Relatório Sobre o Trabalho *"Cadastro de Instrumentos – Banda Molejo"*

Alunos: Gabriel Oliveira Costa e Isadora Laviola Oliveira

1. ***Definição das Bibliotecas.***:

Não usamos nenhuma biblioteca além das comumente usadas.

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

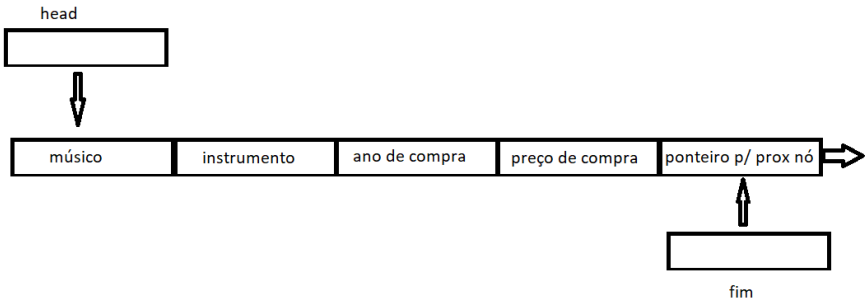
2. ***Definição das estruturas:***

- **struct banda**: Contém os dados que serão armazenados sobre os instrumentos e um ponteiro para o próximo elemento da lista. A chamamos de "tipoNo" para ficar mais claro em nossas cabeças o desenho da lista em si.

- **struct lista**: Estrutura do tipo "tipoNo" que terá um ponteiro para o primeiro elemento e um para o último. O primeiro irá auxiliar na inserção, localização e remoção dos elementos e o último, na remoção.

```c
typedef struct banda{
    char nomeDoInstrumento [50];
    char nomeDoMusico [100];
    int anoDeCompra;
    float precoDeCompra;
    struct banda *proximo;
}tipoNo;

typedef struct lista{
    tipoNo *head;
    tipoNo *fim;
}tipoLista;
```



3. ***Declaração das funções que serão usadas no decorrer do programa***:

Declaramos as funções que serão usadas antes do main e as definimos depois do mesmo.

```c
void cadastrar(tipoLista *lista);
int menu();
void escolha(tipoLista *lista, int opc);
void imprimirInstrumentos(tipoLista *lista);
void calculaMedia(tipoLista *lista);
void buscarMusico(tipoLista *lista);
void exlcuirInstrumento(tipoLista *lista);
```

4. ***Main:***

Criamos a lista dentro do main para facilitar a passagem por parâmetro e iniciamos o menu para que o usuário navegue pelo programa, até que por fim o usuário digite '6' para sair do programa.

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

   tipoLista *lista = (tipoLista*)malloc(sizeof(tipoLista));
    if(!lista){
        printf("Impossível alocar lista");
    }else {
        lista->head = NULL;
        lista->fim = NULL;
    }

   int opcao;

   do{
    opcao = menu();
    escolha(lista, opcao);
   }while(opcao != 6);

    return 0;
}
```

5. *Definição das funções:*

Definimos então as funções usadas durante a execução do programa.

- **Menu de opções:**

```c
int menu(){
  int opc;

    printf("\n\n---------- CADASTRO DE INSTRUMENTOS - BANDA MOLEJO ----------\n");
    printf("\nDigite um número para escolher a opção desejada: ");
    printf("\n1 - Cadastrar instrumento");
    printf("\n2 - Deletar instrumento");
    printf("\n3 - Imprimir todos os instrumentos cadastrados");
    printf("\n4 - Buscar por músico");
    printf("\n5 - Saber idade média dos instrumentos cadastrados");
    printf("\n6- Sair do programa\n\n");
    scanf("%d%*c", &opc);
    system("cls || clear");

    return opc;
}
```

- **Função para Cadastro.**

Alocamos dinamicamente memória para um nó, lemos os dados e fazemos a ligação na lista.

