Curso : Engenharia de Computação

Disciplina : Programação de Computadores e Algoritmos

Aluno: Adevan Neves Santos

Período :2020/2

96. Crie um algoritmo que, usando uma função, retorne o valor da divisão de dois

números inteiros para o programa principal e mostre o resultado.

Função Principal: pedirá os dois números inteiros, chamará a função DIVISÃO e

exibirá o resultado.

Função Divisão: dividirá os números e retornará o valor da divisão para a função

principal.

Função dividir(a,b : inteiro ) : real

VAR

div : real

Início

div= a/b

Retorne div

Fim-Função

Algoritmo dividirDoisNumeros

VAR

a,b : inteiro

divisao : real

Início

Imprimir (“ Digite os números a e b separados por vírgulas :”)

Leia(a,b)

divisao= dividir (a,b)

Imprimir (“ A divisão é :”)

Imprimir (divisao)

Fim-Algoritmo

97. Criar um algoritmo que apresente um menu de seleções no algoritmo principal.

Esse menu deverá dar ao usuário a possibilidade de escolher uma entre quatro

operações aritméticas. Escolhida a opção desejada, deverá ser solicitada a entrada

de dois números. Processada a operação, o resultado deverá ser exibido. O

algoritmo deverá ser executado enquanto o usuário quiser.

Função Principal:

• Chamará o procedimento Desenha\_Menu que mostrará as opções na tela.

• Receberá a opção do usuário.

• Enquanto a opção do usuário for diferente de “0 – sair”, pedirá dois números

inteiros.

• Utilizará a instrução ESCOLHA chamar a função que corresponda à opção do

usuário.

Função Desenha\_Menu: Mostrará um menu de opções (“0 – sair”), (“1 – dividir”),

(“2 – somar”), (“3 – multiplicar”) e (“4 – subtrair”).

Função Dividir: dividirá os valores lidos e mostrará o resultado.

Função Somar: somará os valores lidos e mostrará o resultado.

Função Multiplicar: multiplicará os valores lidos e mostrará o resultado

Função Desenha\_Menu ( ) :

VAR

Início

Imprimir (“ (“0 – sair”), (“1 – dividir”),

(“2 – somar”), (“3 – multiplicar”) e (“4 – subtrair”)”)

Fim-Função

Função soma(a,b : real ) : vetor[50] de caractere

VAR

soma : real

resposta : vetor[50] de caratere

Início

soma=a+b

resposta=“A soma resultou em :”+Numpcarac(soma:real)

Retornar resposta

Fim-Função

Função sub (a,b : real) : vetor [50] de caractere

VAR

subtracao : real

resposta : vetor[50] de caractere

Início

subtracao=a/b

resposta=“ A subtração resultou em :”+Numpcarac(subtracao:real)

Retornar resposta

Fim-Função

Função mult(a,b : real ) : real

VAR

mult : real

Início

mult=a\*b

resposta=“A multiplicação foi “+Numpcarac(mult:real)

Retornar resposta

Fim-Função

Função div (a , b : real ) : real

VAR

div

Início

div=a/b

resposta=A divisão resultou em :”+Numpcarac(div : real

Fim-Função

Algoritmo Calculadora

VAR

op : inteiro

a,b : real

Início

Faça

Desenha\_Menu( )

Leia(op)

Selecione (op)

Caso 1

Imprimir (div(a,b))

Caso 2

Imprimir (soma(a,b))

Caso 3

Imprimir (mult(a,b))

Caso 4

Imprimir (sub(a,b))

Fim-Selecione

Enquanto (op!=0)

98. Faça um algoritmo que leia os valores dos salários atuais dos funcionários de uma

empresa e imprima os valores com aumento. A função main deve pedir ao usuário

para digitar o salário de um funcionário e chamar a função Calcula\_Aumento( ),

receber o retorno desta função, imprimir o novo valor do salário e perguntar se o

usuário deseja digitar outro salário ou não. Se o usuário responder que sim, o ciclo

deve se repetir até que o usuário responda que não. A função Calcula\_Aumento

deve seguir a seguinte regra para calcular o novo salário: se o salário for menor ou

igual a R$500,00, deve aplicar um aumento de 0.10 (10%) e se for maior que

R$500,00, o aumento deve ser de 0.08 (8%). Essa função deve calcular o salário e

retornar o valor para a função principal.

