Atividade Prática 03
Keysorting

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

Sua implementação do programa de reuso de arquivos para o professor M foi um sucesso. Com a automatização do processo via operações de arquivos, foi possível manter os registros sempre atualizados. Mas ... vocês sabem como é vida de programar: entregou uma funcionalidade, sempre surge algo novo. O professor M já colocou a mente para funcionar a todo o vapor.

Como todo bom "computeiro", ele é um nerd convicto e gosta muito de quadrinhos e heróis em geral. De todas as HQs, quadrinhos, animes e outros materiais que ele já consumiu, ele mantém uma lista com as principais informações dos heróis que mais curte. Porém, essa lista não está organizada, pois ele sempre adiciona novos valores conforme seus gostos vão mudando ao longo dos anos. Vocês mesmos já devem ter visto essa lista em algum momento.

Bom, a ideia do professor M é fazer uma solução que ordene o arquivo com as informações dos heróis. Como vocês são alunos espertos, já ouviram nas aulas do professor que o método de ordenação de chaves (*Keysorting*) realiza a mágica necessária. O desafio é implementar o método, porém ele reaproveita grande parte dos conceitos que vocês já aprenderam em sala.

Sendo, assim, mãos a obra. Ajudem o professor M e implementem um programa que faça o *Keysorting* dos registros de herói e grave o resultado da operação em um novo arquivo ordenado.

2 Entradas do programa

O programa receberá **dois** arquivos texto como parâmetros: um de entrada e um de saída. Exemplos de arquivos manipulados pela aplicação podem ser vistos na Figura 1. Abaixo, iremos detalhar cada um deles.

2.1 Arquivo de entrada

Os arquivos de entrada são arquivos texto contendo os registros dos heróis anotados pelo professor M (Figura 1a). Durante a execução podem ser fornecidos **N** registros, esse número é variável. O arquivo de entrada é codificado usando **registros de tamanho fixo e campos de tamanho variável**. Há um registro de cabeçalho (*header*) que contém algumas informações importantes para a execução do programa. Essas informações estão sumarizadas na Tabela 1.

Tabela 1: Parâmetros contidos no cabeçalho do arquivo de entrada

Parâmetro	**Descrição**	**Opções Válidas**
SIZE	tamanho dos registros (bytes/caracteres)	136
TOP	índice da posição do topo da pilha lógica de reuso	-1
QTDE	quantidade de registros contidos no arquivo de entrada	[1, ...]
SORT	método de ordenação a ser usado pelo processo de *KeySorting*	{Q, M, H, I}
ORDER	o sentido da ordenação	{C, D}

No parâmetro SIZE, os registros terá valor fixo em 136. Ou seja, consumindo todos os caracteres possíveis ou não, cada registro (linha) vai ter 136 caracteres. O parâmetro TOP controla o reuso dos registros, depois de operações de leitura e escrita. Nessa atividade o valor será constante em -1, pois não iremos modificar os arquivos, apenas ordená-los. Na opção SORT, os valores possíveis são: Q - Quick Sort; H - Heap Sort; M - Merge Sort; e I - Insertion Sort. Valores diferentes destes indicam opções inválida para execução do programa. Já os valores possíveis para ORDER são: C para crescente ou D para decrescente. Além disso, cada registro contido em uma linha contém algumas informações sobre os heróis:

Considerem que na representação dos registros, os correspondentes campos estarão separados por delimitadores fixos. Use o caractere pipe (|) para separar campos de um mesmo registro, e um caractere especial de quebra de linha para identificar o fim de um registro. É importante ser capaz de obter a quantidade de registros do arquivo pela leitura do cabeçalho, e não percorrendo os registros do arquivo.

```
// Estrutura de Heroi
typedef struct {
```

input1.txt

```
SIZE=133 TOP=-1 QTDE=22 SORT=Q ORDER=C
12|James|Logan|Wolverine|fator cura|jean gray|Canada|alcoolatra
37|T'challa|King|Pantera Negra|roupa|sem a roupa|Wakanda|rei
62|Bruce|Banner|Hulk|forte para caralho|viuva negra|nova york|fisico
05|Virgil|Hawkins|Super Choque|choque|borracha|Dakota|estudante
27|Hal|Jordan|Lanterna Verde|Anel|Lanterna amarelo|Coast City|piloto
29|Diana|Prince|Mulher Maravilha|força para caralho|Steve turner|Themyscira|guerreira
81|Wade|Wilson|Deadpool|fator de cura|maluco|Canada|mercenario
68|Barry|Allen|Flash|correr|cadarço|Star city|policial forense
33|Steve|Rogers|Capitão América|super soldado|freezer|Nova York|soldado
72|Bruce|Wayne|Batman|rico|pobre|Gothan city|magnata
66|Doug|Funny|Homem Codorna|Cinto na cabeça|Paty Maionese|States|estudante
14|Bros|Mario|Super Mario|cogumelo|princesa|No cano|encanador
51|Bros|Luigi|Mario Verde|pular mais alto|fantasma|No cano|ser irmão do mario
46|Marc|Spector|Cavaleiro da Lua|traje|esquizofrenia|Egito|mercenário
79|Charles|Xavier|Professor X|mega mente|pernas|mansão maromba|psicólogo
54|Shark|Boy|Shark Boy|Shark|boy|Oceano|pescador
07|Lava|Girl|Lava Girl|Lava|Agua|Vulcao|bombeira
48|Link|Link|Zelda|Triforce|Ganondorf|Hyrule|duende
64|Ernie|Sereia|Homem Sereia|Sereia|Homem|Fenda do biquini|aposentado
94|Mendigo|Dos Mares|Mendigo do Mares|Mendigo|Riqueza|Mares|Lixeiro
96|Lloyd|Garmadon|Green Ninja| Verde|Lord Garmadon|Ninjago|Ninja
40|Mestre|Dos Magos|Mestre dos Magos|Sumir|Aparecer|O Reino|Guia
```

