Universidade Federal do Espírito Santo - Departamento de Informática

1° Trabalho Prático

PlayED

Estruturas de Dados (INF15974) - 2024/1

20 de maio de 2024

Alunas: Aline Mendonça Mayerhofer Manhães e Marcela Carpenter da Paixão

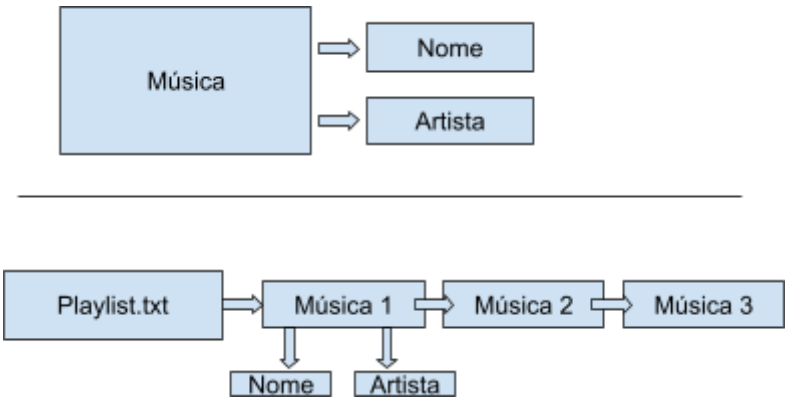
Introdução: O PlayED consiste em criar uma lista de pessoas, no arquivo “amizade.txt”, e correlacionar pessoas através da relação de amizade, também lendo pelo arquivo “amizade.txt”. A amizade é uma relação de reciprocidade, isto é, se Aline é amiga de Marcela, então Marcela também é amiga de Aline.

Além de uma lista de amigos, cada pessoa também deve ter uma lista de playlist, e uma playlist pode ser definida como uma lista de músicas. As informações relacionadas ao número de playlists que cada pessoa tem e seus nomes podem ser encontradas no arquivo “playlists.txt”.

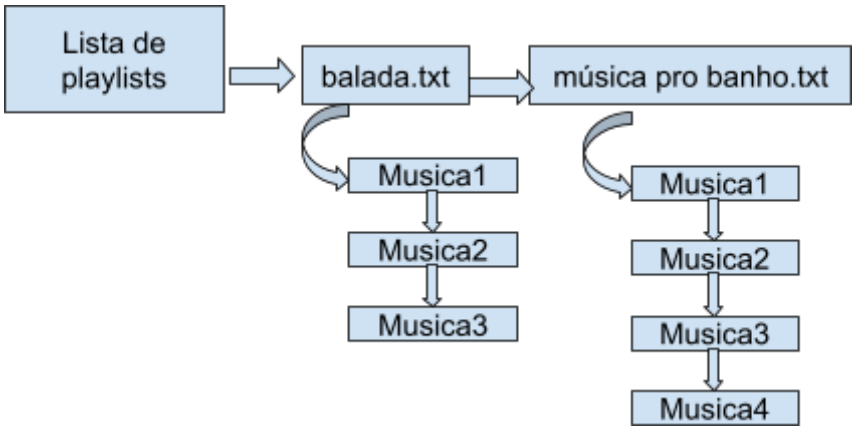
A partir disso, devemos ler cada arquivo de playlist informado e armazenar suas músicas dentro da pessoa correspondente. Por último, devemos implementar duas funcionalidades: criar novas playlists, organizadas por artista, e encontrar o número de músicas que cada relacionamento de amizade tem em comum.

