aula 07/03

#include <stdio.h>  
int main(void)  
{  
 int nota=0;  
 int total=0;  
 float media=0;  
 int i=1;  
  
 while (i<11){  
 printf("Nota do aluno %d:\n", i);  
 scanf("%d",&nota);  
 total+=nota; // total = total+nota;  
 i++; // i =i+1;  
 }  
 media=(float)total/i;  
 printf("Média=%.2f \n",media);  
   
 return 0;  
}

#include<stdio.h>  
int main(void){  
int nota=0;  
  
printf("Informe a nota entre 1 e 5\n");  
scanf("%d", &nota);  
 switch(nota){  
  
 case 1:  
 printf("Péssimo!\n");  
 break;  
 case 2:  
 printf("Menos ruim!\n");  
 break;  
 case 3:  
 printf("Médio!\n");  
 break;  
 case 4:   
 printf("Bom!\n");  
 break;  
 case 5:  
 printf("Muito bom!\n");  
 break;  
 default:  
 printf("Incorreto!\n");  
 }  
  
return 0;  
}

#include <stdio.h>  
int main(void)  
{  
 int nota=0;  
 printf("Informe a nota\n");  
 scanf("%d",&nota);  
 printf("%s\n", nota >=7?"Aprovado": "Quase...\n");  
 return 0;  
}

#include <stdio.h>  
int main(void)  
{  
 int nota=0;  
 printf("Informe uma nota\n");  
 scanf("%d",&nota);  
 nota >=7? printf("Aprovado\n"); printf("Não...\n");  
 return 0;  
}

#include <stdio.h>  
  
int main(void)  
{  
 int nota=0;  
 float s=0;  
 int i;  
  
 for(i=0;i<10;i++){  
 printf("Aluno %d\t", i+1);  
 scanf("%d", &nota);  
   
 s+=nota;  
 }  
 printf("média=%.2f\n",s);  
  
 return 0;  
  
}

14/03

#include <stdio.h>  
#include <math.h>  
int cubo(int n);  
int quadrado(int n);  
int main(void){  
int a=0;  
printf("Valor?\t");  
scanf("%d",&a);  
a = cubo(a);  
printf("cubo = %d\n", a);  
return 0;  
  
}  
  
int cubo(int n){  
int c=0;  
c=quadrado(n)\*n;  
return c;  
}  
int quadrado(int n){  
return n\*n;  
}

#include<stdio.h>  
#include<math.h>  
  
int main(void){  
 double saldo = 1000.0;  
 double t = 0.05;  
 int anos;  
 double vf = 0.0;  
 int n=0;  
 for(n=1; n <= 10; n++){  
 printf("Informe o número de anos\n");  
 scanf("%d",&anos);  
 for(n=1; n <= anos; n++){  
 vf = saldo\*pow(1+t, n);   
 printf("O saldo em %d é %.2lf\n",n,vf);  
 }  
 }  
  
return 0;  
  
}

#include<stdio.h>  
#include<math.h>  
double hipotenusa(double a, double b);  
int main(void){  
  
 double v1=0;  
 double v2=0;  
 double c=0;  
 printf("Informe os cateto\n");  
 scanf("%lf%lf", &v1, &v2);  
 c = hipotenusa(v1,v2);  
 printf("c=%.2lf\n", c);  
 return 0;  
 }  
 double hipotenusa(double a, double b){  
 double c=0;  
 c= pow(a,2)+pow(b,2);  
 c= sqrt(c);  
 return c;  
  
 }

16/03

#include <stdio.h>  
#include <stdlib.h>  
#include <time.h>  
#define FIM 7  
  
int jogar\_dados(int dados, int faces);  
enum status{ VENCEU, PERDEU, CONTINUAR};  
  
int main(void){  
enum status jogo;  
int ponto;  
int dados;  
srand(time(NULL));  
dados = jogar\_dados(2,6);  
printf("Dados = %d\n", dados);   
switch(dados){  
 case 7:  
 case 11:  
 jogo = VENCEU;  
 break;  
 case 2:  
 case 3:  
 case 12:  
 jogo = PERDEU;  
 break;  
 default:  
 ponto=dados;  
 jogo = CONTINUAR;   
 }  
  
printf("Vamos apostar amigos!\n");  
while(jogo == CONTINUAR){  
 printf("Seu ponto é %d\n", dados);  
 getchar();  
 dados = jogar\_dados(2,6);  
 printf("Você tirou %d\n", dados);  
 if(dados == ponto){  
 jogo=VENCEU;  
 } else if(dados == FIM){  
 jogo=PERDEU;  
 }  
}  
  
if(jogo == VENCEU){  
 printf("Você ganhou!\n");  
}else{  
 printf("Tente novamente!\n");   
}  
return 0;  
}  
int jogar\_dados(int dados, int faces){  
int i;  
int total=0;  
int aux=0;  
 for(i=0; i<dados; i++){  
 aux = 1 + rand()%faces;  
 total += aux;  
 }  
return total;  
  
}

21/03

#include <stdio.h>  
  
int expo (int base, int expoente);  
int main(void){  
 int basex=0;  
 int expoentey=0;  
 int valor=0;  
 printf ("Digite o valor da base:\n");  
 scanf("%d", &basex);  
 printf ("Digite o valor do exponete:\n");  
 scanf("%d", &expoentey);  
 valor = expo(basex,expoentey);  
 printf ("%d elevado ao expoente %d é igual á: %d\n", basex, expoentey,valor);  
   
   
 return 0;  
 }  
int expo(int base, int expoente){  
if(expoente == 0){  
 return 1;  
 }  
 return base \* expo(base,expoente-1);   
  
  
}

#include <stdio.h>  
  
int main(void){  
 int n=0;  
 int t=0;  
 int i=0;  
 printf ("Digite o número\n");  
 scanf("%d", &n);  
 for (i=0; i<=n; i++){  
 t= t + i;  
 }  
 printf("O resultado é de: %d\n", t);  
 return 0;  
 }

#include <stdio.h>  
  
long int fatorial(long int n);  
  
 int main(void){  
 long int N=0;  
 long int fatoN=0;  
 long int P=0;  
 long int fatoP=0;  
 long int nep=0;  
 long int fatonep=0;  
 long int S=0;  
 printf("O valor de N .\n");  
 scanf("%li", &N);  
 printf("O valor de P .\n");  
 scanf("%li", &P);  
 fatoN=fatorial(N);  
 fatoP=fatorial(P);  
 nep = N-P;  
 fatonep = fatorial(nep);  
 S = fatoN/(fatoP\*fatonep);  
 printf("Valor de S = %li\n",S);  
 return 0;  
}  
long int fatorial(long int n){  
 if(n==1){  
 return 1;  
 }  
 return n\*fatorial(n-1);  
}

