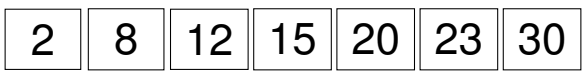
AULA 15
ESTRUTURA DE DADOS

Árvores – Conceitos Básicos

Norton T. Roman & Luciano A. Digiampietri

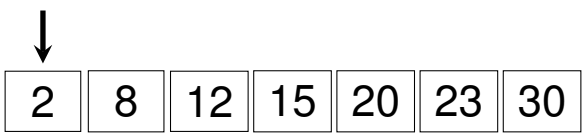
Buscando um Elemento

Imagine que queremos buscar um elemento (15) no arranjo abaixo. Como fazer?



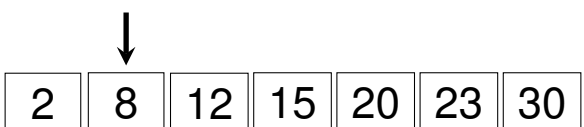
Buscando um Elemento

Imagine que queremos buscar um elemento (15) no arranjo abaixo. Como fazer?



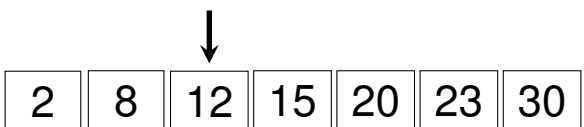
Buscando um Elemento

Imagine que queremos buscar um elemento (15) no arranjo abaixo. Como fazer?



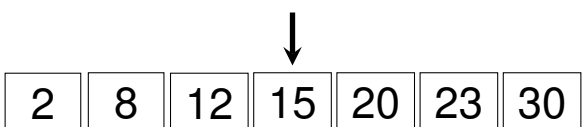
Buscando um Elemento

Imagine que queremos buscar um elemento (15) no arranjo abaixo. Como fazer?



Buscando um Elemento

Imagine que queremos buscar um elemento (15) no arranjo abaixo. Como fazer?

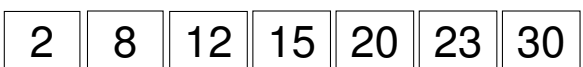


Buscando um Elemento

Imagine que queremos buscar um elemento (15) no arranjo abaixo. Como fazer?

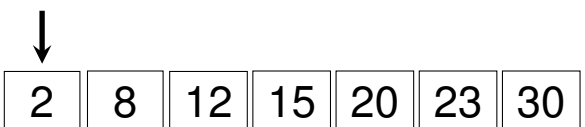
Buscando um Elemento

E o 16?



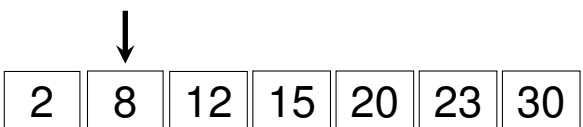
Buscando um Elemento

E o 16?



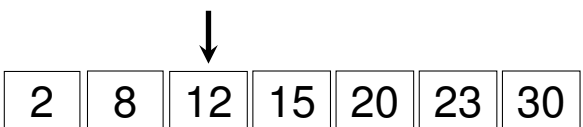
Buscando um Elemento

E o 16?



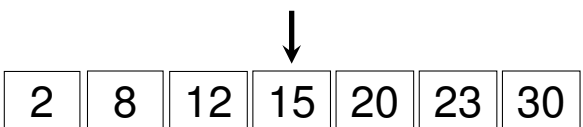
Buscando um Elemento

E o 16?



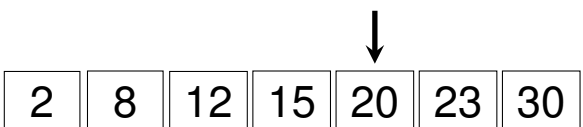
Buscando um Elemento

E o 16?



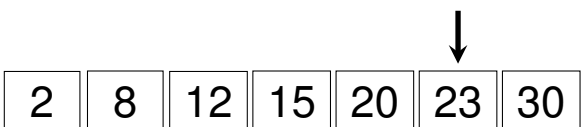
Buscando um Elemento

E o 16?



