REMINDERS

**-procedimento = void, X return**

**-função = qlqr um**

**-char (%c) = letra/nome**

**-float(%f)  mlr q double (%lf) -> float = números quebrados com ponto**

**-perguntando é =, afirmando é = =**

**-scanf = qnd pergunta // X scanf = qnd afirma**

**-& = só qnd usa scanf ou mostrar/pegar o endereço do ponteiro/variavel (chamando função - recursividade)**

PONTEIRO

PASSAGEM POR REFERÊNCIA

**Int func (int \*va, int \*vb) {}**

**Int main {**

**func(&a, &b); }**

**—**

**-\*p -> conteúdo**

**-&p -> endereço**

-int variavel=250; //variavel inteira c valor 250

int \*ponteiro; //ponteiro para a variavel

int exibe; //variavel q vai exibir o valor 250

ponteiro = &variavel; //ponteiro esta apontando p o endereço da variavel

exibe = \*ponteiro; //a variavel exibe recebe o conteudo do end p a qual o ponteiro aponta - 250

**-int a;**

**int \*n = &a;**

**-printf(“valor de a: %d”, a);**

**-printf(“endereço de a: %p”, &a);**

**-printf(“valor do ponteiro de a: %d”, \*n);**

**-printf(“endereço do ponteiro de a: %p”, &n);**

**-no int do ponteiro, se ta escrito por exemplo: troca1 = \*valor1; \*valor1 = \*valor2; \*valor2= \*valor3; \*valor3 = troca1; significa que o valor valor2 vai ser escrito no lugar do valor1; que o valor3 vai estar escrito no lugar do valor 2 e que o valor da troca (valor1) vai estar escrito no lugar do valor3**

Escreva um programa em C que declare variáveis para armazenar 1 valor inteiro, 1 valor real e 1 caracter. Deve existir no programa ponteiros associados a cada um deles. O programa deve solicitar novos dados para as variáveis e elas devem ser modificadas usando os respectivos ponteiros. Exiba os endereços e os conteúdos de todas as variáveis e ponteiros

antes e após a alteração.

#include <stdio.h>

int main(void) {

  int valori = 3;

  float valorr = 2.2;

  char car = 't';

  int \*vi = &valori; //td q fizer no \*vi acontece no valor

  float \*vr = &valorr;

  char \*c = &car;

  //valor ANTES da variável

printf("\no valor do valor inteiro da variável é: %d\n", vi);

printf("o valor do valor real da variável é: %.2f\n", vr);

printf("o valor do caracter da variável é: %c\n", c);

//endereço ANTES da variável

printf("\no endereço do valor inteiro da variável é: %p\n", &vi);

printf("o endereço do valor real da variável é: %p\n", &vr);

printf("o endereço do caracter da variável é: %p\n", &c);

//valor ANTES do ponteiro da variável

printf("\no valor do valor inteiro do ponteiro da variável é: %d\n", \*valori);

printf("o valor do valor real do ponteiro da variável é: %.2f\n", \*valorr);

printf("o valor do caracter do ponteiro da variável é: %c\n", \*ca);

//endereço ANTES do ponteiro da variável

printf("\no endereço do valor inteiro do ponteiro da variável é: %p\n", &valori);

printf("o endereço do valor real do ponteiro da variável é: %p\n", &valorr);

printf("o endereço do caracter do ponteiro da variável é: %p\n", &ca);

//NOVOS valores

printf("\ndigite um NOVO valor inteiro: ");

scanf("%d", valori);

printf("digite um NOVO valor real: ");

scanf("%f", valorr);

printf("digite um NOVO caracter: ");

scanf(" %c", ca);

//valor DEPOIS da variável

printf("\no NOVO valor do valor inteiro da variável é: %d\n", vi);

printf("o NOVO valor do valor real da variável é: %.2f\n", vr);

printf("o NOVO valor do caracter da variável é: %c\n", c);

