



CENTRO UNIVERSITÁRIO INTERNACIONAL UNINTER
ESCOLA SUPERIOR POLITÉCNICA
BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO
DISCIPLINA DE LÓGICA DE PROGRAMAÇÃO E ALGORITMOS

ATIVIDADE PRÁTICA

KELE PÓLVORA CAIRES – RU: 2707381

VINICIUS BORIN

IGUAÍ – BAHIA

2019

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; Utilize como base o código de listas da AULA PRÁTICA 2 da disciplina. Código está disponível no Github do professor. O link está na aula prática 2;

Solução do aluno:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int menu();
void InserirInicio(char nome[], char artista[], int m, int s);
void InserirFim(char nome[], char artista[], int m, int s);
void InserirMeio(char nome[], char artista[], int m, int s, int posicao);
int Remover(char nome[], char artista[], int m, int s);
void Listar();

//Variáveis
struct Musicas {
    char nome[100];
    char artista[100];
    int min, seg;
    struct Musicas *prox;
} *Head;

Musicas *inicio, *fim, *aux;

//Fazendo while para armazenagem de dados e funções para execução de acordo o escolhido
int main() {
    int op, pos, c, m, s;
    char nome[100];
    char artista[100];
    Head = NULL;

    while (1) {
        op = menu();
        switch (op) {

            case 1:
                printf("Digite o nome da musica: ");
                gets_s(nome);
                while ((c = getchar()) != '\n' && c != EOF) {} //USADO PRA LIMPAR OS DADOS

                printf("Digite o nome do artista: ");
                gets_s(artista);
                while ((c = getchar()) != '\n' && c != EOF) {}

                printf("Digite a duracao da musica (mm ss): ");
                scanf_s("%d %d", &m, &s);
                while ((c = getchar()) != '\n' && c != EOF) {}
```

```

        InserirInicio( nome, artista, m, s);
        break;
    case 2:
        printf("Digite o nome da musica desejado: ");
        scanf_s("%d", &nome);
        printf("Digite o nome do artista desejado: ");
        scanf_s("%d", &artista);
        printf("Digite o minuto desejado: ");
        scanf_s("%d", &m);
        printf("Digite o segundo desejado: ");
        scanf_s("%d", &s);
        while ((c = getchar()) != '\n' && c != EOF) {}
        InserirFim(nome, artista,m,s);
        break;
    case 3:
        printf("Digite o nome desejado: ");
        scanf_s("%d", &nome);
        printf("Digite o nome do artista desejado: ");
        scanf_s("%d", &artista);
        printf("Digite o minuto desejado: ");
        scanf_s("%d", &m);
        printf("Digite o segundo desejado: ");
        scanf_s("%d", &s);
        while ((c = getchar()) != '\n' && c != EOF) {}
        printf("Digite a posicao que deseja inserir: ");
        scanf_s("%d", &pos);
        while ((c = getchar()) != '\n' && c != EOF) {}
        InserirMeio(nome, artista, m, s,pos);
        break;
    case 4:
        int Remover();
        break;
    case 5:
        Listar();
        break;
    case 6:
        return 0;
    default:
        printf("Invalido\n");
    }
}
return 0;
}

//Opções para escolha do usuário
int menu() {
    int op, c;
    system("Cls");

    printf("1.Inserir musica no inicio da lista\n");
    printf("2.Inserir no fim da lista encadeada simples\n");
    printf("3.Inserir no meio da lista encadeada simples\n");
    printf("4.Remover da lista encadeada simples\n");
    printf("5.Listar a lista encadeada simples\n");
    printf("6.Sair\n");
    printf("Digite sua escolha: ");

    scanf_s("%d", &op);
    while ((c = getchar()) != '\n' && c != EOF) {}

    system("Cls");
    return op;
}

// funções
void InserirInicio(char nome[], char artista[], int m, int s)
{
    Musicas *NovoElemento;

```

```

NovoElemento = (struct Musicas *)malloc(sizeof(struct Musicas));
strcpy_s(NovoElemento->nome, nome);
strcpy_s(NovoElemento->artista, artista);
NovoElemento->min = m;
NovoElemento->seg = s;

if (Head == NULL)
{
    Head = NovoElemento;
    Head->prox = NULL;
}
else
{
    NovoElemento->prox = Head;
    Head = NovoElemento;
}
}

void Listar()
{
    Musicas *ElementoVarredura;
    ElementoVarredura = (struct Musicas *)malloc(sizeof(struct Musicas));