Função CALCULA\_AUMENTO( salario\_atual : real ) : real

VAR

acrescimo, salario\_final : real

Início

Se (salario\_atual<=500)

acrescimo=salario\_atual\*0.10

salario\_final= salario\_atual+acrescimo

Retornar (salario\_final)

Senão

acrescimo=salario\_atual\*0.80

salario\_final=salario\_atual+acrescimo

Retornar (salario\_final)

Fim-Se

Fim-Função

Função cadeia main()

VAR

cadeia salario

cadeia salario\_novo

Início

Imprimir (“ Digite o seu salário atual:”)

Leia(salario)

salario\_novo=CALCULA\_AUMENTO(salário)

Imprimir (“ O seu salário recebeu um aumento e agora vale :”)

Imprimir (salario\_novo)

Fim-Função

Algoritmo Aumento\_de\_salario

VAR

resposta : vetor[4] de caractere

Início

Faça

main()

Imprimir (“Deseja continuar ? (sim/não)”)

Leia(resposta)

Enquanto (resposta!=”não”)

Fim-Algoritmo

99. Escreva uma função que exiba as componentes de um vetor na ordem inversa

daquela em que foram armazenadas.

#include <stdio.h>

#include <stdlib.h>

void Reverso(int n, int \* v, int \*r)

{

int i;

int j;

j=n-1;

for (i=0;i<n;i++){

r[i]=v[j];

j--;

}

}

int main(void)

{

/\*99.Escreva uma função

que exiba as componentes de

um vetor na ordem inversa

daquela em que foram

armazenadas.\*/

int \*v;

int \*r;

int n;

int i;

printf("n:");

scanf("%d",&n);

v=(int \*) malloc(n\*sizeof(int));

r=(int \*) malloc(n\*sizeof(int));

for (i=0;i<n;i++){

printf("Vetor %d:",i);

scanf("%d",&v[i]);

}

Reverso(n,v,r);

printf("Vetor Reverso:\n");

for (i=0;i<n;i++){

printf ("reverso[%d]: %d\n",i,r[i]);

}

free(v);

free(r);

return 0;

}

100.Um vetor é palíndromo se ele não se altera quando as posições das componentes são invertidas. Por exemplo,o vetor v = {1, 3, 5, 2, 2, 5, 3, 1} é palíndromo. Escreva uma função que verifique se um vetor é palíndromo.

#include <stdio.h>

#include <stdlib.h>

void Palin(int n, int \* v, int \*r)

{

int i;

int j, palindromo;

j=n-1;

for (i=0; i<n; i++) {

r[i]=v[j];

j--;

}

for (j=0;j<n;j++){

if (v[j]!=r[j])

palindromo=0;

else

palindromo=1;}

if (palindromo)

printf("\*\*\*É um palindromo\*\*\*\n");

else

printf("\*\*\*Não palindromo\*\*\*\n");

}

int main(void)

{

int \*v;

int \*r;

int n;

int i,j,palindromo;

printf("n:");

scanf("%d",&n);

v=(int \*) malloc(n\*sizeof(int));

r=(int \*) malloc(n\*sizeof(int));

for (i=0; i<n; i++) {

printf("Vetor %d:",i);

scanf("%d",&v[i]);

}

Palin(n,v,r);

printf("Vetor Reverso:\n");

for (i=0; i<n; i++) {

printf ("reverso[%d]: %d\n",i,r[i]);

}

free(v);

free(r);

return 0;

}

101.Escreva uma função que receba um vetor e o decomponha em dois outros vetores,

um contendo as componentes de ordem ímpar e o outro contendo as componentes

de ordem par. Por exemplo, se o vetor dado for v = {3, 5, 6, 8, 1, 4, 2, 3, 7}, o vetor

deve gerar os vetores u = {3, 6, 1, 2, 7} e w = {5, 8, 4, 3}.

