

SIN 211 - Algoritmos e Estruturas de Dados (Árvores Binárias)

Prof^o: Joelson Antônio dos Santos

Universidade Federal de Viçosa
Instituto de Ciências Exatas e Tecnológicas
Campus de Rio Paranaíba - MG

joelsonn.santos@gmail.com
Sala: BBT 233

12 de junho de 2018

Aula de Hoje

1 Árvores Binárias

- Tipos de implementações
 - Estática
 - Dinâmica
- Percursos
 - Pré ordem
 - Em ordem
 - Pós ordem

Árvores Binárias

- Assim como as estruturas vista até o momento, árvores podem ser implementadas de duas maneiras:
 - Estática
 - Dinâmica

Árvores Binárias

- Algumas operações que são comuns em árvores:
 - Criar/Inicializar árvore;
 - Inserir elemento (à direita e à esquerda);
 - Remover elemento (à direita e à esquerda);
 - Buscar elemento;
 - Impressão de percursos;
 - Verificar se está vazia
 - Retornar o número de nós, altura, níveis, grau e etc;

Árvores Binárias - Implementação Estática

- É utilizado um vetor de tamanho fixo para representar a árvore estática.
- Uma estrutura para o tipo de informação pode ser criada.
- Uma estrutura para o tipo **árvore** pode ser criada, limitada a um número máximo de elementos.

```
#include <stdio.h>
#define TAM 50

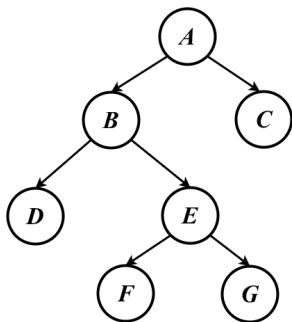
typedef struct sInfo{
    <tipo> info;
}NO;

typedef struct sArv{
    NO elemento[TAM];
}ARVORE;
```

Árvores Binárias - Implementação Estática

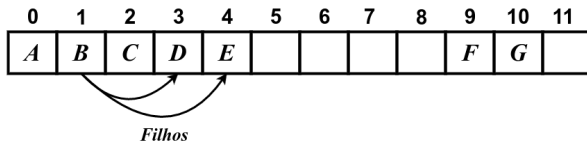
- Duas funções são utilizadas para calcular as posições no vetor para os filhos de qualquer (nó) pai.
- Sendo assim, os filhos de um nó na posição i de um vetor são:
 - Filho à esquerda: $f_{esq}(i) = 2i + 1$;
 - Filho à direita: $f_{dir}(i) = 2i + 2$;

Árvores Binárias - Implementação Estática



$$f_{esq} = 2i + 1$$

$$f_{dir} = 2i + 2$$



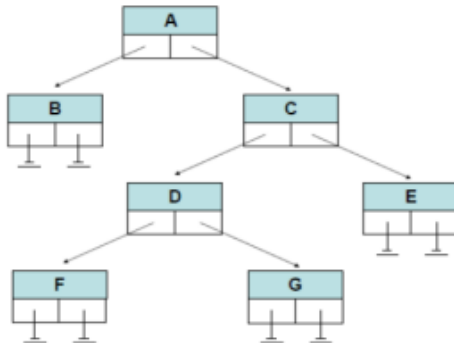
- Qual o problema deste tipo de implementação?

Árvores Binárias - Implementação Dinâmica

- É utilizado o recurso de alocação dinâmica;
- Cada nó da árvore é tratado como um ponteiro alocado dinamicamente a medida que os nós são inseridos;
- Podemos definir uma estrutura para o tipo de dado e uma estrutura para o tipo de nós da árvore;

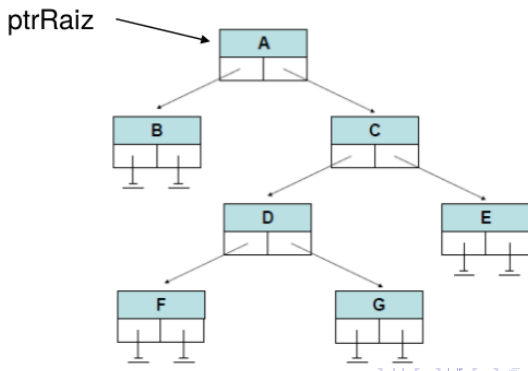
Árvores Binárias - Implementação Dinâmica

```
typedef struct sInfo{  
    <tipo> info;  
}ELEMENTO;  
  
typedef struct sArv{  
    ELEMENTO info;  
    struct sArv *esq;  
    struct sArv *dir;  
}ARVORE;
```



Árvores Binárias - Implementação Dinâmica

- Para guardar o primeiro nó da árvore é necessário utilizar um **ponteiro para ponteiro**. O objetivo é poder atualizar a raiz da árvore caso seja necessário.