Implementação: Os arquivos foram estruturados em:

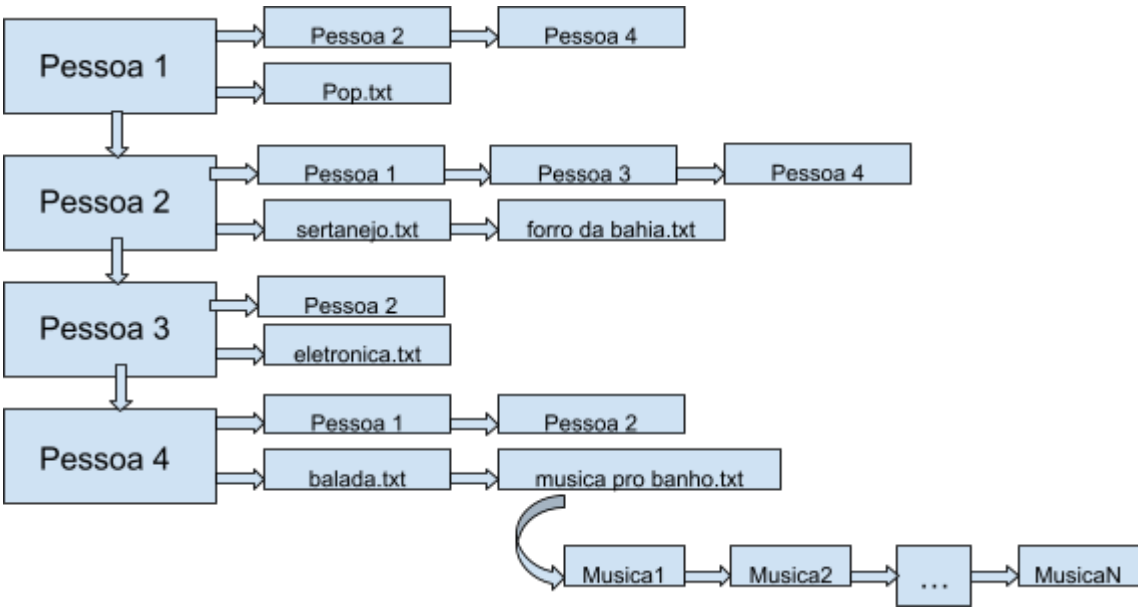
- *musicas.h/ musicas.c*: onde estão as músicas e a lista de músicas (que também pode ser chamada de playlist), além das suas funções. Cada música possui nome e artista, alocado dinamicamente. E uma playlist possui , além de um nome, músicas, que possuem nome e artista. A estrutura pode ser visualizada abaixo. Damos o exemplo da playlist de nome “Playlist.txt”, que possui as músicas “Música 1”, “Música 2” e “Música 3”.



- *playlists.h/ playlists.c*: onde estão as listas de playlists. Abaixo temos um exemplo da estrutura de lista de playlists, que contém playlists que por sua vez contém músicas.



- *pessoas.h/ pessoas.c*: onde estão pessoas e lista de pessoas, além de suas funções. Uma lista de pessoas pode representar um conjunto de amigos ou então apenas um conjunto de pessoas para controle da *main*, onde ficam todas as pessoas do programa. Cada pessoa possui nome, lista de amizades (ou lista de pessoas) e lista de playlists (ou lista de lista de músicas). Na imagem abaixo, representamos essa estrutura mencionada. As pessoas 1, 2, 3 e 4 estão em uma lista de pessoas (controlada pela *main*). Por sua vez, cada pessoa possui uma lista de amizades (que também é uma lista de pessoas) e uma lista de playlists (considerando que playlists são listas de músicas).



- *main.c*: onde temos controle das pessoas existentes no programa (através de uma lista), realizamos a leitura dos arquivos iniciais e chamamos as funções para o funcionamento do programa. Por último, liberamos a memória alocada.

- *Makefile*: ao rodar “make” no terminal executa a compilação através do comando do terminal “gcc *.c -o main”, que compila todos os .c, transformando em .o, e depois gera o executável main. Além disso, para executar o programa, o Makefile roda o executável através de “./main”.