#include <stdio.h>

#include <stdlib.h>

int main(void)

{

int n,i,ip=0,imp=0;

int \* v;

int \* vp;

int \* vimp;

printf("n: ");

scanf("%d",&n);

v = (int \*) malloc(n\*sizeof(int));

vp = (int \*) malloc(n\*sizeof(int));

vimp = (int \*) malloc(n\*sizeof(int));

for(i=0;i<n;i++){

printf("v[%d]: ",i);

scanf("%d",&v[i]);

}

for(i=0;i<n;i++){

if (v[i]%2==0){

vp[ip]=v[i];

ip++;

}

else{

vimp[imp]=v[i];

imp++;

}

}

printf("Vetor Par:\n");

for (i=0;i<ip;i++){

printf("vetor par [%d] %d\n",i,vp[i]);

}

for (i=0;i<imp;i++){

printf("vetor impar[%d] %d\n",i,vimp[i]);

}

return 0;

}

102.Um vetor do Rn é uma n-upla de números reais v = {x1, x2, ..., xn}, sendo cada xi

chamado de componente. A norma de um vetor v = {x1, x2, ..., xn} é definida por

Escreva uma função que receba um vetor do Rn

, n dado, e forneça sua norma.

#include <math.h>

#include <stdio.h>

#include <stdlib.h>

void Norma(int n, float \*v, float \*norma){

int i;

float acum=0.0F;

for(i=0;i<n;i++){

acum=acum+v[i]\*v[i];

}

\*norma=sqrt(acum);

}

int main()

{

int n,i;

float \* v ;

float norma=0.0F;

printf("Quant de elementos: ");

scanf("%d",&n);

v = (float \*) malloc(n\*sizeof(float));

for (i=0;i<n;i++){

printf("v[%d]: ",i);

scanf("%f",&v[i]);

}

Norma(n,v,&norma);

printf("A norma foi :%.2f",norma);

free(v);

return 0;

}

103.O produto escalar de dois vetores do Rn é a soma dos produtos das componentes

correspondentes. Isto é, se u = {x1, x2, ..., xn} e v = {y1, y2, ..., yn}, o produto

escalar é x1.y1 + x2.y2 ... + xn.yn. Escreva uma função que receba dois vetores do

Rn, n dado, e forneça o produto escalar deles.

#include <math.h>

#include <stdio.h>

#include <stdlib.h>

float Produto\_Escalar(int n, float \*v, float \*u){

int i;

float prod=0.0F;

for(i=0;i<n;i++){

prod=prod+v[i]\*u[i];

}

return prod;

}

int main(void)

{

int n,i;

float \* v ;

float \* u;

float prod=0.0F;

printf("Quant de elementos: ");

scanf("%d",&n);

v = (float \*) malloc(n\*sizeof(float));

u = (float \*) malloc(n\*sizeof(float));

for (i=0;i<n;i++){

printf("v[%d]: ",i);

scanf("%f",&v[i]);

printf("\n");

printf("u[%d]: ",i);

scanf("%f",&u[i]);

}

prod=Produto\_Escalar(n,v,u);

printf("A norma foi :%.2f",prod);

free(v);

return 0;

}

104.A amplitude de uma relação de números reais e a difereça entre o maior e o

menor valores da rela��o. Por exemplo, a amplitude da rela��o 5, 7, 15, 2, 23 21, 3,

6 � 23 - 2 = 21. Escreva uma fun��o que receba uma rela��o de n�meros e forne�a

sua amplitude

#include <stdio.h>

#include <stdlib.h>

float Amplitude(int n, float \*v)

{

int i;

float h, maior, menor;

maior=v[0];

menor=v[0];

for(i=0;i<n;i++){

if (v[i]>=maior){

maior=v[i];

}

else if(v[i]<=menor){

menor=v[i];

}

}

h = maior - menor;

return h;

}

int main(void){

int n, i;

float amp;

float \*v;

printf("Quantidade de elementos: ");

scanf("%d",&n);

v = (float \*) malloc(n\*sizeof(float));

for(i=0;i<n;i++){

printf("v[%d]: ",i);

scanf("%f",&v[i]);

}

amp=Amplitude(n,v);

printf("Amplitude: %f",amp);

free(v);

return 0;

}

105.O desvio padrão de uma relação de números reais é a raiz quadrada da média

aritmética dos quadrados dos desvios. Escreva uma função que receba uma

relação de números reais e forneça o seu desvio padrão.

