#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include "dicionario.h"

int main(void)

{

inicializar\_dicionario();

char \*\*dicionario;

dicionario = pegar\_dicionario();

char palavra[10000];

printf("digite uma palavra\n");

scanf("%s", palavra);

int tam\_dic = tamanho\_dicionario();

int tam=0;

tam = strlen(palavra)-1; //mede o tamanho da string;

char primeiraletra, ultimaletra;

primeiraletra = palavra[0];

ultimaletra = palavra[tam];

int i=0;

int contador=0;

char palavrasselecionadas[1000][1000];

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

{

if( (primeiraletra == dicionario[i][0])){

if( ultimaletra == dicionario[i][strlen(dicionario[i]) - 1]){

printf("A letra inicial e final da %s é igual.\n", dicionario[i]);

strcpy(palavrasselecionadas[contador], dicionario[i]); //copia todos os char de uma string para outra;

contador++;

}

}

if(i==tam\_dic -1 && contador == 0)

{

printf(" \n");

}

}

int m=0,y=0,cont=0;

char palavrasselecionadas2[10][100];

for(y=0;y<contador;y++)

{

//printf("%s\n",palavrasselecionadas[y]);

int n=0;

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

{

if (palavrasselecionadas[y][n] == palavra[m])

{

//printf("%c do dicionario = %c da palavra\n", palavrasselecionadas[y][n],palavra[m]);

n++;

if(n==strlen(palavrasselecionadas[y]))

{

printf("A palavra %s foi selecionada\n", palavrasselecionadas[y]);

strcpy(palavrasselecionadas2[cont], palavrasselecionadas[y]);

//printf("%s\n",palavrasselecionadas2[cont]);

cont++;

}

else if(m==tam){

printf(" \n");

}

}

}

}

if(cont==1)

{

printf("%s\n",palavrasselecionadas2[0] );

}

else if(cont>1)

{

int xp=0;

xp = strcmp(palavrasselecionadas2[cont-1],palavrasselecionadas2[cont-2]);

if(xp<0)

{

printf("%s\n",palavrasselecionadas2[cont-1] );

}

else{

printf("%s\n",palavrasselecionadas2[cont-2] );

}

}

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);