//endereço DEPOIS da variável

printf("\no NOVO endereço do valor inteiro da variável é: %p\n", &vi);

printf("o NOVO endereço do valor real da variável é: %p\n", &vr);

printf("o NOVO endereço do caracter da variável é: %p\n", &c);

//valor DEPOIS do ponteiro da variável

printf("\no NOVO valor do valor inteiro do ponteiro da variável é: %d\n", \*valori);

printf("o NOVO valor do valor real do ponteiro da variável é: %.2f\n", \*valorr);

printf("o NOVO valor do caracter do ponteiro da variável é: %c\n", \*ca);

//endereço DEPOIS do ponteiro da variável

printf("\no NOVO endereço do valor inteiro do ponteiro da variável é: %p\n", &valori);

printf("o NOVO endereço do valor real do ponteiro da variável é: %p\n", &valorr);

printf("o NOVO endereço do caracter do ponteiro da variável é: %p\n", &ca);

return 0;

}

Método para fazer a troca de 2 variáveis

#include <stdio.h>

void troca (int \*a, int\*b){

  int temp;

  temp = \*a;

  \*a = \*b;

  \*b = temp;

}

int main(void) {

  int x = 5, y = 3;

  printf(“%d e %d”, x, y);  // 5 e 3

  troca (&x, &y);

  printf(“%d e %d”, x, y);  // 3 e 5

}

Implemente 1 procedimento que calcule o comprimento e a área de 1 circunferência de raio R. Esse procedimento deve obedecer ao cabeçalho a seguir:

void calcCircunferencia (float R, float \*compr, float \*area). No programa principal faça a leitura do raio, acione o procedimento e exiba os resultados do comprimento e área calculados por ele.

#include <stdio.h>

void calcCircunferencia(float R, float \*compr, float \*area) {

  \*compr = 2 \* 3.14 \* R;

  \*area = 3.14 \* (R\*R);

//esse módulo recebe o endereço das variáveis que

estão no main

}

int main(void) {

  float ra, comp, a;

  printf("digite o valor do raio: ");

  scanf("%f", &ra);

  calcCircunferencia(ra, &comp, &a);

  printf("o valor do comprimento é: %.2f\n", comp);

  printf("o valor da área é: %.2f", a);

  return 0;

}

Faça 1  programa que leia 2 valores inteiros e chame 1 função que receba estes 2 valores de entrada e retorne o maior valor na primeira variável e o menor valor na segunda variável. Escreva o conteúdo das 2 variáveis na tela no programa principal.

#include <stdio.h>

int check (int \*va, int \*vb) {

  int troca=0;

  if (\*va<\*vb) {

    troca=\*va;

    \*va=\*vb;

    \*vb=troca;

    return 1;

  } else if (\*va==\*vb) {

    return 2;

  } else {

    return 0;

  }

}

int main(void) {

  int a, b;

  printf("Digite 2 valores inteiros:\n");

  scanf("%d %d", &a, &b);

  if (check(&a,&b)==1) {

    printf("b é maior que a:\nb: %d\na: %d\n", a, b);

  } else if (check(&a,&b)==2) {

    printf("a e b são iguais:\na: %d\nb: %d", a, b);

  } else if (check(&a,&b)==0) {

    printf("a é maior que b:\na: %d\nb: %d\n", a, b);

  }

  return 0;

}

VETOR (1 for) E MATRIZ (2 for) / ESTRUTURAS HOMOGÊNEAS

**-diagonal principal -> if (l == c)**

**-diagonal secundária -> if (l + c == n-1), onde n é o tamanho da matriz**

Faça 1 procedimento que receba e preencha 1 vetor

com as notas de uma turma de 10 alunos. Faça um

outro procedimento que receba 1 vetor preenchido com

as notas, calcule a média da turma e conte quantos

alunos obtiveram nota acima da média. Esse

procedimento deve exibir a média e o resultado da

contagem. Faça um programa que declare as devidas

variáveis e acione os procedimentos.

