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

Entrega: 14/04/2021 via e-mail para jarbas_joaci@yahoo.com.br – **Obs.**: o trabalho não será recebido após a data mencionada. Preferencialmente fazer o trabalho usando a IDE Dev-C++. Enviar todos os arquivos do projeto, exceto os executáveis (.exe).

1ª) Implemente o Tipo Abstrato de Dados (TAD) "lista.h" (ver slides sobre Listas Encadeadas) e acrescente as seguintes funções:

a) função que verifique se duas listas `l1` e `l2` possuem os mesmos valores (não necessariamente na mesma ordem). Essa função deve obedecer ao protótipo:

```
int duas_lst_mesmo_val(Lista* l1, Lista* l2);
```

Obs. o retorno deve ser 1 – verdadeiro ou 0 – falso.

b) função para criar uma lista que é formada apenas pelos nós da lista `l` que possuem o campo `info` divisível por `n`. Na **nova** lista os elementos devem estar em ordem crescente. Essa função deve obedecer ao protótipo:

```
Lista* lst_div_n(Lista* l, int n);
```

Obs. a lista `l` permanece inalterada após a execução da função.

A seguir, execute o seguinte programa.

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

int main(void){
  Lista* l1 = lst_cria();
  l1 = lst_insere(l1,6);
  l1 = lst_insere(l1,13);
  l1 = lst_insere(l1,25);
  l1 = lst_insere(l1,28);
  l1 = lst_insere(l1,40);
  lst_imprime(l1);

  Lista* l2 = lst_cria();
  l2 = lst_insere(l2,25);
  l2 = lst_insere(l2,13);
  l2 = lst_insere(l2,6);
  l2 = lst_insere(l2,40);
  l2 = lst_insere(l2,28);
```

```
  lst_imprime(l2);

  Lista* l3= lst_div_n(l1,2);
  lst_imprime(l3);

  printf("mesmos valores %d\n",duas_lst_mesmo_val(l1,l2));
  printf("mesmos valores %d\n",duas_lst_mesmo_val(l1,l3));

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

2. Implemente a TAD "arvb.h" (Árvore Binária de Buscas) e acrescente as seguintes funções:

a) função que verifique se duas árvores `arv1` e `arv2` possuem os mesmos valores (não necessariamente na mesma ordem). Essa função deve obedecer ao protótipo:

```
int duas_arv_mesmo_val(ArvB* a1, ArvB* a2);
```

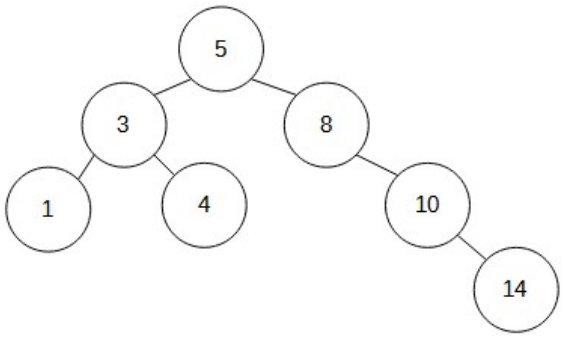
Obs. o retorno deve ser 1 – verdadeiro ou 0 – falso.

b) função que imprima a quantidade de folhas em cada nível de uma árvore binária de busca. Essa função deve obedecer ao protótipo :

```
void impressao_arv_folhas_niveis(ArvB* a);
```

Por exemplo, na árvore da figura abaixo, a impressão deve ser:

nível 0 – nº de folhas: 0
nível 1 – nº de folhas: 0
nível 2 – nº de folhas: 2
nível 3 – nº de folhas: 1



A seguir, execute o seguinte programa.

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


int main(void){

  ArvB* arv1 = arvb_cria_vazia();
```

```
  arv1=arvb_insere(arv1,41);
  arv1=arvb_insere(arv1,7);
  arv1=arvb_insere(arv1,55);
  arv1=arvb_insere(arv1,71);
  arv1=arvb_insere(arv1,40);
  arv1=arvb_insere(arv1,10);
  arv1=arvb_insere(arv1,6);
  arv1=arvb_insere(arv1,66);

  ArvB* arv2 = arvb_cria_vazia();

  arv2=arvb_insere(arv2,5);
  arv2=arvb_insere(arv2,8);
  arv2=arvb_insere(arv2,9);
  arv2=arvb_insere(arv2,11);
  arv2=arvb_insere(arv2,71);

  ArvB* arv3 = arvb_cria_vazia();

  arv3=arvb_insere(arv3,9);
  arv3=arvb_insere(arv3,71);
  arv3=arvb_insere(arv3,11);
  arv3=arvb_insere(arv3,5);
  arv3=arvb_insere(arv3,8);

  printf("mesmos val. %d\n",duas_arv_mesmo_val(arv1,arv2));
  printf("mesmos val. %d\n",duas_arv_mesmo_val(arv2,arv3));
  printf("mesmos val. %d\n",duas_arv_mesmo_val(arv1,arv3));

  impressao_arv_folhas_niveis(arv1);
  impressao_arv_folhas_niveis(arv2);
  impressao_arv_folhas_niveis(arv3);

  arvb_libera(arv1);
  arvb_libera(arv2);
  arvb_libera(arv3);

  system("PAUSE");
  return 0;

}
```