**Questão 1**. Faça os testes de mesa e apresente os valores das variáveis de saída para o código abaixo:

#include <stdio.h>

int main() {

int valor;

int \*p1;

float temp;

float \*p2;

char aux;

char \*nome = "ABCDEFGHIJ";

char \*p3;

int idade;

int vetor[3];

int \*p4;

int \*p5;

/\* (a) \*/

valor = 10;

p1 = &valor;

\*p1 = 20;

printf("(a) %d \n", valor);

/\* (b) \*/

temp = 29.5;

p2 = &temp;

\*p2 = 23.0;

printf("(b) %.1f \n", temp);

/\* (c) \*/

p3 = &nome[0];

aux = \*p3;

printf("(c) %c \n", aux);

/\* (d) \*/

p3 = &nome[4];

aux = \*p3;

printf("(d) %c \n", aux);

/\* (e) \*/

p3 = nome;

printf("(e) %c \n", \*p3);

/\* (f) \*/

p3 = p3 + 4;

printf("(f) %c \n", \*p3);

/\* (g) \*/

p3--;

printf("(g) %c \n", \*p3);

/\* <h> \*/

vetor[0] = 33;

vetor[1] = 25;

vetor[2] = 7;

p4 = vetor;

idade = \*p4;

printf("(h) %d \n", idade);

/\* (i) \*/

p5 = p4 + 1;

idade = \*p5;

printf("(i) %d \n", idade);

/\* (j) \*/

p4 = p5 + 1;

idade = \*p4;

printf("(j) %d \n", idade);

/\* (l) \*/

p4 = p4 - 2;

idade = \*p4;

printf("(l) %d \n", idade);

/\* (m) \*/

p5 = &vetor[2] - 1;

printf("(m) %d \n", \*p5);

/\* (n) \*/

p5++;

printf("(n) %d \n", \*p5);

return(0);

}

(a) 20

(b) 23.0

(c) A

(d) E

(e) A

(f) E

(g) D

(h) 33

(i) 25

(j) 7

(l) 33

(m) 25

(n) 7

**Questão 2**. Faça uma função para escrever os valores de uma LLDE. A função receberá uma LLDE e deverá escrever os valores de seus elementos em ordem inversa (i.e: do último para o primeiro)

void mostrarInvertidoLLDE(LLDE l){

LLDE aux=l;

if(l){ // Existe a lista -- Tem elementos

if(!l->prox) printf("L->[x|%i|x]\n",aux->info); //Apenas um elemento

else for(aux=l;aux->prox!=NULL;aux=aux->prox); // Mais de um elemento, vá para o final

printf("L->[x|%i|-]->",aux->info);

aux=aux->ant;

while(aux&&aux->ant!=NULL){

printf("<-[-|%i|-]->",aux->info);

aux=aux->ant;

}

if(aux)printf("<-[-|%i|x]\n",aux->info);

} else printf("L->x\n"); // Lista Vazia -- Não tem elementos

}

**Questão 3**. Defina uma estrutura de dados para um programa de leilão de carros. O programa deverá registrar o Modelo, a Marca, o Ano de fabricação e o preço do lance vencedor em reais. O programa manipulará os dados na forma de uma lista linear simplesmente encadeada.

struct carro{

char modelo[40];

char marca[40];

int ano;

float lanceVencedor;

struct carro \*prox;

};

**Questão 4**. Escreva uma função para encontrar o carro mais caro vendido no leilão descrito na questão anterior. Considere que a lista de carros já foi preenchida previamente.

struct carro{

char modelo[40];

char marca[40];

int ano;

float lanceVencedor;

struct carro \*prox;

};

typedef struct carro \*CARROS;

CARROS carroMaisCaro(CARROS c){

CARROS aux=c, maiorLance=(CARROS) malloc(sizeof(struct carro));

maiorLance->lanceVencedor=0.0;

if(aux){

while (aux) {

if(aux->lanceVencedor>maiorLance->lanceVencedor){

strcpy(maiorLance->modelo, aux->modelo);

strcpy(maiorLance->marca, aux->marca);

maiorLance->ano = aux->ano;

maiorLance->lanceVencedor = aux->lanceVencedor;

}

aux=aux->prox;

}

return maiorLance;

} else return NULL;

}

**Boa Prova.**