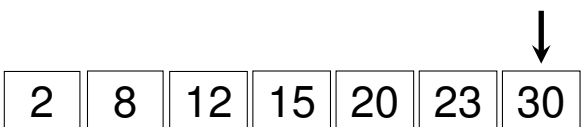
Buscando um Elemento

E o 16?



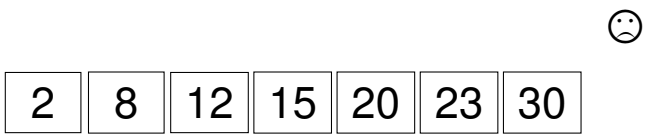
Buscando um Elemento

E o 16?



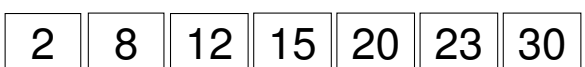
Buscando um Elemento

E o 16?



Buscando um Elemento

Será que não poderíamos aproveitar o fato de que o arranjo está ordenado? Ex: buscando o 16

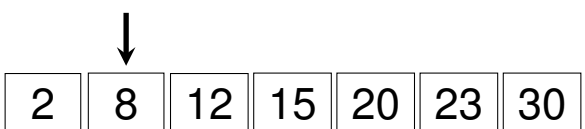


Buscando um Elemento

Será que não poderíamos aproveitar o fato de que o arranjo está ordenado? Ex: buscando o 16

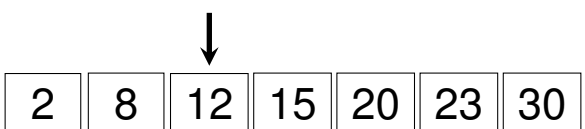
Buscando um Elemento

Será que não poderíamos aproveitar o fato de que o arranjo está ordenado? Ex: buscando o 16



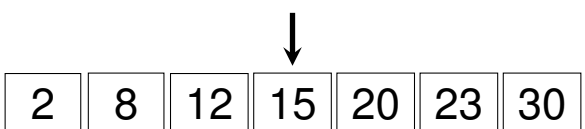
Buscando um Elemento

Será que não poderíamos aproveitar o fato de que o arranjo está ordenado? Ex: buscando o 16



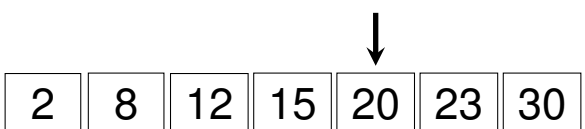
Buscando um Elemento

Será que não poderíamos aproveitar o fato de que o arranjo está ordenado? Ex: buscando o 16



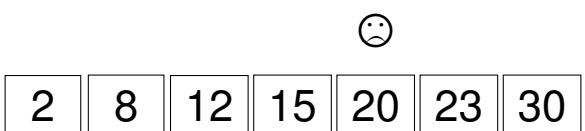
Buscando um Elemento

Será que não poderíamos aproveitar o fato de que o arranjo está ordenado? Ex: buscando o 16



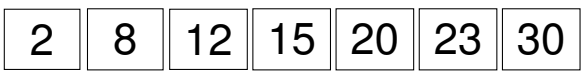
Buscando um Elemento

Será que não poderíamos aproveitar o fato de que o arranjo está ordenado? Ex: buscando o 16



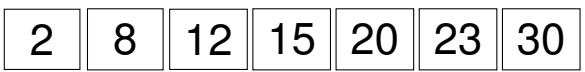
Buscando um Elemento

Mas, se o número buscado for maior que todos (ex: 31), ainda corremos o arranjo inteiro. Teria como melhorar?



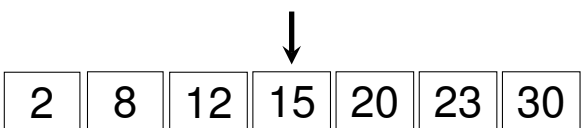
Buscando um Elemento

Mas, se o número buscado for maior que todos (ex: 31), ainda corremos o arranjo inteiro. Teria como melhorar? Busca binária...



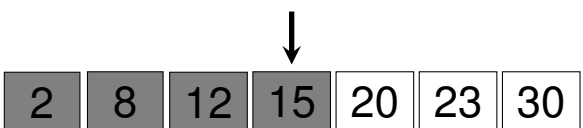
Buscando um Elemento

Mas, se o número buscado for maior que todos (ex: 31), ainda corremos o arranjo inteiro. Teria como melhorar? Busca binária...