(a) Exemplo de arquivo de entrada.

output1.txt

```
SIZE=133 TOP=-1 QTDE=22 SORT=Q ORDER=C
05|Virgil|Hawkins|Super Choque|choque|borracha|Dakota|estudante
07|Lava|Girl|Lava Girl|Lava|Agua|Vulcao|bombeira
12|James|Logan|Wolverine|fator cura|jean gray|Canada|alcoolatra
14|Bros|Mario|Super Mario|cogumelo|princesa|No cano|encanador
27|Hal|Jordan|Lanterna Verde|Anel|Lanterna amarelo|Coast City|piloto
29|Diana|Prince|Mulher Maravilha|força para caralho|Steve turner|Themyscira|guerreira
33|Steve|Rogers|Capitão América|super soldado|freezer|Nova York|soldado
37|T'challa|King|Pantera Negra|roupa|sem a roupa|Wakanda|rei
40|Mestre|Dos Magos|Mestre dos Magos|Sumir|Aparecer|O Reino|Guia
46|Marc|Spector|Cavaleiro da Lua|traje|esquizofrenia|Egito|mercenário
48|Link|Link|Zelda|Triforce|Ganondorf|Hyrule|duende
51|Bros|Luigi|Mario Verde|pular mais alto|fantasma|No cano|ser irmão do mario
54|Shark|Boy|Shark Boy|Shark|boy|Oceano|pescador
62|Bruce|Banner|Hulk|forte para caralho|viuva negra|nova york|fisico
64|Ernie|Sereia|Homem Sereia|Sereia|Homem|Fenda do biquini|aposentado
66|Doug|Funny|Homem Codorna|Cinto na cabeça|Paty Maionese|States|estudante
68|Barry|Allen|Flash|correr|cadarço|Star city|policial forense
72|Bruce|Wayne|Batman|rico|pobre|Gothan city|magnata
79|Charles|Xavier|Professor X|mega mente|pernas|mansão maromba|psicólogo
81|Wade|Wilson|Deadpool|fator de cura|maluco|Canada|mercenario
94|Mendigo|Dos Mares|Mendigo do Mares|Mendigo|Riqueza|Mares|Lixeiro
96|Lloyd|Garmadon|Green Ninja| Verde|Lord Garmadon|Ninjago|Ninja
```

(b) Exemplo de arquivo de saída **depois** do *Keysorting*.

Figura 1: Valores de entrada e correspondentes arquivos de saída gerado pelo programa.

```
char chave[3];          // chave numerica pode ter até 3 digitos
char primeiroNome[16];  // primero nome do heroi
char sobrenome[16];     // seu sobrenome
```

```
    char nomeHeroi[16];      // nome do Heroi, alias
    char poder[16];          // poder
    char fraqueza[21];       // fraqueza
    char cidade[21];         // cidade que defende
    char profissao[21];      // profissao da sua identidade secreta
} Heroi;
```

2.2 Arquivo de saída

Um arquivo texto contendo o estado resultante do programa após realizar a ordenação das chaves do arquivo de entrada. Se o cabeçalho do arquivo possuir ORDER=C, os registros devem ser apresentados na forme crescente. Caso contrário, ORDER=D, apresentar os registros na forma descrente das chaves. A Figura 1b) mostra um exemplo do arquivo de saída.

3 Rodando o programa

Para rodar o programa por linha de comando, manipular os argumentos **argc** e **argv** da função `main`. Para executar o programa por linha de comando, deve-se obedecer o seguinte padrão:

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

Exemplo de execução de um programa chamado `keysorting.c`:

```
./keysorting entrada1.txt saida1.txt
```

4 Orientações gerais

Além da funcionalidade 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 uso no cabeçalho);
- 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 compila e executa;
5. uso de `argc` e `argv` para controle dos arquivos de teste;
6. implementar a leitura dos dados de entrada via arquivo texto;
7. implementação correta das estruturas necessárias (campos, registros e sua manipulação, ordenação das chaves);
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 (.c) seguindo o padrão de nome especificado:

```
ED2-<ANO>-<SEMESTRE>-AT03-Keysorting-<NOME>.c
```

Exemplo:

```
ED2-2022-1-AT03-Keysorting-RafaelMantovani.c
```

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

7 Links úteis

- Arquivos em C:
  - https://www.inf.pucrs.br/~pinho/LaproI/Arquivos/Arquivos.htm
  - https://www.geeksforgeeks.org/basics-file-handling-c/
  - https://www.programiz.com/c-programming/c-file-input-output
- 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 em C(argc e argv):
  - https://www.tutorialspoint.com/cprogramming/c_command_line_arguments.htm
  - http://linguagemc.com.br/argumentos-em-linha-de-comando/
  - http://www.univasf.edu.br/~marcelo.linder/arquivos_pc/aulas/aula19.pdf
  - http://www.inf.ufpr.br/cursos/ci067/Docs/NotasAula/notas-31_Argumentos_linha_comando.html
  - http://www.dca.fee.unicamp.br/cursos/EA876/apostila/HTML/node145.html
- 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/

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.