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

double Media(int n,double \*v)

{

int i;

double media\_simples=0.000;

for (i=0;i<n;i++){

media\_simples=media\_simples+v[i];

}

media\_simples=media\_simples/n;

return media\_simples;

}

double desvio\_padrao(int n, double \*v)

{

int i;

double desvio=0.0000, media=0.000;

media=Media(n,v);

for (i=0;i<n;i++){

desvio=desvio+(v[i]-media)\*(v[i]-media);

}

desvio=sqrt(desvio/n);

return desvio;

}

/\* %g especifica um double (ou float ) no formato mais apropriado(%f ou %e) printf\*/

/\* %lf,%le,%lg especificam um double\*/

int main(void)

{

int n,i;

double \* v, desvio;

printf("Digite a quantidade de termos: ");

scanf("%d",&n);

v = (float \*) malloc(n\*sizeof(double));

for (i=0;i<n;i++){

printf("v[%d]: ",i);

scanf("%lf",&v[i]);

}

desvio=desvio\_padrao(n,v);

printf("O desvio padrão é: %g",desvio);

free(v);

return 0;

}

106.Escreva uma função que forneça as componentes distintas de um vetor dado. Por

exemplo, se o vetor dado for v = {3, 2, 1, 3, 4, 1, 5, 5, 2} a função deve fornecer v =

{3, 2, 1, 4, 5}.

#include <stdlib.h>

#include <stdio.h>

void vetor\_unico(int n, int \*v,int \*vu,int \*nps)

{

int i, cont,nao\_ta\_dentro;

vu[0]=v[0];

for (i=0;i<n;i++){

for(cont=0;cont<= \*nps;cont++){

if (v[i]==vu[cont]){

nao\_ta\_dentro=0;

break;

}

else{

nao\_ta\_dentro=1;

}

}

if (nao\_ta\_dentro){

\*nps=\*nps+1;

vu[\*nps]=v[i];

}

}

}

int main(void)

{

int n,i,np=0;

int \* v;

int \* vu;

printf("n: ");

scanf("%d",&n);

v=(int \*) malloc(n\*sizeof(int));

vu=(int \*) malloc(n\*sizeof(int));

for (i=0;i<n;i++){

printf("v[%d]: ",i);

scanf("%d",&v[i]);

}

printf("Vetor Único de v\n");

vetor\_unico(n,v,vu,&np);

for(i=0;i<=np;i++){

printf("\nvetor\_unico[%d]: %d",i,vu[i]);

}

free(vu);

free(v);

return 0;

}

107.Algumas empresas que realizam sorteios de prêmios entre seus clientes o fazem

através dos sorteios da loteria federal, sendo ganhador o número formado pelos

algarismos das casas das unidades dos números sorteados nos cinco prêmios da

referida loteria. Por exemplo, se o sorteio da loteria federal deu como resultado os

números 23451, 00234, 11236, 01235 e 23452, o prêmio da tal empresa seria dado

ao cliente que possuísse o bilhete de número 14652. Escreva uma função que

receba os números sorteados pela loteria federal e forneça o número que ganhará

o prêmio de acordo com as regras acima.

#include <stdio.h>

#include <stdlib.h>

int main(void)

{

int sort1[5],sort2[5],sort3[5],sort4[5],sort5[5],premio[5];

int n=5,i;

for(i=0;i<n;i++){

printf("\nsorteio1[%d]: ",i);

scanf("%d",&sort1[i]);

}

for(i=0;i<n;i++){

printf("\nsorteio2[%d]: ",i);

scanf("%d",&sort2[i]);

}

for(i=0;i<n;i++){

printf("\nsorteio3[%d]: ",i);

scanf("%d",&sort3[i]);

}

for(i=0;i<n;i++){

printf("\nsorteio4[%d]: ",i);

scanf("%d",&sort4[i]);

}

for(i=0;i<n;i++){

printf("\nsorteio5[%d]: ",i);

scanf("%d",&sort5[i]);

}

premio[0]=sort1[4];

premio[1]=sort2[4];

premio[2]=sort3[4];

premio[3]=sort4[4];

premio[4]=sort5[4];

printf("O prêmio é :%d%d%d%d%d",premio[0],premio[1],premio[2],premio[3],premio[4]);

return 0;

}

108.Escreva uma função que insira um valor dado num vetor numa posição dada. Por

exemplo, se o vetor for v = {3, 8, 5, 9, 12, 3}, o valor dado for 10 e a posição dada for 4,

a função deve fornecer v = {3, 8, 5, 10, 9, 12, 3}.