#include <stdio.h>

#include <stdlib.h>

void recebe (int v[10]){

  printf("as notas são:\n");

  for (int i = 0; i < 10; i++){

    v[i] = rand() % 10;

      printf("%d\n", v[i]);

  }

}

void calcule (int v[10]){

  int sum = 0;

  int media = 0;

  int cont = 0;

  for (int i = 0; i < 10; i++){

    sum += v[i];

  }

  printf("a soma é: %d\n", sum);

  media = sum/10;

  printf("a média das notas dos alunos é: %d", media);

  for (int i = 0; i < 10; i++){

    if (v[i] > media){

      cont++;

    }

  }

  printf("\no número de alunos acima da média é: %d", cont);

}

int main(void) {

  int v[10];

  recebe(v);

  calcule(v);

  return 0;

}

Faça 1 procedimento que preencha 1 matriz M 5 x 5. Faça uma função que receba 1 matriz

preenchida, calcule e retorne cada 1 das somas a seguir (uma função para cada letra abaixo):

#include <stdio.h>

int m = 5;

int M[5][5];

int sum = 0;

float soma\_linha\_4(float M[5][5]) {

  sum = 0;

  for (int j = 0; j < m; j++) {

    sum += M[3][j];

  }

  return sum;

}

float soma\_coluna\_2(float M[5][5]) {

  sum = 0;

  for (int i = 0; i < m; i++) {

    sum += M[1][i];

  }

  return sum;

}

float soma\_diagonal\_principal(float M[5][5]) {

  sum = 0;

  for (int i = 0; i < m; i++) {

    for (int j = 0; j < m; j++) {

      if (i == j)

        sum += M[i][j];

    }

  }

  return sum;

}

float soma\_diagonal\_secundaria(float M[5][5]) {

  sum = 0;

  for (int i = 0; i < m; i++) {

    for (int j = 0; j < m; j++) {

      if (i + j == 4)

        sum += M[i][j];

    }

  }

  return sum;

}

float soma\_todos\_os\_elemenos(float M[5][5]) {

  sum = 0;

  for (int i = 0; i < m; i++) {

    for (int j = 0; j < m; j++) {

      sum += M[i][j];

    }

  }

  return sum;

}

int main(void) {

  float M[m][m];

  float somali, somacol, somadiag, somadiagsec,

somatudo;

  for (int i = 0; i < m; i++) {

    for (int j = 0; j < m; j++) {

      M[i][j] = rand() % 10;

      printf("%.0f ", M[i][j]);

    }

    printf("\n");

  }

  float soma = soma\_linha\_4(M);

  printf("\na soma da linha 4 é: %.0f\n", soma);

  soma = soma\_coluna\_2(M);

  printf("a soma da coluna 2 é: %.0f\n", soma);

  soma = soma\_diagonal\_principal(M);

  printf("a soma da diagonal principal é: %.0f\n",

soma);

  soma = soma\_diagonal\_secundaria(M);

  printf("a soma da diagonal secundária é: %.0f\n”

, soma);

  soma = soma\_todos\_os\_elemenos(M);

  printf("a soma de todos os elementos é: %.0f\n",

soma);

  return 0;

}

Ordem dos elementos em linhas e colunas

[0,0] [0,1] [0,2] [0,3]

[1,0] [1,1] [1,2] [1,3]

[2,0] [2,1] [2,2] [2,3]

Passar matriz para vetor - fabo

#include <stdio.h>

#include <stdlib.h>

void preenche (int m[4][4]){

  for (int l=0; l<4; l++){

    for (int c=0; c<4; c++){

      m[l][c] = rand() % 10;

      printf("%d ", m[l][c]);

    }

    printf("\n");

  }

int v[4];

printf("vetor da 3° linha:\n");

  for (int i=0; i<4; i++){

    v[i] = m[2][i];

    printf("%d ", v[i]);

  }

}

int main(void) {

  int m[4][4];

  preenche(m);

  return 0;

}

3 6 7 5

3 5 6 2

9 1 2 7

0 9 3 6

vetor da 3° linha:

9 1 2 7

Faça um procedimento que preencha 2 vetores X

e Y com 10 elementos cada um (ocupando as

posições de 0 a 9 em cada vetor). Faça 1 outro

procedimento que receba 2 vetores preenchidos

e  gera 1 novo vetor com os elementos desses 2

vetores intercalados de tal forma que nas

posições ímpares do novo vetor estejam os elem

do primeiro vetor e nas posições pares os elem

do segundo vetor recebido por parâmetro. Faça

1 procedimento  que  receba  e  exiba  o  conteúdo

de 1 vetor.  Faça  um  programa  que  faça  as

devidas declarações e acione os módulos para

exemplificar o seu uso.

#include <stdio.h>

#include <stdlib.h>

void preenche (int x[10], int y[10]){

for (int i = 0; i < 10; i++){

x[i] = rand() % 10;

y[i] = rand() % 10;

}

}

void recebe (int x[10], int y[10], int v[10]){

for (int i = 0; i < 10; i++){

if (i % 2 != 0){

v[i] = x[i]; //o primeiro que muda

} else {

v[i] = y[i];

}

}

}

void imprime (int v[10]){

for (int i = 0; i < 10; i++){

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

}

}

int main(void) {

int v1[10], v2[10], v3[10];

preenche (v1, v2);

recebe(v1, v2, v3);

imprime(v3);

return 0;

}

Em uma cidade, sabe-se hipoteticamente que, em janeiro ,

não ocorreu temperatura inferior a 15°C, nem superior a

40°C. Faça um programa que armazene as temperaturas

de cada dia de janeiro em um vetor (de 31 posições),

calcule e imprima

#include <stdio.h>

int main(void) {

int v[31] = {15, 18, 33, 34, 29, 39, 28, 18, 17, 27, 36, 30,

35, 25, 19, 17, 16, 28, 29, 40, 23, 22, 21, 20, 20, 23, 24,

27, 33, 32, 31};

int tmin = 40;

int tmax = 15;

int sum = 0, media = 0, cont = 0;

for (int i = 0; i < 31; i++){

if (v[i] < tmin){

tmin = v[i];

}

if (v[i] > tmax){

tmax = v[i];

}

sum += v[i];

}

media = sum/31;

for (int i = 0; i < 31; i++){

if (v[i] < media){

cont++;

}

}

printf("a menor temperatura foi de: %d\n", tmin);

printf("a maior temperatura foi de: %d\n", tmax);

printf("a temperatura média foi de: %d\n", media);

printf("o número de dias nos quais a temperatura foi

abaixo da média foi de: %d dias", cont);

return 0;

}

ESTRUTURAS HETEROGÊNEAS STRUCT

- if (strcmp(c[n].pagina, "s")==0) //testa se as strings são iguais

-%s = qnd é string = + de 1 caracter

(nome/endereço)

**-%s NÃO usa & no scanf**

Dados os seguintes campos de um registro: nome, dia de aniversário e mês de aniversário, desenvolver um algoritmo que mostre em cada um dos meses do ano quem são as pessoas que fazem aniversário, exibir também o dia. Considere um conjunto de 40 pessoas

#include <stdio.h>

typedef struct Data {

  int dia, mes;

} Data;

typedef struct Pessoa {

  Data dtnasc;

  char nome[20];

} Pessoa;

int main() {

  int n = 4; //p fzr 40 pessoas era só colocar 40 aqui

  Pessoa p[n];

  for (int i = 0; i < n; i++) {

    printf("\ndigite seu nome: ");

    scanf("%s", p[i].nome);

    printf("digite seu dia de aniversário: ");

    scanf("%d", &p[i].dtnasc.dia);

    printf("digite seu mês de aniversário: ");

    scanf("%d", &p[i].dtnasc.mes);

  }

  for (int m = 1; m <= 12; m++) {

    printf("\naniversariantes do mês %d:\n", m);

    for (int i = 0; i < n; i++) {

      if (m == p[i].dtnasc.mes)

        printf("- %s, no dia %d\n", p[i].nome, p[i].dtnasc.dia);

    }

  }

}

1 pessoa cadastrou um conjunto de 15 registros contendo o nome da loja, telefone e preço de um eletrodoméstico. Desenvolver 1 algoritmo que permita exibir qual foi a média dos preços cadastrados e uma relação contendo o nome e o telefone das lojas cujo preço estava abaixo da média.