#include <stdio.h>  
  
int fibonacci(int n);  
  
int main(void){  
 int x=0;  
 int y=0;  
 printf("Informe um inteiro\n");  
 scanf("%d", &x);  
 y = fibonacci(x);  
 printf("Fibonacci(%d)=%d\n",x,y);  
 return 0;  
}  
int fibonacci(int n){  
 if(n<=1){  
 return n;  
 }  
 return fibonacci(n-1)+fibonacci(n-2);  
}

#include <stdio.h>  
  
long int fatorial(long int n);  
  
 int main(void){  
 long int x=0;  
 long int y=0;  
 printf("Número inteiro por favor.\n");  
 scanf("%li", &x);  
 y=fatorial(x);  
 printf("%li != %li\n",x,y);  
 return 0;  
}  
long int fatorial(long int n){  
 if(n==1){  
 return 1;  
 }  
 return n\*fatorial(n-1);  
}

23/03

#include<stdio.h>  
#include<stdlib.h>  
#include<time.h>  
#define TAM 1000  
  
int main(void){  
 int vetor[TAM] = {0};  
 int i=0;  
 int valor = 0;  
 srand(time(NULL));  
 for(i=0; i < TAM; i++){  
 valor = rand()%10;  
 ++vetor[valor];  
 }  
 for(i=0; i < 10; i++){  
 printf("%8d\t%8d\n",i,vetor[i]);  
 }  
  
return 0;  
}

#include <stdio.h>  
int contador(char v[]);  
  
int main(void)  
{  
 char nome[0]="LPG0001 2017/1";  
 int i; int TAM=0;  
 printf("nome=%s\n",nome);  
 TAM=contador(nome);  
 printf("Tamanho=%d\n",TAM);  
 for(i=0;i<TAM/i++){  
 printf("nome[%d]=%c\n",i,nome[i]);  
 }  
 return 0;  
}  
  
int contador(char v[]){  
 int i;  
 for (i=0;v[i]!='\0';i++);  
 return i;  
  
  
}

28/03

#include <stdio.h>  
#include "notas.h"  
  
int main (void){  
 int mat[alunos][exames]={{0}};  
 int i,j;  
 int nmin = 0;  
 int nmax = 0;  
 float nmedia;  
 printf("informe os valores\n");  
 for (i=0;i<alunos;i++){  
 for (j=0;j<exames;j++){  
 printf("nota para posicao\t[%d][%d]\n",i,j);  
 scanf("%d",&mat[i][j]);  
 }  
}  
  
 nmin = minimo(mat);  
 nmax = maximo(mat);  
 printf("minimo=%d e max=%d\n",nmin,nmax);  
 for(i=0;i<alunos;i++){  
 nmedia = media(mat[i]);  
 printf("media de %d = %.2lf\n",i,nmedia);  
 }  
return 0;  
}

#include <stdio.h>  
#include "notas.h"  
  
int minimo(int m[][exames]){  
 int i,j,ret=100;  
for(i=0;i<alunos;i++){  
 for(j=0;j<exames;j++){  
 if(m[i][j]<ret){  
 ret=m[i][j];  
 }  
 }  
}  
return ret;  
}  
int maximo(int m[][exames]){  
 int i,j,ret=0;  
 for(i=0;i<alunos;i++){  
 for(j=0;j<exames;j++){  
 if(m[i][j]>ret){  
 ret=m[i][j];  
 }  
 }  
 }  
return ret;  
}  
  
float media(int v[]){  
 int i;  
 float ret=0;  
 for(i=0;i<exames;i++){  
 ret+=v[i];  
 }  
return ret/exames;  
}

#ifndef notas\_h  
#define notas\_h  
#define alunos 10  
#define exames 3  
  
int minimo (int m[][exames]);  
int maximo (int m[][exames]);  
float media(int v[]);  
#endif

30/03

#include<stdio.h>  
  
int main(void){  
  
int a=100;  
int b=10;  
int \*aP=NULL;  
pA = &;  
\*pA=500;  
int \*pB = NULL;  
pB=&b;  
  
}

#include <stdio.h>  
int main (void){  
 int i=0;  
 int limite=100;  
 LPG: printf("i=%d\n",i);  
 i++;  
 if (i<limite){  
 printf("goto\n");  
 goto LPG;  
 }  
 printf("fim\n");  
 return 0;  
  
  
}

#include <stdio.h>  
  
void varlocal(int x);//escopo protótipo  
void varstaticlocal(int x);//funcao  
void varglobal(int );  
  
int x;//global o arquivo  
  
int main(void){  
 int x=100;  
 printf("x=%d\n",x);  
 {//escopo bloco  
 int x=3;  
 int y=10;  
 {  
 int x=5;  
 printf("x=%d\n",x);  
 x+=y;  
 }  
 printf("x=%d\n",x);  
 x+=100;  
 }  
 varlocal(x);  
 varstaticlocal(x);  
 varglobal(x);  
  
 varlocal(x);  
 varstaticlocal(x);  
 varglobal(x);  
 printf("x=%d\n",x);  
 return 0;  
}  
  
void varlocal(int y){  
 int x=10;  
 x+=y;  
 printf("x=%d\n",x);  
}  
  
void varstaticlocal(int y){  
 static int x=10;  
 x+=y;  
 printf("x=%d\n",x);  
}  
  
void varglobal(int y){  
 x=10;  
 y=y+10;  
 x+=y;  
 printf("x=%d\n",x);  
}

04/04+

#include <stdio.h>  
  
int cubo1(int n);  
int cubo2(int \*p);  
  
int main(void){  
 int x=0;  
 int y=0;  
 printf("Informe um inteiro pls\n");  
 scanf("%d", &x);  
 x=cubo1(x);  
 printf("cubo = %d\n", x);  
 printf("Informe um inteiro\n");  
 scanf("%d", &y);  
 cubo2(&y);  
 printf("cubo = %d\n", y);  
 return 0;  
}  
int cubo1(int n){  
 int r;  
 r =n\*n\*n;  
 return r;  
}  
int cubo2(int \*p){  
 \*p=(\*p) \* (\*p) \* (\*p);   
}