As relação às funções do programa, as principais, apresentadas na *main*, foram:

- *LeArquivosDePlaylist*: essa função recebe uma lista de pessoas e o diretório dos arquivos como parâmetro. Para todas as pessoa da lista ela lê seus respectivos arquivos de playlist e insere seu conteúdo dentro da lista de playlists de cada um;

- *RelacionaAmigos*: recebe dois nomes e uma lista de pessoas como parâmetro, e compara esses nomes até encontrar quais pessoas eles são. Depois disso, a função insere um na lista de amigos do outro, uma vez que a amizade é recíproca;

- *OrganizaPessoaPorArtista*: reorganiza as playlists de cada pessoa da lista de pessoas recebida como parâmetro na função, de forma que, as

playlists originais da pessoa, com músicas de artistas/bandas diversos, são substituídas por uma playlist específica para cada artista/banda. As playlists antigas são excluídas e as novas, organizadas por artista, não possuem músicas repetidas;

- *GeraPlayedRefatorada*: gera o arquivo “played-refatorada.txt” listando, para cada pessoa da lista recebida como parâmetro, seu nome, a quantidade de playlists que essa pessoa passou a ter após a reorganização por artista/banda (função descrita acima), e o nome dessas playlists;

- *VerificaSimilaridades*: essa é a função responsável por verificar as similaridades existentes entre as músicas de cada pessoa e de seus amigos. Nela, foi usado um loop dentro de outro para percorrer a lista de pessoas, e, para cada pessoa, percorrer sua lista de amigos. Assim, com uso de funções auxiliares, cada playlist dessa pessoa é comparada com as playlists de seus amigos, ou seja, TODAS as músicas dela são comparadas com TODAS as músicas de TODOS os seus amigos. Ao final da função, é gerado um arquivo “similaridades.txt” que apresenta em cada linha um par de nomes de pessoas analisadas (dois amigos) e a quantidade de similaridades encontradas (sem ocorrer repetição de pares de nome, mesmo que em ordem inversa).

- *CriaNovosArquivosPessoa:* para cada pessoa da lista de pessoas recebida como parâmetro, gera os arquivos .txt das novas playlists, listando cada uma de suas músicas, com nome do artista e da música;

- *RealizaMergePlaylists*: essa função combina as playlists de música de uma pessoa com as de seus amigos, caso existam artistas/bandas similares nas playlists de ambos. No final, cada pessoa terá uma playlist atualizada que inclui as suas músicas originais somadas às músicas (de mesmo artista/banda) das listas de músicas de seus amigos, sem que ocorra repetição;

- *CriaArquivoMergePessoa*: para cada pessoa da lista de pessoas recebida como parâmetro, gera os arquivos .txt das novas playlists, atualizadas após o merge com as playlists de seus amigos, listando cada uma de suas músicas, com nome do artista e da música;

Além das listadas acima, a main também apresentou outras funções necessárias para gerenciar as listas e variáveis, como funções de inicialização, inserção de novas células à lista, e liberação da memória ao final do programa.

Conclusão: Nós duas achamos o trabalho muito interessante e divertido de ser feito. Acreditamos que evoluímos enquanto programadoras e fixamos o conteúdo aprendido em Estrutura de Dados até agora.

Onde encontramos maior dificuldade foi para fazer o arquivo played-refatorada.txt, ao analisar se uma playlist de determinado artista já existia, e se caso sim, se uma determinada música já estava dentro dela para não ter músicas repetidas em uma playlist. Também gastamos um tempo para pensar em como ler o nome de uma música e seu artista, já que o que separava essas duas strings era um hífen (“ - ”) e o artista também poderia ter hífen em seu nome. Assim, fizemos uma leitura que lê a linha toda ao mesmo tempo e depois é feita a separação do que é artista e nome de música ao encontrar um espaço e um hífen, nessa ordem.

Bibliografia: O trabalho foi feito com base nos conhecimentos adquiridos através da disciplina de Estrutura de Dados, ministrada pela professora Patrícia Dockhorn Costa. Não foram utilizados sites de pesquisa.