

CENTRO UNIVERSITÁRIO INTERNACIONAL UNINTER

ESCOLA SUPERIOR POLITÉCNICA

TECNOLOGIA EM ANALISE E DESENVOLVIMENTO DE SISTEMAS

DISCIPLINA DE LINGUAGEM DE PROGRAMAÇÃO

ATIVIDADE PRÁTICA

ALYSSON DUMA – RU: 3050279

PROFESSOR:VAGNER

CURITIBA – PARANÁ

2019

**Exercício 1**: Escreva um algoritmo em linguagem C com as seguintes instruções:

1. Declare três variáveis (inteiro, real e char);

2. Declare três ponteiros;

3. Associe as variáveis aos ponteiros;

4. Modifique os valores de cada variável indiretamente usando os ponteiros associados. Para armazenar os valores nas variáveis, armazene na variável char a primeira letra do seu nome, na variável inteira os dois últimos dígitos do seu RU e na variável real os 4 últimos dígitos do seu RU, sendo os 2 últimos os valores com virgula;

5. Imprima na tela os valores das variáveis antes e após a modificação.

#include <stdio.h>

#include <stdlib.h>

int main() {

//DECLARAR VARIAVEL

int RU;

float RUltimo;

char letra;

//DECLARAR PONTEIRO DA VARIAVEL

int \*pRU;

float \*pRUltimo;

char \*pletra;

//PONTEIRO RECEBE VARIAVEL

pRU=&RU;

pRUltimo=&RUltimo;

pletra=&letra;

//USANDO A VARIAVEL INT

RU=79;

printf("RU=%d\n", RU);

//RECEBE 30 PARA ONDE ESTA APONTANDO

\*pRU=30;

printf("RU=%d\n", RU);

//USANDO A VARIAVEL CHAR

letra='A';

printf("primeira letra:%c\n", letra);

//RECEBE D PARA ONDE ESTA APONTANDO

\*pletra='D';

printf("primeira letra:%c\n", letra);

//USANDO A VARIAVEL FLOAT

RUltimo=5279;

printf("ultimos numeros do RU:%.2f\n", RUltimo);

//RECEBE 9725 PARA ONDE ESTÁ APONTANDO

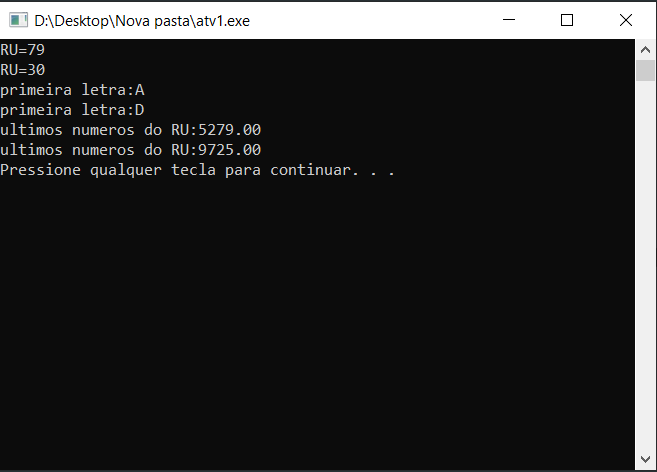
\*pRUltimo=9725;

printf("ultimos numeros do RU:%.2f\n", RUltimo);

system("pause");

return 0;

}



**Exercício 2:** Escreva um algoritmo em LINGUAGEM C que armazene na memória o seu RU e o valor 1234567, ambos digitados pelo usuário na tela. Em seguida, imprima na tela ambos RU usando ponteiros. O algoritmo também vai ter que comparar os dois RU usando ponteiros e imprimir na tela qual é o maior.

#include <stdio.h>

#include <stdlib.h>

int main() {

//DECLARAR VARIAVEL

int RU;

int valor;

printf("digite seu RU:");

scanf("%d", &RU); //ARMAZENA NA MEMORIA VARIAVEL 'RU'

printf("digite um valor:");

scanf("%d", &valor);//ARMAZENA NA MEMORIA VARIAVEL 'VALOR'

//DECLARAR PONTEIRO VARIAVEL

int \*pRU;

int \*pvalor;

pRU = &RU;

pvalor = &valor;

//MOSTRAR CONTEUDO PONTEIRO VARIAVEL

printf("RU= %d\n", \*pRU);

printf("valor=%d\n", \*pvalor);

//COMANDO DE SELECAO SE-ENTAO USANDO PONTEIROS

if (\*pRU > \*pvalor)

printf("RU eh maior = %d\n", \*pRU);

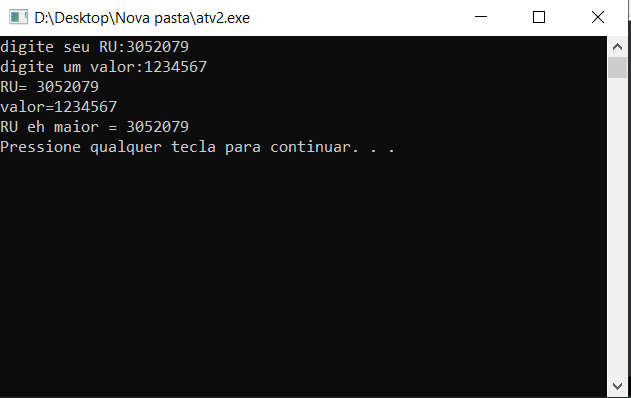
else

printf("valor eh maior = %d\n", \*pvalor);

system("pause");

return 0;

}



**Exercício 3**: Faça um algoritmo em linguagem C com as seguintes funcionalidades:

• Receba um registro, com dois campos, como dados de entrada.

