

CENTRO UNIVERSITÁRIO INTERNACIONAL UNINTER

ESCOLA SUPERIOR POLITÉCNICA

BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO

DISCIPLINA DE ESTRUTURA DE DADOS

ATIVIDADE PRÁTICA

Raphaell maciel de sousa – RU: 2479021

Prof. Vinicius Borin

cajazeiras – paraiba

2020

# EXERCÍCIO 1

ENUNCIADO: Faça um algoritmo em linguagem C que emule as características de um player de músicas sendo executado em modo texto, via prompt de comando.

Solução do aluno:

#include<stdio.h>

#include<stdlib.h>

int menu();

void InserirMusica();

void Listar();

struct ElementoDaLista\_Simples {

char nomeDaMusica[50];

char nomeDoArtista[50];

float duracao;

struct ElementoDaLista\_Simples\* prox;

} \*Head;

int main() {

int op, num, pos, c;

Head = NULL;

char musica[50];

while (1) {

op = menu();

switch (op) {

case 1:

InserirMusica();

break;

case 2:

Listar();

break;

case 3:

return 0;

default:

printf("Invalido\n");

}

}

return 0;

}

//Função Menu

int menu() {

int op, c;

system("Cls");

printf("1. Inserir Musica\n");

printf("2. Listar Musicas\n");

printf("3. Sair\n");

printf("Digite sua escolha: ");

scanf\_s("%d", &op);

while ((c = getchar()) != '\n' && c != EOF) {} // sempre limpe o buffer do teclado.

system("Cls");

return op;

}

//Insere músicas no início da lista

void InserirMusica()

{

int c;

ElementoDaLista\_Simples\* NovoElemento;

NovoElemento = (struct ElementoDaLista\_Simples\*)malloc(sizeof(struct ElementoDaLista\_Simples));

printf("Inserir nome da musica: ");

fgets(NovoElemento->nomeDaMusica, 50, stdin); //Leitura do nome

fflush(stdin);

printf("Inserir nome do artista: ");

fgets(NovoElemento->nomeDoArtista, 50, stdin); //Leitura do nome

fflush(stdin);

printf("Inserir duracao da musica: ");

scanf\_s("%f", &NovoElemento->duracao);

while ((c = getchar()) != '\n' && c != EOF) {} // sempre limpe o buffer do teclado.

if (Head == NULL)

{

Head = NovoElemento;

Head->prox = NULL;

}

else

{

NovoElemento->prox = Head;

Head = NovoElemento;

}

}

//Função para apresentar a lista no terminal

void Listar() { //listar

ElementoDaLista\_Simples\* ElementoVarredura;

ElementoVarredura = (struct ElementoDaLista\_Simples\*)malloc (sizeof(struct ElementoDaLista\_Simples));

ElementoVarredura = Head;

if (ElementoVarredura == NULL) {

printf("\n Lista vazia!" );

return;

}

while (ElementoVarredura != NULL) {

printf("\n###### MINHA PLAYLIST ATUAL ######\n\n");

while (ElementoVarredura != NULL) {

printf("Musica: %s", ElementoVarredura->nomeDaMusica);

printf("Artista: %s", ElementoVarredura->nomeDoArtista);

printf("Duracao: %f\n\n", ElementoVarredura->duracao);

ElementoVarredura = ElementoVarredura->prox;

}

}

printf("\n");

system("pause");

return;

}

Imagem do código funcionando no seu computador:

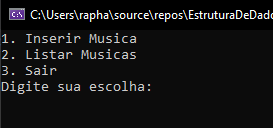


Figura 1: prompt com menu inicial.

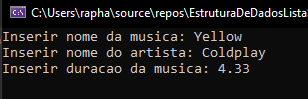


Figura 2: prompt com opção de inserção de música.

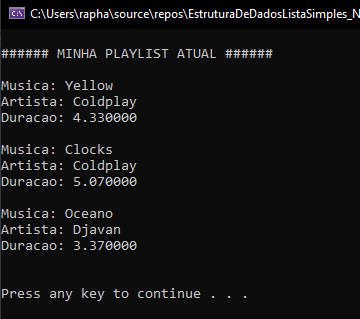


Figura 3: prompt com a exibição da lista de músicas cadastradas.

# EXERCÍCIO 2

ENUNCIADO: Faça um algoritmo em linguagem C que realiza a busca de um aluno da UNINTER no AVA. A busca deve ser realizada utilizando uma estrutura de dados bastante eficiente para esta tarefa.

Defina a estrutura de dados que você irá utilizar para fazer esta implementação e JUSTIFIQUE em texto porque você escolheu ela;

**Resposta**: Para a solução deste problema, foi utilizada a estrutura de dados não linear do tipo árvore binária. Visto que a busca de dados é uma das aplicações mais importantes das árvores binárias. Visto que o referido tipo de estrutura de dados possui um padrão quando se refere à profundidade dos ramos percorridos.