11/04

#include<stdio.h>  
  
int main(void){  
int vetor[10] = {5,6,0,1,3,8,9,2,4};  
int i=0;  
int \*p = NULL;  
p = vetor;  
printf("vetor\n");  
for(i=0; i<10; i++){  
printf("vetor[%d] = %d\n", i,vetor[i]);  
}  
printf("Ponteiro e Vetor\n");  
for(i=0; i<10; i++){  
printf("p[%d] = %d\n", i,p[i]);  
}  
printf("ponteiros e vetores\n");  
for(i=0; i<10; i++){  
printf("\*(p+%d) = %d\n", i, \*(p+1));  
}  
printf("Vetor mais deslocamento\n");  
for(i=0; i<10; i++){  
printf("\*(vetor+%d)=%d\n", i, \*(vetor+i));  
}  
return 0;  
}

20/04

#include <stdio.h>  
  
int crescente(int a, int b);  
int decrescente(int a, int b);  
  
void bubble\_sort(int \*v, int (\*f) (int,int));  
  
int main(void){  
 int op=0;  
 int vetor[10]={3,2,9,8,7,4,5,2,7,4};  
 int i;  
 printf("Informe 1 para crescente\n 2 para decrescente\n");  
 scanf("%d",&op);  
 printf("vetor original\n");  
 for(i=0;i<10;i++){  
 printf("%d=%d\n",i,vetor[i]);  
  
 }  
 if (op==1){  
 bubble\_sort(vetor,crescente);  
 }  
 else {  
 bubble\_sort(vetor,decrescente);  
 }  
 printf("vetor original\n");  
 for(i=0;i<10;i++){  
 printf("%d=%d\n",i,vetor[i]);  
   
 }  
  
  
 return 0;  
}  
  
int crescente(int a,int b)  
{  
 return a>b;  
}  
  
int decrescente(int a,int b)  
{  
 return a<b;  
}  
void bubble\_sort(int \*v, int (\*f)(int,int))  
{  
 int i,j,aux;  
 for(i=0;i<10;i++){  
 for(j=0;j<9;j++){  
 if((\*f)(v[j],v[j+1])){  
 aux=v[j];  
 v[j]=v[j+1];  
 v[j+1]=aux;  
 }  
 }  
 }  
}

#include <stdio.h>  
 void funcao1(int a);  
 void funcao2(int b);  
 void funcao3(int c);  
 void funcao4(int x);  
 int menu(void);  
 int main(void){  
 void(\*p[])(int) = {funcao1,funcao2,funcao3,funcao4};  
 int i;  
 i = menu();  
 (\*p[i])(i);  
 return 0;  
 }  
  
 int menu(void){  
 int op=-1;  
 printf("Informe 0 ou 1 ou 2 ou 3\n");  
 while((op<0)||(op>3)){  
 scanf("%d",&op);  
 }  
 return op;  
 }  
 void funcao1(int n){  
 printf("funcao%d\n",n);  
 }  
  
 void funcao2(int n){  
 printf("funcao%d\n",n);  
 }  
  
 void funcao3(int n){  
 printf("funcao%d\n",n);  
 }  
  
 void funcao4(int n){  
 printf("funcao%d\n",n);  
 }

#include <stdio.h>  
#include <stdlib.h>  
  
int main(void){  
 int i;  
 printf("Informe o tamanho do vetor");  
 scanf("%d",&i);  
 int \*vetor = NULL;  
 vetor = (int \*)malloc(i\*sizeof(int));  
 int j=0;  
 printf("Informa ae poha");  
 for(j=0; j < i; j++){  
 scanf("%d",&vetor[j]);  
  
 }  
 free(vetor);  
 return 0;  
}

25/04

#include <stdio.h>  
#include <stdlib.h>  
  
int main(void){  
 int \*vetor = NULL;// esse ponteiro é pra fazer um vetor dinamicamente, NULL =0  
 int i=0;  
 int tam=0;  
 printf("Informe o tamanho\n");  
 scanf("%d", &tam);  
 if (!(vetor=(int\*)calloc(tam,sizeof(int)))){ // verifica se eu coloquei um valor valido para memoria, (! --> indica ideia contrária)  
//CALLOC(reserva um espaço de memória do tamanho do que eu digitei, fora da main, e coloca tudo como NULL).  
 printf("erro ao alocar\n");  
 exit(EXIT\_FAILURE);  
 }  
 for (i=0; i<tam;i++){  
 \*(vetor+i)=i\*i;  
 }  
 printf("vetor\n");  
 for(i=0;i<tam;i++){  
 printf("%d=%d\n", i, vetor[i]);  
 }  
 free(vetor);//já usei o vetor, não quero mais, ele libera o espaço de memoria que o Calloc usou.  
 vetor=NULL;  
 return 0;  
  
}  
// Calloc x Malloc, (Calloc é mais facil pro progamador mas ele é mais lento)!

#include <stdio.h>  
#include <stdlib.h>// pra pega as funções que manipulam memória   
// exemplo = Malloc  
//e outro tipos de funções que não lembro o que ele falo.  
  
int main(void){  
 float \*vetor = NULL;// esse ponteiro é pra fazer um vetor dinamicamente, NULL =0  
 int i=0;  
 int tam=0;  
 printf("Informe o tamanho\n");  
 scanf("%d", &tam);  
 if (!(vetor=(float\*)malloc(sizeof(float)\*tam))){ // verifica se eu coloquei um valor valido para memoria, (! --> indica ideia contrária)  
//MALLOC(reserva um espaço de memória do tamanho do que eu digitei, fora da main).  
 printf("erro ao alocar\n");  
 exit(EXIT\_FAILURE);  
 }  
 for (i=0; i<tam;i++){  
 \*(vetor+i)=666;  
 }  
 printf("vetor\n");  
 for(i=0;i<tam;i++){  
 printf("%d=%.2f\n", i, vetor[i]);  
 }  
 free(vetor);//já usei o vetor, não quero mais, ele libera o espaço de memoria que o Malloc usou.  
 vetor=NULL;  
 return 0;  
  
}

#include <stdio.h>  
#include <stdlib.h>// pra pega as funções que manipulam memória   
// exemplo = Malloc  
//e outro tipos de funções que não lembro o que ele falo.  
  
int main(void){  
 int \*vetor = NULL;// esse ponteiro é pra fazer um vetor dinamicamente, NULL =0  
 int i=0;  
 int tam=0;  
 printf("Informe o tamanho\n");  
 scanf("%d", &tam);  
 if (!(vetor=(int\*)malloc(sizeof(int)\*tam))){ // verifica se eu coloquei um valor valido para memoria, (! --> indica ideia contrária)  
//MALLOC(reserva um espaço de memória do tamanho do que eu digitei, fora da main).  
 printf("erro ao alocar\n");  
 exit(EXIT\_FAILURE);  
 }  
 for (i=0; i<tam;i++){  
 \*(vetor+i)=i\*i;  
 }  
 printf("vetor\n");  
 for(i=0;i<tam;i++){  
 printf("%d=%d\n", i, vetor[i]);  
 }  
 free(vetor);  
 vetor=NULL;  
 return 0;  
  