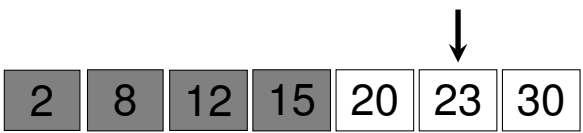
Buscando um Elemento

Mas, se o número buscado for maior que todos (ex: 31), ainda corremos o arranjo inteiro. Teria como melhorar? Busca binária...



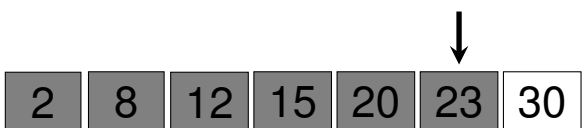
Buscando um Elemento

Mas, se o número buscado for maior que todos (ex: 31), ainda corremos o arranjo inteiro. Teria como melhorar? Busca binária...



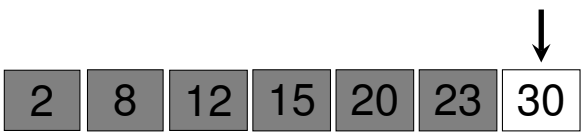
Buscando um Elemento

Mas, se o número buscado for maior que todos (ex: 31), ainda corremos o arranjo inteiro. Teria como melhorar? Busca binária...



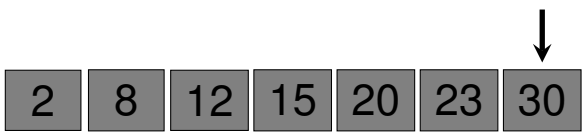
Buscando um Elemento

Mas, se o número buscado for maior que todos (ex: 31), ainda corremos o arranjo inteiro. Teria como melhorar? Busca binária...



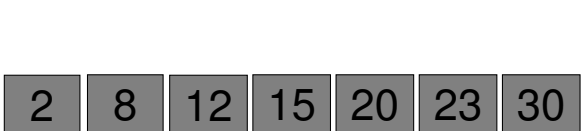
Buscando um Elemento

Mas, se o número buscado for maior que todos (ex: 31), ainda corremos o arranjo inteiro. Teria como melhorar? Busca binária...



Buscando um Elemento

Mas, se o número buscado for maior que todos (ex: 31), ainda corremos o arranjo inteiro. Teria como melhorar? Busca binária...



Buscando um Elemento

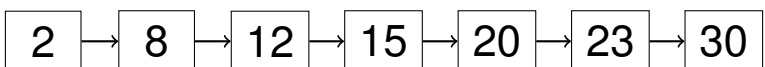
Busca binária é mais eficiente...

Buscando um Elemento

Busca binária é mais eficiente... mas depende de arranjos estáticos.

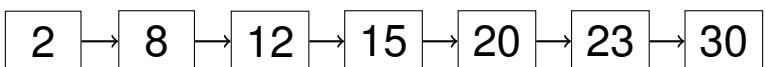
Buscando um Elemento

Busca binária é mais eficiente... mas depende de arranjos estáticos.
E se tivéssemos uma lista ligada?



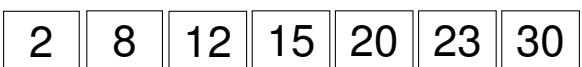
Buscando um Elemento

Busca binária é mais eficiente... mas depende de arranjos estáticos.
E se tivéssemos uma lista ligada? Ops!



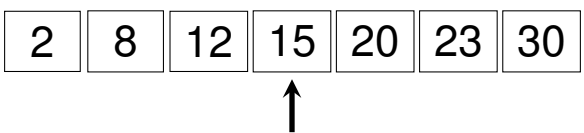
Buscando um Elemento

Será que não podemos ter uma estrutura dinâmica que nos ajude nessa tarefa?



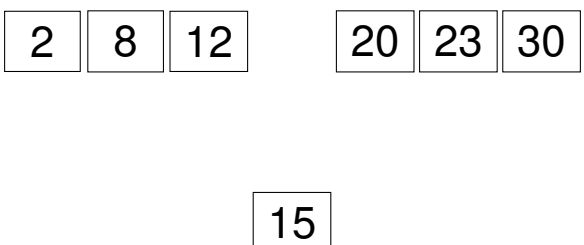
Buscando um Elemento

Será que não podemos ter uma estrutura dinâmica que nos ajude nessa tarefa?



