CENTRO UNIVERSITÁRIO INTERNACIONAL UNINTER
ESCOLA SUPERIOR POLITÉCNICA
CST ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - DISTÂNCIA
DISCIPLINA DE ESTRUTURA DE DADOS

ATIVIDADE PRÁTICA

FELIPE MARCHI GUIMARÃES – RU: 1371185
VINICIUS POZZOBON BORIN

RIO CLARO - SP
2021

1 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.

1. Deve-se criar uma *playlist* das músicas utilizando uma lista encadeada. A lista encadeada poderá ser simples ou dupla, circular ou não circular. Fica a critério do aluno decidir.

2. Deve-se armazenar o nome de cada música, do artista/banda e a duração da faixa. Para o armazenamento utilize uma estrutura heterogênea de dados.

3. Para inserção dos dados, você pode criar uma leitura dos dados através de um menu na tela ou já deixá-los armazenados em um arquivo texto no seu computador e só carregar este arquivo ao executar o programa. Ou ambas soluções. Decida também como você irá implementar a inserção (no início, no fim ou no meio da lista encadeada);

4. Deve existir um menu na tela. Este menu deve permitir a inserção de novas músicas (caso optado pela inserção manual de dados), deve ter a opção de listar todas as músicas da *playlist* (listagem de uma lista encadeada) na tela e encerrar o programa;

SOLUÇÃO DO ALUNO:

```c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<locale.h>

// declaração das funções
int menu();
void tocando();
void inserir_track(int cod, char* musica, int duracao, char* artista, char* album, int ano);
void tocar_anterior();
void tocar_proxima();
void listar();

// declaração das structs
typedef struct _Track {
	int cod_musica;
	char musica[40];
	int duracao;
	char artista[40];
	char album[40];
	int ano;
};
```

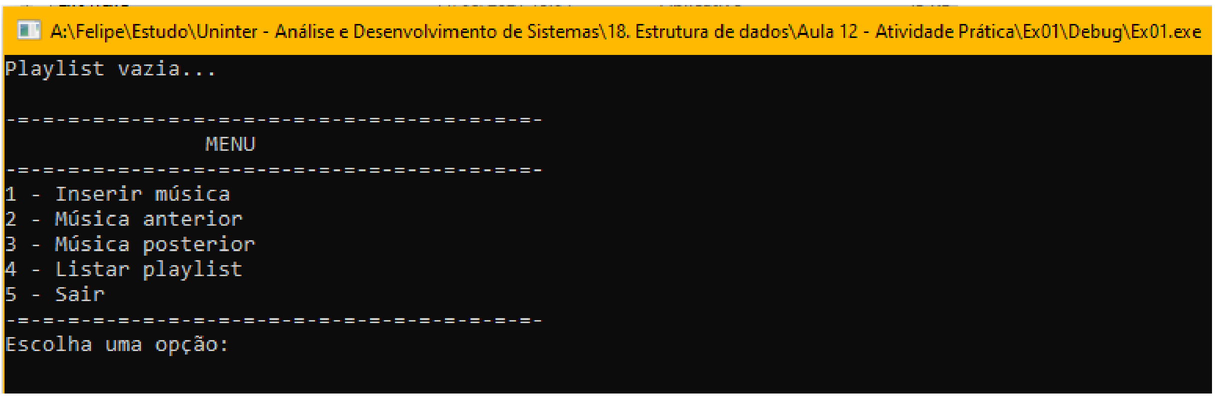
```c
typedef struct _List {
	_Track Track;
	_List* ant;
	_List* prox;
};

_List* Head, * Tail, * Aux = NULL;


// Programa principal
int main() {
	setlocale(LC_ALL, "Portuguese");

	int op, c;
	int cod = 0, duracao, ano;
	char musica[40], artista[40], album[40];

	while (true) {
		op = menu();
		switch (op) {
		case 1: // inserir track
			cod++;
			printf("Informe o nome da música: ");
			gets_s(musica);
			printf("Informe a duração da música em segundos: ");
			scanf_s("%d", &duracao);
			while ((c = getchar()) != '\n' && c != EOF) {}
			printf("Informe o nome do artista/banda: ");
			gets_s(artista);
			printf("Informe o nome do álbum: ");
			gets_s(album);
			printf("Informe o ano de lançamento da música: ");
			scanf_s("%d", &ano);
			while ((c = getchar()) != '\n' && c != EOF) {}
			inserir_track(cod, musica, duracao, artista, album, ano);
			break;

		case 2:
			tocar_anterior();
			break;

		case 3:
			tocar_proxima();
			break;

		case 4:
			listar();
			break;

		case 5: // sair
```

