BCC202 – Estruturas de Dados I (2019-02)

Departamento de Computação - Universidade Federal de Ouro Preto - MG

Trabalho Prático 02

Data de entrega: 08/11/2019 até 17:00.
Atenção: *Trabalho individual.*

Procedimento para a entrega:.

1. Para cada questão, implemente sua solução.
2. Para testar suas soluções, implemente um único método *main()*, que poderá conter, por exemplo, um *Menu* de interações e possibilidades do usuário especificar os dados de entrada.
3. Especifique o *Makefile* com as instruções necessárias para compilação e execução do seu código, sendo que o *Makefile* deve conter também o redirecionamento da entrada e da saída (e.g., **`./prog.exe < input.txt > output.txt`**.)
4. Compacte em um único diretório o seu código fonte juntamente com o *Makefile*, o arquivo de entrada e o arquivo de saída usados para testes (e.g., *input.txt, output.txt*).
5. Faça a entrega do arquivo compactado, obrigatoriamente em formato *.zip*, no *RunCodes*, na tarefa correspondente.

- **Não utilize caracteres acentuados ou especiais para nomes de pastas, arquivos e na especificação de comentários no código**.
- **Especifique funções para modularizar seu código.**

- Bom trabalho!

Questão 01 (4 pontos)

A representação de informação por um ponteiro nos permite simular listas, pilhas e filas heterogêneas, isto é, as informações armazenadas podem diferir de nó para nó. Como exemplo, vamos considerar uma aplicação que necessite manipular pilhas de objetos geométricos planos para cálculo de áreas. Para simplificar, vamos considerar que os objetos podem ser apenas quadrados, losangos ou trapézios. Nesse contexto, deve ser definido um tipo para cada objeto representado (*i.e.,* quadrado, losango ou trapézio). O nó da pilha, por sua vez, deve então ser composto por três campos:

- um identificador de qual objeto está armazenado no nó;
- um ponteiro para a estrutura que contém a informação;
- um ponteiro para o próximo nó da pilha.

É importante salientar que, a rigor, a pilha é homogênea, no sentido de que todos os nós contêm os mesmos campos. O ponteiro para a informação deve ser do tipo genérico, pois não sabemos a princípio para qual estrutura ele irá apontar: pode apontar para um quadrado, um losango ou um trapézio. Um ponteiro genérico em C é representado pelo tipo **void ***. Uma variável do tipo ponteiro genérico pode representar qualquer endereço de memória, independente da informação de fato armazenada no espaço. No entanto, de posse de um ponteiro genérico, não podemos acessar a memória por ele apontada, já que não sabemos o tipo de informação armazenada. Por esta razão, o nó de uma pilha genérica deve guardar explicitamente um identificador do tipo de objeto de fato armazenado. Consultando esse identificador, podemos converter, utilizando uma variável auxiliar e ***casting***, o ponteiro genérico no ponteiro específico para o objeto em questão e, então, acessar campos do objeto. Como identificador de tipo, podemos usar uma enumeração (pesquise pelo conceito de ***enum*** em C). Assim, na criação de nós, armazenamos o identificador de tipo correspondente ao objeto a ser representado.

Considerando os conceitos acima expostos, implemente uma pilha dos objetos geométricos mencionados utilizando ***enum***. Sua pilha deve contemplar as seguintes operações.

- Criar pilha vazia.
- Inserir um elemento na pilha (*push)*.
- Remover um elemento da pilha (*pop*).
- Verificar se a pillha está vazia.
- Liberar a estrutura de pilha.
- Calcular a área de todos os objetos geométricos armazenados na pilha.

Implemente também as operações para criar, ler e atualizar cada uma dos objetos geométricos planos mencionados.

Questão 02 (3 pontos)

Sobre Campeonato de Futebol:

a) Escreva um programa que, dados os resultados de um campeonato de futebol, imprime a classificação correspondente seguindo o formato especificado abaixo. Vitória, empate e derrota valem três, um e zero pontos, respectivamente. O critério de classificação é o número de pontos marcados, seguido pelo saldo de gols (gols marcados - gols sofridos) e o número de gols marcados. Quando mais de um time possui o mesmo número de pontos, o mesmo saldo de gols e o mesmo número de gols marcados, considera-se que esses times ocupam a mesma posição na tabela de classificação.

b) Qual a complexidade do seu algortimo no melhor e pior caso? Justifica sua resposta.