#include <stdlib.h>

#include <stdio.h>

int main(void)

{

int n,i,pos,valor,temp1,temp2;

int \*v;

printf("n: ");

scanf("%d",&n);

v = (int\*) malloc(n\*sizeof(int));

for (i=0;i<n;i++){

printf("v[%d]: ",i);

scanf("%d",&v[i]);

}

printf("Posição que deseja alterar: ");

scanf("%d",&pos);

v=(int\*)realloc(v,(n+1)\*sizeof(int));

printf("valor: ");

scanf("%d",&valor);

/\*Realizar a primeira troca\*/

temp1=v[pos-1];

v[pos-1]=valor;

for(i=pos;i<(n+1);i++){

/\*temp2 guarda o valor

do vetor antes da mudança\*/

temp2=v[i];

/\*temp1 está guardando o valor

uma posição antes da atual\*/

v[i]=temp1;

/\*agora eu atualizo temp1 pois temp2 é

anterior a próxima iteração\*/

temp1=temp2;

}

for(i=0;i<(n+1);i++){

printf("\nv[%d]: %d",i,v[i]);

}

free(v);

return 0;

}

109.Escreva uma função que insira um valor dado num vetor ordenado de modo que o

vetor continue ordenado. Por exemplo, se o vetor dado for v = {2, 5, 7, 10, 12, 13} e

o valor dado for 6, a função deve fornecer o vetor v = {2, 5, 6, 7, 10, 12, 13}.

#include <stdio.h>

#include <stdlib.h>

void inserir(int n,int elemento,int \*v)

{

/\*Irei percorrer a o vetor até

encontrar um valor maior que o

elemento dado, após isto eu

colocarei o elem no lugar deste valor\*/

int temp1,temp2,ind,i;

for (i=0;i<n;i++){

if (v[i]>=elemento){

ind=i;

temp1=v[i];

v[i]=elemento;

break;}

}

for(i=ind+1;i<(n+1);i++){

/\*temp2 guarda o valor

do vetor antes da mudança\*/

temp2=v[i];

/\*temp1 está guardando o valor

uma posição antes da atual\*/

v[i]=temp1;

/\*agora eu atualizo temp1 pois temp2 é

anterior a próxima iteração\*/

temp1=temp2;

}

}

int main(void)

{

int n,i,elemento;

int \*v;

printf("n: ");

scanf("%d",&n);

v=(int \*)malloc((n+1)\*sizeof(int));

for(i=0;i<n;i++){

printf("v[%d]: ",i);

scanf("%d",&v[i]);

}

printf("elemento: ");

scanf("%d",&elemento);

inserir(n,elemento,v);

for(i=0;i<(n+1);i++){

printf("\nv[%d]: %d",i,v[i]);

}

free(v);

return 0;

}

110.Escreva uma função que remova uma componente de ordem dada de um vetor

dado. Por exemplo, se o vetor dado for v = {2, 5, 7, 10, 12, 13} e a componente a

ser removida for a de ordem 4, programa deve fornecer o vetor v = {2, 5, 7, 12, 13}.

#include <stdio.h>

#include <stdlib.h>

void remover(int n,int pos,int \*v,int \*vn)

{

int i;

pos=pos-1;

for(i=0;i<n;i++){

if(i<pos){

vn[i]=v[i];

}

else if(i==pos){

continue;

}

else{

vn[i-1]=v[i];

}

}

}

int main(void)

{

int n,i,pos;

int \*v;

int \*vn;

printf("n: ");

scanf("%d",&n);

v=(int \*) malloc(n\*sizeof(int));

vn=(int \*)malloc((n-1)\*sizeof(int));

for (i=0;i<n;i++){

printf("v[%d]: ",i);

scanf("%d",&v[i]);

}

printf("Posição do elemento a retirar: ");

scanf("%d",&pos);

remover(n,pos,v,vn);

for(i=0;i<(n-1);i++){

printf("\nv[%d]: %d",i,vn[i]);

}

free(v);

free(vn);

return 0;

}

111.Escreva uma função que, dadas duas relações de números, cada uma delas com

números distintos, forneça os números que aparecem nas duas listas. Por exemplo,

se as relações forem u = {9, 32, 45, 21, 56, 67, 42, 55} e w = {24, 42, 32, 12, 45, 11,

67, 66, 78}, a função deve fornecer o vetor v = {32, 45, 67, 42}.