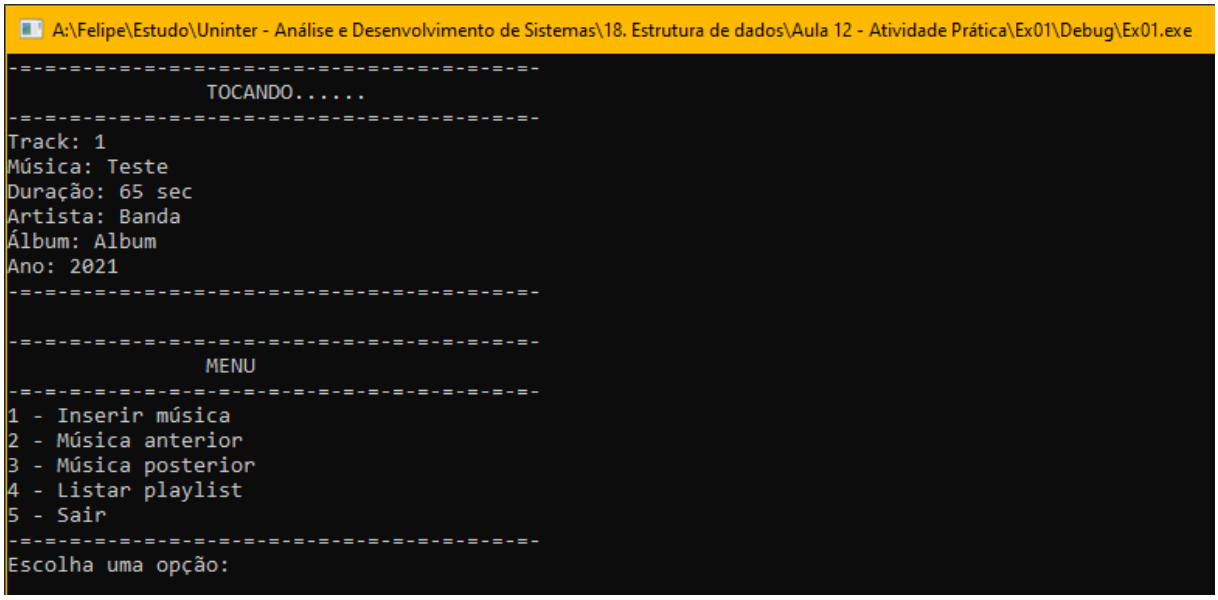
```
                printf("Saindo...\n\n\n");
                system("pause");
                return 0;

            default:
                printf("Opção inválida!\n\n\n");
                system("pause");
                break;
            }
        }


        printf("\n\n\n");
        system("pause");
        return 0;
}

// função para escolher uma opção
int menu() {
        int op, c;

        system("cls");
        tocando();
        printf("-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-\n");
        printf("\t\tMENU\n");
        printf("-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-\n");
        printf("1 - Inserir música\n");
        printf("2 - Música anterior\n");
        printf("3 - Música posterior\n");
        printf("4 - Listar playlist\n");
        printf("5 - Sair\n");
        printf("-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-\n");
        printf("Escolha uma opção: ");
        scanf_s("%d", &op);
        while((c = getchar()) != '\n' && c != EOF) {}
        system("cls");

        return op;
}

// mostra a música atual
void tocando() {
        if (Head == NULL) {
                printf("Playlist vazia...\n\n");
        }
        else {
                printf("-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-\n");
                printf("\t\tTOCANDO.....\n");
                printf("-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-\n");
                printf("Track: %d\n", Aux->Track.cod_musica);
```

