

Atividade Prática 04

Estruturas de Índices

Universidade Tecnológica Federal do Paraná (UTFPR), campus Apucarana
Curso de Engenharia de Computação
Disciplina de Estrutura de Dados 2 - EDCO4B
Prof. Dr. Rafael Gomes Mantovani

Instruções:

- Leia todas as instruções corretamente para poder desenvolver sua atividade/programa;
- Evite plágio (será verificado por meio de ferramentas automatizadas). Faça seu programa com os seus nomes de variáveis e lógica de solução. Plágios identificados anularão as atividades entregues de todos os envolvidos.
- Adicione comentários nos códigos explicando seu raciocínio e sua tomada de decisão. Porém, não exagere nos comentários, pois a própria estrutura do programa deve ser auto-explicativa.
- Salve sua atividade em um arquivo único, com todas as funções e procedimentos desenvolvidos. É esse **arquivo único** que deverá ser enviado ao professor.

1 Descrição da atividade

Depois de alguns semestres tendo aula com o professor M vocês perceberam que ele é uma pessoa bem normal. E como uma pessoa normal, muitas vezes seu humor varia ao longo dos dias: há dias mais desanimados e outros mais animados. Porém, uma estratégia para sempre tentar abstrair e ficar bem é ouvir música. É comum o professor M colocar um som de fundo nas suas aulas para que tanto ele, como os alunos, possam se distrair um pouco. E como todo viciado em música, professor M ficou curioso em descobrir as músicas mais tocadas em 2023 pelos usuários do Spotify. Felizmente, ele conseguiu encontrar essa informação com poucos cliques.

1.1 Dataset

Nesta atividade vocês irão manipular o dataset *Most Streamed Spotify Songs 2023*¹. Este dataset contém uma lista abrangente das músicas mais famosas de 2023 listadas no *Spotify*

¹<https://www.kaggle.com/datasets/nelgiriyeewithana/top-spotify-songs-2023/>

(total de 953 músicas). O conjunto de dados oferece uma variedade de recursos além do que normalmente está disponível em conjuntos de dados semelhantes. Ele fornece informações sobre os atributos, popularidade e presença de cada música em várias plataformas musicais. O conjunto de dados inclui informações como: **nome da faixa, nome do(s) artista(s), data de lançamento, listas de reprodução e paradas do *Spotify*, estatísticas de *streaming*, presença do *Apple Music*, presença do *Deezer*, paradas do *Shazam* e vários recursos de áudio**. A [Tabela 1](#) apresenta uma descrição detalhada de todos os campos existentes na descrição de cada música.

Tabela 1: Tabela de características disponíveis no arquivo “spotify-2023.csv”

Nro	Feature	Descrição
1	track_name	Nome da música
2	artist(s)_name	Nome do(s) artista(s) da música
3	artist_count	Número de artistas contribuindo na música
4	released_year	Ano em que a música foi lançada
5	released_month	Mês que a música foi lançada
6	released_day	Dia do mês que a música foi lançada
7	in_spotify_playlists	Número de playlists no <i>Spotify</i> que contém a música
8	in_spotify_charts	Presença e classificação da música na parada do <i>Spotify</i>
9	streams	Número total de reproduções no <i>Spotify</i>
10	in_apple_playlists	Número de playlists do <i>Apple Music</i> nas quais a música está incluída
11	in_apple_charts	Presença e classificação da música nas paradas musicais da <i>Apple</i>
12	in_deezer_playlists	Número de playlists do <i>Deezer</i> em que a música está incluída
13	in_deezer_charts	Presença e classificação da música nas paradas da <i>Deezer</i>
14	in_shazam_charts	Presença e classificação da música nas paradas do <i>Shazam</i>
15	bpm	Batidas por minuto, uma medida do andamento da música
16	key	Chave da música
17	mode	Modo da música (maior ou menor)
18	danceability_%	Porcentagem que indica quão adequada a música é para dançar
19	valence_%	Positividade do conteúdo musical da música
20	energy_%	Nível de energia percebido da música
21	acousticness_%	Quantidade de som acústico na música
22	instrumentalness_%	Quantidade de conteúdo instrumental na música
23	liveness_%	Presença de elementos de performance ao vivo
24	speechiness_%	Quantidade de palavras faladas na música