    ElementoVarredura = Head;
    if (ElementoVarredura == NULL) {
        return;
    }
    while (ElementoVarredura != NULL) {
        printf("Nome da musica: %s\n", ElementoVarredura->nome);
        printf("Artista: %s\n", ElementoVarredura->artista);
        printf("Duracao: %d:%d\n\n", ElementoVarredura->min, ElementoVarredura->seg);
        ElementoVarredura = ElementoVarredura->prox;
    }
    printf("\n");

    system("pause");
    return;
}

void InserirFim(char nome[], char artista[], int m, int s)
{
    ElementoDaLista_Simples *NovoElemento;
    NovoElemento = (struct ElementoDaLista_Simples *)malloc(sizeof(struct ElementoDaLista_Simples));
    ElementoDaLista_Simples *ElementoVarredura;
    ElementoVarredura = (struct ElementoDaLista_Simples *)malloc(sizeof(struct
ElementoDaLista_Simples));

    NovoElemento->dado = nome;
    NovoElemento->dado = artista;
    NovoElemento->dado = m;
    NovoElemento->dado = s;

    if (Head == NULL)
    {
        Head = NovoElemento;
        Head->prox = NULL;
    }
    else
    {
        ElementoVarredura = Head;
        while (ElementoVarredura->prox != NULL)
            ElementoVarredura = ElementoVarredura->prox;

        ElementoVarredura->prox = NovoElemento;
        NovoElemento->prox = NULL;
    }
}

void InserirMeio(char nome[], char artista[], int m, int s, int posicao)

```

```

{
    ElementoDaLista_Simples *NovoElemento;
    NovoElemento = (struct ElementoDaLista_Simples *)malloc(sizeof(struct ElementoDaLista_Simples));
    ElementoDaLista_Simples *ElementoVarredura;
    ElementoVarredura = (struct ElementoDaLista_Simples *)malloc(sizeof(struct
ElementoDaLista_Simples));
    ElementoDaLista_Simples *ElementoAuxiliar;
    ElementoAuxiliar = (struct ElementoDaLista_Simples *)malloc(sizeof(struct
ElementoDaLista_Simples));

    NovoElemento->dado = nome;
    NovoElemento->dado = artista;
    NovoElemento->dado = m;
    NovoElemento->dado = s;

    if (posicao == 0)
    {
        Head = NovoElemento;
        Head->prox = NULL;
    }
    else
    {
        ElementoVarredura = Head;
        for (int i = 0; i < posicao - 1; i++)
            ElementoVarredura = ElementoVarredura->prox;

        ElementoAuxiliar = ElementoVarredura->prox;
        ElementoVarredura->prox = NovoElemento;
        NovoElemento->prox = ElementoAuxiliar;
    }
}

int Remover(char nome[], char artista[], int m, int s)
{
    ElementoDaLista_Simples *ElementoVarredura;
    ElementoVarredura = (struct ElementoDaLista_Simples *)malloc(sizeof(struct
ElementoDaLista_Simples));
    ElementoDaLista_Simples *Anterior;
    Anterior = (struct ElementoDaLista_Simples *)malloc(sizeof(struct ElementoDaLista_Simples));

    ElementoVarredura = Head;
    while (ElementoVarredura != NULL) {
        if (ElementoVarredura->dado == nome) {
            if (ElementoVarredura == Head) {
                Head = ElementoVarredura->prox;
                free(ElementoVarredura);
                return 1;
            }
            else {
                Anterior->prox = ElementoVarredura->prox;
                free(ElementoVarredura);
                return 1;
            }
        }
        else {
            Anterior = ElementoVarredura;
            ElementoVarredura = ElementoVarredura->prox;
        }
    }
    return 0;
}

void localizarMusica()
{
    aux = inicio;

    if (inicio == NULL)