}

#include <stdio.h>  
#include <stdlib.h>// pra pega as funções que manipulam memória   
// exemplo = Malloc  
//e outro tipos de funções que não lembro o que ele falo.  
  
int main(void){  
 int \*vetor = NULL;// esse ponteiro é pra fazer um vetor dinamicamente, NULL =0  
 int i=0;  
 int tam=0;  
 printf("Informe o tamanho\n");  
 scanf("%d", &tam);  
 if (!(vetor=(int\*)malloc(sizeof(int)\*tam))){ // verifica se eu coloquei um valor valido para memoria, (! --> indica ideia contrária)  
//MALLOC(reserva um espaço de memória do tamanho do que eu digitei, fora da main).  
 printf("erro ao alocar\n");  
 exit(EXIT\_FAILURE);  
 }  
 for (i=0; i<tam;i++){  
 \*(vetor+i)=i\*i;  
 }  
 printf("vetor\n");  
 for(i=0;i<tam;i++){  
 printf("%d=%d\n", i, vetor[i]);  
 }  
  
 int\*aux=NULL;  
 int novo=0;  
  
 printf("INFORME O NOVO TAMANHO\n");  
 scanf("%d", &novo);  
  
 aux=(int\*)realloc(vetor,novo\*sizeof(int));  
 //Função realloc vai tenta fazer um puxadinho, se ela não conseguir fazer o puxadinho ela vai fazer outro e mover tudo. vc   
 if(!aux){  
 printf("memória não alterada\n");  
  
 } else{  
 vetor=aux;  
  
 for (i=tam-1;i<novo;i++){  
 \*(vetor+i)=i\*i;  
 }  
 }  
 tam=novo;  
 for(i=0;i<novo;i++){  
 printf("%d=%d\n",i,vetor[i]);  
 }  
 free(vetor);  
 //free(aux)precisamos  
 return 0;  
  
}

**27/04**

**#include <stdio.h>  
#include <stdlib.h>  
#include <time.h>  
#include "matriz.h"  
  
int \*\*criar\_matriz(int l, int c)  
{  
 int \*\*ret=NULL;  
 int i=0;  
 // (ret recebe o endereço de um ponteiro pra ponteiro, reserve x de memoria.)  
 if(!(ret=(int\*\*)malloc(sizeof(int\*)\*l))){  
 return NULL;  
 }  
 for (i=0; i<l; i++){  
 // ret recebe um ponteiro que vai apontar para X de memória  
 ret[i]=(int\*)malloc(sizeof(int)\*c);  
 if(!ret[i]){//if(ret[i]==NULL)  
 return NULL;  
 }  
 }  
 return ret;  
}  
  
void iniciar\_matriz(int \*\*m,int l, int c){  
 int i,j;  
 srand(time(NULL));  
 for(i=0;i<l;i++){  
 for(j=0;j<c;j++){  
 m[i][j]=rand()%100;  
 }  
 }  
  
}  
  
void imprimir\_matriz(int \*\*m, int l, int c){  
 int i, j;  
 for(i=0;i<l;i++){  
 for(j=0;j<c;j++){  
 printf("[%d][%d]=%d\n", i,j,m[i][j]);  
 }  
 }  
  
}  
  
void liberar\_matriz(int \*\*m, int l, int c){  
 int i;  
 for (i=0; i<l; i++){  
 free(m[i]);  
 }  
 free(m);  
}**

**#ifndef MATRIZ\_H  
#define MATRIZ\_H  
  
int \*\*criar\_matriz(int linhas, int colunas);  
// se eu tivesse colocado int (\*\*cria\_matriz), iria ser um ponteiro de função,   
//porém sem parenteses é uma função que retorna um ponteiro  
void iniciar\_matriz(int \*\*m,int linhas, int colunas);  
void imprimir\_matriz(int \*\*m, int linhas, int colunas);  
void liberar\_matriz(int \*\*m, int linhas, int colunas);  
  
#endif**

**#include <stdio.h>  
#include "matriz.h"  
  
int main(void)  
{  
 int \*\*mat=NULL;  
 int linhas = 0;  
 int colunas = 0;  
 printf("Informe o número de linhas e colunas \n");  
 scanf("%d%d", &linhas,&colunas);  
 mat=criar\_matriz(linhas,colunas);  
 iniciar\_matriz(mat,linhas,colunas);  
 imprimir\_matriz(mat, linhas, colunas);  
 liberar\_matriz(mat,linhas,colunas);  
 return 0;  
}**

04/05

#include <stdio.h>  
  
char \*criar\_palavra(int tam){// essa função vai criar o espaço de armazenamento da palavra  
 char \*palavra;   
 palavra=(char\*)malloc(sizeof(char)\*tam); // (char\*) --> isso é pra forçar o bagulho a funciona  
 return palavra;  
   
}  
  
int tamanho\_palavra(char \*p){ // essa função retorna o tamanho da palavra, o laço conta a quantidade de quadradinhos até chega no \0  
 int i;  
 for(i=0; p[i] !='\0';i++);  
 return i;  
}  
  
char \*redimensionar\_palavra(char \*p, int novo\_tam){ // essa função faz o realloc, onde ela faz o puxadinho de memoria  
 char \*aux;  
 aux=(char\*)realloc(p,novo\_tam\*sizeof(char));  
 return aux;  
}  
  
void copiar\_palavra(char \*origem,char \*destino){ // essa função faz o destino receber a origem.  
 int i, n;  
 n = tamanho\_palavra(origem);  
 for (i=0; i < n;i++){  
 destino[i] = origem[i];  
 }  
 return destino;  
  
}

#ifndef PALAVRA\_H  
#define PALAVRA\_H  
  
char \*criar\_palavra(int tam);  
int tamanho\_palavra(char \*p);  
void liberar\_palavra(char \*p);  
char \*redimensionar\_palavra(char \*p, int novo\_tam);  
void copiar\_palavra(char \*origem, char \*destino);  
#endif

#include <stdio.h>  
#include "palavra.h"  
  
int main(void)  
{  
 int i,j;  
 char \*nome=NULL;  
 char \*alteracao=NULL;  
 printf ("Informe o tamanho do nome:\n");  
 scanf ("%d", &i);  
 nome=criar\_palavra(i);  
 printf("Informe o nome\n");  
 scanf("%s", nome);  
 j=tamanho\_palavra(nome);  
  
 j=j\*2;  
 alteracao=redimensionar\_palavra(nome,j);  
 if(!alteracao){  
 printf("não alterado\n");  
 } else {  
 nome = alteracao;  
 }  
 char vetor[]="teste";  
 alteracao = criar\_palavra(j);  
 copiar\_palavra(vetor, alteracao);  
 printf("%s\n%s\n%s\n", vetor, alteracao, nome);  
 //... liberar  
 free(nome);  
 free(alteracao);  
   
 return 0;  
}

aula dia 16/05

#define TAM\_NOME 80

typedef struct {

int dia;

int mes;

int ano;

char cidade [TAM\_NOME];