#include <stdio.h>

typedef struct Registro {

char nome[20];

int tel;

double preco;

} Registro;

int main(void) {

  int n = 3;

  double sum = 0, media = 0;

  Registro r[n];

  for (int i = 0; i < n; i++) {

    printf("\ndigite o nome da loja %d: ", i+1);

    scanf("%s", r[i].nome);

    printf("digite o telefone da loja: ");

    scanf("%d", &r[i].tel);

    printf("digite o preço de um eletrodoméstico: ");

    scanf("%lf", &r[i].preco);

    sum += r[i].preco;

  }

  media = sum/n;

  printf("\na média dos preços é: %.2lf\n", media);

  printf("\nlojas com preços abaixo da média:\n");

  for (int i = 0; i < n; i++) {

    if (r[i].preco < media) {

      printf("- %s, de telefone %d\n", r[i].nome, r[i].tel);

    }

  }

  return 0;

}

1 provedor de acesso à Internet mantém o seguinte cadastro de clientes: código do cliente, número de horas  de  acesso, página (S-sim ou  N-não). Elaborar  1 algoritmo que calcule e mostre um relatório contendo o valor a pagar  por cada cliente, sabendo- se que as primeiras 20 horas de acesso é R$35,00 e as horas que excederem tem o custo de R$2,50 por hora. Para os clientes que têm página, adicionar R$40,00.

#include <stdio.h>

#include <string.h>

typedef struct Acesso {

int cod;

int horas;

char pg;

float pagar;

} Acesso;

int main(void) {

  int m = 2;

  double extra, valor;

  double vf;

  int s, n;

  double pge;

  Acesso a[m];

  for (int i = 0; i < m; i++) {

    printf("\nCLIENTE %d:\ndigite seu código de cliente: ", i+1);

    scanf("%d", &a[i].cod);

    printf("digite a quantidade de horas de acesso: ");

    scanf("%d", &a[i].horas);

    printf("você quer uma página? digite s- sim ou n-não: ");

    scanf(" %c", &a[i].pg);

  }

  for (int i = 0; i < m; i++) {

    a[i].pagar = 0;

    if (a[i].horas <= 20) {

      a[i].pagar = 35.00;

      printf("\no preço a pagar pelo cliente %d é: R$35.00", i+1);

    } else if (a[i].horas > 20) {

      extra = a[i].horas - 20;

      valor = extra \* 2.50;

      a[i].pagar = 35.00 + valor;

      printf("\no preço a pagar pelo cliente %d é: R$%.2lf", i+1, a[i].pagar);

    }

  }

  for (int i = 0; i < m; i++) {

    if (a[i].pg == 's') {

      pge = a[i].pagar + 40;

    } else {

      pge = a[i].pagar;

    }

    printf("\no NOVO preço a pagar após a adição da página pelo cliente %d é: R$%.2lf", i+1, pge);

  }

  return 0;

}

FUNÇÃO RECURSIVA QUE CALCULA DIVISÃO USANDO SUBTRAÇÕES SUCESSIVAS:

#include <stdio.h>

int divisao (int numerador, int denominador) {

  int subt=numerador-denominador;

  if(subt != 0) { //diferente de 0

    return (1+divisao(subt, denominador));

  } else {

    return 1;

  }

}

int main(void) {

  int n, d;

  printf("Insira dois números inteiros\n");

  scanf("%d %d", &n, &d);

  divisao(n, d);

  printf("O resultado é %d", divisao(n, d));

  return 0;

}