1.2 O que fazer?

Pensando em melhorar o seu processo de consulta às músicas, o professor M pediu para vocês desenvolverem **índices secundários** que permitam consultas dos valores existentes

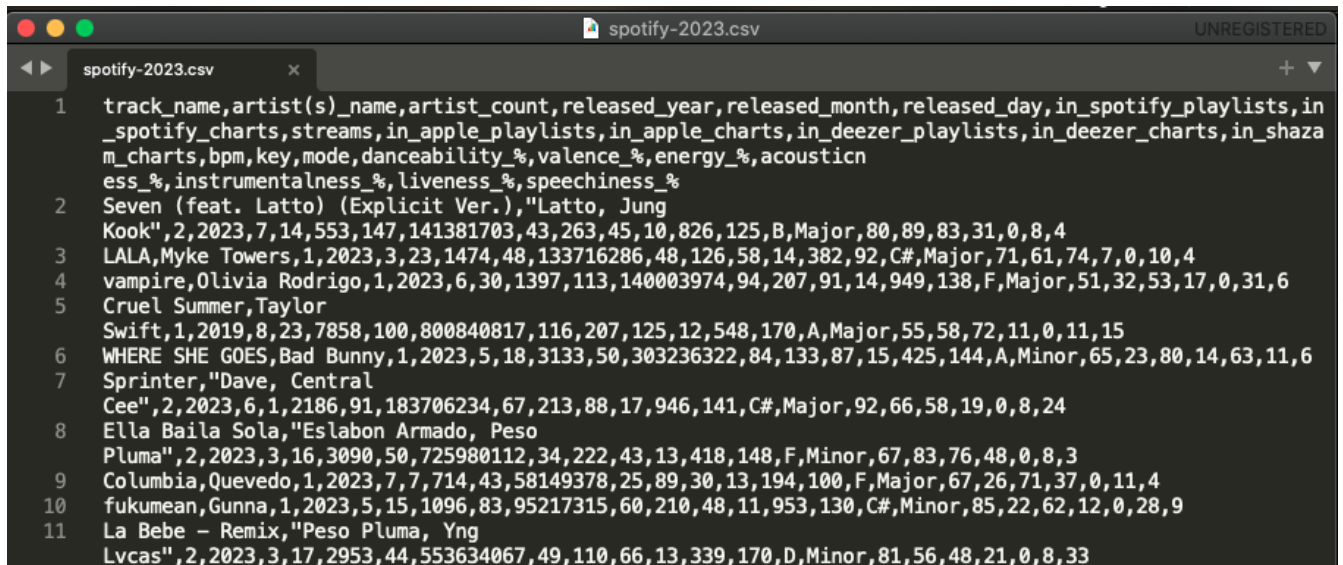
no arquivo. Por exemplo: retornar todas as músicas de um artista específico, retornar todas as músicas lançadas em 2023, ou ainda retornar todas as músicas que possuem um índice de dançabilidade (*danceability*) acima de 10%. Essa é sua nova **missão**: desenvolver um programa com índices secundários que satisfaça as consultas de músicas do arquivo do professor M.

2 Entradas do programa

O programa receberá **três** arquivos texto como parâmetros: um arquivo de dados com as músicas a serem manipuladas (“spotify-2023.csv”), um arquivo de consulta, e um arquivo de saída. Abaixo, iremos detalhar cada um deles.