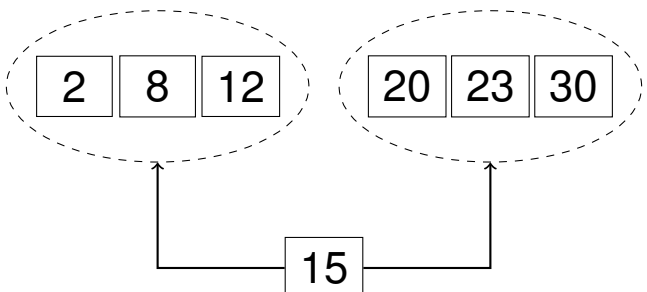
Buscando um Elemento

Será que não podemos ter uma estrutura dinâmica que nos ajude nessa tarefa?



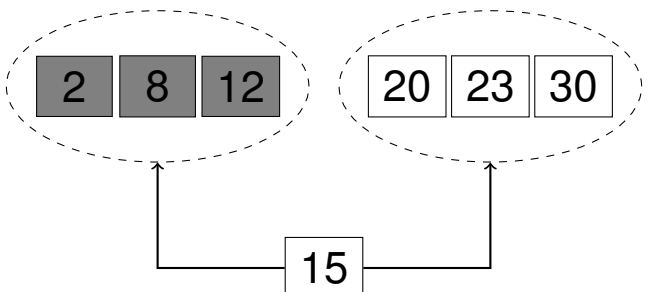
Buscando um Elemento

Será que não podemos ter uma estrutura dinâmica que nos ajude nessa tarefa?



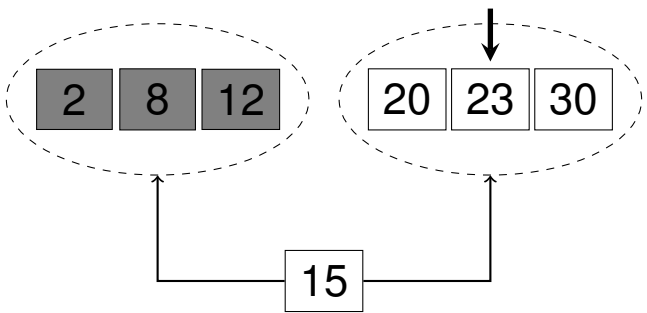
Buscando um Elemento

Será que não podemos ter uma estrutura dinâmica que nos ajude nessa tarefa?



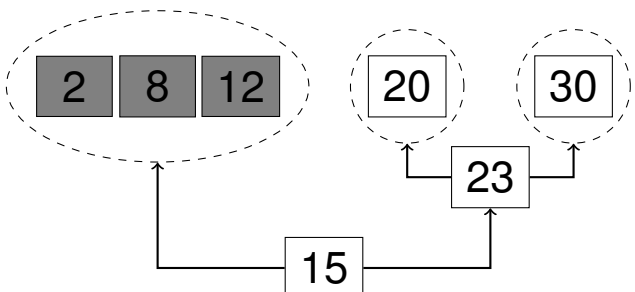
Buscando um Elemento

Será que não podemos ter uma estrutura dinâmica que nos ajude nessa tarefa?



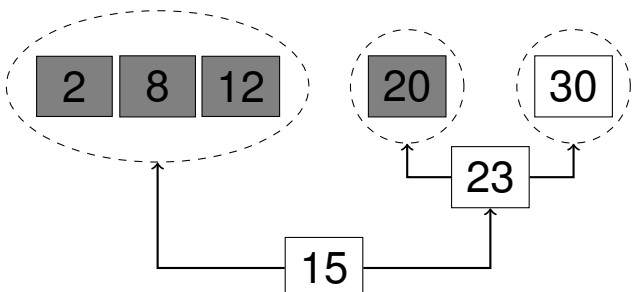
Buscando um Elemento

Será que não podemos ter uma estrutura dinâmica que nos ajude nessa tarefa?



