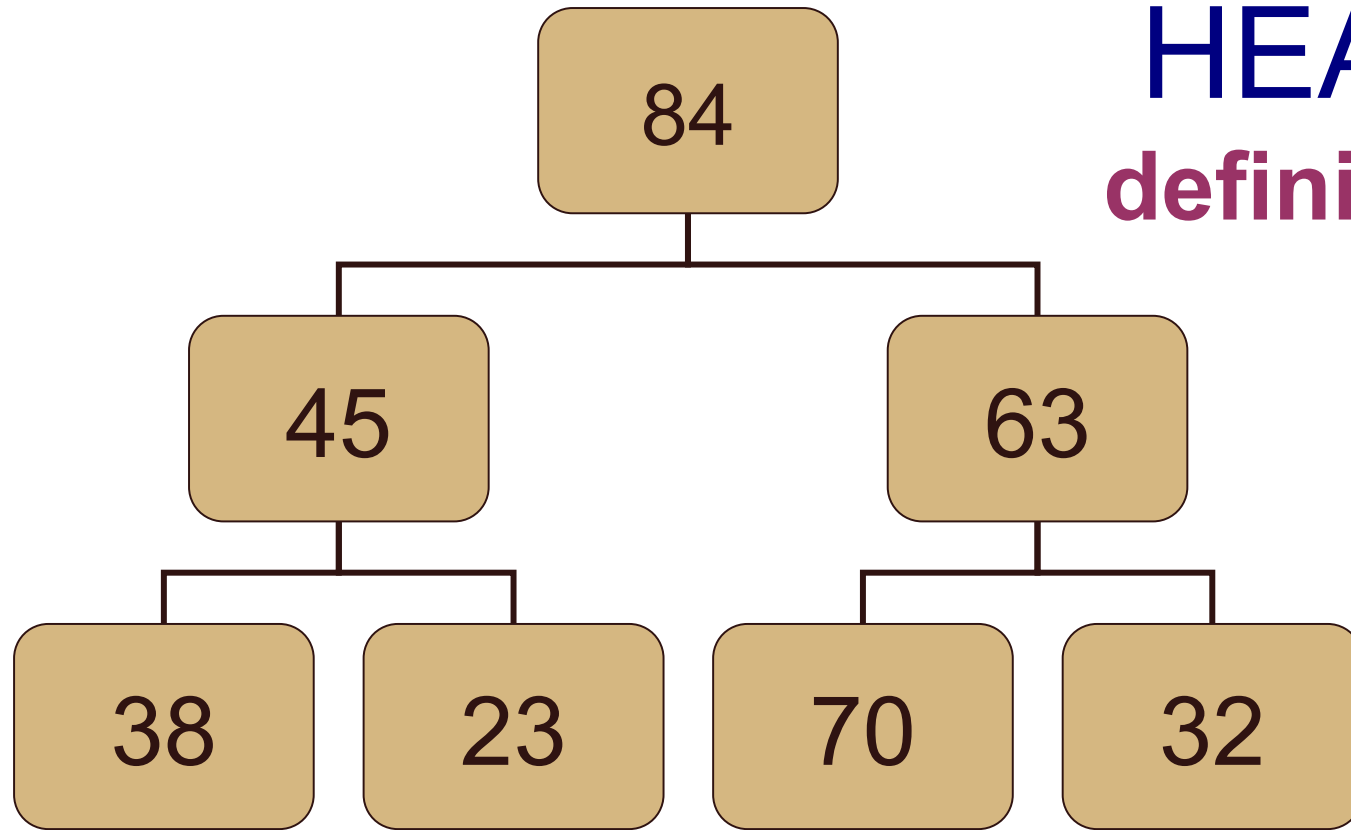
definição



*Sendo **heap** uma árvore binária completa, onde o valor contido em cada nó é maior que os valores de seus sucessores; a estrutura dada corresponde a um heap?*

HEAP

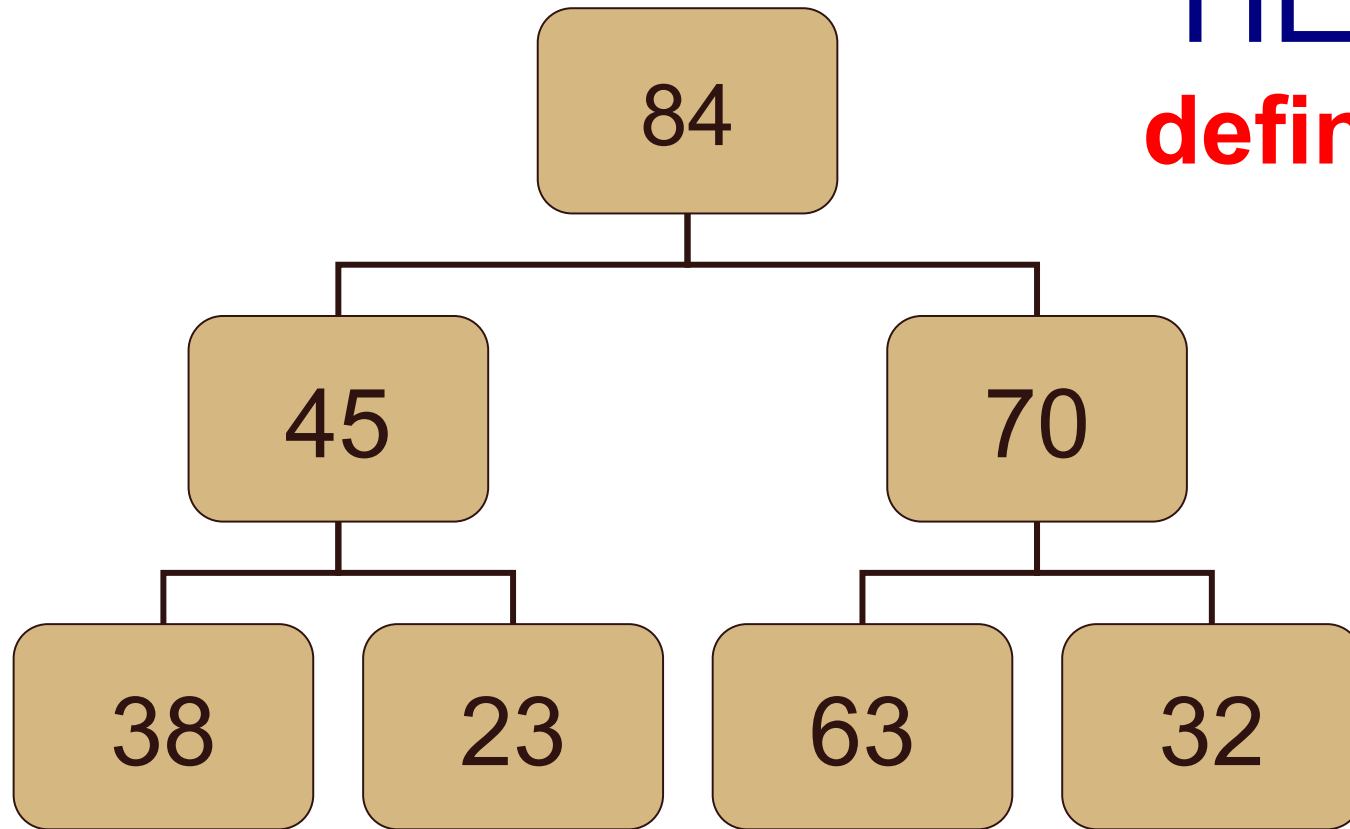
definição



Esta não constitui um **heap** porque o nó que contém 63 viola a regra de que o valor contido em cada nó deve ser maior que os valores de seus sucessores (tem descendente com valor 70).

HEAP

definição



Esta é **heap** - árvore binária completa onde o valor contido em cada nó é maior que os valores mantidos em seus sucessores.

HEAP

inserção

No processo de inserção todas as propriedades do heap devem ser preservadas:

- Consistir numa árvore binária.
- O valor contido em cada nó deve ser maior que os valores de seus sucessores.
- Ser completa; se n é um nó do heap H com subárvore vazia, então n se localiza no penúltimo ou no último nível de H .

HEAP

inserção

Montando um heap com os dados:

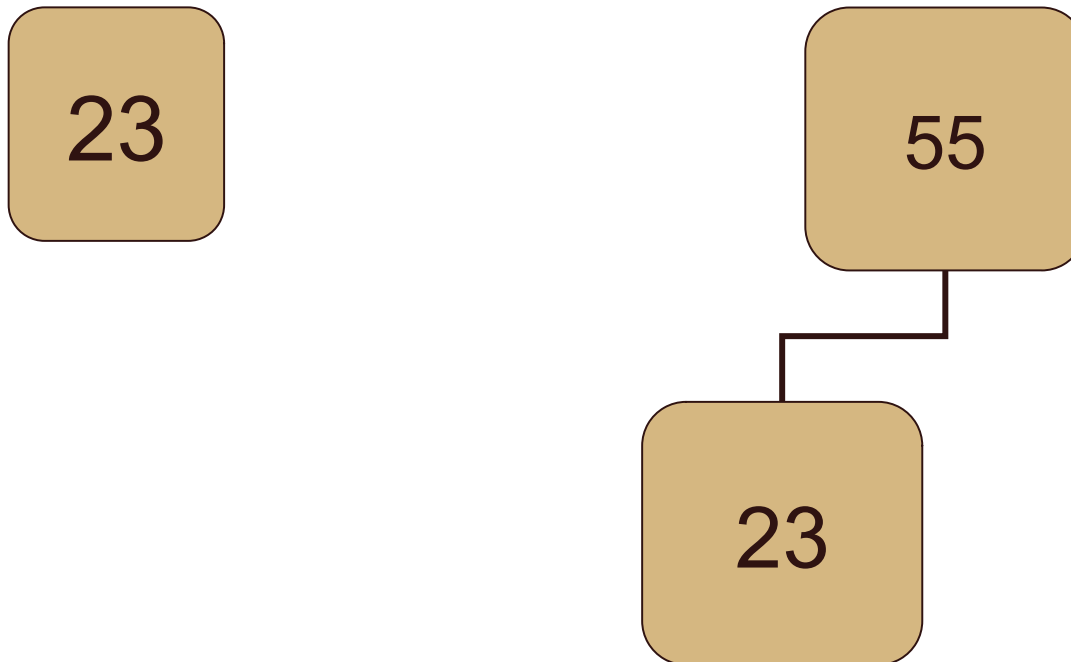
23 – 55 – 46 – 35 – 10 – 90

23

O primeiro elemento constitui a raiz do heap.