2.1 Arquivo de dados

O primeiro parâmetro é o arquivo de dados, o mesmo cuja estrutura é apresentada na [Tabela 1](#). O armazenamento das músicas é feito em um arquivo de extensão “.csv” (*comma separated values*). Cada registro possui um tamanho fixo, porém os campos são de tamanhos variáveis. Cada campo é separado por uma vírgula (‘,’) e os registros finalizados por uma quebra de linha. Há um registro de cabeçalho (*header*) que contém os nomes dos campos (atributos) dos registros. A [Figura 1](#) mostra uma imagem do arquivo de dados aberto em um editor de textos comum. Além disso, uma sugestão de classe/estrutura para codificar um registro (música) é apresentada na [Figura 2](#).



```

1 track_name,artist(s)_name,artist_count,released_year,released_month,released_day,in_spotify_playlists,in
2 _spotify_charts,streams,in_apple_playlists,in_apple_charts,in_deezer_playlists,in_deezer_charts,in_shaza
3 m_charts,bpm,key,mode,danceability_%,valence_%,energy_%,acousticn
4 ess_%,instrumentalness_%,liveness_%,speechiness_%
5 Seven (feat. Latto) (Explicit Ver.),"Latto, Jung
6 Kook",2,2023,7,14,553,147,141381703,43,263,45,10,826,125,B,Major,80,89,83,31,0,8,4
7 LALA,Myke Towers,1,2023,3,23,1474,48,133716286,48,126,58,14,382,92,C#,Major,71,61,74,7,0,10,4
8 vampire,Olivia Rodrigo,1,2023,6,30,1397,113,140003974,94,207,91,14,949,138,F,Major,51,32,53,17,0,31,6
9 Cruel Summer,Taylor
10 Swift,1,2019,8,23,7858,100,800840817,116,207,125,12,548,170,A,Major,55,58,72,11,0,11,15
11 WHERE SHE GOES,Bad Bunny,1,2023,5,18,3133,50,303236322,84,133,87,15,425,144,A,Minor,65,23,80,14,63,11,6
12 Sprinter,"Dave, Central
13 Cee",2,2023,6,1,2186,91,183706234,67,213,88,17,946,141,C#,Major,92,66,58,19,0,8,24
14 Ella Baila Sola,"Eslabon Armado, Peso
15 Pluma",2,2023,3,16,3090,50,725980112,34,222,43,13,418,148,F,Minor,67,83,76,48,0,8,3
16 Columbia,Quevedo,1,2023,7,7,714,43,58149378,25,89,30,13,194,100,F,Major,67,26,71,37,0,11,4
17 fukumean,Gunna,1,2023,5,15,1096,83,95217315,60,210,48,11,953,130,C#,Minor,85,22,62,12,0,28,9
18 La Bebe - Remix,"Peso Pluma, Yng
19 Lvcas",2,2023,3,17,2953,44,553634067,49,110,66,13,339,170,D,Minor,81,56,48,21,0,8,33

```

Figura 1: Dados de entrada do arquivo “spotify-2023.csv”

```

# objeto para armazenamento de uma música
class Music:
    __track_name      # nome da música
    __artists_name    # nome do artista ou artistas
    __artist_count    # quantidade de artistas na música
    __released_year   # ano de lançamento da música
    __released_month  # mês de lançamento da música
    __released_day    # dia de lançamento da música

    # ... TODO: adicionar os demais campos, de acordo com Tabela 1

```

Figura 2: Estrutura de música a ser manipulada na aplicação.

2.2 Arquivo de consulta (*query*)

Um arquivo texto contendo as informações necessárias para realizar a busca na estrutura de índices. Esse arquivo pode conter dois tipos de buscas:

- **consulta simples:** baseada em apenas um único tipo de índice. A [Figura 3](#) mostra um exemplo de consulta simples. Nesse caso, o arquivo possui duas linhas com conteúdo: na primeira linha o nome do campo que irá definir o índice secundário, e na segunda linha uma *string* com o conteúdo a ser buscado. No exemplo em questão, queremos criar um índice secundário com as informações dos nomes dos artistas e retornar todas aquelas que são da “Taylor Swift”.
- **consulta com operadores booleanos:** dois ou mais operadores booleanos concatenando diferentes índices para uma consulta mais robusta. Um exemplo de consulta booleana é apresentada na [Figura 4](#). Nesse caso, a primeira linha contém os nomes dos campos usados para criar vários índices secundários, separados por operadores booleanos. Os dois operadores válidos são: & que corresponde ao operador AND, e || que corresponde ao operador OU. Na segunda linha são passados os valores correspondentes de consulta em cada índice, separados por vírgulas. Na consulta especificada o usuário irá criar dois índices secundários distintos: um deles baseado no nome dos artistas (*artist_name*), e o segundo baseado no ano de lançamento das músicas (*released_year*). A consulta resultante irá levar em consideração a interseção dos resultados das buscas por Talor Swift no nome dos artistas, e 2020 no ano de lançamento. A ordem de procedência dos operadores é da esquerda para a direita, conforme forem definidos na busca.

É importante salientar que podem ser criados índices para todas as colunas disponíveis no dataset. Caso qualquer arquivo de consulta possua uma *string* indicando a criação de tipo de índice de campos inválido, erros devem ser capturados e indicados no arquivo de saída.

