2ª Trabalho

Curso: Engenharia de Computação
Disciplina: Estruturas de Dados
Prof. Dr. Jarbas Joaci de Mesquita Sá Junior
Universidade Federal do Ceará – UFC/Sobral

Entrega: 25/01/2016 via e-mail para jarbas_joaci@yahoo.com.br – **Obs 1**: Não receberei o trabalho após a data mencionada. **Obs 2:** O trabalho é "individual". **Obs 3:** plágio acarretá diminuição de pontos ou anulação do trabalho (ou seja, aconselho "fortemente" que não mostrem o seu código ao colega).

1ª) Implemente a TAD "arvb.h" (Árvore Binária de Buscas) e acrescente as seguintes funções: (6,0 pontos)

a) função que retorne a quantidade de folhas de uma árvore binária de busca que possuem no campo `info` um número primo. Essa função deve obedecer ao protótipo:

```
int folhas_primos(ArvB* a);
```

b) função que retorne a quantidade de nós de uma árvore binária de busca que possuem os dois filhos (campos `dir` e `esq` diferentes de `NULL`). Essa função deve obedecer ao protótipo:

```
int dois_filhos(ArvB* a);
```

c) função que, dada uma árvore binária de busca, retorne a quantidade de nós cujas subárvores esquerda e direita tenham igual altura. Essa função deve obedecer ao protótipo:

```
int nos_igual_altura(ArvB* a);
```

d) função que compare se duas árvores binárias de busca são iguais. Essa função deve obedecer ao protótipo:

```
int iguais(ArvB* a, ArvB* b);
```

Obs: 1 – verdadeiro; 0 – falso.

A seguir, execute o seguinte programa.

```
#include <stdio.h>
#include <stdlib.h>
#include "arvb.h"


int main(void){

  Arv* arvA = arvb_cria_vazia();
  arvA=arvb_insere(arvA,3);
  arvA=arvb_insere(arvA,5);
  arvA=arvb_insere(arvA,2);
```

```
  arvA=arvb_insere(arvA,4);
  arvA=arvb_insere(arvA,7);
  arvA=arvb_insere(arvA,0);
  arvA=arvb_remove(arvA,4);
  printf(''Altura da árvore %d\n'',arv_altura(arvA));
  printf(''Qtd folhas primos %d\n'',folhas_primos(arvA));
  printf(''Qtd de nós dois filhos %d\n'',dois_filhos(arvA));
  printf(''Nós igual altura %d\n'',nos_igual_altura(arvA));

  Arv* arvB = arvb_cria_vazia();

  arvB=arvb_insere(arvB,8);
  arvB=arvb_insere(arvB,9);
  arvB=arvb_insere(arvB,11);

  arvb_imprime(arvA); //impressao em ordem simétrica
  arvb_imprime(arvB); //impressao em ordem simétrica

  int comp = iguais(arvA,arvB);

  printf(''arvores iguais %d\n'',comp);

  arvb_libera(arvA);
  arvb_libera(arvB);

  system(''PAUSE'');
  return 0;
}
```

2ª) Implemente os algoritmos **BubbleSort**, **InsertionSort**, **QuickSort, MergeSort** e **HeapSort** e calcule o tempo médio de cada um para ordenar vetores com valores aleatórios de tamanho 10^2, 10^3, 10^4, 10^5 e 10^6. Elabore um relatório com os dados obtidos. (4,0 pontos)

Obs: O tempo deve ser dado em milissegundos.

Boa diversão!