```c
void cadastrar(tipoLista *lista){

    tipoNo *novoNo = (tipoNo*)malloc(sizeof(tipoNo));
    if(!novoNo){
        printf("Impossível alocar");
    } else{
        printf("Músico: ");
        gets(novoNo->nomeDoMusico);
        fflush(stdin);
        printf("Nome do instrumento: ");
        gets(novoNo->nomeDoInstrumento);
        fflush(stdin);
        volta: printf("Ano de compra do instrumento (yyyy): ");
        scanf("%4d%*c", &novoNo->anoDeCompra);
        if (novoNo->anoDeCompra > 2020){
            printf("ANO INVÁLIDO\n");
            goto volta;
        }else{
        printf("Preço de compra: ");
        scanf("%f%*c", &novoNo->precoDeCompra);
        }
        fflush(stdin);

    novoNo->proximo = lista->head;
    lista->head = novoNo;
    }
}
```

- **Função para escolha do menu (Switch Case):**
  Recebe a lista como parâmetro e direciona para a função solicitada, conforme a opção digitada pelo usuário.

```
95  void escolha(tipoLista *lista, int opc){
96      switch(opc){
97          case 1:
98              cadastrar(lista);
99              break;
100
101         case 2:
102             exlcuirInstrumento(lista);
103             break;
104
105
106         case 3:
107             imprimirInstrumentos(lista);
108             break;
109
110         case 4:
111             buscarMusico(lista);
112             break;
113
114         case 5:
115             calculaMedia(lista);
116             break;
117
118         case 6:
119             printf("\nPROGRAMA ENCERRADO COM SUCESSO!\n");
120             break;
121
122         default:
123             printf("\nCÓDIGO INVÁLIDO!\n");
124             break;
125     }
126 }
127
```

- **Impressão da Lista de Instrumentos.**
  Aqui é exibida a lista completa de instrumentos cadastrados. A função “getchar()” aparece aqui pela primeira vez e será usada mais vezes na execução do programa. Optamos pelo seu uso para ajudar a ter a tela mais limpa durante impressão dos dados. O menu só volta a aparecer quando o usuário pressiona alguma tecla aleatória.

```
95  void escolha(tipoLista *lista, int opc){
127
128 void imprimirInstrumentos(tipoLista *lista){
129
130     printf("\n\n ---------- LISTAGEM DE INSTRUMENTOS ----------");
131     tipoNo *aux = lista->head;
132     if(lista->head == NULL){
133         printf("\nNENHUM ELEMENTO NA LISTA!\nPOR FAVOR, ESCOLHA A OPÇÃO 1 PARA CADASTRO\n");
134     }else{
135         while(aux != NULL){
136             printf("\n - %s", aux->nomeDoInstrumento);
137             aux = aux->proximo;
138         }
139         printf ("\n\nPressione qualquer tecla para continuar...\n");
140         getchar();
141     }
142 }
```

- **Cálculo da Média:**
  Não achamos difícil a calcular a média da idade dos instrumentos, pois já havíamos feito um exercício do tipo com a professora Camila.

```
144 void calculaMedia(tipoLista *lista){
145     int cont = 0;
146     float soma = 0;
147     tipoNo *aux = lista->head;
148 
149     if(aux == NULL){
150         printf("\nNENHUM ELEMENTO NA LISTA!\nPOR FAVOR, ESCOLHA A OPÇÃO 1 PARA CADASTRO\n");
151         printf ("\nPressione qualquer tecla para continuar...\n");
152         getchar();
153     }else{
154         while(aux != NULL){
155         soma += aux->anoDeCompra;
156         cont++;
157         aux = aux->proximo;
158     }
159     printf("\n\nIdade média dos instrumentos: %.1f anos", soma/cont);
160     printf ("\nPressione qualquer tecla para continuar...\n");
161     getchar();
162     }
163 }
```

- **Função de Busca por Músico:**
  Aqui, o usuário encontra um filtro por nome, para localizar todos os instrumentos relacionados a um músico, dado o nome dele. Dentre as funções citadas até agora, essa foi a mais trabalhosa, pois não tínhamos em mente como comparar a string digitada pelo usuário com a já existente na lista. Além disso, estávamos colocando um else para o if dentro do while para mostrar a mensagem de “Dado não encontrado. Verifique o nome a ser procurado” e caso os dados realmente estivessem errados, a mensagem aparecia mas, com um loop infinito da mesma. Depois de pesquisas, descobrimos o uso de “flags”, que são variáveis booleanas.