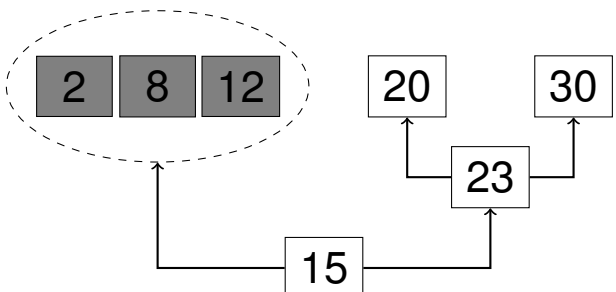
Buscando um Elemento

Será que não podemos ter uma estrutura dinâmica que nos ajude nessa tarefa?



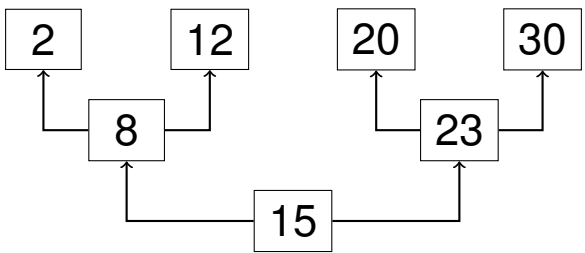
Buscando um Elemento

Será que não podemos ter uma estrutura dinâmica que nos ajude nessa tarefa?



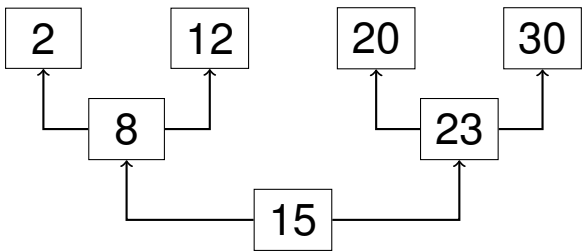
Buscando um Elemento

Será que não podemos ter uma estrutura dinâmica que nos ajude nessa tarefa?



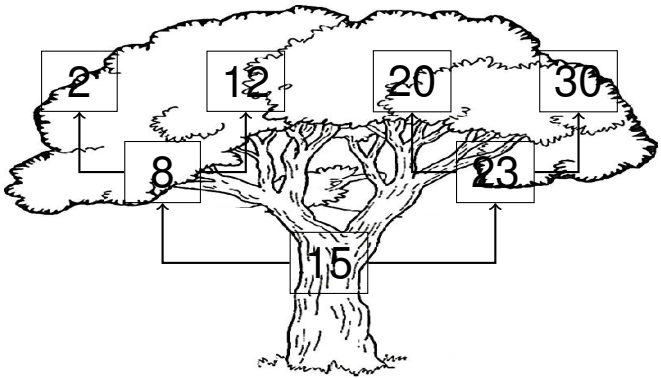
Buscando um Elemento

Eis nossa estrutura. Que nome daremos a ela?



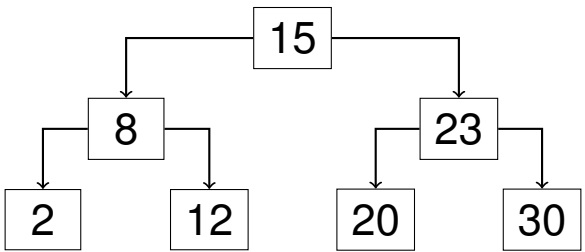
Buscando um Elemento

Eis nossa estrutura. Que nome daremos a ela?



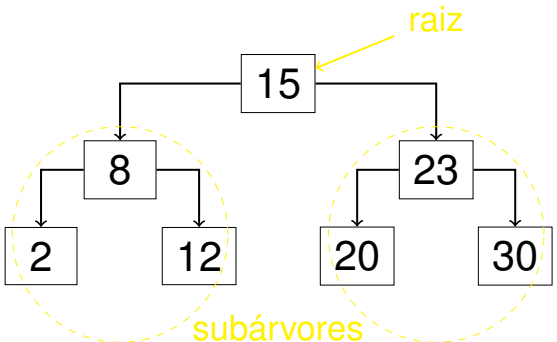
Árvores

Uma observação: em computação costumamos representar a árvore de forma invertida...