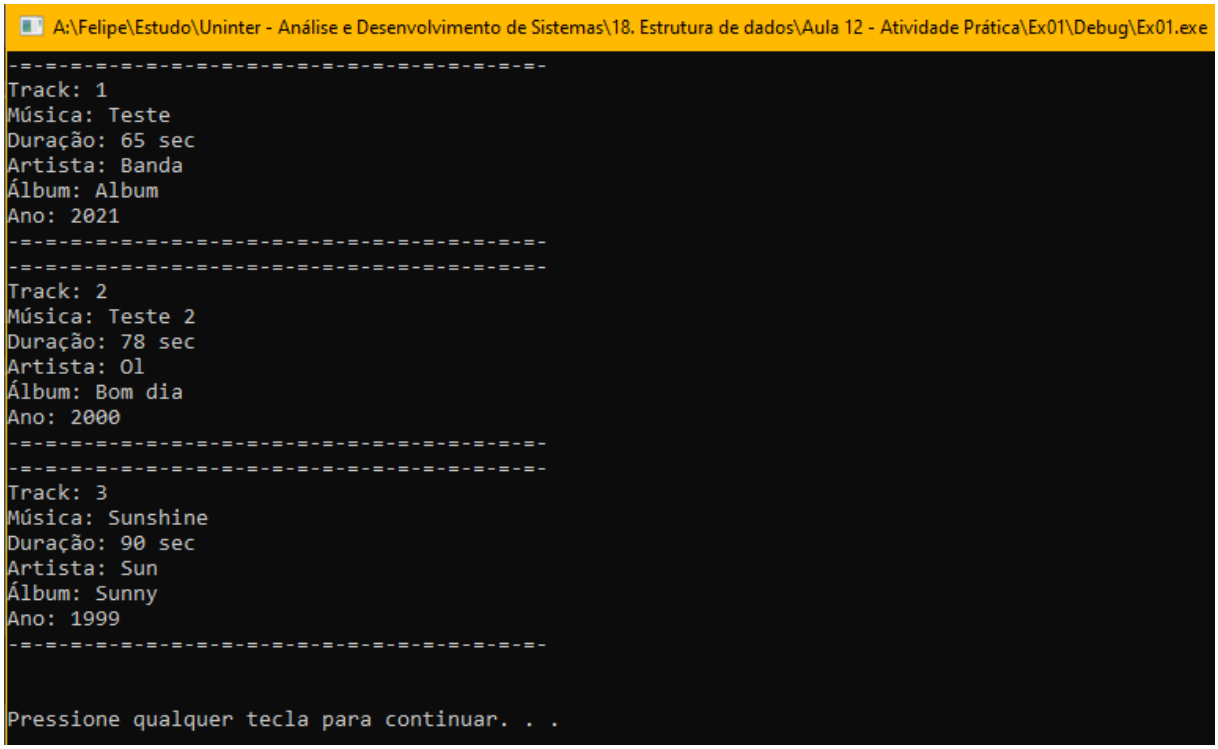
```c
			printf("Música: %s\n", Aux->Track.musica);
			printf("Duração: %d sec\n", Aux->Track.duracao);
			printf("Artista: %s\n", Aux->Track.artista);
			printf("Álbum: %s\n", Aux->Track.album);
			printf("Ano: %d\n", Aux->Track.ano);
			printf("-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-\n\n");
		}
}

// função para inserir uma música no final da lista
void inserir_track(int cod, char* musica, int duracao, char* artista, char* album, int ano)
{
	_List* novaFaixa = (_List*)malloc(sizeof(_List));

	novaFaixa->Track.cod_musica = cod;
	strcpy_s(novaFaixa->Track.musica, musica);
	novaFaixa->Track.duracao = duracao;
	strcpy_s(novaFaixa->Track.artista, artista);
	strcpy_s(novaFaixa->Track.album, album);
	novaFaixa->Track.ano = ano;

	int flag = 0;

	if (Head == NULL) {
		Head = novaFaixa;
		Tail = novaFaixa;
		novaFaixa->ant = Head;
		novaFaixa->prox = Head;
		flag = 1;
		Aux = Head;
	}
	else {
		Tail->prox = novaFaixa;
		novaFaixa->ant = Tail;
		Tail = novaFaixa;
		Tail->prox = Head;
		Head->ant = Tail;
		flag = 1;
	}

	if (flag == 0)
		printf("ERRO!\n\n");
	else
		printf("Música inserida com sucesso!\n\n");
	system("pause");
}

// função para tocar música anterior
void tocar_anterior() {
	if (Head != NULL)
```

```
			Aux = Aux->ant;
}

// função para tocar música posterior
void tocar_proxima() {
	if (Head != NULL)
		Aux = Aux->prox;
}

// função para listar a playlist
void listar() {
	if (Head == NULL) {
		printf("Playlist vazia...\n\n");
	}
	else {
		_List* varredura = Head;
		do {
			printf("-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-\n");
			printf("Track: %d\n", varredura->Track.cod_musica);
			printf("Música: %s\n", varredura->Track.musica);
			printf("Duração: %d sec\n", varredura->Track.duracao);
			printf("Artista: %s\n", varredura->Track.artista);
			printf("Álbum: %s\n", varredura->Track.album);
			printf("Ano: %d\n", varredura->Track.ano);
			printf("-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-\n");
			varredura = varredura->prox;
		} while (varredura != Head);
	}
	printf("\n\n");
	system("pause");
}
```