```
165 void buscarMusico(tipoLista *lista){
166     char nome [100];
167     int flag = 0;
168 
169     tipoNo *aux = lista->head;
170     if(lista->head == NULL){
171         printf("\nNENHUM ELEMENTO NA LISTA!\nPOR FAVOR, ESCOLHA A OPÇÃO 1 PARA CADASTRO\n");
172     }else{
173         printf("\n\nPor qual músico gostaria de procurar? ");
174         gets(nome);
175 
176         while(aux != NULL){
177             if (strcmp(aux->nomeDoMusico, nome) == 0){
178                 printf("------------------");
179                 printf("\nNome do instrumento: %s", aux->nomeDoInstrumento);
180                 printf("\nAno de compra: %d", aux->anoDeCompra);
181                 printf("\nPreço de compra: %1.f", aux->precoDeCompra);
182                 printf("\n");
183                 flag = 1;
184             }
185             aux = aux->proximo;
186         }
187         if(!flag){
188             printf("Dado não encontrado. Vereifique o nome a ser procurado.");
189         }
190         printf ("\nPressione qualquer tecla para continuar...\n");
191         getchar();
192     }
193 }
```

- **Excluir Elemento da Lista:**
  Comparar strings em funções anteriores e o usar uma variável booleana nos auxiliou muito nesta função que é a mais complicada de todas. Desenhamos muitas vezes a lista para entender como seria feita a remoção de um elemento sem que a conexão entre os nós fosse perdida. Após entendermos as ligações pelos desenhos feitos e pesquisas em fóruns, assistimos a um vídeo (clique aqui) que nos auxiliou muito na implementação dessa função. Tínhamos pensado em usar recursividade mas após tentativas falhas, optamos por uma maneira diferente.

```
195 void exlcuirInstrumento(tipoLista *lista){
196     char nomeInstrumento[50];
197     char nomeMusico[100];
198     int flag = 0;
199 
200     tipoNo *anterior, *atual;
201 
202     printf("\nDigite o nome do instrumento que deseja deletar: ");
203     gets(nomeInstrumento);
204     fflush(stdin);
205     printf("\nDe qual músico é esse instrumento? ");
206     gets(nomeMusico);
207     fflush(stdin);
208 
209     if(lista->head == NULL){
210         printf("\nLISTA SEM NENHUM INSTRUMENTO CADASTRADO!");
211     }else {
212        anterior = lista->head;
213        atual = lista->head;
214        while(atual != NULL){
215            if (strcmp(atual->nomeDoMusico, nomeMusico) == 0 && strcmp(atual->nomeDoInstrumento, nomeInstrumento)  == 0){
216                if(atual == lista->head){
217                    lista->head = lista->head->proximo;
218                    free(atual);
219                    printf("\nINSTRUMENTO DELETADO COM SUCESSO!");
220                    flag = 1;
221                    break;
222                }else{
223                    if(atual == lista->fim){
224                        lista->fim = anterior;
225                    }
226                    anterior->proximo = atual->proximo;
227                    free(atual);
228                    printf("\nINSTRUMENTO DELETADO COM SUCESSO!");
229                    flag=1;
230                    break;
231                }
232            }else{
233                anterior = atual;
234                atual = atual->proximo;
235            }
236        }
237        if(!flag){
238            printf("\nNENHUM DADO ENCONTRADO!");
239        }
240         printf ("\nPressione qualquer tecla para continuar...\n");
241         getchar();
242     }
243 }
244 
```

6. *Fontes Consultadas :*

- Link
- Link
- Link
- Link
- Link
- Link
- Link
- Link
- Link
- Link
- Link
- Link
- Link
- Link
- Link
- Link
- Link
- Link

7. ***Dificuldades Encontradas:***

- Primeiramente, tínhamos o desenho da lista em nossa mente mas ter que estruturar todo o pensamento em funções foi um pouco complicado, uma vez que, não tínhamos até então um conhecimento sólido sobre o uso das mesmas. Por isso, foi um pouco embaraçoso esse processo.

- Houve também uma demora no entendimento dos ponteiros, principalmente na função de remoção, visto que em alguns casos a ligação entre os nós era perdida após remoção de algum item. Após bastante estudo, conseguimos chegar a uma função que funciona corretamente, com todas as funcionalidades.

- Houve também algumas dificuldades na formatação das informações, comandos, funções e desempenho em diferentes compiladores.

Em resumo, foi um trabalho complexo devido a absorção de conhecimento sólido, porém agora olhamos para o código e suas funções e conseguimos compreender o funcionamento de cada um deles.

//