

Trabalho 01

Versão: 28 de agosto de 2018

## 1 Instruções:

- O trabalho é **individual**. O nome do integrante devem estar no cabeçalho comentado no arquivo.
- Será aplicado detector de plágio! Uma vez detectado plágio, a **nota da disciplina** será **zerada**.
- Data de entrega: **17 de setembro de 2018 (segunda-feira)** no sistema do moodle até as 17:00.
- A entrega deve ser no formato ZIP com o seguinte conteúdo:
  - + Apenas 1 arquivo como código-fonte;
- 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.
- Para validação do código será utilizado um ambiente GNU/Linux com o compilador GCC (versão 7+) dado os seguintes parâmetros<sup>1</sup>:  
`gcc -Wall -Wextra -Werror -Wpedantic`
- Para verificação de vazamento de memória e gerência dos ponteiros, será utilizado o software Valgrind<sup>2</sup>.

## 2 Correção

Serão utilizados os seguintes critérios para correção:

### 2.1 Compilação

- Compilação com parâmetros requeridos.

### 2.2 Entrada/Saída

- Cada entrada gera uma saída correta.

### 2.3 Organização do código

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

### 2.4 Gerência de memória

- Teste de alocação.
- Remoção correta da memória alocada.
- Vazamentos de memória.
- Controle de ponteiros.

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

---

<sup>1</sup>Leia mais sobre em: <https://gcc.gnu.org/onlinedocs/gcc-6.4.0/gcc/Warning-Options.html>.

<sup>2</sup><http://valgrind.org>

## 3 Simulação de uma AGÊNCIA – PARTE I

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

Para iniciar a implementação da agência precisamos implementar nossas estruturas de dados. Nesta etapa do desenvolvimento iremos implementar os guichês de atendimento da simulação. Desta forma, deveremos ser capazes de executar simulação que serão definidas através de uma entrada e ao final a simulação deve gerar 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: Código de pessoa física (CPF), operação, o bem, CPF de terceiro envolvido na operação e sua prioridade. 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;

A 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  $k \bmod 3$ , então o presente cliente de ser alocado no guichê  $k + 1 \bmod 3$  ficando a seguinte ordem dos guichês selecionados: 0, 1, 2, 0, 1, 2, 0, ..., 2, 0, 1, 2, 0, 1, 2, ...

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  $1 \leq N \leq 2^{32} - 1$ , 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  $1 \leq CPF \leq 2^{32} - 1$  representa o CPF do cliente,  $1 \leq CPFT \leq 2^{32} - 1$  o CPF do terceiro envolvido,  $O \in \{D, S, T\}$  o caractere referente a operação e  $1 \leq V \leq 2^{32} - 1$  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 por um relatório para 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  $M$  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, CP Ft, O, V]” onde “CPFc” é referente ao CPF do cliente atendido, “CP Ft” 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:

- $M$ : Quantidade total de guichês.
- $P_k$ : Quantidade de documentos armazenados no guichê  $k$ .
- $CPFc_j^i$ : CPF do cliente referente a  $j$ -ésima operação desempilhada do guichê  $i$ .
- $CP Ft_j^i$ : 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  $i$ .

Ademais, teremos a seguinte estrutura:

```
Saída:
- : | RELATÓRIO PARCIAL | : -
M
Guiche 1:  Q1
[CPFc11, CP Ft11, O11, V11]
[CPFc21, CP Ft21, O21, V21]
⋮
[CPFcQ11, CP FtQ11, OQ11, VQ11]
⋮
Guiche M:  QM
[CPFc1M, CP Ft1M, O1M, V1M]
[CPFc2M, CP Ft2M, O2M, V2M]
⋮
[CPFcQMM, CP FtQMM, OQMM, VQMM]
```

## 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 implementar 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 “auxiliar” que irá criar um elemento para ser inserido na pilha. Tal função deve receber os valores por parâmetro e retornar um registro alocado com os dados armazenados. Esse registro 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.

Por fim, espera-se que a simulação implemente as seguintes funções:

1. Função para ler linhas da entrada;
2. Função para “criar” um cliente (descrita acima);
3. Função de enviar cliente para atendimento (guichê);
4. Função para gerar/imprimir relatório parcial;

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



Boa diverSão!!!!