Solução do aluno:

#include<stdlib.h>

#include<stdio.h>

#include <string.h>

struct ElementoDaArvoreBinaria {

char nome[30];

char email[30];

int RU;

struct ElementoDaArvoreBinaria \*direita, \*esquerda; //ponteiro para direita e esquerda do tipo da nossa struct

};

//Declaração de vetores com os dados dos alunos

char nomes[10][30] = { "Jefferson", "Pepi", "Pedro", "Simas", "Alex",

"Ravi", "Mikael", "Jose", "Suiamya", "Raphaell" };

char emails[10][30] = { "jefferson@uninter.com", "pepi@uninter.com",

"pedro@uninter.com", "simas@uninter.com",

"alex@uninter.com", "ravi@uninter.com",

"mikael@uninter.com", "jose@uninter.com",

"suiamya@uninter.com", "raphaell@uninter.com"

};

int RUs[10] = { 2400001, 2400002, 2400003, 2400004, 2400005,

2400006, 2400007, 2400008, 2400009, 2479021 };

ElementoDaArvoreBinaria \*Buscar(ElementoDaArvoreBinaria\*\* ElementoVarredura, int num);

void Consultar\_EmOrdem(ElementoDaArvoreBinaria\*);

void Inserir(ElementoDaArvoreBinaria \*\*ElementoVarredura, char Nome[], char Email[], int num);

struct ElementoDaArvoreBinaria aluno[10];

int main() {

int num, c;

ElementoDaArvoreBinaria \*root;

root = (ElementoDaArvoreBinaria\*)malloc(sizeof(ElementoDaArvoreBinaria)); //alocar um espaço de memória para a raiz

root = NULL; //raiz da árvore, valor inicial nulo

ElementoDaArvoreBinaria\* ElementoBusca;

ElementoBusca = (ElementoDaArvoreBinaria\*)malloc(sizeof(ElementoDaArvoreBinaria));

memset(&aluno, 0x0, sizeof(aluno)); //Inicialização de memória

//Inserindo os dados na struct

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

{

Inserir(&root, nomes[i], emails[i], RUs[i]);

}

while (true)

{

printf("Digite o numero a ser buscado: ");

scanf\_s("%d", &num);

while ((c = getchar()) != '\n' && c != EOF) {} // sempre limpe o buffer do teclado.

ElementoBusca = Buscar(&root, num);

if (ElementoBusca != 0)

{

printf("Valor localizado.\n");

printf("\n\n");

printf("\n\nRU %d\t", ElementoBusca->RU);

printf("\nNome %s\t", ElementoBusca->nome);

printf("\nEmail %s\t", ElementoBusca->email);

printf("\n\n");

}

else

printf("Valor nao encontrado na arvore.\n");

system("pause");

}

return 0;

}

//\*\* significa um ponteiro para um ponteiro para uma variável

// \*ElementoVarredura aponta para \*root, que por sua vez, aponta para um elemento da struct

void Inserir(ElementoDaArvoreBinaria \*\*ElementoVarredura, char Nome[], char Email[], int Num) {

if (\*ElementoVarredura == NULL)

{ //SE ESTÁ VAZIO, COLOCA NA ÁRVORE

ElementoDaArvoreBinaria \*NovoElemento = NULL;

NovoElemento = (ElementoDaArvoreBinaria\*)malloc(sizeof(ElementoDaArvoreBinaria));

NovoElemento->esquerda = NULL;

NovoElemento->direita = NULL;

NovoElemento->RU = Num;

strcpy\_s(NovoElemento->nome, Nome);

strcpy\_s(NovoElemento->email, Email);

\*ElementoVarredura = NovoElemento;

return;

}

if (Num < (\*ElementoVarredura)->RU) //\* para acessar o conteúdo que a variável aponta

{

Inserir(&(\*ElementoVarredura)->esquerda, Nome, Email, Num); //ponteiro para outro ponteiro

}

else

{

if (Num > (\*ElementoVarredura)->RU)

{

Inserir(&(\*ElementoVarredura)->direita, Nome, Email, Num);

}

}

}

ElementoDaArvoreBinaria \*Buscar(ElementoDaArvoreBinaria \*\*ElementoVarredura, int num)

{

if (\*ElementoVarredura == NULL)

return NULL;

if (num < (\*ElementoVarredura)->RU)

{

Buscar(&((\*ElementoVarredura)->esquerda), num);

}

else

{

if (num > (\*ElementoVarredura)->RU)

{

Buscar(&((\*ElementoVarredura)->direita), num);

}

else

{

if (num == (\*ElementoVarredura)->RU)

{

return \*ElementoVarredura;

}

}

}

}

void Consultar\_EmOrdem(ElementoDaArvoreBinaria \*ElementoVarredura)

{

if (ElementoVarredura)

{

Consultar\_EmOrdem(ElementoVarredura->esquerda);

printf("\n%d\t", ElementoVarredura->RU);

printf("\n%c\t", ElementoVarredura->nome);

printf("\n%c\t", ElementoVarredura->email);

Consultar\_EmOrdem(ElementoVarredura->direita);

}

}

Imagem do código funcionando no seu computador:

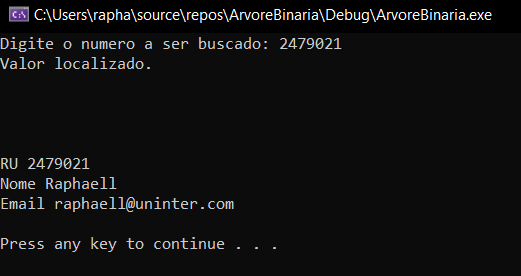


Figura 4: prompt com a impressão dos dados encontrados.