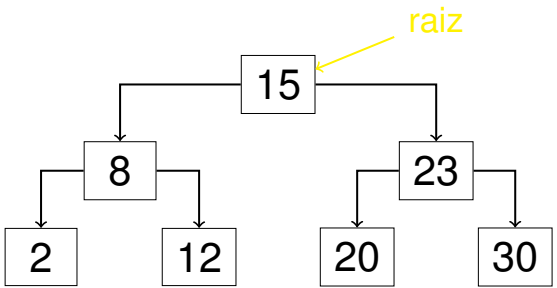
Árvores

Uma árvore é um conjunto de nós consistindo de um nó chamado raiz, abaixo do qual estão as subárvores que compõem essa árvore.



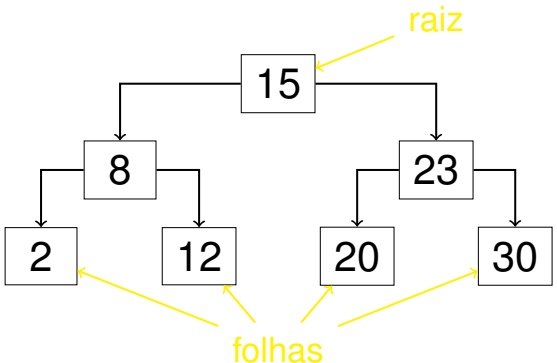
Árvores

O número de subárvores de cada nó é chamado de grau desse nó.
No exemplo ao lado, todo nó tem grau 2, exceto os da base, que têm grau 0.



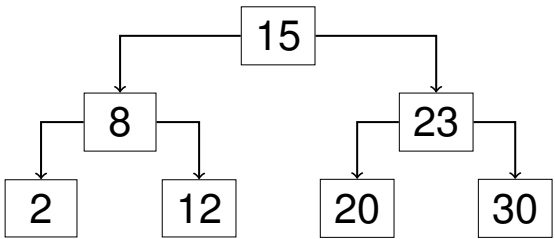
Árvores

Nós de grau zero são chamados de nós externos ou folhas.
Os demais são chamados de nós internos.



Árvores

Nós abaixo de um determinado nó são seus descendentes

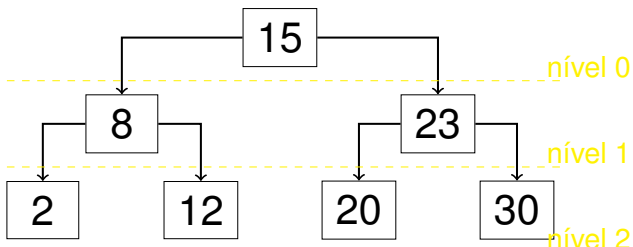


Descendentes do 8: 2 e 12

Descendentes do 15: todos os demais

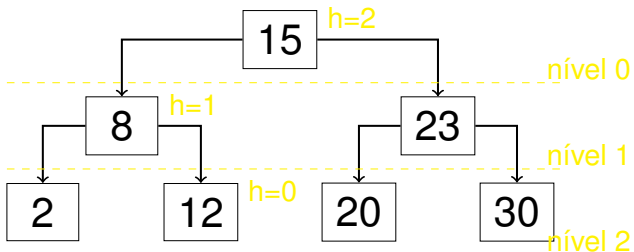
Árvores

O nível do nó raiz é 0



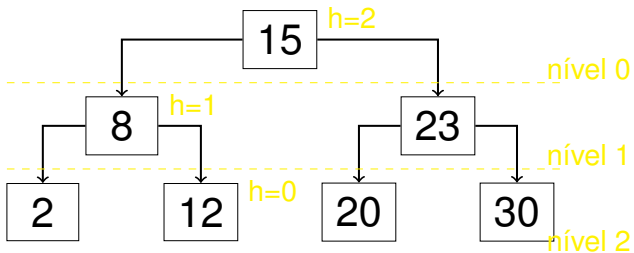
Árvores

O nível do nó raiz é 0
A altura (h) de um nó é o comprimento do caminho mais longo entre ele e uma folha.



