Universidade Federal Fluminense

Matéria Estrutura de Dados

Professor Dalessandro Soares

Alunos: Alessandro Sampaio e Tatiane Sousa

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

typedef struct generico{

void \*info;

char tipo;

}Generico;

typedef struct aluno{

char nome[81];

int mat;

}Aluno;

typedef struct professor{

char nome[81];

int mat;

char titul[30];

int dep;

}Professor;

Generico \*Inicializar (Generico \*vet, int n){

int i;

vet =(Generico\*) malloc(n\*sizeof(Generico));

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

vet[i].info = NULL;

vet[i].tipo = 'v';

}

return vet;

}

Generico\* InserirProf (Generico \*vet, int n, int i){

int dep, mat;

char info;

char nome[81];

char titul[21];

if( vet[i].tipo == 'a'){

free(vet[i].info);

}

printf("Digite o Nome: \n");

scanf(" %80[^\n]", nome);

printf("Digite a matricula: \n");

scanf("%d", &mat);

printf("Digite a titulacao: \n");

scanf(" %20[^\n]", titul);

printf("Digite a quantidade de dependentes: \n");

scanf("%d", &dep);

Professor \*no = (Professor\*)malloc(sizeof(Professor));

no-> mat = mat;

strcpy(no->nome,nome);

strcpy(no->titul,titul);

no-> dep = dep;

vet[i].info = no;

vet[i].tipo = 'p';

return vet;

}

Generico\* InserirAluno (Generico \*vet, int n, int i){

int mat;

char nome[81];

if( vet[i].tipo == 'p'){

free(vet[i].info);

}

Aluno \*no = (Aluno\*)malloc(sizeof(Aluno));

printf("Digite o Nome: \n");

scanf(" %80[^\n]", nome);

printf("Digite a matricula: \n")

scanf("%d", &mat);

no-> mat = mat;

strcpy(no->nome,nome);

vet[i].info = no;

vet[i].tipo='a';

return vet;

}

void ImprimirAluno(Generico \*vet, int n){

int i;

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

if(vet[i].tipo == 'a'){

Aluno \*a = (Aluno\*)vet[i].info;

printf(" Aluno: %s \n", a->nome);

printf(" Matricula: %d \n", a->mat);

}

}

}

void ImprimirProf(Generico \*vet, int n){

int i;

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

if(vet[i].tipo == 'p'){

Pofessor \*p =(Professor\*) vet[i].info;

printf(" prof: %s \n", p->nome);

printf(" Matricula: %d \n", p->mat);

printf(" Titulacao: %s \n", p->titul);

printf(" Dependentes: %d \n", p->dep);

}

}

}

void ImprimirTodos(Generico \*vet, int n){

int i;

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

if(vet[i].tipo != 'v'){

if(vet[i].tipo == 'a'){

Aluno \*a = (Aluno\*)vet[i].info;

printf(" aluno: %s \n", a->nome);

printf(" Matricula: %d \n", a->mat);

}

else{

Professor \*p =(Professor\*) vet[i].info;

printf(" prof: %s \n", p->nome);

printf(" Matricula: %d \n", p->mat);

printf(" Titulacao: %s \n", p->titul);

printf(" Dependentes: %d \n", p->dep);

}

}

}

}

int QuantAlunos(Generico \*vet, int n){

int cont=0, i;

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

if(vet[i].tipo == 'a' || vet[i].tipo == 'A')

cont++;

}

return cont;

}

int QuantProf(Generico \*vet, int n){

int cont=0, i;

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

if(vet[i].tipo == 'p')

cont++;

}

return cont;

}

Generico\* ExcluirProf (Generico \*vet, int n, int mat){

int i;

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

Professor \*p =(Professor\*)vet[i].info;

if(vet[i].tipo == 'p'&& p->mat == mat){

free(p);

vet[i].info = NULL;

vet[i].tipo = 'v';

}

return vet;

}

}

int main(void){

Generico \*vet;

int i=0, n, conta=0, contp=0, x, x1;

char nome[81];

int mat, dep;

char titul[20];

printf("Digite o tamanho do vetor \n");

scanf("%d", &n);

while(x!=6){

system("cls");

printf("[ - - Menu de opcoes - - ]\n\n");

printf("1 - Inicializar vetor \n");

printf("2 - Inserir Elemento \n");

printf("3 - Imprimir \n");

printf("4 - Quantidade de alunos e profs \n");

printf("5 - Excluir professor \n");

printf("6 - Sair");

printf("\n");

scanf("%d", &x);

system("cls");

switch (x){

case 1:{

vet=Inicializar(vet, n);

printf("Seu vetor foi inicializado com sucesso \n");

system("pause");

break;

}

case 2:{

printf("Digite:\n");

printf("1 - Inserir Aluno\n");

printf("2 - Inserir Professor \n");

scanf("%d", &x1);

switch (x1){

case 1:{

printf("digite a posicao que deseja inserir o aluno: \n");

scanf("%d", &i);

vet=InserirAluno(vet, n, i);

printf("Aluno inserido com sucesso \n");

break;

}

case 2:{

printf("digite a posicao que deseja inserir o prof \n");

scanf("%d", &i);

vet=InserirProf(vet, n, i);

printf("Professor inserido com sucesso \n");

break;

}

}

system("pause");

break;

}

case 3:{

printf("Digite a opcao\n:");

printf("1 - Imprimir Aluno\n");

printf("2 - imprimir Professor \n");

scanf("%d", &x1);

switch (x1){

case 1:{

ImprimirAluno(vet, n);

break;

}

case 2:{

ImprimirProf(vet, n);

break;

}

case 3:{

ImprimirTodos(vet, n);

break;

}

}

system("pause");

break;

}

case 4:{

printf("Digite\n:");

printf("1 - Qauntidade de Aluno\n");

printf("2 - Quantidade de Professor \n");

scanf("%d", &x1);

switch (x1){

case 1:{

conta = QuantAlunos(vet, n);

printf("quantidade de alunos %d \n", conta);

break;

}

case 2:{

contp = QuantProf(vet, n);

printf("quantidade de profs %d \n", contp);

break;

}

}

system("pause");

break;

}

case 5:{

printf("digite a matricula do professor que vc deseja excluir \n");

scanf("%d", &mat);

vet=ExcluirProf(vet, n, mat);

printf("Professor excluido com sucesso");

system("pause");

break;

}

case 6:{

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

free(vet[i].info);

return 0;

}

}

}

return 0;

}