• O primeiro campo é um vetor que vai armazenar o nome do aluno.

• O segundo campo é uma variável do tipo inteiro que vai armazenar o RU do aluno.

• Imprime na tela os dados armazenados na estrutura.

#include <stdlib.h>

#include <stdio.h>

//DECLARAR STRUCT DO ALUNO

struct aluno {

int ru;

char nome[101];

};

//DECLARAR VARIAVEL

int main() {

struct aluno aluno1;

printf("Digite seu Nome:"); // IMPRIMA NA TELA

fflush(stdin); //COMANDO PARA LIMPAR BUFFER DO TECLADO

fgets(aluno1.nome, 100, stdin); // ALOCANDO OS DADOS DO ALUNO

printf("Digite seu RU:"); // IMPRIMA NA TELA

scanf("%d", &aluno1.ru); //ARMAZENA NA MEMORIA VARIAVEL

printf("nome=%s\n", aluno1.nome); // IMPRIMA NA TELA

printf("RU=%d\n", aluno1.ru); // IMPRIMA NA TELA

system("pause");

return 0;}



**Exercício 4:** Replique o exercício 3. Porém, agora, declare um ponteiro para a estrutura de dados heterogênea. No momento da leitura dos dados e da impressão na tela, use o ponteiro para buscar o conteúdo dos campos. Imprima na tela também o seu RU na tela.

#include <stdlib.h>

#include <stdio.h>

//DECLARAR STRUCT DO ALUNO

struct dados\_aluno

{

int ru;

char \*nome;

};

int main(void)

{

typedef struct dados\_aluno aluno;//CRIAR UM TIPO NOVO

aluno aluno1;

aluno \*aluno2=&aluno1; //ENDERECO DO PONTEIRO

aluno2->nome=(char\*)malloc(sizeof(char)\*100); //ALOCAR ESPACO PARA O BLOCO

printf("Digite seu Nome:"); // IMPRIMA NA TELA

fflush(stdin); //COMANDO PARA LIMPAR BUFFER DO TECLADO

fgets(aluno2->nome, 100, stdin); // ALOCANDO OS DADOS DO ALUNO

printf("Digite seu RU:"); // IMPRIMA NA TELA

scanf("%d", &aluno2->ru); //ARMAZENA NA MEMORIA VARIAVEL

printf("nome=%s\n", aluno2->nome); // IMPRIMA NA TELA

printf("RU=%d\n", aluno2->ru); // IMPRIMA NA TELA

system("pause");

return 0;

}



**Exercício 5:** Faça um algoritmo em linguagem C que contenha dois números inteiros digitados na tela pelo usuário:

a. O primeiro número marca um início;

b. O segundo número marca um fim;

O algoritmo vai contar quantos números existem entre o início (primeira entrada) e o fim (segunda entrada). A impressão na tela do usuário deve ser realizada de duas formas:

a. Iterativa;

b. Recursiva;

Ao colocar no seu relatório uma imagem do seu código funcionando, coloque ele rodando utilizando como valor de inicio os 2 últimos valores do seu RU e valor final o número 99.

#include<stdio.h>

#include<stdlib.h>

int impressao(int inicio, int fim);

int main()

{

int inicio, fim, i;

printf("digite o primeiro numero:"); //IMPRESSAO NA TELA

scanf\_s("%d", &inicio);  //LEITURA NA TELA E ARMAZENAR VARIAVEL

printf("digite o segundo numero:"); //IMPRESSAO NA TELA

scanf\_s("%d", &fim); // LEITURA NA TELA E ARMAZENAR VARIAVEL

printf ("\nForma Recursiva\n");//IMPRESSAO NA TELA

for (i = inicio; i <= fim; i++) //LAÇO DE REPETIÇAO

{

printf("%d ", i); //IMPRESSAO NA TELA

}

printf("\n\n");//IMPRESSAO NA TELA

printf("Forma Interativa\n");//IMPRESSAO NA TELA

impressao(inicio, fim);

if (inicio < fim) { //TESTA CONDICIONAL

printf("%d", inicio); //IMPRESSAO NA TELA

impressao(inicio + 1, fim); //SOMA 1 A VARIAVEL

}

printf("\n\n");

system("PAUSE");

return 0;

}

int impressao(int inicio, int fim) //DECLARACAO DA VARIAVEL

{

if (inicio <= fim)

{

printf("%d ", inicio); //IMPRESSAO NA TELA

impressao(inicio + 1, fim); //SOMA 1 A VARIAVEL

}

return 0;

}