#include <stdio.h>

#include <stdlib.h>

int conjIntersecao(int n,int m, int \*v, int \*u,int \*intersec)

{

int i,j,k=0;

/\*Preciso percorrer o vetor v todo e

com cada elemento dele eu comparo com

todos os elementos do vetor u\*/

for (i=0;i<n;i++){

for (j=0;j<m;j++){

if (v[i]==u[j]){

intersec[k]=v[i];

k++;

break;

}

}

}

return k;

}

int main(void)

{

int i,n,m,k;

int \* v;

int \* u;

int \*intersec;

printf("n: ");

scanf("%d",&n);

printf("m: ");

scanf("%d",&m);

v= (int \*)malloc(n\*sizeof(int));

u= (int \*)malloc(m\*sizeof(int));

if(n>m){

intersec=(int \*) malloc(m\*sizeof(int));

}

else{

intersec=(int \*)malloc(n\*sizeof(int));

}

for(i=0;i<n;i++){

printf("v[%d]: ",i);

scanf("%d",v+i);

}

printf("\n");

for(i=0;i<m;i++){

printf("u[%d]: ",i);

scanf("%d",u+i);

}

k=conjIntersecao(n,m,v,u,intersec);

for(i=0;i<k;i++){

printf("\nintersec[%d]: %d",i,intersec[i]);

}

return 0;

}

112.Uma avaliação escolar consiste de 50 questões objetivas, cada uma delas com 5

opções, v = {1, 2, 3, 4 e 5}, sendo apenas uma delas verdadeira. Escreva uma

função que receba a sequência de respostas corretas, o gabarito, e corrija um

cartão-resposta dado.

#include <stdlib.h>

#include <stdio.h>

void vetor\_gabarito(int \*vgabar)

{

int i,n;

n=50;

printf("insira as respostas no vetor gabarito:\n");

for(i=0;i<n;i++){

printf("\nvgab[%d]: ",i);

scanf("%d",vgabar+i);

}

}

void corrigir\_prova(int \*vgabar,int \*vcorrig,int \*vprova)

{

int i,resp,n=50;

printf("\n");

for(i=0;i<n;i++){

if(vprova[i]==vgabar[i]){

resp=1;

vcorrig[i]=resp;

}

else{

resp=0;

vcorrig[i]=resp;

}

}

for(i=0;i<n;i++){

printf("\nvcorrig[%d]: %d",i,vcorrig[i]);

}

}

int main(void){

int n=50,i;

int vgabar[n];

int vcorrig[n];

int vprova[n];

vetor\_gabarito(vgabar);

printf("\nInsira o cartão resposta:");

for(i=0;i<n;i++){

printf("\nvprova[%d]: ",i);

scanf("%d",vprova+i);

}

corrigir\_prova(vgabar,vcorrig,vprova);

printf("\n(0) --> Não acertou (1) -->acertou");

return 0;

}

113.Escreva uma função que forneça o valor numérico de um polinômio P(x) dado,

para um valor de x dado. Por exemplo, se o polinômio dado for P(x) = x3 + 2x - 1 e o

valor de x dado for 2, a função deve fornecer P(2) = 23 + 2x2 - 1 = 11.

114.Escreva uma função que forneça a transposta de uma matriz dada.