Árvores

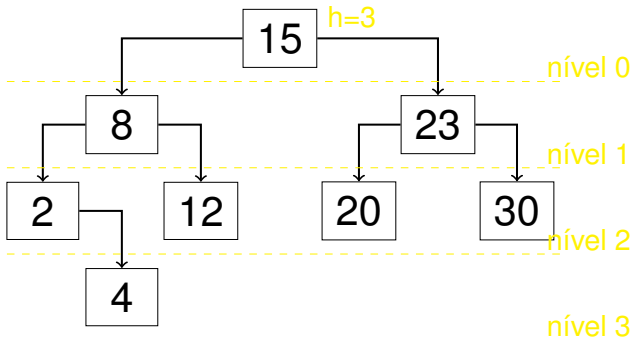
O nível do nó raiz é 0
A altura (h) de um nó é o comprimento do caminho mais longo entre ele e uma folha.



Vale notar que a árvore é percorrida da raiz às folhas.

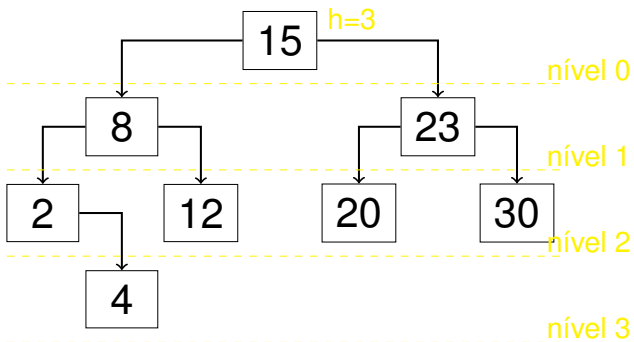
Árvores

Nem sempre a árvore estará perfeitamente balanceada. Ainda assim, as definições de altura, nível etc valem.



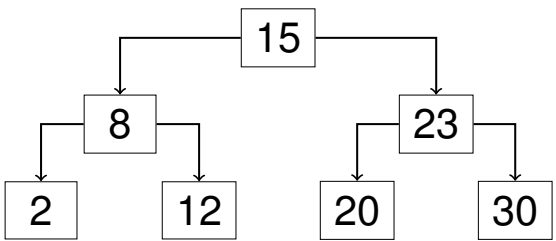
Árvores

A altura de uma árvore é a altura do nó raiz.
Da mesma forma, o endereço de uma árvore na memória será o endereço de seu nó raiz.



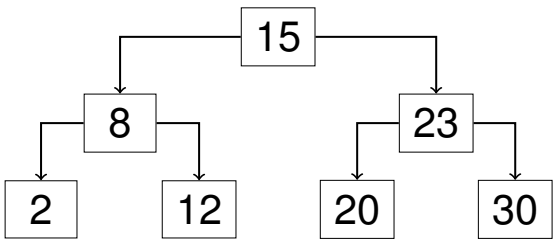
Árvores

A profundidade de um nó é a distância percorrida da raiz a esse nó.
Profundidade de 15: 0
Profundidade de 8: 1
Profundidade de 12: 2



Árvores Binárias

Uma árvore binária é uma árvore em que, abaixo de cada nó, existem no máximo duas subárvores.



Árvores Binárias

Como representamos computacionalmente uma árvore binária?

Árvores Binárias

Como representamos computacionalmente uma árvore binária? Unindo nós.

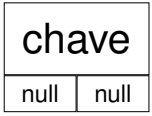
Árvores Binárias

Como representamos computacionalmente uma árvore binária? Unindo nós.
E como representamos os nós?

Árvores Binárias

Como representamos computacionalmente uma árvore binária? Unindo nós.
E como representamos os nós?

Com 2 ponteiros: um para a subárvore da esquerda e um para a subárvore da direita.
Além de um campo para a chave e dados



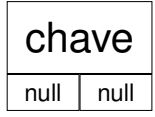
Árvores Binárias

```
#include <stdio.h>
#include <stdlib.h>
#define true 1
#define false 0

typedef int bool;
typedef int TIPOCHAVE;

typedef struct aux {
  TIPOCHAVE chave;
  /* Dados armazenados vão aqui */
  struct aux *esq, *dir;
} NO;
```

```
typedef NO* PONT;
```



AULA 15
ESTRUTURA DE DADOS

Árvores – Conceitos Básicos

Norton T. Roman & Luciano A. Digiampietri