<u>IMAGEM DO CÓDIGO FUNCIONANDO:</u>




A:\Felipe\Estudo\Uninter - Análise e Desenvolvimento de Sistemas\18. Estrutura de dados\Aula 12 - Atividade Prática\Ex01\Debug\Ex01.exe
TOCANDO.....
Track: 1
Música: Teste
Duração: 65 sec
Artista: Banda
Álbum: Album
Ano: 2021
MENU
1 - Inserir música
2 - Música anterior
3 - Música posterior
4 - Listar playlist
5 - Sair
Escolha uma opção:


A:\Felipe\Estudo\Uninter - Análise e Desenvolvimento de Sistemas\18. Estrutura de dados\Aula 12 - Atividade Prática\Ex01\Debug\Ex01.exe
Track: 1
Música: Teste
Duração: 65 sec
Artista: Banda
Álbum: Album
Ano: 2021
Track: 2
Música: Teste 2
Duração: 78 sec
Artista: Ol
Álbum: Bom dia
Ano: 2000
Track: 3
Música: Sunshine
Duração: 90 sec
Artista: Sun
Álbum: Sunny
Ano: 1999
Pressione qualquer tecla para continuar. . .

2 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 de busca. Dentre as estruturas que podem ser empregados estão: árvore binária ou hash.

1. Deve-se armazenar o nome do aluno, seu e-mail e seu RU. Para o armazenamento utilize uma estrutura heterogênea de dados.

2. Não é necessário fazer a leitura dos dados dos dados dos alunos manualmente. Você já pode deixar pré-cadastrado os dados no seu código. Cadastre pelo menos uns 10 contatos de alunos na sua estrutura de dados. Um dos contatos deverá ser o seu próprio nome e o seu RU da UNINTER;

3. Em um menu na tela, peça para o usuário digitar um RU. O programa deverá realizar a busca por este RU na estrutura de dados e, caso localize o RU, deverá mostrar o nome correspondente do aluno e o e-mail deste contato. Caso não localize, uma mensagem de erro deve ser apresentada.