} Nascimento;

typedef struct{

int matricula;

char nome[TAM\_NOME];

Nascimento nasc;

} pessoa;

int ord\_dia\_ascendente (pessoa a, pessoa b);

int ord\_dia\_descendente (pessoa a, pessoa b);

int ord\_mes\_ascendente (pessoa a, pessoa b);

int ord\_mes\_descendente (pessoa a, pessoa b);

int ord\_ano\_ascendente (pessoa a, pessoa b);

int ord\_ano\_descendente (pessoa a, pessoa b);

int ord\_alfabetica\_nome\_ascendente (pessoa a, pessoa b);

int ord\_alfabetica\_nome\_descendente (pessoa a, pessoa b);

int ord\_idade\_ascendente (pessoa a, pessoa b);

int ord\_idade\_descendente (pessoa a, pessoa b);

void bubble\_sort(pessoa \*p, int tam, int (\*f)(pessoa, pessoa));

void insertion\_sort(pessoa \*p, int tam, int (\*f)(pessoa, pessoa));

void imprimir\_vetor\_1 (pessoa \*p, int tam);

void imprimir\_vetor\_2 (pessoa \*p, int tam);

void preencher\_vetor(pessoa \*p, int tam);

main.c

#include <stdio.h>

#include <string.h>

#include <time.h>

#include <stdlib.h>

#include "pessoa.h"

#define TAM\_VETOR 100

int main(void)

{

pessoa vetor[TAM\_VETOR];

int op\_ordem, op\_alg, ok = 0;

preencher\_vetor(vetor, TAM\_VETOR);

while(!ok)

{

printf("1 para ordenar pelo dia de nascimento (asc)\n");

printf("2 para ordenar pelo dia de nascimento (desc)\n");

printf("3 para ordenar pelo mes de nascimento (asc)\n");

printf("4 para ordenar pelo mes de nascimento (desc)\n");

printf("5 para ordenar pelo ano de nascimento (asc)\n");

printf("6 para ordenar pelo ano de nascimento (desc)\n");

printf("7 para ordenar pelo nome de nascimento (asc)\n");

printf("8 para ordenar pelo nome de nascimento (desc)\n");

printf("9 para ordenar pelo idade de nascimento (asc)\n");

printf("10 para ordenar pelo idade de nascimento (desc)\n");

}

}

a.c

#include <stdio.h>

#include <string.h>

#include <time.h>

#include <stdlib.h>

#include "pessoa.h"

int ascendente(int a, int b)

{

return a > b;

}

int descendente(int a, int b)

{

return a<b;

}

int ord\_dia\_ascendente (pessoa a, pessoa b)

{

return 0;

}

int ord\_dia\_descendente (pessoa a, pessoa b)

{

return 0;

}

int ord\_mes\_ascendente (pessoa a, pessoa b)

{

return 0;

}

int ord\_mes\_descendente (pessoa a, pessoa b)

{

return 0;

}

int ord\_ano\_ascendente (pessoa a, pessoa b)

{

return 0;

}

int ord\_ano\_descendente (pessoa a, pessoa b)

{

return 0;

}

int ord\_alfabetica\_nome\_ascendente (pessoa a, pessoa b)

{

return 0;

}

int ord\_alfabetica\_nome\_descendente (pessoa a, pessoa b)

{

return 0;

}

int ord\_idade\_ascendente (pessoa a, pessoa b)

{

return 0;

}

int ord\_idade\_descendente (pessoa a, pessoa b)

{

return 0;

}

void bubble\_sort(pessoa \*p, int tam, int (\*f)(pessoa, pessoa))

{

}

void insertion\_sort(pessoa \*p, int tam, int (\*f)(pessoa, pessoa))

{

}

void imprimir\_vetor\_1 (pessoa \*p, int tam)

{

}

void imprimir\_vetor\_2 (pessoa \*p, int tam)

{

}

void preencher\_vetor(pessoa \*p, int tam)

{

char aux\_nome[TAM\_NOME];

char\* cidade[] = {"Joinville" , "Garuva" , "Florianopolis" , "Blumenau" , "Jaragua"};

srand(time(NULL));

for(i = 0; i < tam; i++)

{

sprintf(aux\_nome, "Pessoa %d",i);

strcpy(p[i].nome , aux\_nome);

p[i].matricula = rand()%(1000-5000) + 1000; //matricula

p[i].nasc.dia = rand()%(1-30)+1; //dia de nascimento

p[i].nasc.mes = rand()%(1-12)+1; //mes de nascimento

p[i].nasc.ano = rand()%(1990-2015)+1990; //ano de nascimento

strcpy(p[i].nasc.cidade , cidades[rand()%(0-4) + 0]); //cidade de nascimento

}

}

Aula sobre matrizes

prog.c

#include <stdio.h>

#include <stdlib.h>

#include "matriz.h"

int main(void){

TMatriz \*mat = NULL;

int op, l, c;

printf("Informe 1 para bloco e 2 para distribuida\n");

scanf("%d", &op);

printf("Informe o número de linhas e colunas\n");

scanf("%d%d", &l, &c);

if(op == 1){

mat = criar\_matriz(l, c, dados\_bloco);

} else if(op == 2){

mat = criar\_matriz(l, c, dados\_distribuidos);

} else {

printf("Opção inválida\n");

exit(EXIT\_FAILURE);

}

iniciar\_matriz(mat);

imprimir\_matriz(mat);

if( op == 1){

liberar\_matriz(mat, liberar\_dados\_bloco);

} else {

liberar\_matriz(mat, liberar\_dados\_distribuidos);

}

return 0;

}

matriz.c

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include "matriz.h"

void liberar\_matriz(TMatriz \*mat, void (\*f)(int \*\*, int))

{

if (!mat)

return;

(\*f)(mat->m, mat->linhas);

free(mat);

}

void liberar\_dados\_bloco(int \*\*m, int l)

{

if (!m)

return;

free(m[0]);

free(m);

}

void liberar\_dados\_distribuidos(int \*\*m, int l)

{

int i;

if (!m)

return;

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

if (!m[i])

return;

free(m[i]);

}

free(m);

}

void iniciar\_matriz(TMatriz \*mat)