Árvores Binárias - Percursos

- Uma operação muito comum em árvore é a maneira em que a mesma é percorrida;
- Geralmente, o percurso é feito nó a nó uma única vez. Nesse caso, cada nó possuirá:
 - Nó predecessor;
 - Nó sucessor;

Árvores Binárias - Percursos

- Existem alguns tipos de percursos utilizados para processar os nós de uma determinada árvore:
 - Pré ordem (raiz, esquerda, direita);
 - Em ordem (esquerda, raiz, direita);
 - Pós ordem (esquerda, direita, raiz);
- Cada percurso **visita** um nó em uma ordem diferente;
- “Visita” pode ser interpretado como impressão, acesso/modificação, remoção de um elemento, dentre outras;

Árvores Binárias - Percursos

- **Pré ordem:**

- Visitar a raiz da árvore ou sub-árvore;
- Visitar recursivamente sua sub-árvore à esquerda;
- Visitar recursivamente sua sub-árvore à direita;

Árvores Binárias - Percursos

- **Em ordem:**

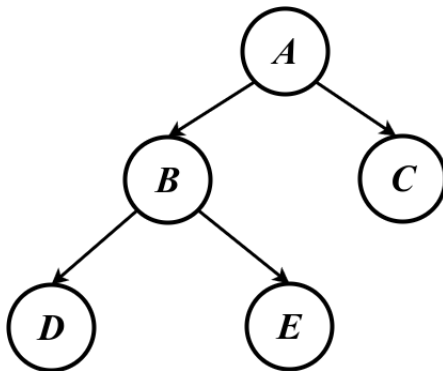
- Visitar recursivamente sua sub-árvore à esquerda;
- Visitar a raiz da árvore ou sub-árvore;
- Visitar recursivamente sua sub-árvore à direita;

- **Pós ordem:**

- Visitar recursivamente sua sub-árvore à esquerda;
- Visitar recursivamente sua sub-árvore à direita;
- Visitar a raiz da árvore ou sub-árvore;

Árvores Binárias - Percursos

- Dada uma árvore A com três elementos:
- Qual seria a saída para os percursos, pré ordem, em ordem e pós ordem?



Árvores Binárias - Percursos


- **Pré ordem:**
 - A, B, D, E, C
- **Em ordem:**
 - D, B, E, A, C
- **Pós ordem:**
 - D, E, B, C, A

Árvores Binárias - Percursos

- Como ficaria a implementação do percurso **pré ordem** para impressão dos elementos?

Árvores Binárias - Percursos


- Como ficaria a implementação do percurso **pré ordem** para impressão dos elementos?



```
void preOrdem(ARVORE *raiz){  
    if(raiz != NULL){  
        visita(raiz);  
        preOrdem(raiz->esq);  
        preOrdem(raiz->dir);  
    }  
}
```

Árvores Binárias - Percursos

- Como ficaria a implementação do percurso **pré ordem** para impressão dos elementos?




```
void preOrdem(ARVORE *raiz){  
    if(raiz != NULL){  
        visita(raiz);  
        preOrdem(raiz->esq);  
        preOrdem(raiz->dir);  
    }  
}
```

- O que a função visita faz nesta função?

Árvores Binárias - Percursos

- Como ficaria a implementação do percurso **pré ordem** para impressão dos elementos?

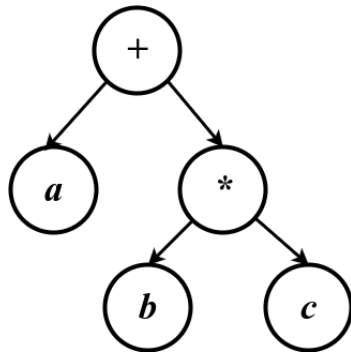


```
void preOrdem(ARVORE *raiz){  
    if(raiz != NULL){  
        visita(raiz);  
        preOrdem(raiz->esq);  
        preOrdem(raiz->dir);  
    }  
}
```

- O que a função visita faz nesta função?
- Como seriam as implementações dos percursos **em ordem** e **pós ordem**?