Dados: 23 – 55 – 46 – 35 – 10 – 90

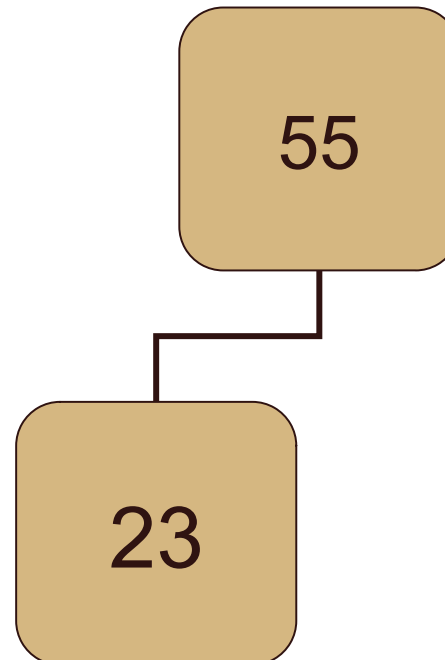
Para incluir 55, compara-se com a raiz 23, sendo o novo elemento 55 maior que 23, 23 (menor elemento) é deslocada para o próximo nível, compondo uma folha da estrutura e 55 assume a antiga posição do 23.



HEAP
inserção



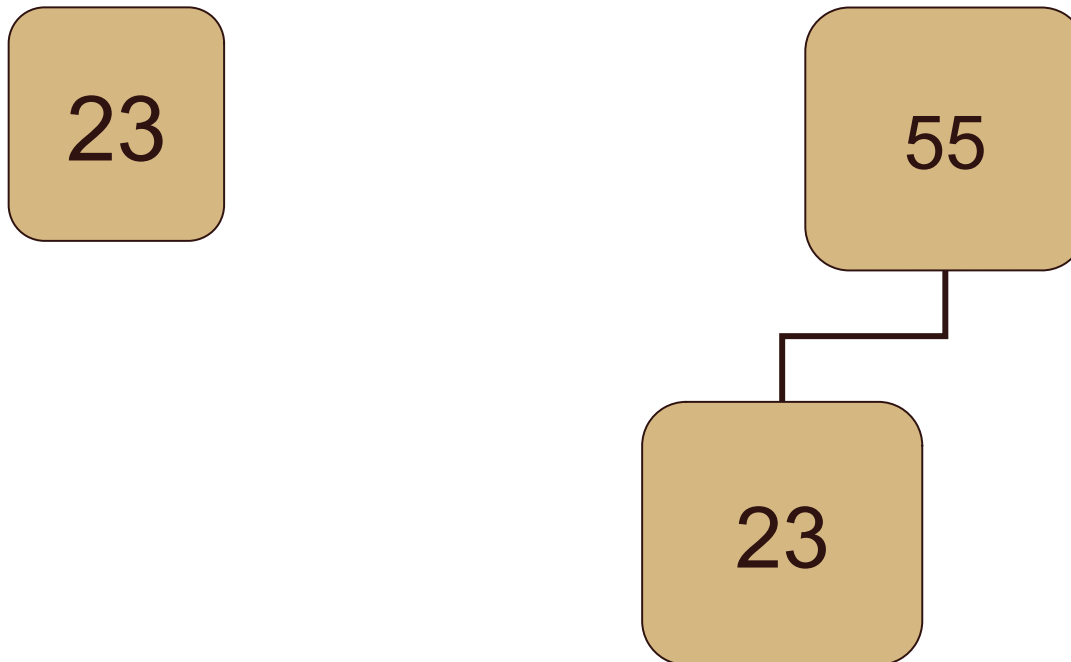
23 – 55 – 46 – 35 – 10 – 90 *Por que a inclusão do 55 provoca o deslocamento do nó 23 para o próximo nível?*



HEAP
inserção

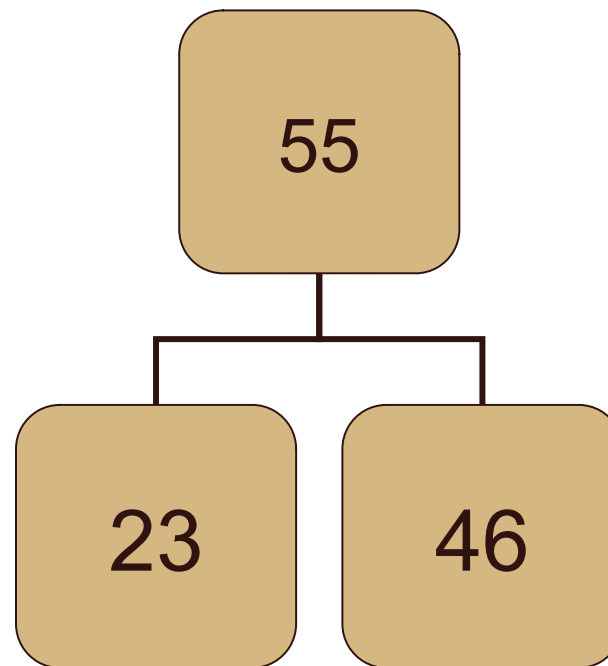
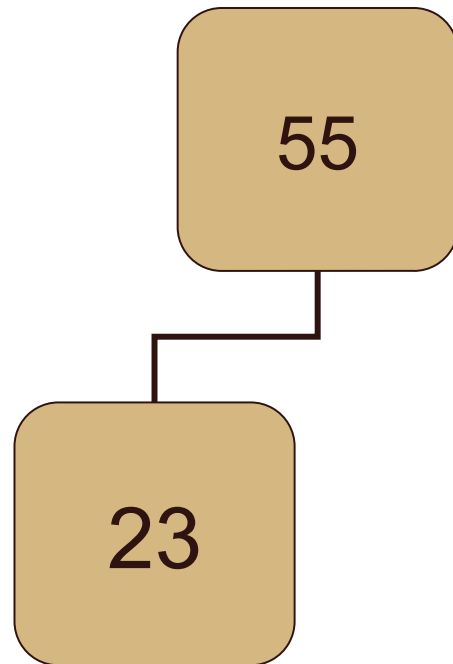
Dados: 23 – 55 – 46 – 35 – 10 – 90

A inclusão do 55 provoca o deslocamento do nó 23 para o próximo nível para garantir que todo nó N seja maior ou igual aos seus sucessores.



HEAP
inserção

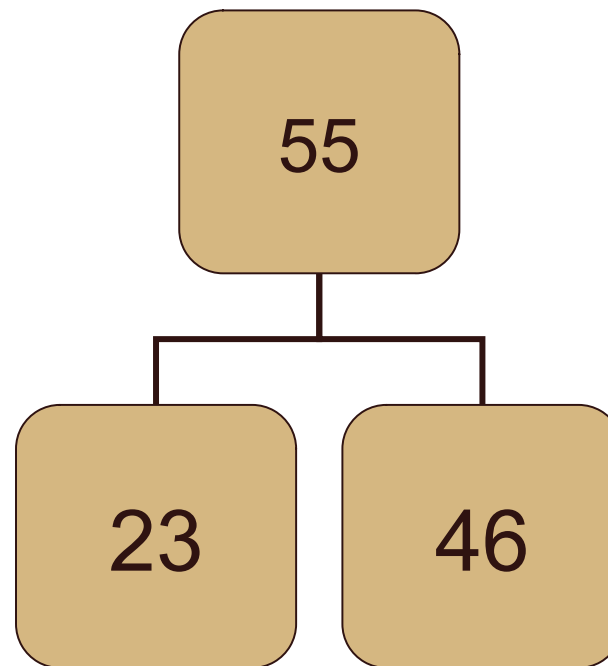
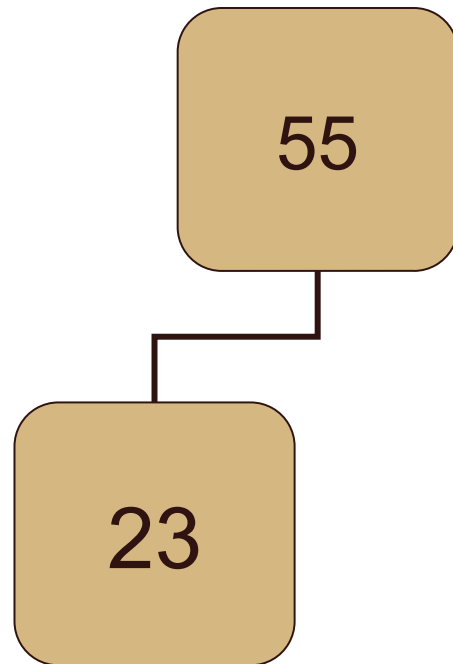
Dados: 23 – 55 – 46 – 35 – 10 – 90
Ao incluir 46, compara-se com a raiz 55, sendo o novo elemento 46 menor que 55, 46 é descolado para o próximo nível da estrutura, à direita do último nó incluído.



HEAP
inserção

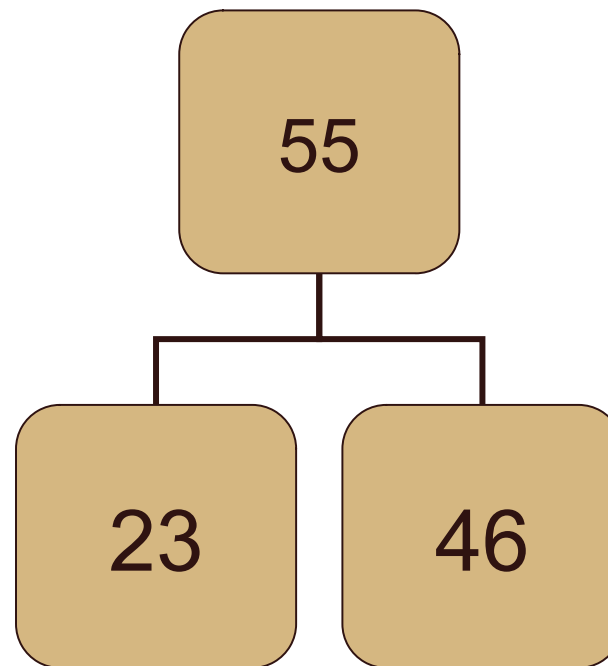
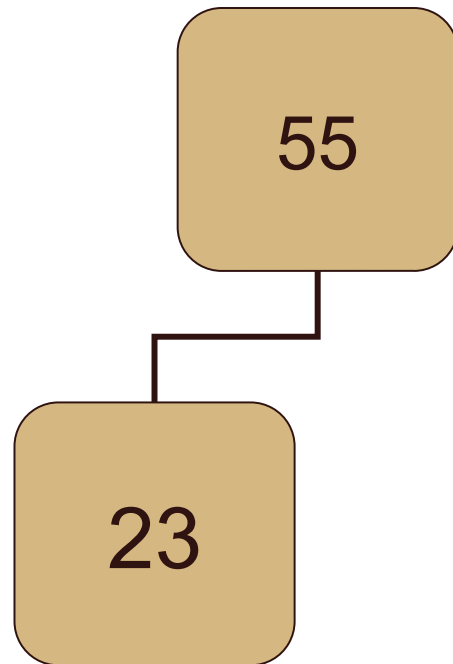
Dados: 23 – 55 – 46 – 35 – 10 – 90

Por que ao incluir 46, este é alocado na estrutura à direita do último nó incluído?