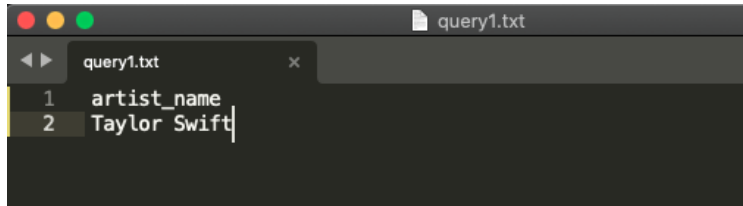


Figura 3: Exemplo de arquivo de consulta simples.

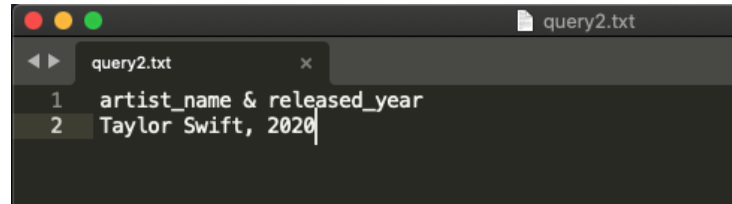


Figura 4: Exemplo de arquivo de consulta com operador booleano.

2.3 Arquivo de saída

Um arquivo texto contendo a busca realizada pelo programa após criar o correspondente índice secundário. A Figura 5 mostra um exemplo do arquivo de saída onde são retornadas todas as músicas do gênero pop contidas no arquivo de dados (base de dados) para o arquivo de busca “query1.txt” da Figura 3. No arquivo de saída aparecem apenas 16 dos 38 registros retornados quando “Taylor Swift” é o valor de procura em um índice secundário baseado no(s) nome(s) do(s) artista(s) das músicas contidas no dataset. Já a Figura 6 mostra a saída quando o arquivo de busca “ query2.txt” da Figura 4 é executado. Tenham em mente que as buscas podem retornar zero ou mais resultados. Além disso, vale destacar que, em caso de consultas inválidas, arquivos corrompidos ou inválidos, o arquivos de saída devem conter uma mensagem indicando o erro capturado.

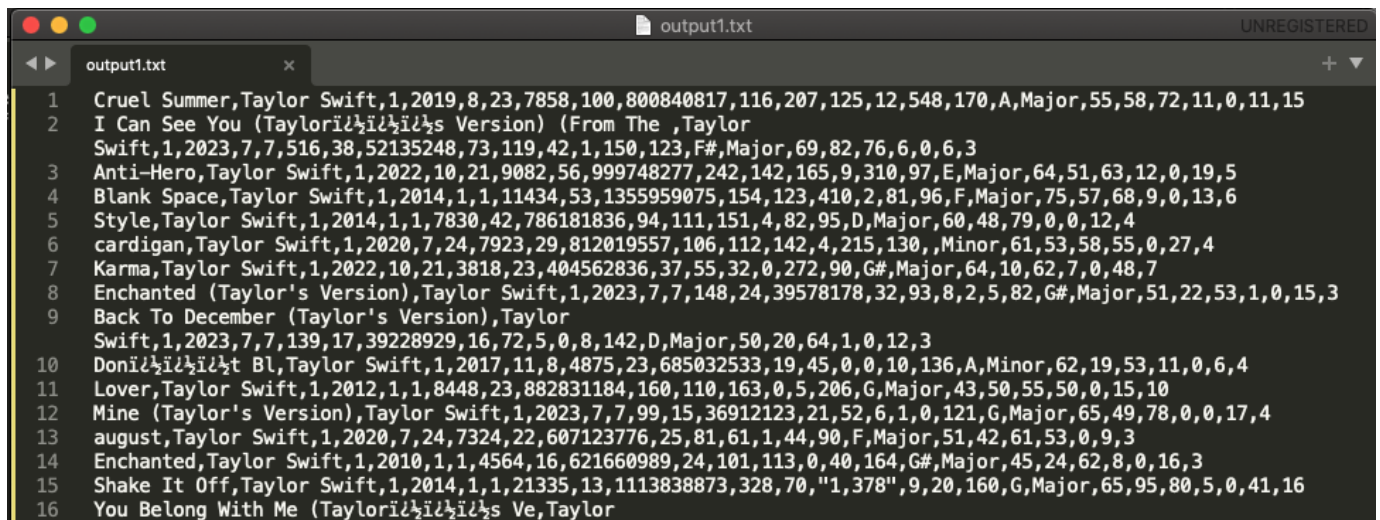
3 Rodando o programa

Para rodar o programa por linha de comando você deve manipular os argumentos em *Python* por meio do parâmetro **sys.argv**. Para executar o programa por linha de comando, deve-se obedecer o seguinte padrão:

```
[nome do programa] [arquivo de dados] [arquivo de entrada] [arquivo de saída]
```