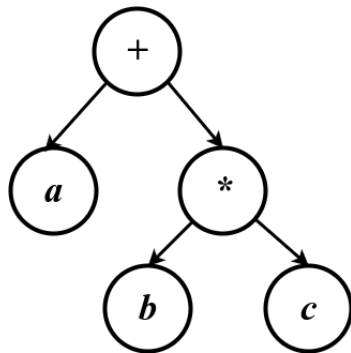
Árvores Binárias - Percursos

- Percurso em expressões aritméticas;
- O percurso utilizado pode definir o tipo de notação utilizada:
 - Pré ordem:



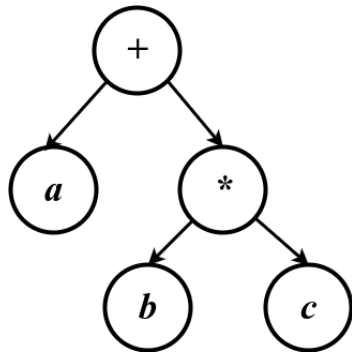
Árvores Binárias - Percursos

- Percurso em expressões aritméticas;
- O percurso utilizado pode definir o tipo de notação utilizada:
 - Pré ordem:
 - $+a * bc$
notação pré-fixa;



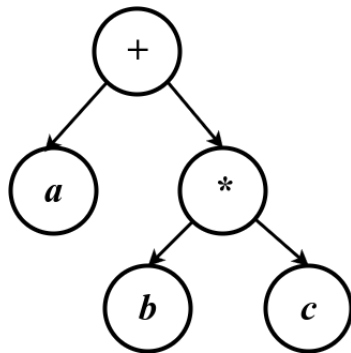
Árvores Binárias - Percursos

- Percurso em expressões aritméticas;
- O percurso utilizado pode definir o tipo de notação utilizada:
 - Em ordem:



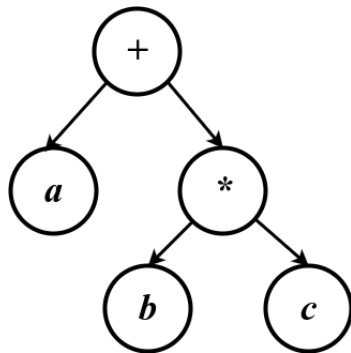
Árvores Binárias - Percursos

- Percurso em expressões aritméticas;
- O percurso utilizado pode definir o tipo de notação utilizada:
 - Em ordem:
 - $a + b * c$
notação infixa;



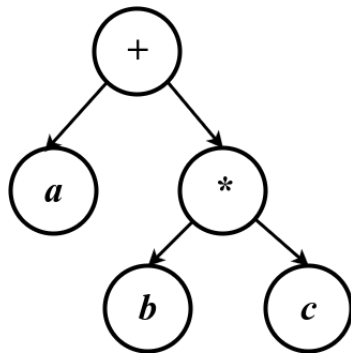
Árvores Binárias - Percursos

- Percurso em expressões aritméticas;
- O percurso utilizado pode definir o tipo de notação utilizada:
 - Pós ordem:



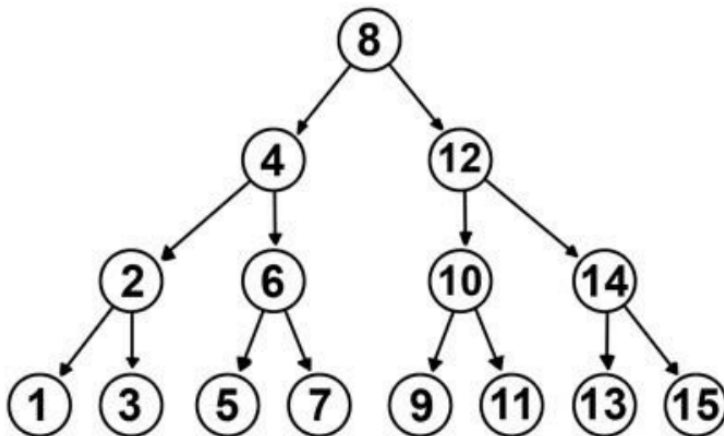
Árvores Binárias - Percursos

- Percurso em expressões aritméticas;
- O percurso utilizado pode definir o tipo de notação utilizada:
 - Pós ordem:
 - $abc * +$
notação pós-fixa;



Exercícios

- Escreva os percursos vistos em aula para a seguinte árvore:



Bibliografia Básica

- DROZDEK, Adam. Estrutura de Dados e Algoritmos em C++. Editora Pioneira Thomson Learning, 2005.
- <https://goo.gl/gk01D5> - Acessado em 7 de novembro de 2017.
- Estrutura de dados descomplicada em linguagem C, CAPÍTULO 11 - André Ricardo Backes, <https://www.evolution.com.br/epubreader/estrutura-de-dados-descomplicada-em-linguagem-c-1ed>