DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
MATA40 – Estruturas de Dados e Algoritmos I – 2019.2
Prof. Roberto Freitas Parente
roberto.parente@ufba.br

Trabalho 01
Versão: 9 de agosto de 2019

1 Instruções:

1. O trabalho é para ser feita em até **4 (quatro) pessoas**. O nome dos integrantes devem estar no cabeçalho comentado no arquivo enviado.
2. Será aplicado detector de plágio! Uma vez detectado plágio, a **nota da disciplina** será **zerada**.
3. Data de entrega: **19 de agosto d9 2018 (segunda-feira)** no sistema do moodle até as 23:50.
4. A entrega deve ser no **formato ZIP** com o seguinte conteúdo: Apenas 1 arquivo como código-fonte;
5. O código deve estar escrito em `Linguagem C` e as únicas bibliotecas permitidas são: `stdio.h`, `stdlib.h` e `string.h` caso contrário o trabalho será **invalidado**.
6. Para validação do código será utilizado um ambiente GNU/Linux com o compilador `GCC` (versão 7+) dado os seguintes parâmetros[1]:
   `gcc -Wall -Wextra -Werror -Wpedantic`
7. Para verificação de vazamento de memória e gerência dos ponteiros, será utilizado o software `Valgrind`[2].

2 Correção

Serão utilizados os seguintes critérios para correção:

2.1 Compilação

• Compilação com parâmetros requeridos.
(Não compilação com parâmetro corretos implica trabalho **invalidado**.).

2.2 Organização do código

• Funções separadas para cada uma das operações.
• Comunicação por retorno de função.

2.3 Entrada/Saída

• Cada entrada gera uma saída correta.

2.4 Gerência de memória

• Teste de alocação (verificar se a memória foi alocada).
• Remoção correta da memória alocada.
• Vazamentos de memória e controle de ponteiros.

Observação: A correção será em boa parte automatizada. Procure deixar o programa funcionando de acordo com as especificações.

[1] Leia mais sobre em: `https://gcc.gnu.org/onlinedocs/gcc-6.4.0/gcc/Warning-Options.html`.
[2] `http://valgrind.org`

3 Simulação de uma AGÊNCIA – PARTE I

Nosso objetivo é desenvolver uma simulação de uma agência de banco simplificada. Tal simulação gerencia o saldo dos clientes através de saques, transferências e depósitos realizados nos guichês da agência.

Para implementar da agência precisamos primeiro implementar nossas estruturas de dados. Na primeira etapa do desenvolvimento implementaremos os guichês de atendimento da simulação. Desta forma, devemos ser capazes de executar simulações simplificadas que serão definidas através de uma entrada e uma saída que será um relatório parcial informando a situação de cada guichê.

3.1 A simulação

Inicialmente para cada simulação, informamos que a agência contém 3 **(três)** guichês e recebe N clientes durante um dia. Cada cliente tem os seguintes dados: CPF do cliente atendido (CPF), operação, o valor e o CPF de outra pessoa envolvida na operação. Cada dado tem a seguinte forma:

- CPF cliente: inteiro;
- Operação: caractere 'D' para depósito, 'S' para saque, 'T' para transferência;
- Valor: inteiro;
- CPF terceiros: inteiro;

Para cada cliente recebido deve-se alocar um registro dinamicamente e enviá-lo a um guichê disponível utilizando a estratégia de round–robin iniciando do guichê 0 (zero), ou seja, se o cliente anterior foi alocado no guichê $k \mod 3$, então o presente cliente de ser alocado no guichê $k + 1 \mod 3$ ficando a seguinte ordem dos guichês selecionados: $0, 1, 2, 0, 1, 2, 0 \ldots, 2, 0, 1, 2, 0, 1, 2 \ldots$.

Cada cliente realiza apenas uma operação, mas não existe restrição da quantidade de vezes que um cliente é atendido. Cada guichê atende a um cliente por vez e armazena os documentos de cada cliente em uma pilha. Ao finalizar os atendimentos, deve ser impresso o relatório de cada guichê, onde será informado a quantidade de documentos armazenados no guichê seguido dos dados de cada documento.

3.2 Entrada

A entrada é constituída por uma única simulação enviada pela entrada padrão. A primeira linha contém um inteiro N, onde $1 \leq N \leq 10^{10}$, e N representa a quantidade de clientes envolvidos na simulação. As próximas N linhas deve conter os seguintes valores CPF, $CPFT$, O e V onde $1 \leq CPF \leq 10^{10}$ representa o CPF do cliente, $1 \leq CPFT \leq 10^{10}$ o CPF do terceiro envolvido, $O \in \{D, S, T\}$ o caractere referente a operação e $1 \leq V \leq 2^{16}$ a quantidade de dinheiro envolvido na transação.