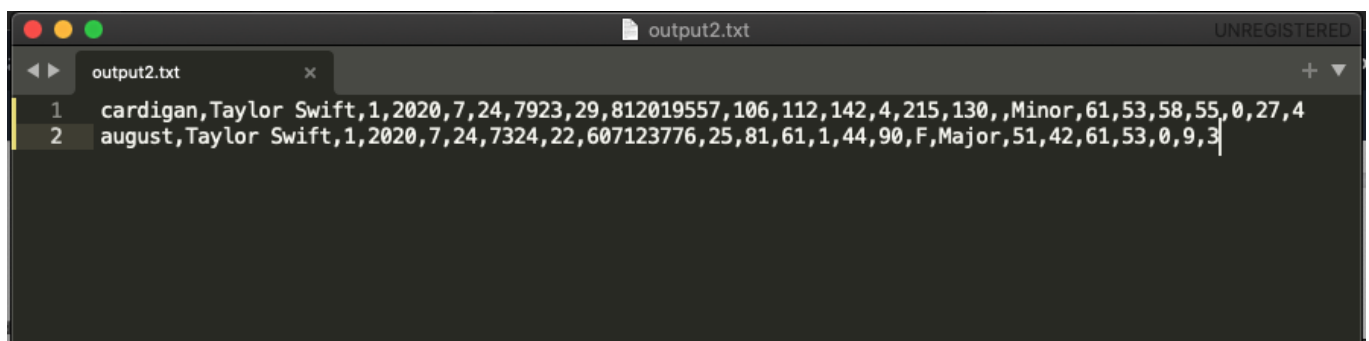
Exemplo de execução de um programa chamado `indice.py`:

```
python3 indice.py spotify-2023.csv query01.txt saida01.txt  
python3 indice.py spotify-2023.csv query02.txt saida02.txt
```



```
1 Cruel Summer,Taylor Swift,1,2019,8,23,7858,100,800840817,116,207,125,12,548,170,A,Major,55,58,72,11,0,11,15
2 I Can See You (Taylor's Version) (From The ,Taylor
Swift,1,2023,7,7,516,38,52135248,73,119,42,1,150,123,F#,Major,69,82,76,6,0,6,3
3 Anti-Hero,Taylor Swift,1,2022,10,21,9082,56,999748277,242,142,165,9,310,97,E,Major,64,51,63,12,0,19,5
4 Blank Space,Taylor Swift,1,2014,1,1,11434,53,1355959075,154,123,410,2,81,96,F,Major,75,57,68,9,0,13,6
5 Style,Taylor Swift,1,2014,1,1,7830,42,786181836,94,111,151,4,82,95,D,Major,60,48,79,0,0,12,4
6 cardigan,Taylor Swift,1,2020,7,24,7923,29,812019557,106,112,142,4,215,130,,Minor,61,53,58,55,0,27,4
7 Karma,Taylor Swift,1,2022,10,21,3818,23,404562836,37,55,32,0,272,90,G#,Major,64,10,62,7,0,48,7
8 Enchanted (Taylor's Version),Taylor Swift,1,2023,7,7,148,24,39578178,32,93,8,2,5,82,G#,Major,51,22,53,1,0,15,3
9 Back To December (Taylor's Version),Taylor
Swift,1,2023,7,7,139,17,39228929,16,72,5,0,8,142,D,Major,50,20,64,1,0,12,3
10 Don't Blame Me,Taylor Swift,1,2017,11,8,4875,23,685032533,19,45,0,0,10,136,A,Minor,62,19,53,11,0,6,4
11 Lover,Taylor Swift,1,2012,1,1,8448,23,882831184,160,110,163,0,5,206,G,Major,43,50,55,50,0,15,10
12 Mine (Taylor's Version),Taylor Swift,1,2023,7,7,99,15,36912123,21,52,6,1,0,121,G,Major,65,49,78,0,0,17,4
13 august,Taylor Swift,1,2020,7,24,7324,22,607123776,25,81,61,1,44,90,F,Major,51,42,61,53,0,9,3
14 Enchanted,Taylor Swift,1,2010,1,1,4564,16,621660989,24,101,113,0,40,164,G#,Major,45,24,62,8,0,16,3
15 Shake It Off,Taylor Swift,1,2014,1,1,21335,13,1113838873,328,70,"1,378",9,20,160,G,Major,65,95,80,5,0,41,16
16 You Belong With Me (Taylor's Version),Taylor
```

