Algoritmos e Estrutura de Dados

Aula 15 – Estrutura de Dados:
Pesquisa em Árvores Binárias

Prof. Tiago A. E. Ferreira

Pesquisa em Árvore Binária

- A operação mais comum em uma árvore de pesquisa binária:
  - Pesquisa por um elemento ou chave!
    - Função: **SEARCH**
- Além da função de busca (search), uma árvore binária ainda admite as funções:
  - **MINIMUM**: valor mínimo
  - **MAXIMUM**: valor máximo
  - **SUCCESSOR**: sucessor de um valor
  - **PREDECCESSOR**: predecessor de um valor

Função Pesquisa - SEARCH

- Como Pesquisar:
  - O procedimento abaixo utiliza um ponteiro *x* para a raiz da árvore e um valor *k* como chave de busca.
  - Este irá retornar um ponteiro para o nodo da árvore que contiver a chave *k*
  - Caso não exista nenhum nodo com a chave *k*, então o procedimento retorna um ponteiro NULO (None ou Nill)

```
TREE-SEARCH(x, k)
1  if x = NIL or k = chave[x]
2      then return x
3  if k < chave[x]
4      then return TREE-SEARCH(esquerda[x], k)
5      else return TREE-SEARCH(direita[x], k)
```

Função Pesquisa - SEARCH

- A pesquisa começa pela RAIZ da árvore binária, e traça um caminho descendente pela árvore
- Para cada nodo x da árvore há uma comparação:
  - K==CHAVE[X]
    - Se K = CHAVE[X], então Fim da pesquisa
    - Se K < CHAVE[X], então vá para sub-árvore esquerda
    - Se K > CHAVE[X], então vá para sub-árvore direita
- No pior caso, o custo em tempo T(n) = O(n), onde n é a profundidade da árvore!

Função Pesquisar - SEARCH

- Há ainda uma forma iterativa para se montar a mesma função de pesquisa, que na maior parte das situações tende a ser mais rápida
  - Função Iterativa:

```
ITERATIVE-TREE-SEARCH(x, k)
1  while x ≠ NIL e k ≠ chave[x]
2      do if k < chave[x]
3          then x ← esquerda[x]
4          else x ← direita[x]
5  return x
```

Função Mínimo - MINUMUM

- Em uma árvore binária, por construção, o nodo com o valor mínimo é aquele encontrado mais a esquerda.
  - A função TREE-MINIMUM(X) irá procurar o elemento mais a esquerda e retorná-lo:

```
TREE-MINIMUM(x)
1  while esquerda[x] ≠ NIL
2      do x ← esquerda[x]
3  return x
```

Função Máximo - MAXIMUM

- Em uma árvore binária, por construção, o nodo com o valor máximo é aquele encontrado mais a direita.
  - A função TREE-MAXIMUM(X) irá procurar o elemento mias a direita e retorná-lo:

```
TREE-MAXIMUM(x)
1  while direita[x] ≠ NIL
2      do x ← direita[x]
3  return x
```

Função Sucessor - SUCCESSOR

- Como em uma árvore binária todas as chaves (ou elementos) são distintos, dado um elemento x o seu sucessor será o nodo que tiver o menor elemento maior que o elemento x
  - Função: (p[x] – é o pai de x)

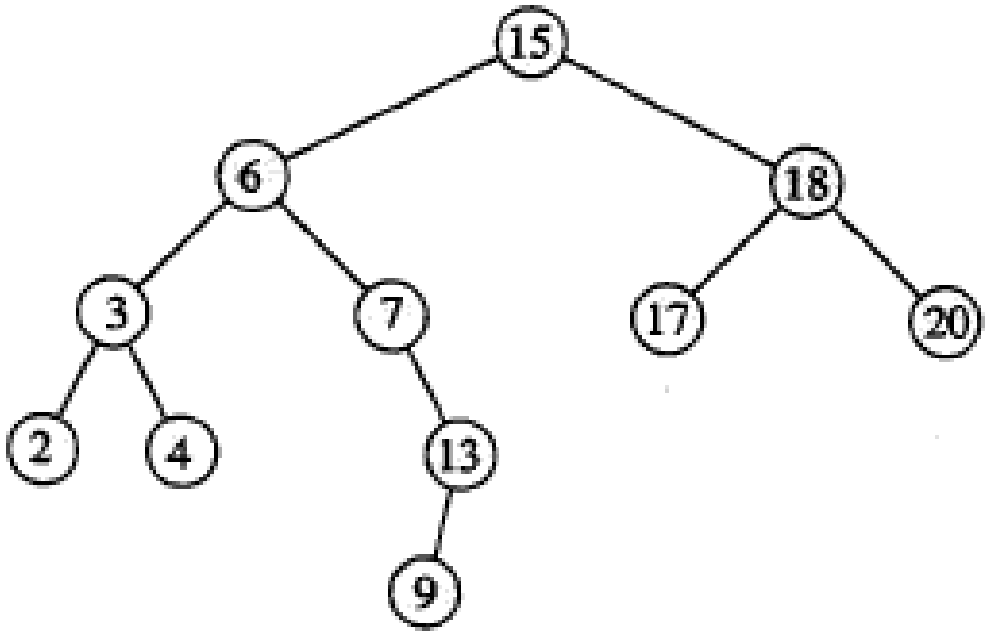
```
TREE-SUCCESSOR(x)
1  if direita[x] ≠ NIL
2     then return TREE-MINIMUM(direita[x])
3  y ← p[x]
4  while y ≠ NIL e x = direita[y]
5     do x ← y
6        y ← p[y]
7  return y
```

Função Sucessor - SUCCESSOR

- Se a sub-árvore direita de X é não nula, então o seu sucessor será o menor elemento da sua sub-árvore direita.
- Caso contrário, o sucessor de X será o ancestral mais baixo de X cujo filho da esquerda também seja um ancestral de X

Função Sucessor - SUCCESSOR

- Exemplo, quem é o sucessor de 13?
  - ***Resp.: 15!***



Função Predecessor - PREDECCESSOR

- Como em uma árvore binária todas as chaves (ou elementos) são distintos, dado um elemento x o seu predecessor será o nodo que tiver o maior elemento menor que o elemento x
  - Se o nó tiver sub-árvore esquerda não nula, então o seu predecessor será o maior elemento da sua sub-árvore esquerda
  - Caso contrário, o seu predecessor será o seu ancestral mais baixo cujo seu filho direito seja também um ancestral de X

Exercício

- ***Elabore a função que determina o elemento predecessor de um dado nodo X em uma árvore binária,***