#include <stdio.h>

#include <stdlib.h>

int \*\* transposta(int m, int n, int \*\* matriz)

{

int i,j;

int \*\*trp;

/\* minha matriz de m linhas e n colunas\*/

/\*vou iniciar as colunas\*/

trp=(int \*\*) malloc(m\*sizeof(int\*));

for (i=0;i<m;i++){

trp[i]=(int \*)malloc(n\*sizeof(int));

}

for(i=0;i<m;i++){

for(j=0;j<n;j++){

trp[i][j]=matriz[j][i];

}

}

return trp;

}

int main(void)

{

int n,m,i,j;

int \*\*matriz;

int \*\*trp;

printf("(m)linhas: ");

scanf("%d",&m);

printf("(n) colunas:");

scanf("%d",&n);

/\*montar a estrutura das matrizes\*/

matriz=(int\*\*)malloc(m\*sizeof(int\*));

trp=(int\*\*)malloc(m\*sizeof(int\*));

for(i=0;i<m;i++){

matriz[i]=(int \*)malloc(n\*sizeof(int));

trp[i]=(int \*)malloc(n\*sizeof(int));

}

printf("Insira os valores: ");

for(i=0;i<m;i++){

for(j=0;j<n;j++){

printf("\nmatriz[%d][%d]: ",i,j);

scanf("%d",&matriz[i][j]);

}

}

trp=transposta(m,n,matriz);

for(i=0;i<m;i++){

for(j=0;j<n;j++){

printf("\ntrp[%d][%d]: %d",i,j,trp[i][j]);

}

}

/\*Liberar memória da matriz\*/

for(i=0;i<m;i++){

free(matriz[i]);

free(trp[i]);

}

free(matriz);

free(trp);

return 0;

}

115.Uma matriz quadrada é dita triangular se os elementos situados acima de sua

diagonal principal são todos nulos. Escreva uma função que receba uma matriz

quadrada e verifique se ela é triangular.

int verif\_triang(int n, int \*\*matriz)

{

int i,j,eh\_triangular=1;

for(i=0;i<n;i++){

for(j=(i+1);j<n;j++){

if(matriz[i][j]!=0){

eh\_triangular=0;

break;

}

}

}

return eh\_triangular;

}

int main(void)

{

int i,j,n,eh\_triangular;

int \*\*matriz;

printf("Ordem(n) da matriz:");

scanf("%d",&n);

matriz=(int \*\*)malloc(n\*sizeof (int\*));

for(i=0;i<n;i++){

matriz[i]=(int \*)malloc(n\*sizeof (int));

}

printf ("Valores:\n");

for(i=0;i<n;i++){

for(j=0;j<n;j++){

printf("\nmatriz[%d][%d]: ",i,j);

scanf("%d",&matriz[i][j]);

}

}

eh\_triangular=verif\_triang(n,matriz);

if(eh\_triangular){

printf("Matriz triangular !!");

}

else{

printf("Não é matriz triagular !!");

}

for(i=0;i<n;i++){

free(matriz[i]);

}

free(matriz);

return 0;

}

116.Escreva uma função que determine o produto de duas matrizes.

#include <stdlib.h>

#include <stdio.h>

/\*116.Escreva uma função que determine

o produto de duas matrizes.\*/

int \*\* comp\_mat(int m,int n)

{

int i,j;

int \*\* mat;

mat=(int \*\*) malloc(m\*sizeof(int\*));

for(i=0;i<m;i++){

mat[i]=(int \*) malloc(n\*sizeof(int));

}

printf("\*\*\*Insira os valores:\*\*\*");

for(i=0;i<m;i++){

for(j=0;j<n;j++){

printf("\nmat[%d][%d]: ",i,j);

scanf("%d",&mat[i][j]);

}

}

return mat;

}

/\* Sendo A m×r1 e B r2×n , com r2=r1

então (AB)ij = (ai1)(b1j) +(ai2)(b2j)

\*/

int \*\* produto(int r,int m,int n,int\*\*A,int \*\* B)

{

int i,j,aux=0,x;

int \*\* AB;

AB=(int \*\*)malloc(m\*sizeof(int\*));

for(i=0;i<m;i++){

AB[i]=(int\*)malloc(n\*sizeof(int));

}

/\*Está iteração irá montar a matriz

m×n AB\*/

for(i=0;i<m;i++){

for(j=0;j<n;j++){

AB[i][j]=0;

for(x=0;x<r;x++){

aux+=A[i][x]\*B[x][j];

}

AB[i][j]=aux;

aux=0;

}

}

return AB;

}

int main(void)

