Estruturas de dados

Sistemas para Internet
UFSM

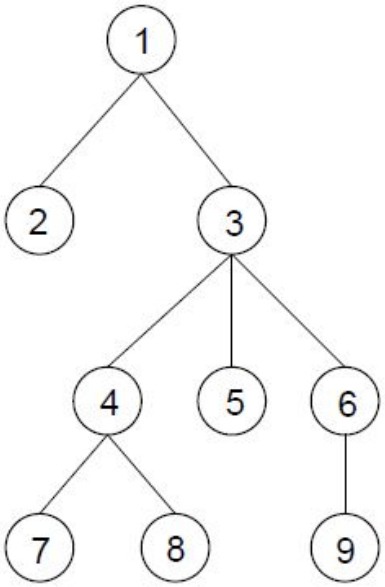
Árvores binárias
Métodos de percurso

Leandro O. Freitas
leandro@politecnico.ufsm.br

Definição de árvores

Sistemas para Internet
UFSM

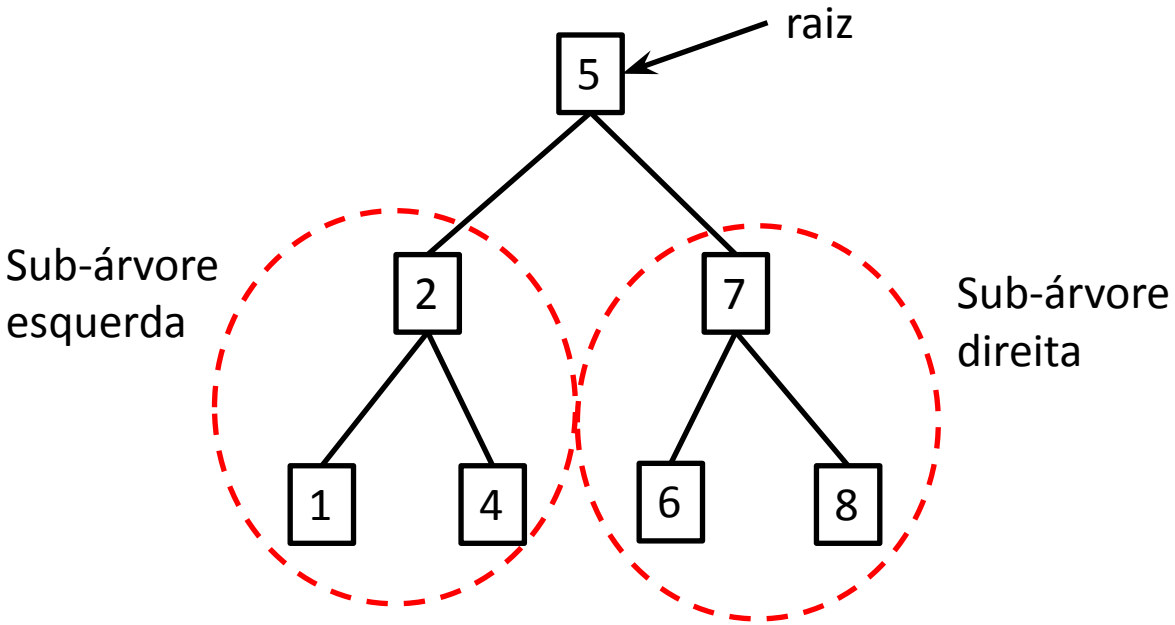
- **Raiz:** nó de origem da árvore.
- **Folhas:** nós que não têm filhos.
- **Grau de um nó:** é o número de sub-árvores (filhos) de um nó.
- **Nível de um nó:** número de nós no caminho da raiz até um nó.
- **Altura da árvore:** é o nível mais alto da árvore.
- **Sub-árvore:** todo nó contido na árvore é a raiz de uma sub-árvore.
- **Caminho:** é o caminho entre dois nós de um árvore.
- **Árvore cheia ou completa:** uma árvore que possui o número máximo de nós, isto é, todos os nós têm número máximo de filhos exceto as folhas, e todas as folhas estão na mesma altura.



Percorrer árvores binárias

Sistemas para Internet
UFSM

- Refere-se ao ato de visitar (acessar, consultar, alterar, etc) cada um dos nós da árvore.
  - Deve-se garantir que cada elemento seja visitado uma única vez por percurso;
  - Deve-se considerar:



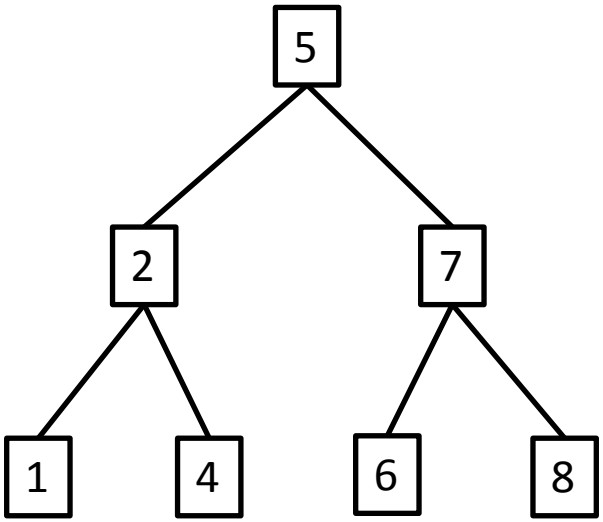
Percorrer árvores binárias

Sistemas para Internet
UFSM

- Prefixado:
  1. Analisa-se a raiz;
  2. Percorre a sub-árvore da esquerda;
  3. Percorre a sub-árvore da direita.

- Simétrico:
  1. Percorre a sub-árvore esquerda;
  2. Analisa-se a raiz;
  3. Percorre a sub-árvore direita;

- Pós-fixado:
  1. Percorre a sub-árvore esquerda;
  2. Percorre a sub-árvore direita;
  3. Analisa-se a raiz.



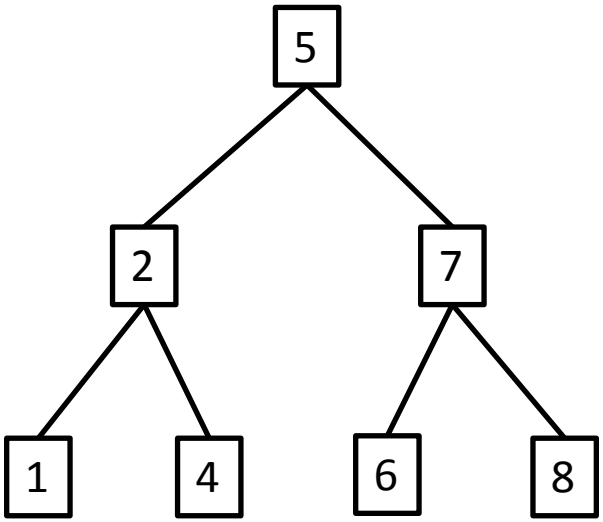
Percorrer árvores binárias

Sistemas para Internet
UFSM

- Prefixado:
  1. Analisa-se a raiz;
  2. Percorre a sub-árvore da esquerda;
  3. Percorre a sub-árvore da direita.
  4. **5, 2, 1, 4, 7, 6, 8.**

- Simétrico:
  1. Percorre a sub-árvore esquerda;
  2. Analisa-se a raiz;
  3. Percorre a sub-árvore direita;
  4. **1, 2, 4, 5, 6, 7, 8.**

- Pós-fixado:
  1. Percorre a sub-árvore esquerda;
  2. Percorre a sub-árvore direita;
  3. Analisa-se a raiz.
  4. **1, 4, 2, 6, 8, 7, 5.**



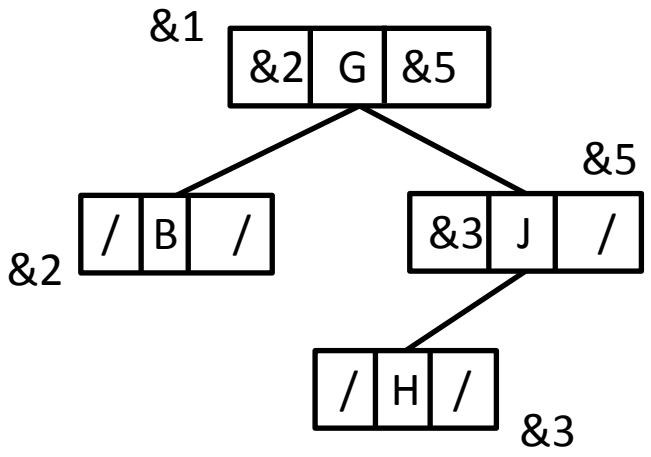
Percorrer árvores binárias

Sistemas para Internet
UFSM

- Exemplo de impressão com recursividade: método pré-fixado.

```
1 void imprime (arv *a)
2 {
3     if (a != NULL)
4     {
5         printf("%c \n", c->info); // mostra o valor da raiz da (sub)árvore
6         imprime(a->esq); // mostra o valor da esquerda
7         imprime(a->dir); // mostra o valor da direita
8     }
9 }
```

```
int main(){
    arv *a;
    // função para criar a árvore
    imprime(a);
    return 0;
}
```



Exercício

- A partir do código do slide anterior e do exercício desenvolvido na aula 9.1, adaptar a função *imprime()* para minimizar o número de chamadas recursivas;

- Implemente também a impressão dos elementos através dos métodos de caminhamento simétrico e pós-fixado.