Teremos a seguinte estrutura geral da entrada:

Entrada:
N
$CPF_1\ CPFT_1\ O_1\ V_1$
$CPF_2\ CPFT_2\ O_2\ V_2$
$\vdots$
$CPF_N\ CPFT_N\ O_N\ V_N$

3.3 Saída

A saída é constituída pelo relatório que devem ser enviado pela saída padrão seguindo a formatação descrita nessa seção.

Relatório Parcial

Antes do relatório parcial deve ser impressa a mensagem "`-:| RELATÓRIO PARCIAL |:-`" e seguido de um inteiro 3 referente a quantidade de guichês envolvidos na simulação, na linha seguinte deve ser impresso o texto "`Guiche` k`:` Q_k" onde k é o número do guichê e um inteiro Q_k referente a quantidade de operações empilhadas no guichê k e nas próximas P_k linhas informará os valores de cada operação na formatação "`[CPFc,CPFt,O,V]`" onde "CPFc" é referente ao CPF do cliente atendido, "CPFt" o cpf do terceiro envolvido na operação, "O" o caractere referente a operação e "V" o valor envolvido na operação. A ordem e impressão dos dados armazenados em cada guichê deve ser a mesma da remoção da pilha individual.

Saída final

Como descrito acima, segue os seguintes parâmetros para formatação da saída:

- Q_k: Quantidade de documentos armazenados no guichê k.
- $CPFc_j^i$: CPF do cliente referente a j-ésima operação desempilhada do guichê i.
- $CPFt_j^i$: CPF de terceiros referente a j-ésima operação desempilhada do guichê i.
- O_j^i: Operação do j-ésima operação desempilhada do guichê i.
- V_j^i: Valor envolvido na j-ésima operação desempilhada do guichê i.

Saída:

-:| RELATÓRIO PARCIAL |:-
3
Guiche 1: Q_1
[CPFc$_1^1$,CPFt$_1^1$,O$_1^1$,V$_1^1$]
[CPFc$_2^1$,CPFt$_2^1$,O$_2^1$,V$_2^1$]
⋮
[CPFc$_{Q_1}^1$,CPFt$_{Q_1}^1$,O$_{Q_1}^1$,V$_{Q_1}^1$]
⋮
Guiche 2: Q_M
[CPFc$_1^2$,CPFt$_1^2$,O$_1^2$,B$_1^2$]
[CPFc$_2^2$,CPFt$_2^2$,O$_2^2$,B$_2^2$]
⋮
[CPFc$_{Q_2}^2$,CPFt$_{Q_2}^2$,O$_{Q_2}^2$,B$_{Q_2}^2$]
⋮
Guiche 3: Q_3
[CPFc$_1^3$,CPFt$_1^3$,O$_1^3$,B$_1^3$]
[CPFc$_2^3$,CPFt$_2^3$,O$_2^3$,B$_2^3$]
⋮
[CPFc$_{Q_3}^3$,CPFt$_{Q_3}^3$,O$_{Q_3}^3$,B$_{Q_3}^3$]

4 Restrições e informações adicionais

Estruturas de dados & programação

1. Cada guichê deve ser uma pilha implementada utilizando alocação encadeada.
2. Não utilizar variável global!

Modularização & Funções

É importante que dentro do arquivo da simulação existam funções específicas para cada uma das operações da simulação e funções específica para cada uma das operações da estrutura de dados pilha. Deve-se implemetar as seguintes funções da pilha:

- Criar: cria a estrutura de dados vazia;
- Empilhar: insere um elemento na pilha;
- Desempilhar: remove um elemento da pilha (deve ser retornado);
- Destruir: destrói a estrutura de dados bem como os dados que ali existir.

Deve existir uma função para criar o nós do elemento a ser inserido na pilha. Tal função deve receber os valores por parâmetro e retornar o ponteiro para o nó com os dados armazenados. Esse nó é quem deve ser inserido na pilha.

É importante que as funções façam suas operações independentes. Ex: não passar por parâmetro o guichê destino. Para um código bem organizado é importante que o mesmo não contenha variáveis globais e que as funções retornem valores que indiquem o seu comportamento. É bastante comum definir padrões para o retorno das funções de acordo com os possíveis comportantes da mesma. Por exemplo: remoção de um elemento: null significa que não houve remoção. Ou mesmo um inteiro que mapeie os possíveis comportamentos.

`Lembre-se: Tudo que é alocado, deve ser desalocado!`

☻ **Boa diverSão!!!!** ☺