{

int m,r1=2,r2=3,n,i,j;

int \*\*A;

int \*\*B;

int \*\*AB;

while (r1!=r2){

printf("(A) tem de ter o número de colunas igual só número de linhas de (B) !\n");

printf("\n(m)linhas(A): ");

scanf("%d",&m);

printf("\n(r1)colunas(A): ");

scanf("%d",&r1);

printf("\n(r2)linhas(B): ");

scanf("%d",&r2);

printf("\n(n)colunas(B): ");

scanf("%d",&n);

}

A=(int \*\*) malloc(m\*sizeof(int\*));

B=(int \*\*)malloc(r2\*sizeof(int\*));

AB=(int \*\*)malloc(m\*sizeof(int\*));

for(i=0;i<m;i++){

A[i]=(int \*) malloc(r1\*sizeof(int));

B[i]=(int \*) malloc(r2\*sizeof(int));

AB[i]=(int \*)malloc(n\*sizeof(int));

}

A=comp\_mat(m,r1);

B=comp\_mat(r2,n);

AB=produto(r1,m,n,A,B);

for(i=0;i<m;i++){

for(j=0;j<n;j++){

printf("\nAB[%d][%d]: %d",i,j,AB[i][j]);

}

}

return 0;

}

117.As companhias de transportes aéreos costumam representar os nomes dos

passageiros no formato último sobrenome/nome. Por exemplo, o passageiro Carlos

Drumond de Andrade seria indicado por Andrade/Carlos. Escreva um programa que

receba um nome e o escreva no formato acima.

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

int main(void)

{

int n,i,j,tam,tamFrm,iv,jv;

char \* nome\_completo;

char \*ultSob;

char \*nome;

char \*nome\_formato;

printf("n: ");

scanf("%d\n",&n);

nome\_completo=(char \*)malloc(n\*sizeof(char));

scanf("%90[^\n]", nome\_completo);

/\*preciso de tamanho desta string,pois o n

é uma estimativa do usuário\*/

tam=strlen(nome\_completo);

/\*usarei o primeiro while para pegar o último índice

sobrenome ao contrário\*/

iv=(tam-1);

while(nome\_completo[iv]!=' '){

iv--;

}

/\*o índice que o IV está quardando é do espaço em branco\*/

jv=0;

while (nome\_completo[jv]!=' '){

jv++;

}

ultSob=(char \*)malloc((tam-iv)\*sizeof(char));

/\*Agora eu posso concatenar pelo índice\*/

j=0;

for(i=(iv+1);i<tam;i++){

ultSob[j]=nome\_completo[i];

j++;

}

nome=(char \*)malloc((tam-jv)\*sizeof(char));

ultSob[j]='/';

ultSob[j+1]='\0';

for(i=0;i<jv;i++){

nome[i]=nome\_completo[i];

}

i=0;

j=0;

tamFrm=strlen(nome)+strlen(ultSob);

nome\_formato=(char \*)malloc((tamFrm+1)\*sizeof(char));

for(i=0;i<strlen(ultSob);i++){

nome\_formato[i]=ultSob[i];

}

for(j=0;j<strlen(nome);j++){

nome\_formato[i]=nome[j];

i++;

}

printf("passageiro:%s",nome\_formato);

free(nome\_completo);

free(ultSob);

free(nome\_completo);

return 0;

}

118.É muito comum que os títulos de documentos como avisos, declarações, atestados

etc., apareçam em letras maiúsculas separadas por um espaço em branco. Escreva

uma função que receba uma palavra e a retorne no formato acima.

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

char maiuscula(char c)

{

if(c>='a' && c<='z'){

c=c-'a'+'A';

}

return c;

}

char \* documento(char \* frase)

{

int i,n=strlen(frase);

char \* formato=(char \*)malloc((n\*2+1)\*sizeof(char));

char temp;

for(i=0;i<n;i++){

temp=maiuscula(frase[i]);

frase[i]=temp;

}

for(i=0;i<n;i++){

formato[2\*i]=frase[i];

formato[2\*i+1]=' ';

}

formato[2\*i+1]='\0';

return formato;

}

int main(void)

{

int n;

char \*frase;

char \*formato;

printf("n: ");

scanf("%d\n",&n);

frase=(char \*)malloc(n\*sizeof(char));

formato=(char \*)malloc((2\*n+1)\*sizeof(char));

scanf("%30[^\n]",frase);

formato=documento (frase);

printf("frase:%s",formato);

return 0;

}