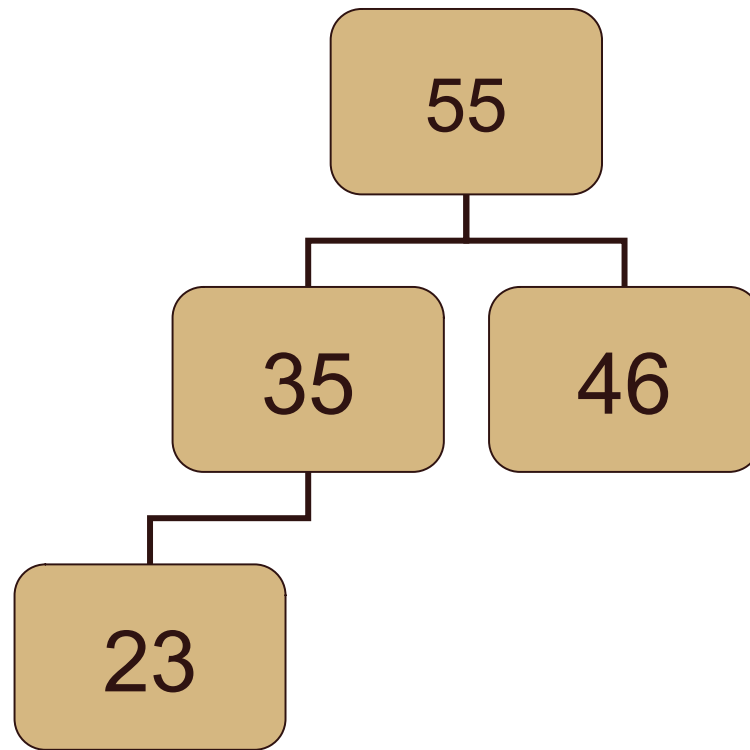
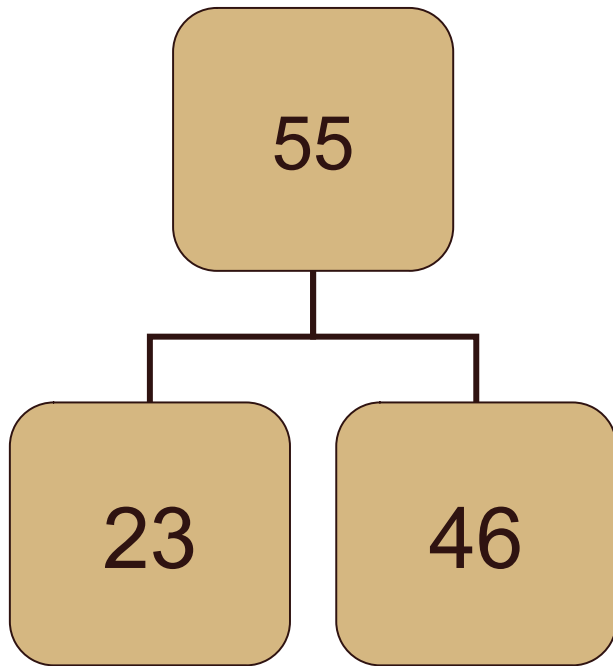
HEAP
inserção

Dados: 23 – 55 – 46 – 35 – 10 – 90
Na inclusão do nó 46 este é alocado à direita do último nó incluído para garantir que a árvore seja completa. Então diz-se que um heap cresce por nível, descendo neste exemplo.



HEAP
inserção

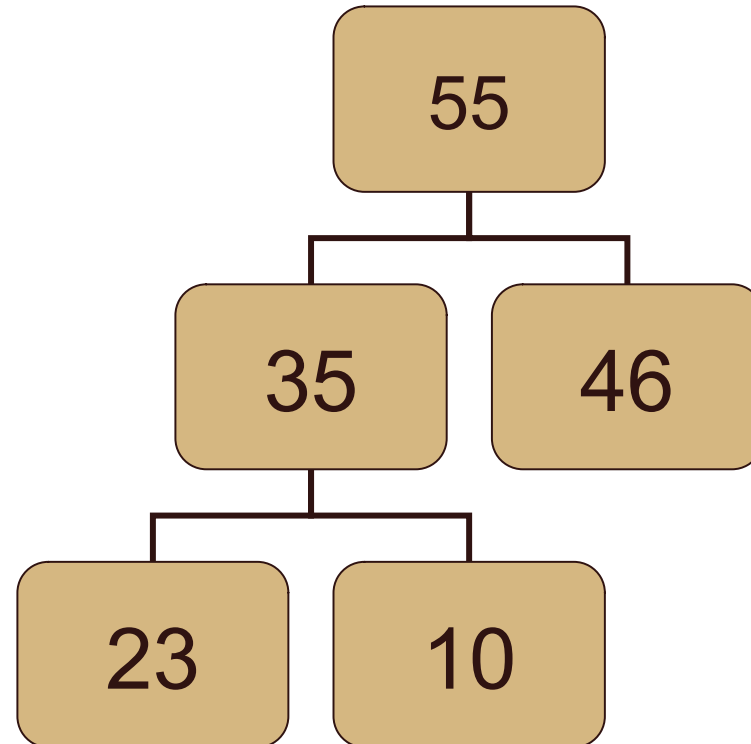
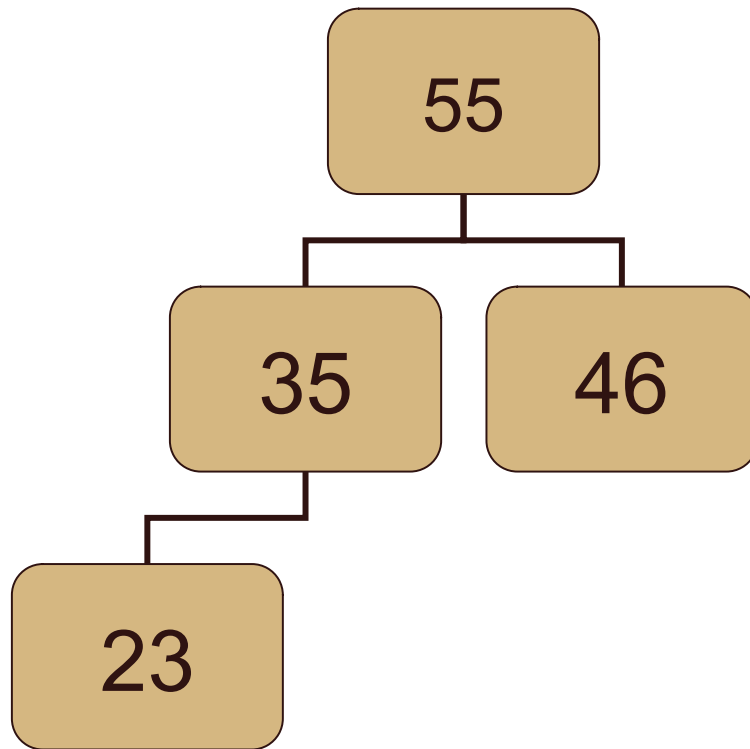
Dados: 23 – 55 – 46 – 35 – 10 – 90
Incluindo 35...



HEAP
inserção

Dados: 23 – 55 – 46 – 35 – 10 – 90

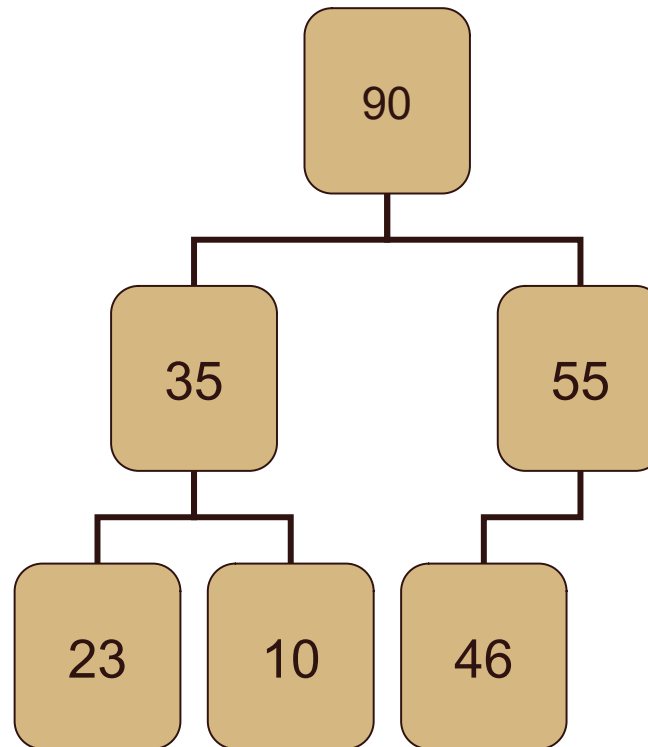
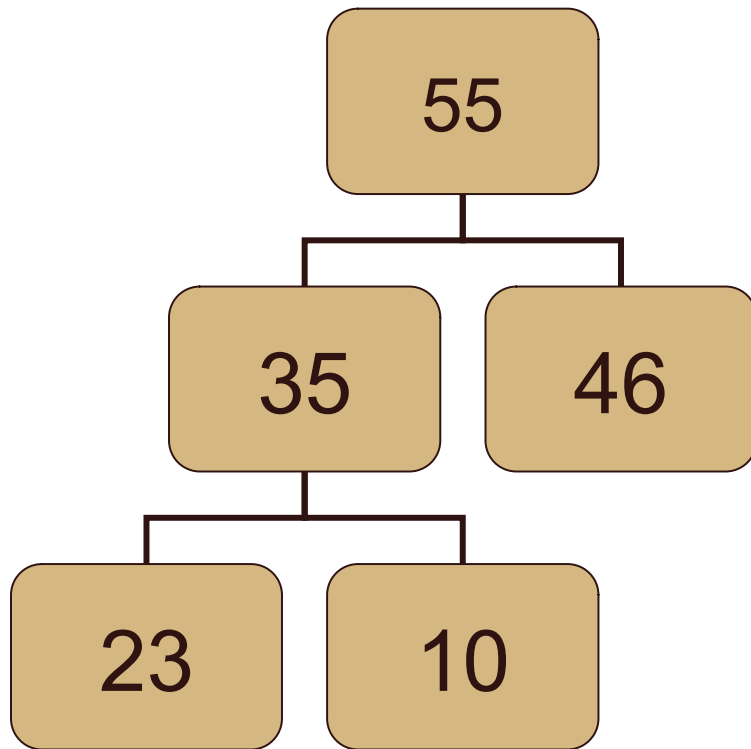
Incluindo 10...