{

int i;

int j;

if (!mat)

return;

srand(time(NULL));

for (i = 0; i < mat->linhas; i++) {

for (j = 0; j < mat->colunas; j++) {

mat->m[i][j] = rand() % 100;

}

}

}

void imprimir\_matriz(TMatriz \*mat)

{

int i;

int j;

if (!mat)

return;

for (i = 0; i < mat->linhas; i++) {

for (j = 0; j < mat->colunas; j++) {

printf("mat[%d][%d] = %d\n", i, j, mat->m[i][j]);

}

}

}

int \*\*dados\_bloco(int l, int c){

int i;

int \*aux = NULL;

int \*\*ret = NULL;

if(!(ret = (int \*\*) malloc(sizeof(int \*)\*l)))

return NULL;

if(!(aux = (int \*) malloc(sizeof(int)\* l \* c))){

free(ret);

return NULL;

}

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

ret[i] = &aux[i \* c];

}

return ret;

}

int \*\*dados\_distribuidos(int l, int c){

int i,j;

int \*\*ret = NULL;

if(!(ret = (int \*\*) malloc(sizeof(int \*) \* l)))

return NULL;

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

if(!(ret[i] = (int \*)malloc(sizeof(int) \* c))){

for(j = 0; j< i; j++)

free(ret[j]);

return(ret);

return NULL;

}

}

return ret;

}

TMatriz \*criar\_matriz(int linhas, int colunas, int \*\*(\*f)(int, int)){

TMatriz \*ret = NULL;

if(!(ret = (TMatriz \*) malloc(sizeof(TMatriz))))

return NULL;

ret->linhas = linhas;

(\*ret).colunas = colunas;

ret->m = (int \*\*)(\*f)(linhas, colunas);

return ret;

}

matriz.h

#ifndef MATRIZ\_H

#define MATRIZ\_H

typedef struct{

int linhas;

int colunas;

int \*\*m;

} TMatriz;

/\* funcoes para manipular a matriz \*/

TMatriz \*criar\_matriz(int linhas, int colunas, int \*\*(\*f)(int, int));

void iniciar\_matriz(TMatriz \* mat);

void imprimir\_matriz(TMatriz \*mat);

void liberar\_matriz(TMatriz \*mat, void (\*f)(int \*\*, int));

/\* funcoes para manipular dados \*/

int \*\*dados\_bloco(int l, int c);

int \*\*dados\_distribuidos(int l, int c);

void liberar\_dados\_bloco(int \*\*m, int l);

void liberar\_dados\_distribuidos(int \*\*m, int l);

#endif

oi.c

#include <stdio.h>

typedef union {

int x;

char v[2];

double y;

} Numero;

int main(void)

{

Numero n;

n.x = 10;

printf("x = %d \n y = %lf\n", n.x, n.y);

n.y = 28.5;

n.x = 5;

printf("x = %d \n y = %lf\n", n.x, n.y);

return 0;

}

***25/05***

***// gabriel lima, guilherme utiama, gustavo daniel;***

***#include <time.h>***

***#include <stdio.h>***

***#include <stdlib.h>***

***#include <string.h>***

***#include "pessoa.h"***

***#define TAM\_VETOR 100***

***int main(void)***

***{***

***Pessoa vetor[TAM\_VETOR];***

***int op\_ordem, op\_alg, ok = 0;***

***preencher\_vetor(vetor,TAM\_VETOR);***

***while(!ok){***

***printf("1 para ordenar pelo dia de nascimento (asc)\n");***

***printf("2 para ordenar pelo dia de nascimento (desc)\n");***

***printf("3 para ordenar pelo mês de nascimento (asc)\n");***

***printf("4 para ordenar pelo mês de nascimento (desc)\n");***

***printf("5 para ordenar pelo ano de nascimento (asc)\n");***

***printf("6 para ordenar pelo ano de nascimento (desc)\n");***

***printf("7 para ordenar pelo nome (asc)\n");***

***printf("8 para ordenar pelo nome (desc)\n");***

***printf("9 para ordenar pela idade (asc)\n");***

***printf("10 para ordenar pela idade (desc)\n");***

***scanf("%d", &op\_ordem);***

***printf("1 para utilizar o Bubble Sort\n");***

***printf("2 para utilizar o Insertion Sort\n");***

***scanf("%d", &op\_alg);***

***if(op\_ordem < 1 || op\_ordem > 10){***

***printf("A opção de ordenação deve ser um valor entre 1 e 10\n\n");***

***ok = 0;***

***} else if(op\_alg < 1 || op\_alg > 2 ){***

***printf("A opção do método de ordenação deve ser 1 ou 2\n\n");***

***ok = 0;***

***} else{***

***ok = 1;***

***}***

***}***

***int (\*p[])(Pessoa, Pessoa) = {***

***ord\_dia\_ascendente, ord\_dia\_descendente, ord\_mes\_ascendente,***

***ord\_mes\_descendente, ord\_ano\_ascendente,***

***ord\_ano\_descendente, ord\_alfabetica\_nome\_ascendente,***

***ord\_alfabetica\_nome\_descendente, ord\_idade\_ascendente, ord\_idade\_descendente};***

***void (\*v[])(Pessoa \*p, int, int (\*f)(Pessoa, Pessoa)) = {bubble\_sort, insertion\_sort};***

***v[op\_alg](vetor,TAM\_VETOR,p[op\_ordem]); //v[j] mostra se é insertion sort ou bubble sort, quanto o p[i] é qual tipo de ordenacao***