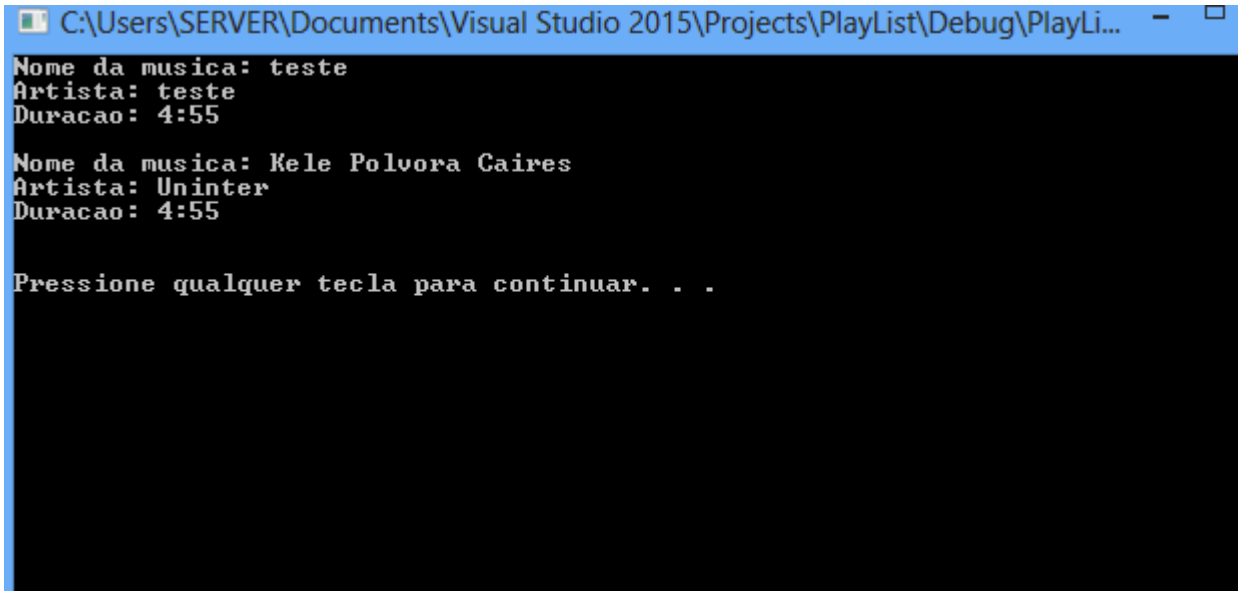
```

```

{
    printf("\nA lista estah vazia!");
}
else
{
    char musica[40];
    printf("Digite a musica que procura: ");
    scanf("%s", &aux);
    while (aux != NULL)
    {
        if (musica == aux->nome)
        {
            printf("\nMusica  %s localizada!\n\n", aux->nome);
            printf("musica %s", aux->nome);
            printf("artista: %s\n\n", aux->artista);
            printf("tempo: %s\n\n", aux->min);
            return;
        }
        else
        {
            aux = aux->prox;
        }
    }
    if (aux == NULL)
    {
        printf("\nMusica  %s nao localizada!\n", musica);
    }
}
printf("\n\n");
}

```

Imagem do código funcionando no computador:



```
C:\Users\SERVER\Documents\Visual Studio 2015\Projects\PlayList\Debug\PlayLi...
Nome da musica: teste
Artista: teste
Duracao: 4:55

Nome da musica: Kele Polvora Caires
Artista: Uninter
Duracao: 4:55

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

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

// funções auxiliares
int check(char *s);
void take_component();

// armazenando os dados no vetor
char databank[][100] = {

    "2707381" , " \n Nome do Aluno:Kele Polvora Caires \n Email:Kellepolvora@gmail.com",
    "2707382" , " \n Nome do Aluno:Carlos Araujo Santos \n Email:carlosaraujo@gmail.com",
    "2707383" , " \n Nome do Aluno:Marcos Silva Santos \n Email:marcossilva@gmail.com",
    "2707384" , " \n Nome do Aluno:Maria Alves Caires \n Email:mariaalves@gmail.com",
    "2707385" , " \n Nome do Aluno:Marcos Vinicius Caires \n Email:marcosvinivius@gmail.com",
```



```

"2707386" , " \n Nome do Aluno:Luciana Castro Polvora \n Email:lucianacastro@gmail.com",
"2707387" , " \n Nome do Aluno:Lucas Santos Nascimento \n Email:lucassantos@gmail.com",
"2707388" , " \n Nome do Aluno:Carla Borges Santos \n Email:carlaborges@gmail.com",
"2707389" , " \n Nome do Aluno:Arthur Braga da Silva \n Email:arthurbraga@gmail.com",
"2707375" , " \n Nome do Aluno:José Castro Caires \n Email:josecastro@gmail.com",

};

//variáveis
char input[100];
char component[100];
char *point;

int main() {

    //variáveis
    int indice;
    char option;

    // pedindo para informar o ru e fazendo a busca dos dados
    do {

        printf("Informe o seu ru para obter os dados:");
        gets_s(input);
        point = input;
        printf("dados:");
        take_component();

        do {

            // se caso não tiver os dados retorna um alerta
            indice = check(component);

            if (indice != -1) printf("%s", databank[indice + 1]);
            else printf("Sem registros");

            take_component();

        } while (*component);

        system("pause");
        return 0;

    } while (option == 's' || option == 'S');

}

//funções

//compara as matrizes e ver se existe ou não
int check(char *s)
{

    int i;

    for (i = 0; *databank[i]; i++)
    {
        if (!strcmp(databank[i], s)) break;
    }
}

```

```

    }

    if (*databank[i]) return(i);
    else return(-1);
}

//lê o próximo componente da matriz
void take_component()
{
    char *q;

    q = component;

    while (*point && *point != ' ')
    {
        *q = *point;
        point++;
        q++;
    }

    if (*point == ' ')point++;
    *q = '\0';
}

```

Imagem do código funcionando no computador:

```

C:\Users\SERVER\Documents\Visual Studio 2015\Projects\Busca\Debug\Busca.exe
Informe o seu ru para obter os dados:2707381
dados:
Nome do Aluno:Kele Polvora Caires
Email:Kellepolvora@gmail.comPressione qualquer tecla para continuar. . .

```