HEAP
inserção

Dados: 23 – 55 – 46 – 35 – 10 – 90

Incluindo 90...



HEAP
inserção

HEAP

inserção

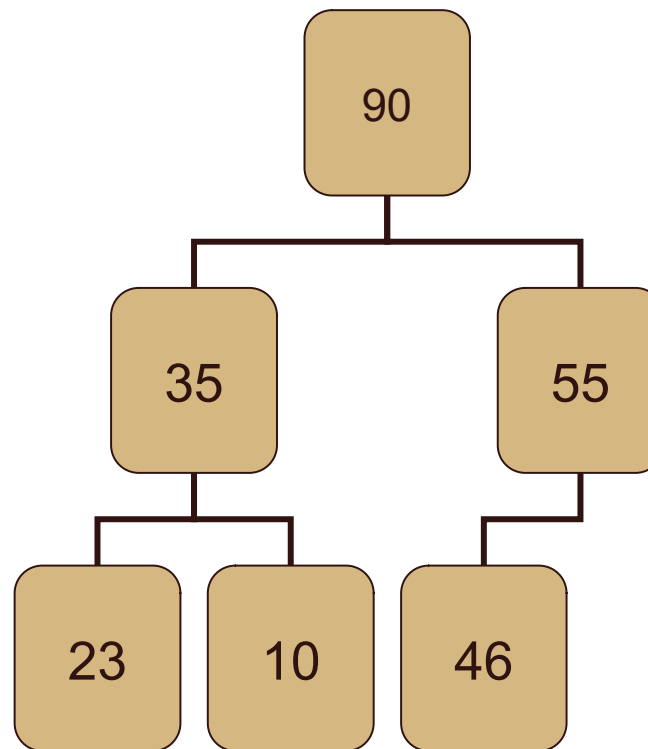
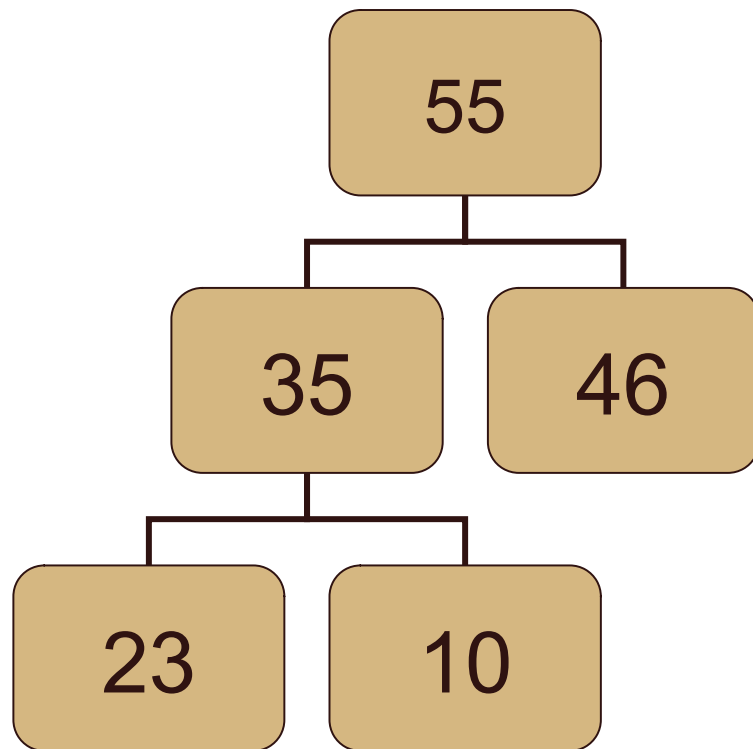
Observar:

- Um heap é montado por nível, para garantir que a árvore obtida seja completa.
- Como consequência de preservar a árvore completa, tem-se uma árvore com **menor altura** possível, e pouco desbalanceada.
- Ademais, por manter o conteúdo dos sucessores de um nó n menor que n , tem-se que os dados encontrados em um caminho da raiz até uma folha são mantidos em ordem não decrescente.



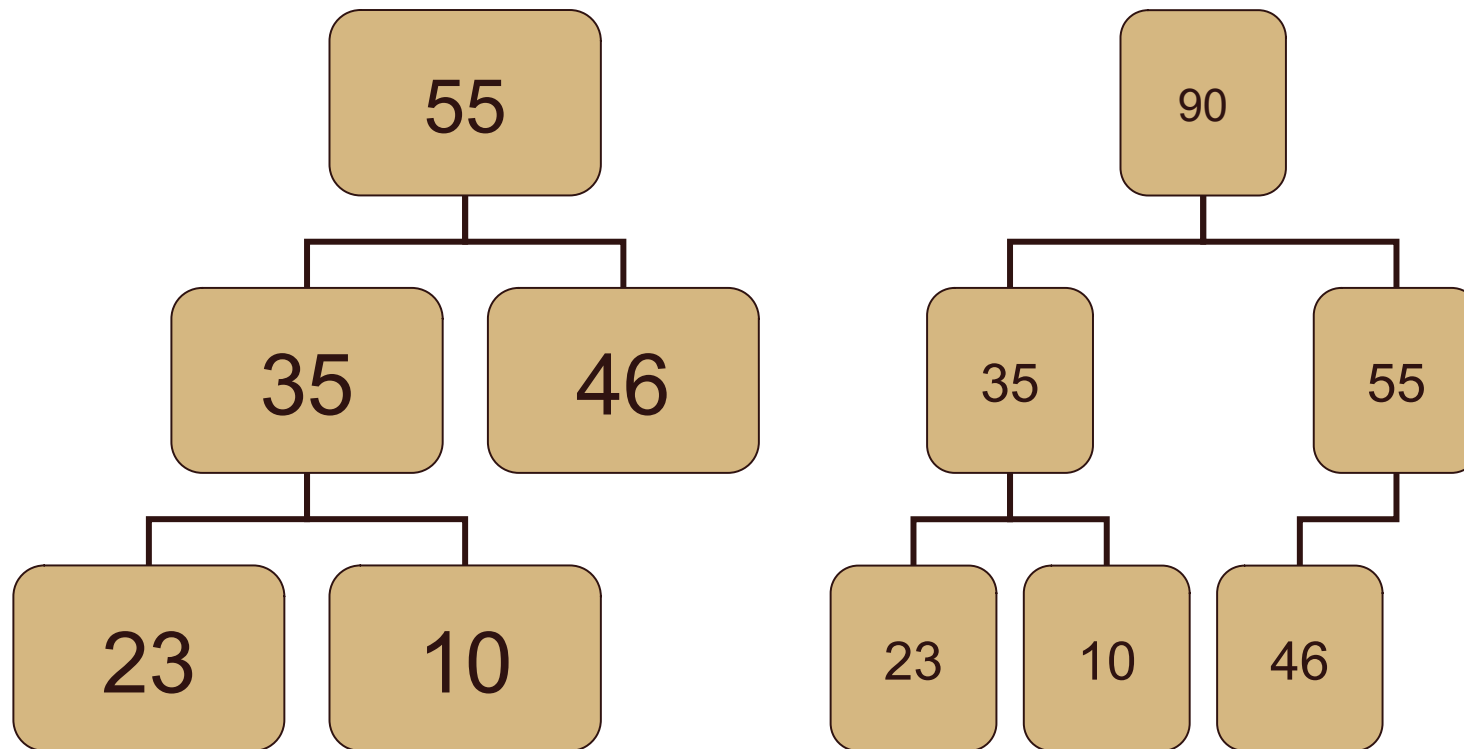
*Qual o custo da operação de
inserção em heap?*

Incluindo 90...



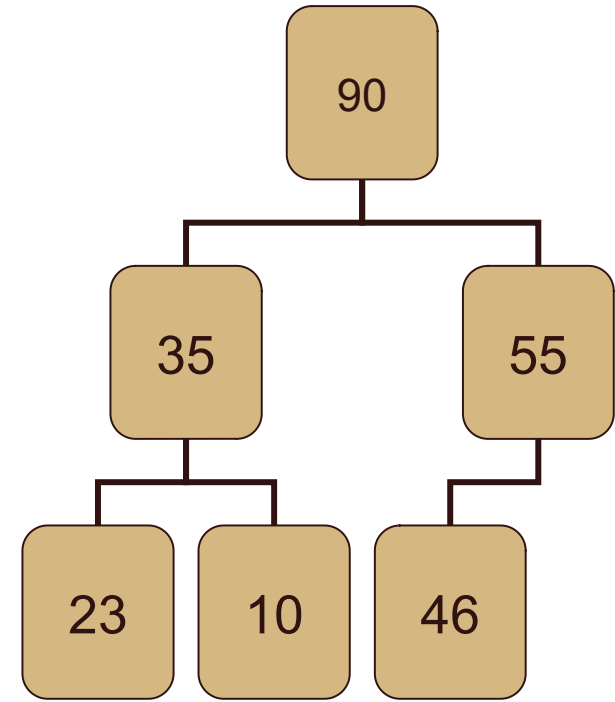
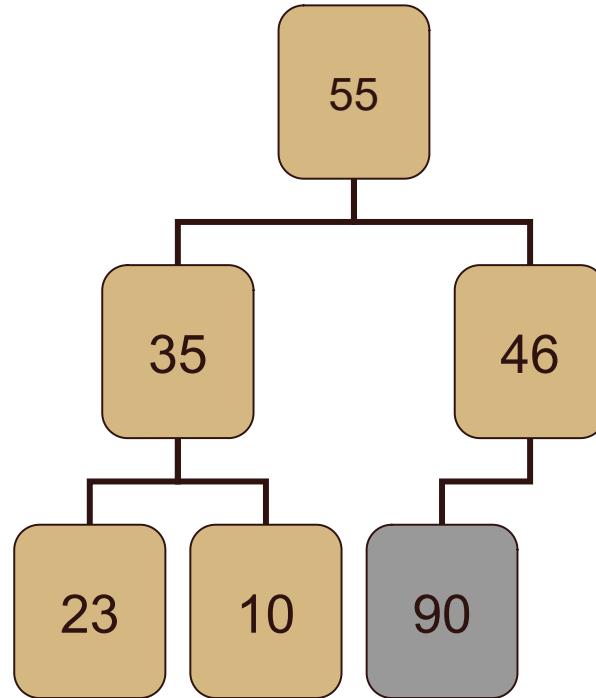
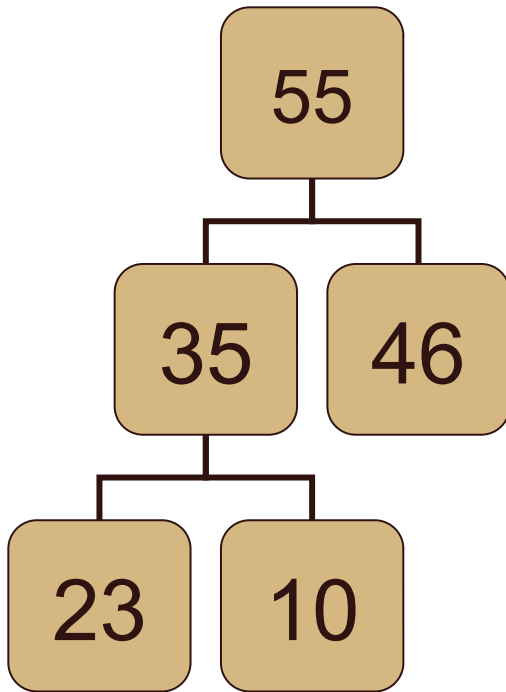
HEAP
inserção

Considerando a necessidade de varredura de um caminho do heap, da raiz até uma folha – ponto de acomodação do novo nó ou algum de seus descendentes, o custo da operação de inserção é $O(\log(n))$.



HEAP
inserção

Há ainda o crescimento/inserção que segue das folhas para a raiz.

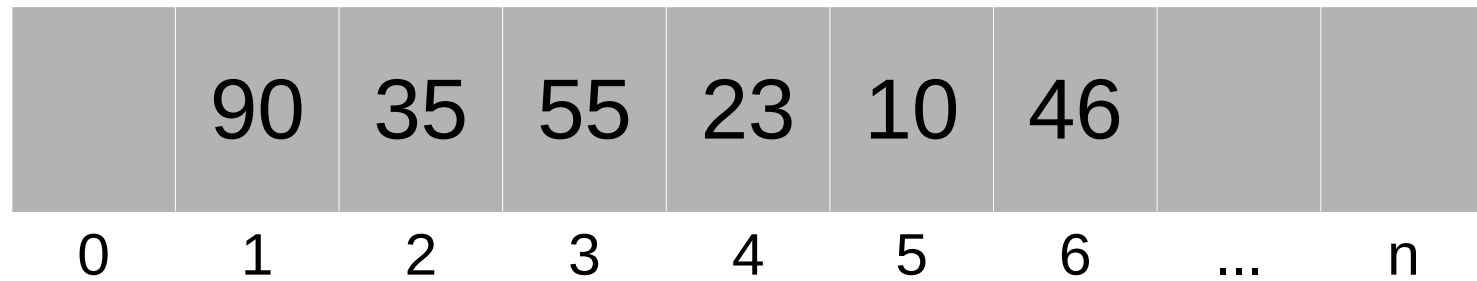
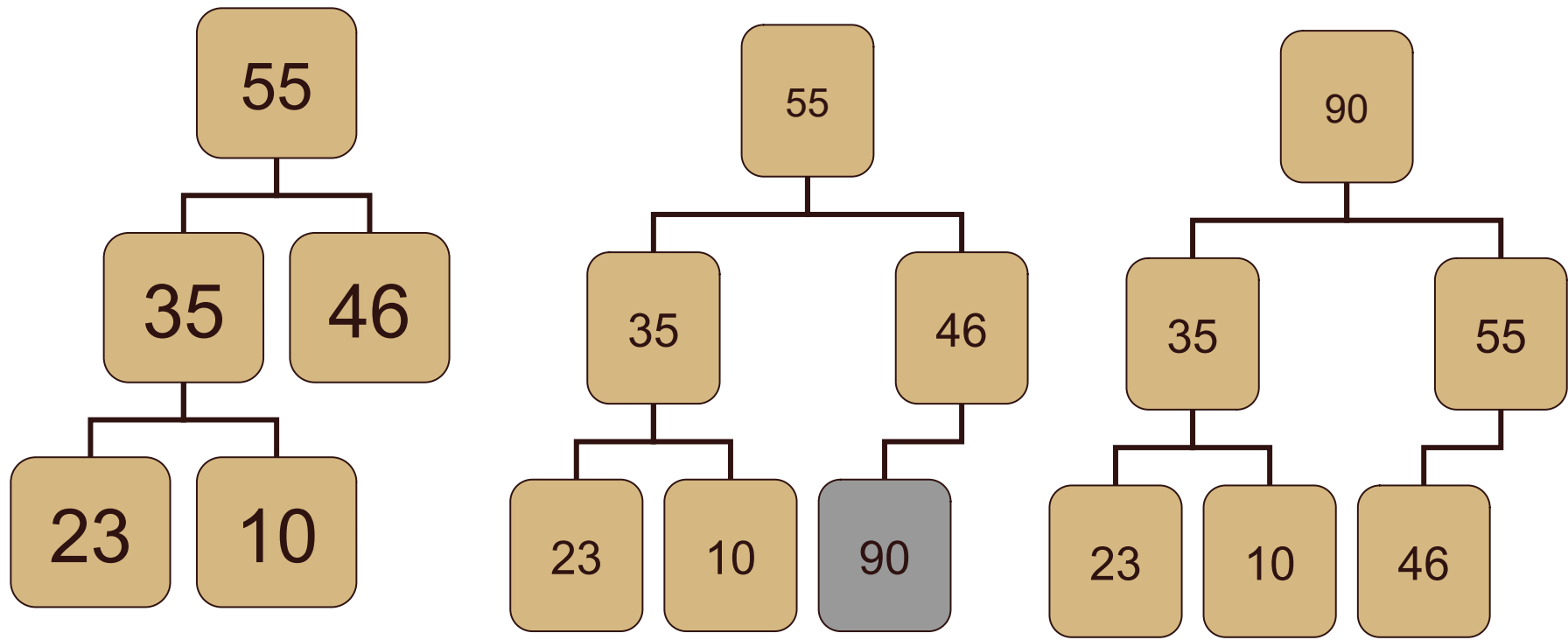


*Há como manter um
heap num array?*

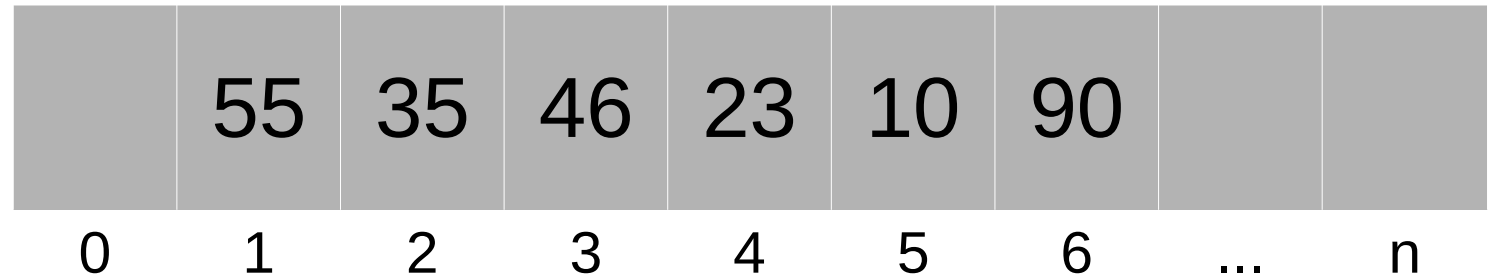
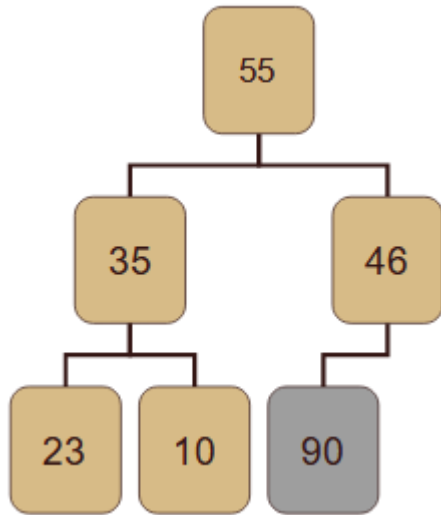


HEAP
inserção

Há ainda o crescimento/inserção que segue das folhas para a raiz.



HEAP
inserção



```

procedimento subir(i)
  j = i/2
  se j >= 1 então
    se T[i].chave > T[j].chave então
      Aux = T[i].chave
      T[i].chave = T[j].chave
      T[j].chave = Aux
      subir(j)
  
```

T...array
 i...índice do valor novo
 j...índice do pai de i

HEAP
 inserção

HEAP

remoção

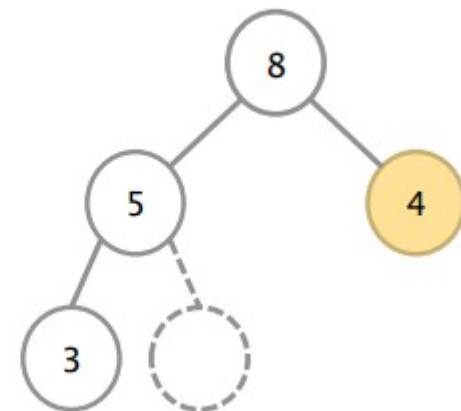
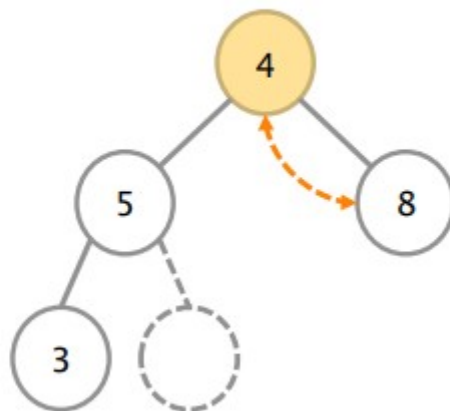
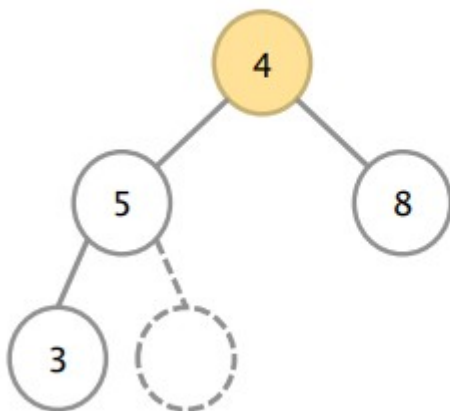
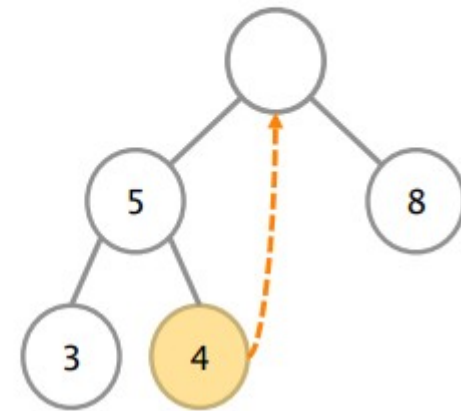
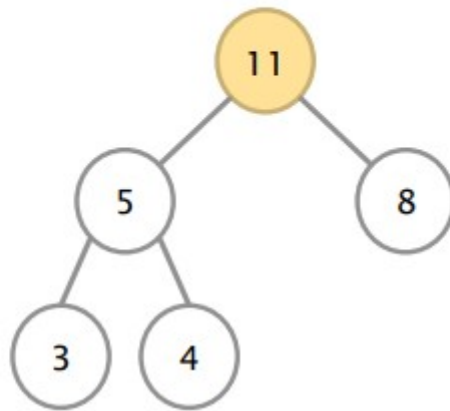
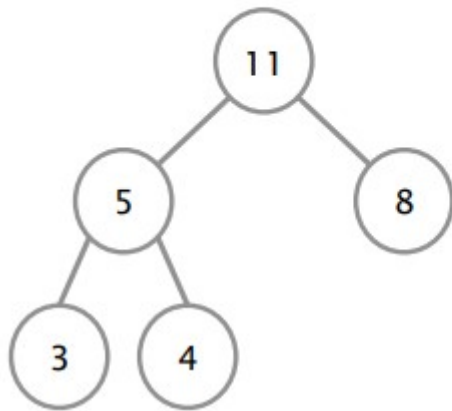
No processo de remoção em heap todas as propriedades do heap também devem ser preservadas:

- Consistir numa árvore binária.
- O valor contido em cada nó deve ser maior que os valores de seus sucessores.
- Ser completa; se n é um nó do heap H com subárvore vazia, então n se localiza no penúltimo ou no último nível de H .

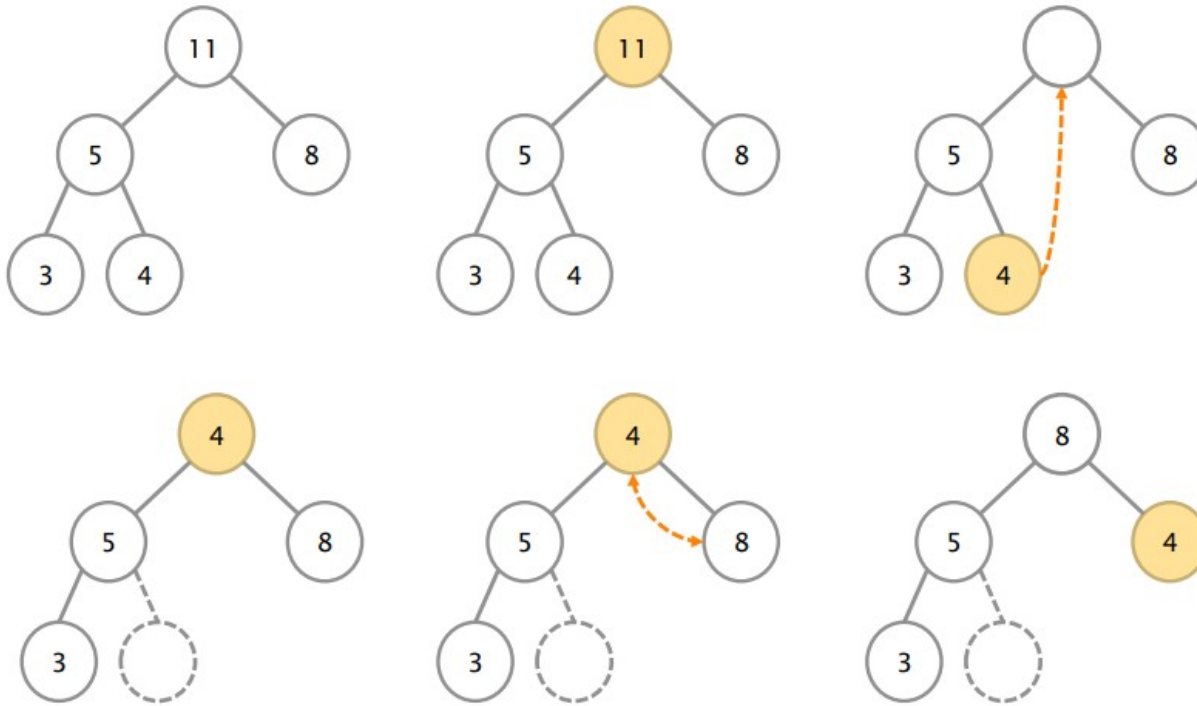
HEAP

remoção

Exemplo de remoção



Exemplo de remoção



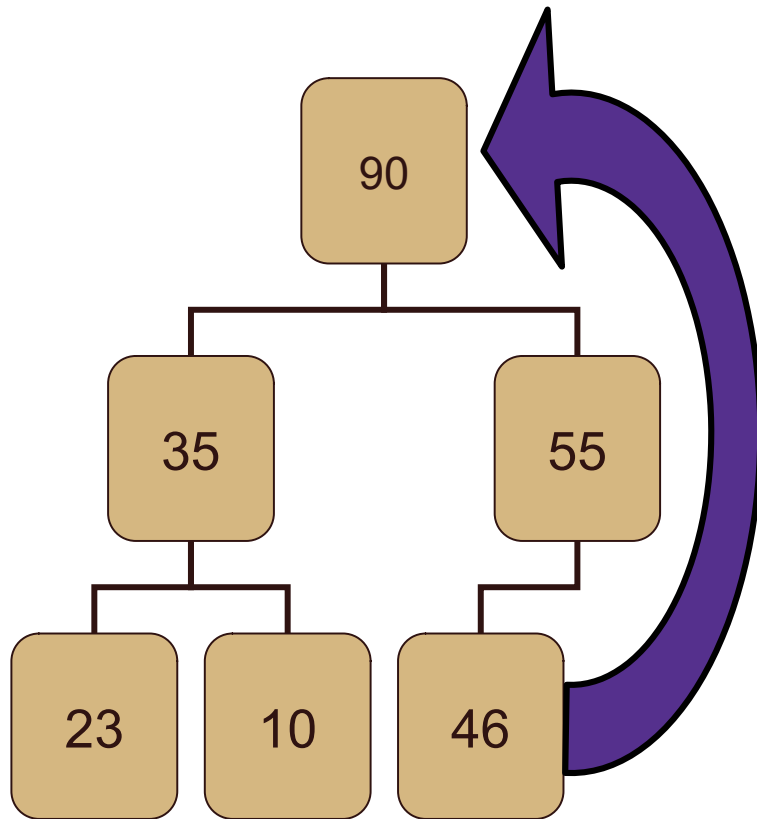
HEAP
remoção

*Por que não substituir o 11 por 8?
Violaria a definição de árvore
completa?*

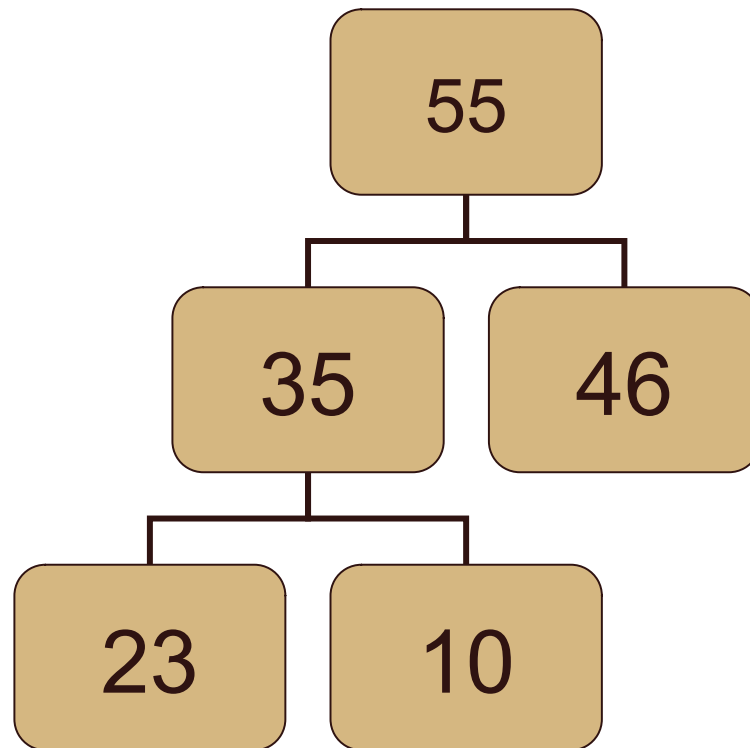


HEAP

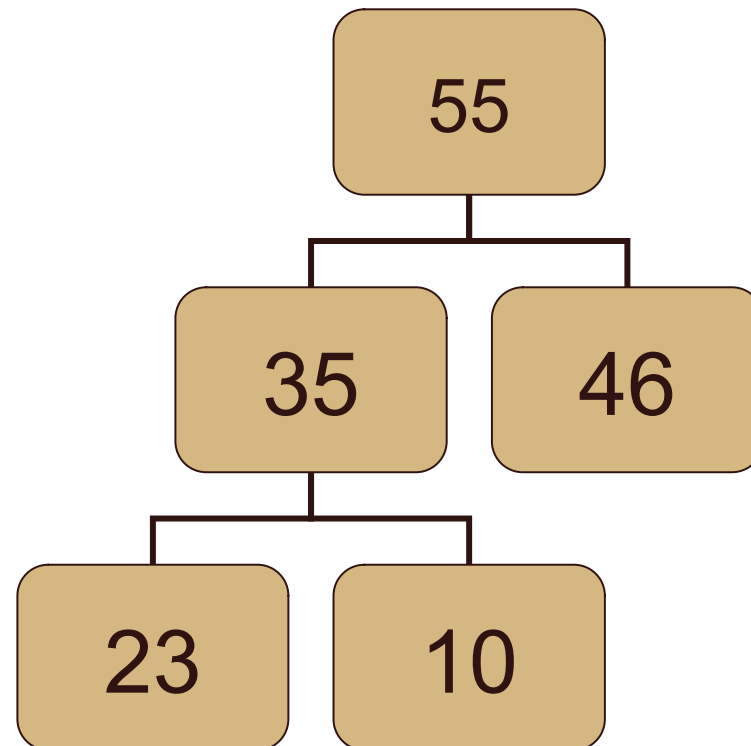
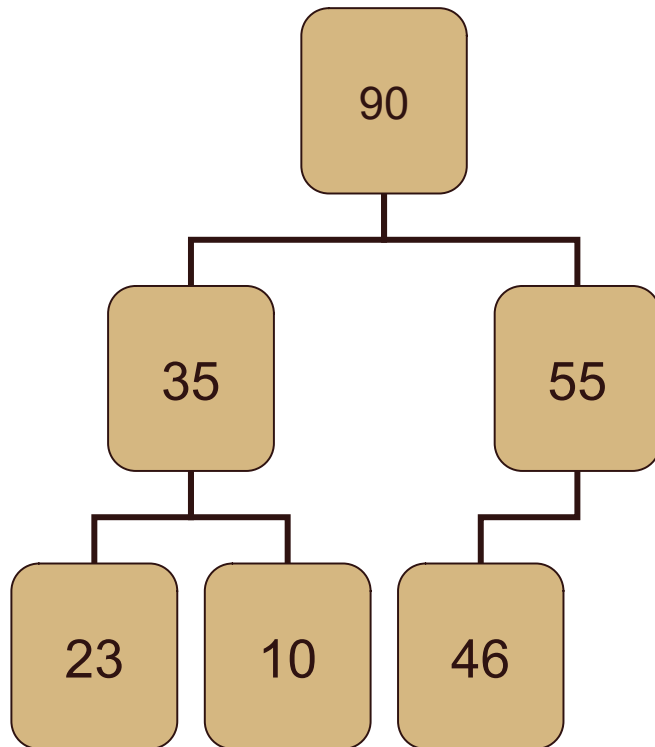
remoção



Removendo, por exemplo, o nó que contém 90:



*Qual o custo da
operação de remoção
em heap?*

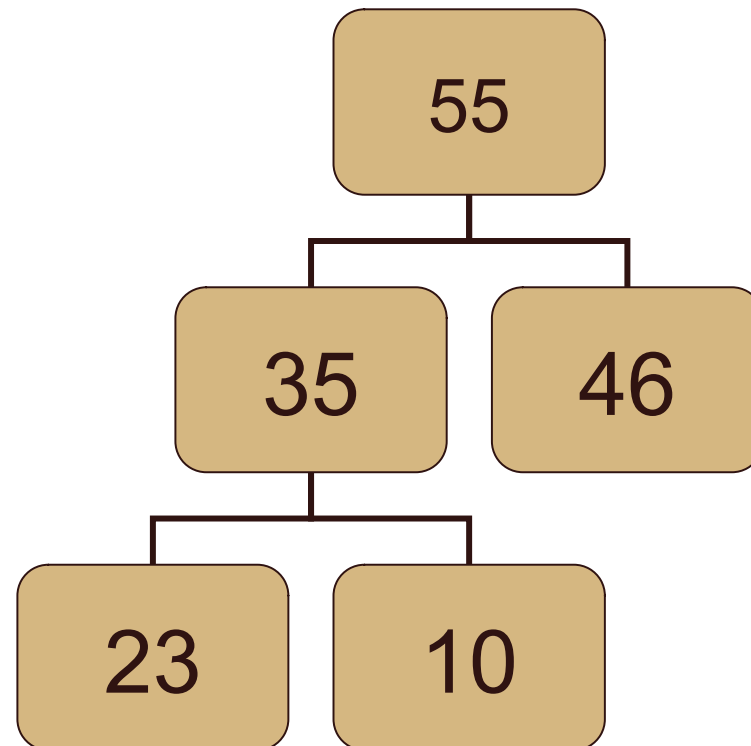
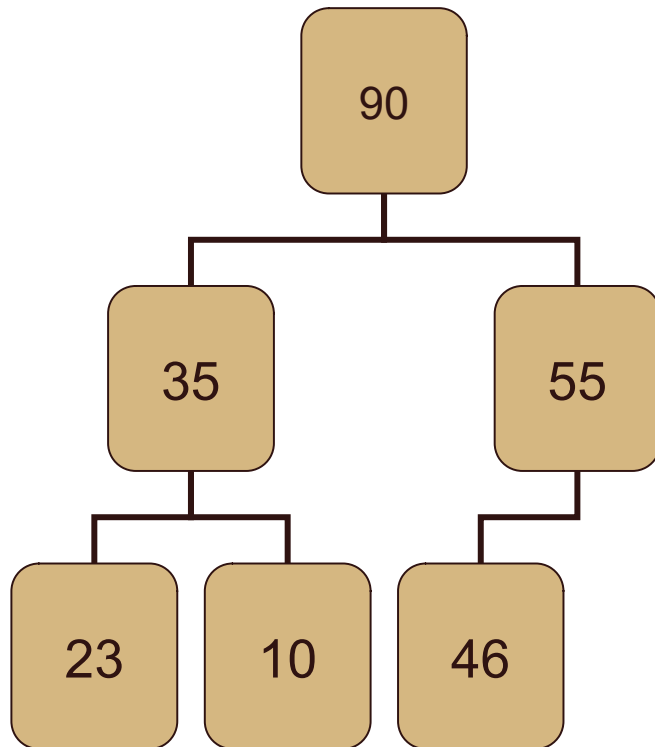


HEAP
remoção

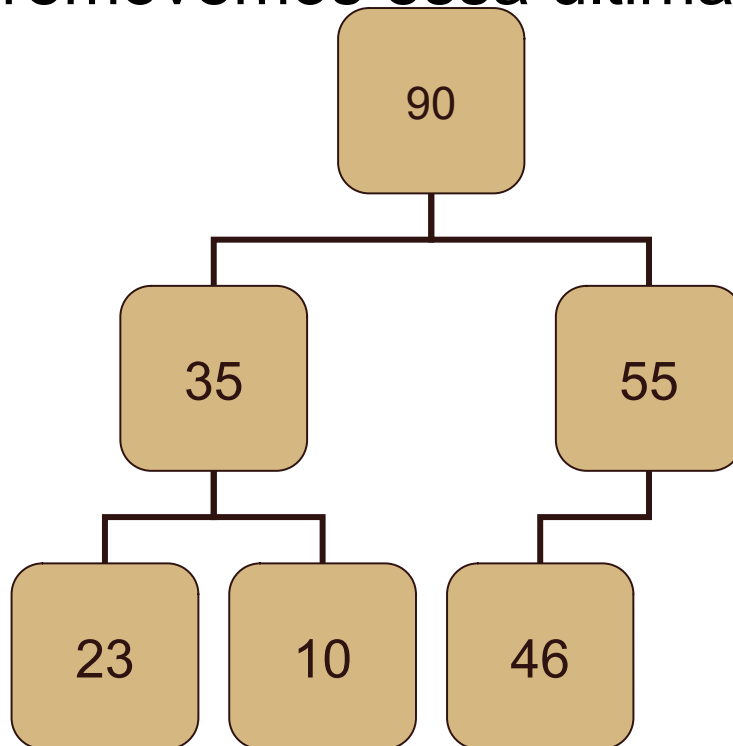
A operação de remoção tem custo logarítmico; já que é necessário ajustar um caminho, da raiz a uma folha, do heap.

HEAP

remoção



O heap pode ser usado para manter prioridades. Neste caso, o elemento removido sempre é o maior, ou seja, sempre a raiz. A remoção em um heap é sempre feita na raiz. Para manter a propriedade de ser completo ou quase-completo da esquerda para a direita, trocamos o valor da raiz com a última folha e removemos essa última folha.

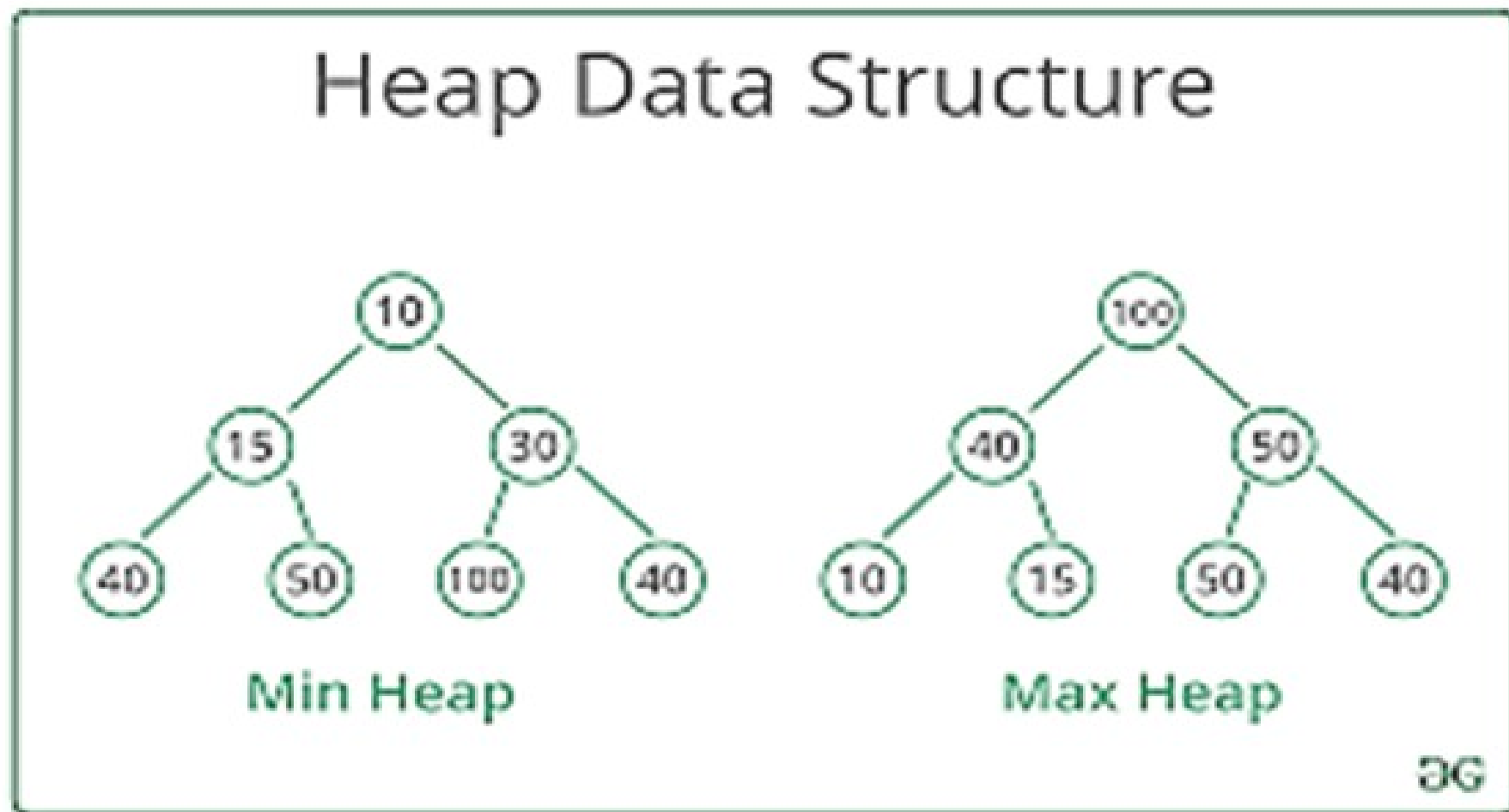


HEAP

remoção

? *Como remover os diversos nós de composição de um heap? E qual o custo?*

HEAP definição



Há dois principais tipos de heaps:

1. **Heap Máximo** – ou Heap Max, apresentado neste. Onde maior item é mantido na raiz.
2. **Heap Mínimo** – ou Heap Min, onde menor item é mantido na raiz.

HEAP

aplicação

1. **Seleção** de elemento de maior prioridade.
2. **Inserção** de novo elemento.
3. **Remoção** de elemento de maior prioridade.
4. **Alteração** de prioridades.



*É possível aplicar heap na resolução das situações problemas consideradas?
Como controla o início e fim de cada fila: vermelha, laranja, amarela, verde e azul?*



Organização da Estrutura

Ordenada Fisicamente
Ordenada por Link
Desordenada

Tipo de Estrutura

Dinâmica
Array
Arquivo

HEAP

aplicação

1. **Seleção** de elemento de maior prioridade.
2. **Inserção** de novo elemento.
3. **Remoção** de elemento de maior prioridade.
4. **Alteração** de prioridades.

? *Que dados manteria em cada nó das filas vermelha, laranja, amarela, verde e azul?*



Organização da Estrutura

Ordenada Fisicamente
Ordenada por Link
Desordenada

Tipo de Estrutura

Dinâmica
Array
Arquivo

HEAP aplicação & custo

Operações	Custos		
	Desordenada	Ordenada	Heap (ORDENADO POR DEFINIÇÃO)
Seleção do elemento de maior prioridade	$O(n)$, no pior caso quando se deve varrer toda a estrutura para identificar a maior prioridade.	$O(k)$, pois o elemento de maior prioridade é mantido no topo.	$O(k)$
Inserção de novo elemento	$O(k)$, já que novo item é inserido no final.	$O(n)$ no pior caso já que é preciso localizar o ponto de inserção conforme a prioridade.	$O(\log n)$ no pior caso
Remoção de elemento de maior prioridade	$O(n)$, no pior caso quando se deve varrer toda a estrutura para identificar a maior prioridade.	$O(k)$, pois o elemento de maior prioridade é mantido no topo.	$O(\log n)$
Alteração de prioridade de dado elemento	$O(n)$ no pior caso quando se deve varrer toda a estrutura para identificar a elemento a ser alterado.	$< O(n)$, pois é pesquisada região menor que a lista toda uma vez que se conhece a prioridade deste. No rearranjo não há deslocamento.	$O(\log n)$ no pior caso

HEAP Outra Vantagem

O heap impede que a solução fique engessada como acontece com a solução que prevê uma fila para cada categoria de urgência hospitalar.

Sendo criada uma nova categoria de classificação de urgência, é necessário mexer no código?

- 1 Necessitam de atendimento imediato.
CASOS DE EMERGÊNCIA
- 2 Necessitam de atendimento praticamente imediato.
CASOS MUITO URGENTES
- 3 Necessitam de atendimento rápido, mas podem aguardar.
CASOS DE URGÊNCIA
- 4 Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS POUCO URGENTES
- 5 Podem aguardar atendimento ou serem encaminhados para outros serviços de saúde.
CASOS NÃO URGENTES

Atividade Periódica II4

Propor situação problema cuja solução ideal, quanto a uso de espaço e tempo de processamento, implica no **uso de Hashing ou Árvores AVL ou Heap**. A descrição deve ter um nível de detalhamento que possibilite identificar com clareza: o objetivo almejado pela solução computacional, as funcionalidades a serem disponibilizadas ao usuário final, os dados a manipular e a estrutura a aplicar.

Para tanto, compor grupo composto por 3 pessoas, **enviar e-mail, até 04/11/2022**, com assunto: ED – T04 – Atividades Periódica II4, no corpo do e-mail (não usar anexo) apresentar nome completo e matrícula dos componentes do grupo seguido da descrição citada no parágrafo anterior.

Complementar Estudos...

Estruturas de Dados e Seus Algoritmos

Szwarcfiter J. L. e Markenzon L.

Capítulo

6 Listas de Prioridade

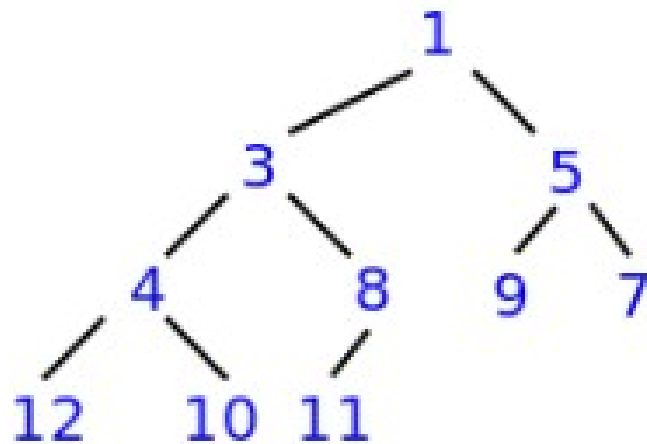
Complementar Estudos...

Estruturas de Dados Descomplicada em Linguagem C
Backes, A.



Capítulo **7** Filas de Prioridade

Complementar Estudos...



1	2	3	4	5	6	7	8	9	10
1	3	5	4	8	9	7	12	10	11



Heaps

www.ic.uff.br/~fabio/Aula-heaps-1.pdf

ime.usp.br/~pf/analise_de_algoritmos/aulas/heap.html

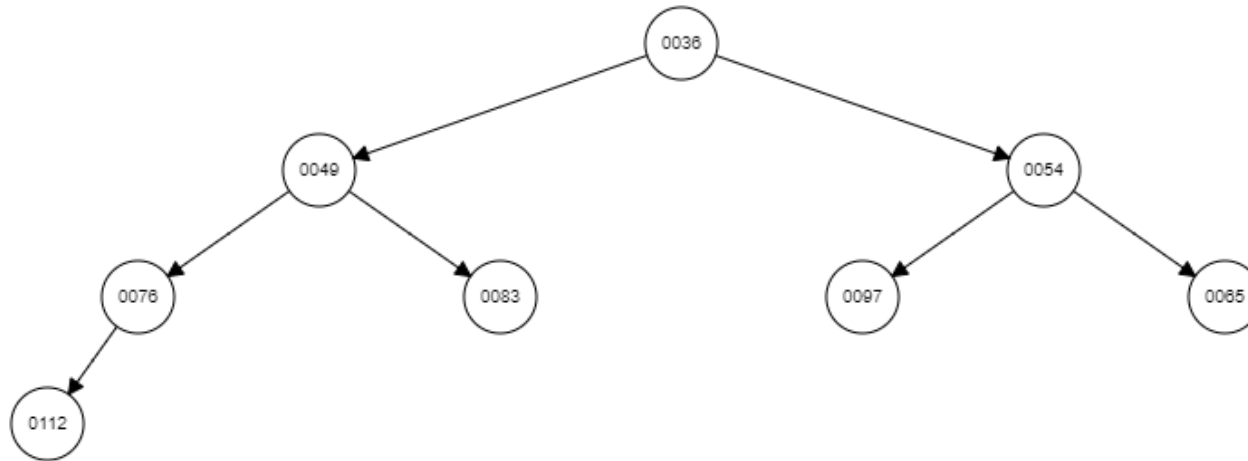
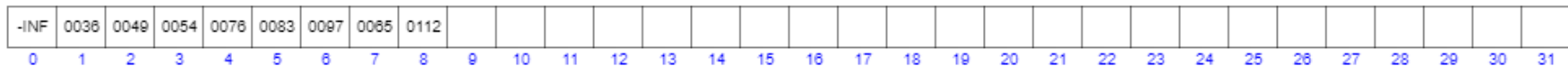
Complementar Estudos...

Min Heap

Remove Smallest

Clear Heap

BuildHeap



Animation Completed

Step Back

Step Forward

Skip Forward

Change Canvas Size

Change Canvas Size Move Controls

Animation Speed

Próximo Passo



Conjuntos
Disjuntos