***imprimir\_vetor\_1(vetor, TAM\_VETOR);***

***return 0;***

***}***

***30/05***

***a.c***

***#include <stdio.h>***

***int main(int argc, char const \* argv[])***

***{***

***int i; //int c = ato1(...);***

***for(i = 1; i < argc; i++)***

***{***

***printf("%s\n",argv[i]);***

***}***

***return 0;***

***}***

***b.c***

***#include <stdio.h>***

***#define TAM 80***

***int main(int argc, char const\*argv[])***

***{***

***int conta;***

***double saldo;***

***char nome[TAM];***

***FILE \*arquivo = 0;***

***if(argc!=2)***

***{***

***printf("execute %s arquivo\n",argv[0]);***

***return 1;***

***}***

***arquivo = fopen(argv[1],"w");***

***if(arquivo == 0){***

***printf("Erro ao criar arquivo\n");***

***return 1;***

***}***

***printf("informe conta, nome e saldo\n");***

***scanf("%d %s %lf", &conta, nome, &saldo);***

***while(!feof(stdin)){***

***fprintf(arquivo,"%d %s %.2lf\n", conta, nome, saldo);***

***scanf("%d %s %lf\n",&conta, nome, &saldo);***

***}***

***fclose(arquivo);***

***return 0;***

***}***

***c.c***

***#include <stdio.h>***

***#define TAM 80***

***int main(int argc, char const\*argv[])***

***{***

***int conta;***

***double saldo;***

***char nome[TAM];***

***FILE \*arquivo = 0;***

***if(argc!=2)***

***{***

***printf("execute %s arquivo\n",argv[0]);***

***return 1;***

***}***

***arquivo = fopen(argv[1],"r");***

***if(arquivo == 0){***

***printf("Erro ao ler arquivo\n");***

***return 1;***

***}***

***printf("%-10s %-20s %10s\n", "conta", "nome", "saldo");***

***fscanf(arquivo,"%d %s %lf", &conta, nome, &saldo);***

***while(!feof(arquivo))***

***{***

***fprintf(stdout, "%-10d %-20s %10.2lf\n", conta, nome, saldo);***

***fscanf(arquivo, "%d %s %lf", &conta, nome, &saldo);***

***}***

***fclose(arquivo);***

***return 0;***

***}***

***1/06***

***arquivo.h***

***#ifndef ARQUIVO\_H***

***#define ARQUIVO\_H***

***int menu(void);***

***int saldo\_zero(double );***

***int credor(double );***

***int devedor(double );***

***void relatorio(FILE\*, int (\*f)(double));***

***#endif***

***principal.c***

***#include <stdio.h>***

***#include "arquivo.h"***

***int main(int argc, char const \*argv[])***

***{***

***int (\*f[])(double) = {saldo\_zero, credor, devedor};***

***FILE \*arquivo = 0;***

***int op=0;***

***if(argc != 2)***

***{***

***printf("use %s <arquivo> \n", argv [0]);***

***return 1;***

***}***

***if(!(arquivo = fopen(argv[1],"r")))***

***{***

***printf("arquivo nao encontrado\n");***

***return 1;***

***}***

***while((op = menu()) != EOF)***

***{***

***relatorio(arquivo,f[op]);***

***}***

***fclose(arquivo);***

***return 0;***

***}***

***arquivo.c***

***#include <stdio.h>***

***#include "arquivo.h"***

***int menu(void)***

***{***

***int ret=-2;***

***while((ret < -1 || ret > 2 ) )***

***{***

***printf("informe 0 para saldo zero\n informe 1 para credores\n ou 2 para devedores\n -1 para sair\n");***

***scanf("%d", &ret);***

***}***

***return ret;***

***}***

***int saldo\_zero(double d)***

***{***

***return d == 0;***

***}***

***int credor (double d)***

***{***

***return d > 0;***

***}***

***int devedor(double d)***

***{***

***return d<0;***

***}***

***void relatorio(FILE \*arquivo, int(\*f)(double))***

***{***

***int conta;***

***double saldo;***

***char nome[80];***

***rewind(arquivo);***

***fscanf(arquivo,"%d %s %lf", &conta, nome, &saldo);***

***while(!feof(arquivo))***

***{***

***if((\*f)(saldo))***

***{***

***printf("%d %30s %10.2lf\n", conta, nome, saldo);***

***}***

***fscanf(arquivo, "%d %s %lf\n", &conta, nome, &saldo);***

***}***

***}***

***lpg.c***

***#include <stdio.h>***

***struct pessoa{***

***int conta;***

***char nome[30];***

***char sobrenome[30];***

***double saldo;***

***};***

***int main(int argc, char const \*argv[])***

***{***

***FILE \* arquivo = NULL;***

***struct pessoa p = {0, "", "", 0.0};***

***if(argc!=2)***

***{***

***printf("use: %s <arquivo>\n",argv[0]);***

***return 1;***

***}***

***if(!(arquivo = fopen(argv[1],"wb")))***

***{***

***printf("erro ao criar arquivo\n");***

***return 1;***

***}***

***int i;***

***for(i = 0; i < 100; i++)***

***{***

***fwrite(&p, sizeof(struct pessoa), 1, arquivo);***

***}***

***fclose(arquivo);***

***return 0;***

***}***

***06/06***

***#include <stdio.h>***

***struct pessoa{***

***int conta;***

***char nome[30];***

***char sobrenome[30];***

***double saldo;***

***};***

***int main(int argc, char const \*argv[])***

***{***

***FILE \* arquivo = NULL;***

***struct pessoa p = {0, "", "", 0.0};***

***if(argc!=2)***

***{***

***printf("use: %s <arquivo.dat>\n",argv[0]);***

***return 1;***

***}***

***if(!(arquivo = fopen(argv[1],"rb+")))***

***{***

***printf("arquivo nao encontrado\n");***

***return 1;***

***}***

***printf("informe a conta entre 1 e 100\n 0 para terminar\t");***

***scanf("%d", &p.conta);***

***while(p.conta != 0)***

***{***

***printf("informe nome, sobrenome e saldo\n");***

***scanf("%s %s %lf", p.nome, p.sobrenome, &p.saldo);***

***fseek(arquivo, (p.conta-1)\*sizeof(struct pessoa), SEEK.SET);***

***fwrite(&p, sizeof(struct pessoa),1, arquivo);***

***printf("[1,100] ou 0\n");***

***scanf("%d", &p.conta);***

***}***

***fclose(arquivo);***

***return 0;***

***}***

***+***

***leitura.c***

***#include <stdio.h>***

***struct pessoa{***

***int conta;***

***char nome[30];***

***char sobrenome[30];***

***double saldo;***

***};***

***int main(int argc, char const \*argv[])***

***{***

***FILE \* arquivo = NULL;***

***struct pessoa p = {0, "", "", 0.0};***

***if(argc!=2)***

***{***

***printf("use: %s <arquivo.dat>\n",argv[0]);***

***return 1;***

***}***

***if(!(arquivo = fopen(argv[1],"rb+")))***

***{***

***printf("arquivo nao encontrado\n");***

***return 1;***

***}***

***rewind(arquivo);***

***while(!feof(arquivo))***

***{***

***fread(&p, sizeof(struct pessoa),1,arquivo);***

***if(p.conta !