Figura 5: Saída correspondente gerada pelo programa para a *query* apresentada na Figura 3.



```
1 cardigan,Taylor Swift,1,2020,7,24,7923,29,812019557,106,112,142,4,215,130,,Minor,61,53,58,55,0,27,4
2 august,Taylor Swift,1,2020,7,24,7324,22,607123776,25,81,61,1,44,90,F,Major,51,42,61,53,0,9,3
```

Figura 6: Saída correspondente gerada pelo programa para a *query* apresentada na Figura 4.

4 Orientações gerais

Além da funcionamento da estrutura de índice secundário desejada, implementar também o controle de erros para lidar com exceções que possam ocorrer, como por exemplo:

- problemas nas aberturas dos arquivos de entrada e saída;
- arquivos de entrada vazio (sem informação);
- arquivos de entrada fora do padrão esperado (opções inválidas para consulta);
- número de argumentos inválido na linha de comando (mais ou menos valores);
- etc.

Opcionalmente, para acompanhamento do desenvolvimento, pode-se criar um repositório individual no [github](#).

5 Critérios de correção

A nota na atividade será contabilizada levando-se em consideração alguns critérios:

1. pontualidade na entrega;
2. não existir plágio;
3. completude da implementação (tudo foi feito);
4. o código executa;
5. uso de `sys.arg` para controle dos arquivos de teste;
6. implementar a leitura dos dados de entrada via arquivo texto (musicas);
7. implementação correta das estruturas necessárias (campos, registros e sua manipulação, ordenação das chaves, estrutura de índice);
8. legibilidade do código (identação, comentários nos blocos mais críticos);
9. implementação dos controles de erros (arquivos de entrada inválidos, e erros no programa principal);
10. controle de memória: chamar o destrutor e desalocar a memória de tudo se usar estruturas dinâmicas, fechar os arquivos, etc;
11. executar corretamente os casos de teste.

Em cada um desses critérios, haverá uma nota intermediária valorada por meio de conceitos:

- **Sim** - se a implementação entregue cumprir o que se esperava daquele critério;
- **Parcial** - se satisfizer parcialmente o tópico;
- e **Não** se o critério não foi atendido.

6 Padrão de nomenclatura

Ao elaborar seu programa, crie um único arquivo fonte (.py) seguindo o padrão de nome especificado:

ED2-AT04-IndiceSecundario-<NOME>.py

Exemplo:

A entrega da atividade será via Moodle: o link será disponibilizado na página da disciplina.

7 Links úteis

- Arquivos em Python:
 - <https://www.geeksforgeeks.org/reading-writing-text-files-python/>
 - https://www.w3schools.com/python/python_file_open.asp
 - <https://www.pythontutorial.net/python-basics/python-read-text-file/>
- Argumentos de Linha de comando no Python:
 - https://www.tutorialspoint.com/python3/python_command_line_arguments.htm
 - <https://realpython.com/python-command-line-arguments/>
 - <http://devfuria.com.br/python/sys-argv/>
- *Most Streamed Spotify Songs 2023*:
 - <https://www.kaggle.com/datasets/nelgiriyeewithana/top-spotify-songs-2023>

Referências

- [1] Michael J. Folk; Bill Zoellick; Greg Riccardi. File Structures, 3rd edition, Addison-Wesley, 1997.
- [2] Thomas H. Cormen,; Ronald Rivest; Charles E. Leiserson; Clifford Stein. Algoritmos - Teoria e Prática - 3ª Ed. Elsevier - Campus, 2012.
- [3] Nivio Ziviani. Projeto de algoritmos com implementações: em Pascal e C. Pioneira, 1999.
- [4] Adam Drozdek. Estrutura De Dados e Algoritmos em C++. Cengage, 2010.