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;

int 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");

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 opcao de ordenacao deve ser um valor entre 1 a 10\n\n");

ok = 0;

}

else

{

ok = 1;

}

}

switch(op\_alg)

{

case 1:

switch(op\_ordem)

{

case 1:

bubble\_sort(vetor, TAM\_VETOR,ord\_dia\_ascendente);

break;

}

case 2:

bubble\_sort(vetor, TAM\_VETOR,ord\_dia\_ascendente);

break;

}

}

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

return 0;

}

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 ascendente(a.nasc.dia,b.nasc.dia);

}

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

{

return descendente(a.nasc.dia,b.nasc.dia);

}

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

{

return ascendente(a.nasc.mes,b.nasc.mes);

}

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

{

return descendente(a.nasc.mes,b.nasc.mes);

}

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

{

return ascendente(a.nasc.ano,b.nasc.ano);

}

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

{

return descendente(a.nasc.ano,b.nasc.ano);

}

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))

{

int i,j;

pessoa aux;

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

{

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

{

if((\*f)(p[j], p[j + i]))

{

aux = p [j];

p[j] = p [i+1];

p[i+1] = aux;

}

}

}

}

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

{

}

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

{

int i=0;

printf("teste");

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

{

printf(" matricula = %d\n nascimento : %d/%d/%d\n cidade : %s ", p[i].matricula, p[i].nasc.dia, p[i].nasc.mes, p[i].nasc.ano, p[i].nasc.cidade);

}

}

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));

int i = 0;

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 , cidade[rand()%(0-4) + 0]); //cidade de nascimento

}

}

pessoa.h

#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);