=0)***

***{***

***printf("Conta = %d, Nome = %s, Sobrenome = %s e Saldo = %.2lf\n ",p.conta,p.nome,p.sobrenome,p.saldo);***

***}***

***}***

***fclose(arquivo);***

***return 0;***

***}***

***08/06***

***banco.h***

***#ifndef BANCO\_H***

***#define BANCO\_H***

***typedef struct***

***{***

***int conta;***

***char nome[30];***

***char sobrenome[30];***

***double saldo;***

***}Registro;***

***int opcoes(void);***

***void novo\_registro(FILE \* arquivo);***

***void atualizar\_registro(FILE \* arquivo);***

***void apagar\_registro(FILE \* arquivo);***

***void relatorio(FILE \* arquivo);***

***#endif***

***principal.c***

***#include <stdio.h>***

***#include "banco.h"***

***int main(int argc, char const \*argv[])***

***{***

***FILE \*arquivo = NULL;***

***if(argc != 2)***

***{***

***printf("Informe: %s [nome\_do\_arquivo.dat]\n", );***

***return 1;***

***}***

***if(!(arquivo=fopen(argv[1],"rb+")))***

***{***

***printf("Arquivo não encontrado\n");***

***return 1;***

***}***

***void (\*f[])(FILE \*) = {novo\_registro, atualizar\_registro, apagar\_registro, relatorio};***

***while ((op = opcoes()) != EOF)***

***{***

***(\*f[op])(arquivo);***

***}***

***fclose(arquivo);***

***return 0;***

***}***

***banco.c***

***#include <stdio.h>***

***#include "banco.h"***

***int opcoes(void)***

***{***

***int r = -2;***

***while(r < -1 || r > 3)***

***{***

***printf("Informe: \n0 para cadastrar\n1 para atualizar\n2 para apagar\n3 para registro\n-1 para sair\n");***

***scanf("%d",%r);***

***}***

***return r;***

***}***

***void novo\_registro(FILE \*arquivo)***

***{***

***Registro r = {0,"","",0.0};***

***int conta;***

***printf("Informe a conta entre 1 e 100\t");***

***scanf("%d", &conta);***

***fseek(arquivo, sizeof(Registro)\*(conta - 1), SEEK\_SET);***

***fread(&r, sizeof(Registro), 1, arquivo);***

***if (r.conta != 0)***

***{***

***printf("Registro ja cadastrado\n");***

***return;***

***}***

***printf("Informe nome, sobrenome e saldo\t");***

***scanf("%s%s%lf", r.nome, r.sobrenome, &r.saldo);***

***r.conta = conta;***

***fseek(arquivo,sizeof(Registro)\*(conta-1), SEEK\_SET);***

***fwrite(&r, sizeof(Registro),1,arquivo);***

***printf("Cadastro concluído com sucesso!\n");***

***}***

***void apagar\_registro(FILE \*arquivo)***

***{***

***Registro r = {0,"","",0.0};***

***Registro l = {0,"","",0.0};***

***int conta;***

***printf("Informe a conta entre 1 e 100\t");***

***scanf("%d", &conta);***

***fseek(arquivo, sizeof(Registro)\*(conta - 1), SEEK\_SET);***

***fread(&r, sizeof(Registro), 1, arquivo);***

***if(r.conta == 0)***

***{***

***printf("Conta inexistente!\n");***

***return;***

***}***

***fseek(arquivo,sizeof(Registro)\*(conta-1), SEEK\_SET);***

***//fseek(arquivo,sizeof(Registro)\*-1, SEEK\_CUR);***

***fwrite(&r, sizeof(Registro),1,arquivo);***

***printf("Cadastro excluído com sucesso!\n");***

***}***

***void atualizar\_regisro(FILE \*arquivo)***

***{***

***Registro r = {0,"","",0.0};***

***int conta;***

***printf("Informe a conta entre 1 e 100\t");***

***scanf("%d", &conta);***

***fseek(arquivo, sizeof(Registro)\*(conta - 1), SEEK\_SET);***

***fread(&r, sizeof(Registro), 1, arquivo);***

***if(r.conta == 0)***

***{***

***printf("Conta inexistente!\n");***

***return;***

***}***

***printf("Olá %s %s, seu saldo atual é:%0.2lf\n",r.nome,r.sobrenome, r.saldo );***

***printf("Informe o valor para deposito(+) ou saque(-)\t");***

***scanf("%lf",&valor);***

***fseek(arquivo, sizeof(Registro)\*(conta-1),SEEK\_SET);***

***fwrite(&r, sizeof(Registro),1,arquivo);***

***printf("Registro atualizado com sucesso!\n");***

***}***

***void relatorio(FILE \*arquivo)***

***{***

***FILE \*saida = NULL;***

***Registro r = {0, "", "", 0.0};***

***if(!(saida = fopen("saida.txt", "w")))***

***{***

***printf("Erro ao criar arquivo\n");***

***return;***

***}***

***rewind(arquivo);***

***while(!feof(arquivo))***

***{***

***fread(&r, sizeof(Registro),1,arquivo);***

***if(r.conta == 0) continue;***

***fprintf(saida, "%s %s %d %.2lf\n",r.nome, r.sobrenome, r.conta, r.saldo);***

***}***

***} fclose(saida);***

***programa.c***

***#include <stdio.h>***

***struct pessoa{***

***int conta;***

***char nome[30];***

***char sobrenome[30];***

***double saldo;***

***};***

***int main(int argc, char const \*argv[])***

***{***

***FILE \* arquivo = NULL;***

***struct pessoa p = {0, "", "", 0.0};***

***if(argc!=2)***

***{***

***printf("use: %s <arquivo.dat>\n",argv[0]);***

***return 1;***

***}***

***if(!(arquivo = fopen(argv[1],"rb+")))***

***{***

***printf("arquivo nao encontrado\n");***

***return 1;***

***}***

***printf("informe a conta entre 1 e 100\n 0 para terminar\t");***

***scanf("%d", &p.conta);***

***while(p.conta != 0)***

***{***

***printf("informe nome, sobrenome e saldo\n");***

***scanf("%s %s %lf", p.nome, p.sobrenome, &p.saldo);***

***fseek(arquivo, (p.conta-1)\*sizeof(struct pessoa), SEEK.SET);***

***fwrite(&p, sizeof(struct pessoa),1, arquivo);***

***printf("[1,100] ou 0\n");***

***scanf("%d", &p.conta);***

***}***

***fclose(arquivo);***

***return 0;***

***}***

***ultima aula::::***

***typedef struct{***

***char \*nome;***

***char \*sobrenome;***

***int matricula;***

***} Aluno;***

***Aluno \*cadastra\_aluno (void)***

***{***

***Aluno \*ret = NULL;***

***ret = (Aluno\* ) malloc(sizeof(Aluno));***

***if(!ret)***

***return NULL;***

***ret -> nome = (char\*) malloc(sizeof(char) \*tam\_nome);***

***if(!ret -> nome)***

***{***

***free(ret);***

***return NULL;***

***ret -> matricula = 0;***

***}***

***if(!ret -> sobrenome)***

***{***

***free (ret ->nome);***

***free(ret);***

***return NULL;***

***}***

***return ret;***