Universidade Federal do ABC
Algoritmos e Estruturas de Dados II 3^o Quadrimestre 2018

Prof. Carlos da Silva dos Santos

Aula prática 03 – Árvore Binária de Busca.

1 - Estude o código dos arquivos `abb.h` e `abb.c`, que contém funções para a criação e manipulação de árvores binárias de busca. Tente entender cada função fornecida. Além das funções já presentes em `abb.c`, note que `abb.h` contém a definição de algumas funções ainda não implementadas, que você deverá criar mais adiante.
2 - Estude o código do arquivo `teste01.c`. Compile e execute o arquivo resultante. Você deve usar um comando de compilação como:

```
gcc -g -Wall -o teste01 teste01.c abb.c
```

3 - Implemente a função `preenche_pai` declarada em `abb.h`:

```
/* Preenche o ponteiro pai de todos nos da arvore, dado o no raiz */
void preenche_pai(no *r);
```

Essa função deve atribuir corretamente o valor do ponteiro `pai` para todos os nós da árvore. No caso do nó raiz, o ponteiro `pai` deve apontar para o próprio nó. *Dica:* a implementação pode ficar mais simples se você usar uma função recursiva. Talvez seja necessário criar uma função auxiliar para ajudar a lidar com o caso do nó raiz.

4 - Retire o comentário do trecho final de `teste01.c` para habilitar o trecho que usa a função `preenche_pai` e teste a sua função.

5 - Estude o código do arquivo `teste02.c`. Para usar esse arquivo, você precisará implementar a função `inserir_no_na_arvore`. Essa função recebe uma árvore e um valor inteiro que deve ser inserido na árvore. Nós vamos restringir a nossa árvore para armazenar apenas valores distintos, então se o usuário tentar inserir uma chave pré-existente na árvore, sua função deve devolver `NULL` e não alterar a árvore. Caso a sua função consiga inserir o novo valor, a função deve devolver o ponteiro para a raiz da árvore. Caso a árvore fornecida seja inicialmente vazia (valor `NULL`), a sua função deve criar um novo nó para guardar o valor inserido e devolver o ponteiro para o nó.