Especificação da Entrada

A entrada consiste de uma série de testes. Cada teste começa com uma linha contendo dois inteiros positivos $1 \leq T \leq 28$ e $G \geq 0$. T é o número de times e G é o número de jogos disputados. Seguem T linhas, cada uma contendo o nome de um time. Nomes de times possuem até 15 caracteres e podem conter somente letras e caracteres de travessão ('-'). Por fim, seguem G linhas contendo o resultado de cada jogo. Os jogos são mostrados no seguinte formato: nome do time da casa, número de gols marcados pelo time da casa, um travessão, número de gols marcados pelo time visitante e nome do time visitante. A entrada termina com um caso de teste onde T = G = 0, o qual não deve ser processado.

Especificação da Saída

O programa deve imprimir as tabelas de classificação correspondentes a cada teste de entrada separadas por uma linha em branco. Em cada tabela, os times aparecem em ordem de classificação ou alfabeticamente, quando eles possuem a mesma posição. As estatísticas de cada time são mostradas em uma única linha contendo: posição do time, número de pontos, número de jogos disputados, número de gols marcados, número de gols sofridos, saldo de gols e porcentagem de pontos ganhos, quando disponível. Note que se vários times estão empatados, somente a posição do primeiro é impressa. Imprima uma linha em branco entre duas saídas. Campos devem ser formatados e alinhados como mostrado no exemplo de saída;

Exemplo

Entrada:

6 10
tA
tB
tC
td
tE
tF
tA 1 - 1 tB
tC 0 - 0 td
tE 0 - 0 tA
tC 0 - 0 tB
td 0 - 0 tE

tA 0 - 0 tC
tB 0 - 0 tE
td 0 - 0 tA
tE 0 - 0 tC
tB 0 - 0 td
2 2
Botafogo
Flamengo
Botafogo 3 - 2 Flamengo
Flamengo 2 - 3 Botafogo
5 10
tA
tB
tC
tD
tE
tA 0 - 0 tB
tC 0 - 0 tD
tE 0 - 0 tA
tC 0 - 0 tB
tD 0 - 0 tE
tA 0 - 0 tC
tB 0 - 0 tE
tD 0 - 0 tA
tE 0 - 0 tC tB 0 - 0 tD
3 2
Quinze-Novembro
Flamengo
Santo-Andre
Quinze-Novembro 6 - 0 Flamengo
Flamengo 0 - 2 Santo-Andre
0 0

Saída

Saída

```
 1.              tA   4   4   1   1   0  33.33
                 tB   4   4   1   1   0  33.33
 3.              tC   4   4   0   0   0  33.33
                 td   4   4   0   0   0  33.33
                 tE   4   4   0   0   0  33.33
 6.              tF   0   0   0   0   0    N/A

 1.        Botafogo   6   2   6   4   2 100.00
 2.        Flamengo   0   2   4   6  -2   0.00

 1.              tA   4   4   0   0   0  33.33
                 tB   4   4   0   0   0  33.33
                 tC   4   4   0   0   0  33.33
                 tD   4   4   0   0   0  33.33
                 tE   4   4   0   0   0  33.33

 1. Quinze-Novembro   3   1   6   0   6 100.00
 2.     Santo-Andre   3   1   2   0   2 100.00
 3.        Flamengo   0   2   0   8  -8   0.00
```

Questão 03 (03 pontos)

Existem elementos em sistemas operacionais que gerenciam a ordem em que os programas devem ser executados. Por exemplo, em um sistema de computação de tempo compartilhado (“time-shared”) mantem um conjunto de processos em uma fila, esperando para serem executados. Para simplificar, considere que cada processo pode ser

representado pelo momento de inserção na fila, um identificador e uma descrição do comando a ser executado. Lembrando que FILA segue o protocolo FIFO (*First in First Out*). Escreva um programa que seja capaz de ler uma série de solicitações para:

- Incluir um novo processo na fila, atualizando seu momento de inclusão;
- Retirar da fila o processo com o maior tempo de espera;
- Imprimir o conteúdo da fila em determinado momento, exibindo o tempo de espera, a descrição do comando e o identificador de cada processo.

Referência Bibliográfica

- Exercícios parcialmente extraídos das seguintes fontes:

  - Introdução a Estrutura de Dados: Com técnicas de programação em C - Waldemar Celes, Renato Cerqueira, José Lucas Rangel. Editora Campus. Elsevier 2016 (Segunda Edição).

  - SPOJ Br (Sphere Online Judge Brasil): https://br.spoj.com/

- Material complementar sobre listas/pilhas/filas "heterogêneas":

  `http://www.decom.ufop.br/anascimento/ensino/bcc202/aulas-teoricas/`