4. Para testar o programa, teste a busca com o seu RU e coloque a captura de dela.

SOLUÇÃO DO ALUNO:

```c
#include<stdio.h>
#include<stdlib.h>
#include<locale.h>
#include<string.h>

// declaração das structs
typedef struct _Aluno {
	int ru;
	char nome[40], email[50];
};

typedef struct _Arvore {
	_Aluno Aluno;
	_Arvore* esq, * dir;
};

// declaração das funções
int menu();
void inserirCadastro(_Arvore** ElementoVarredura, int r, char n[], char e[]);
_Arvore* buscarAluno(_Arvore** ElementoVarredura, int num);

// programa principal
```

```c
int main() {
	setlocale(LC_ALL, "portuguese");

	int op, c, nBuscado;
	_Arvore* Root = (_Arvore*)malloc(sizeof(_Arvore));
	Root = NULL;
	_Arvore* Busca;

	int ru[10] = { 1371180, 1371181, 1371182, 1371183, 1371184, 1371185,
		1371186, 1371187, 1371188, 1371189 };
	char nome[10][30] = { "Ana", "Beto", "Carlos", "Didi", "Eder",
		"Felipe Marchi Guimarães", "Gabi", "Heitor", "Igor", "Julio" };
	char email[10][20] = { "a@a.com", "b@b.com", "c@c.com", "d@d.com",
"e@e.com",
		"fmg.17@outlook.com", "g@g.com", "h@h.com", "i@i.com", "j@j.com" };

	// função para inserir 10 cadastros prévios
	for (int i = 0; i < 10; i++) {
		inserirCadastro(&Root, ru[i], nome[i], email[i]);
	}

	while (true) {
		op = menu();

		switch (op) {
			case 1:
				printf("Informe o RU que deseja buscar: ");
				scanf_s("%d", &nBuscado);
				while ((c = getchar()) != '\n' && c != EOF) {}
				system("cls");

				Busca = buscarAluno(&Root, nBuscado);
				if (Busca == NULL)
					printf("Cadastro não encontrado...\n\n");
				else {
					printf("Encontrei os seguintes dados...\n\n");
					printf("RU: %d\n", Busca->Aluno.ru);
					printf("Nome: %s\n", Busca->Aluno.nome);
					printf("Email: %s\n\n\n", Busca->Aluno.email);
				}
				system("pause");
				break;

			case 2:
				printf("Saindo...\n\n");
				system("pause");
				return 0;

			default:
				printf("Opção inválida!\n\n");
```

```
                                system("pause");
                                break;
                        }
                }

                system("pause");
                return 0;
}

// função do menu
int menu() {
        int op, c;

        system("cls");
        printf("-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=\n");
        printf("1 - Buscar aluno por RU\n");
        printf("2 - Sair\n");
        printf("-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=\n\n");
        printf("Digite uma opção: ");
        scanf_s("%d", &op);
        while((c = getchar()) != '\n' && c != EOF) {}

        system("cls");
        return op;
}

void inserirCadastro(_Arvore** ElementoVarredura, int r, char n[], char e[]) {
        if (*ElementoVarredura == NULL) {
                _Arvore* NovoElemento = NULL;
                NovoElemento = (_Arvore*)malloc(sizeof(_Arvore));
                NovoElemento->dir = NULL;
                NovoElemento->esq = NULL;

                NovoElemento->Aluno.ru = r;
                strcpy_s(NovoElemento->Aluno.nome, n);
                strcpy_s(NovoElemento->Aluno.email, e);

                *ElementoVarredura = NovoElemento;
                return;
        }

        if (r < (*ElementoVarredura)->Aluno.ru) {
                inserirCadastro(&(*ElementoVarredura)->esq, r, n, e);
        }
        else if (r > (*ElementoVarredura)->Aluno.ru) {
                inserirCadastro(&(*ElementoVarredura)->dir, r, n, e);
        }
}

_Arvore* buscarAluno(_Arvore** ElementoVarredura, int num) {
```

```
        if (*ElementoVarredura == NULL)
                return NULL;

        if (num < (*ElementoVarredura)->Aluno.ru)
                buscarAluno(&((*ElementoVarredura)->esq), num);

        else if (num > (*ElementoVarredura)->Aluno.ru)
                buscarAluno(&((*ElementoVarredura)->dir), num);

        else
                return *ElementoVarredura;
}
```

IMAGEM DO CÓDIGO FUNCIONANDO: