Trabalho Prático 1: Simulador Bancário

Valor: 10+2.5 pontos
Entrega: 03/11/2015

1 Introdução

O objetivo deste trabalho é simular uma agência bancária. Especificamente, serão simulados alguns aspectos relacionados ao fluxo de dinheiro (em espécie) e sobre a experiência dos clientes (mensurado pelo tempo de permanência dos mesmos no banco). A finalidade de tal simulação é produzir um relatório que o gerente dessa agência possa utilizar para melhor programar a entrega de dinheiro a sua agência, além de garantir que seus clientes não fiquem sem atendimento por um período maior que o permitido pelas leis municipais (normalmente 15 minutos, exceto em dias de pico, onde muitas vezes passa a ser 30 minutos).

Academicamente, as metas deste trabalho são três: (1) Desenvolver a prática de programação, (2) desenvolver a prática de modelagem arquitetural do software por via da organização do código em tipos abstratos de dados (TADs), e (3) o uso de ferramentas de análise de complexidade em um contexto real.

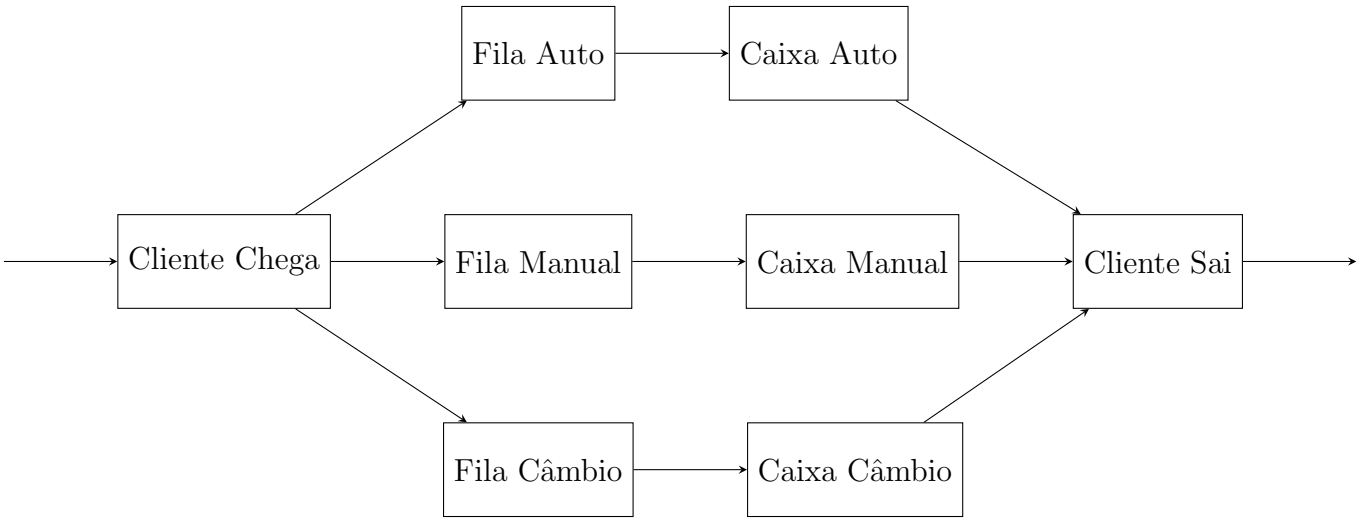
2 O Problema

Para os fins da nossa análise, consideramos que o banco oferece 3 serviços:

- *Caixas automáticos:* Aceitam depósitos de qualquer tamanho e saques de no máximo BRL 300 ou 900 (dependendo se a máquina possui um leitor biométrico).
- *Caixas com atendentes:* Fazem qualquer operação (exceto câmbio).
- *Caixa de Câmbio:* Caixa único que faz câmbio, exclusivamente para USD.

Note que, embora o banco ofereça outros serviços (como pagamento de contas, empréstimos, etc.), consideramos que todos esses ou podem ser expressos como uma operação de saque/depósito, ou não são por hora do interesse do gerente.

O cliente, por sua vez, tem suas ações representadas pelo seguinte fluxograma:



Onde o caminho que o cliente toma é predeterminado pela entrada do programa, composta de seu (1) tempo de chegada, (2) a operação desejada, (3) a quantidade sacada / depositada / cambiada, e (4), o tempo gasto para fazer sua operação uma vez que ele começar a ser atendido (desconsiderando o tempo gasto na fila).

A saída do seu programa será um arquivo `.csv` (*comma-seperated-values* – arquivo de valores separados por vírgulas), contendo primeiro uma lista das operações feitas, junto à variação cumulativa do dinheiro em espécie (BRL e USD), e depois um resumo com algumas informações úteis sobre os dados. O formato específico é detalhado na Seção 3.

A fim de simplificar o trabalho, assumiremos que o banco possui exatamente um caixa de câmbio, 3 caixas manuais, e 25 caixas de atendimento automático, das quais 10 possuem identificadores biométricos (isto é, possuem um limite de saque de 900R$ ao invés dos 300R$ dos demais).

A forma com que as filas funcionam visa copiar a realidade: Um cliente que entra em uma fila deve esperar até que todo mundo que esteja a frente dele seja atendido, e somente será atendido quando um caixa estiver disponível. Note que **há uma exceção**: Se o cliente que estiver na fila de autoatendimento precisar sacar mais de BRL 300, ele deve esperar até que uma máquina com reconhecimento biométrico esteja liberada e, enquanto isso for o caso, deixar que as outras pessoas atrás dele(a) passem na sua frente. Havendo mais de um caixa livre, o cliente do autoatendimento dará preferência ao caixa sem identificador biométrico, se sua operação permitir. Demais empates não afetam nossa análise.

3 Formato de Entrada e Saída

3.1 Entrada

O arquivo de entrada contém uma transação por linha, com os seguintes atributos, separados por vírgula:

- *Tempo de Chegada*: O momento que o cliente pisou no banco em *Unix Time*[1] (um inteiro unsigned de 32 bits que conta o tempo em segundos desde 1970, comunmente usado para representar tempo em bancos de dados).
- *Nome do Cliente*: Atributo útil somente ao gerente

[1] `https://pt.wikipedia.org/wiki/Era_Unix`

- *Tipo de Transação*: `SAQUE`, `DEPOSITO`, `CAMBIO` ou `CAIXA`, observando que ambas as primeiras duas operações se referem aos caixas de atendimento automático (seja com identificador biométrico ou não)
- *Valor em BRL*: Número real com duas casas decimais que representa a variação do dinheiro em reais no banco decorrente da transação do cliente. Saques reduzem essa quantidade (logo são usados valores negativos), depósitos a aumentam (valores positivos), e operações de caixa e câmbio podem fazer ambos.
- *Valor em USD*: Omitido **ou** 0 para qualquer operação exceto câmbio. No caso de operações de câmbio, representa a entrada ou saída de dólares (número real).
- *Tempo de Operação*: O tempo (em segundos) levado para atender o cliente, desconsiderando seu tempo de espera.

Um arquivo de entrada é exemplificado a seguir:

```
1444771699,André Alves,SAQUE,-500.00,,200
1444771730,Junia Jansen,DEPOSITO,300.37,,550
1444771735,Ricardo Romário,CÂMBIO,500.00,-125.44,150
1444771801,Sandrei Sousa,SAQUE,-700.00,,150
1444771805,Giovana Galvão,SAQUE,-100.00,,60
1444771811,Fabio Feliciano,CAIXA,-1500.00,,480
1444771831,Larissa Lousano,CAIXA,500.00,,480
```

Sua implementação deve garantir que o programa não entre em um estado indevido se a entrada for inválida. Você terá que verificar e tratar possíveis erros tais como:

- Verificar que cada linha no arquivo de entrada tenha o número correto de campos.
- Verificar que a operação de saque só pode retirar dinheiro e a operação de deposito só colocar.
- Garantir que seu programa não quebre na presença de campos vazios ou que contenham só espaço em branco.

Caso seu programa encontre uma entrada inválida, o programa deve abortar.

3.2 Saída

O arquivo de saída obedece uma estrutura similar ao de entrada, diferindo somente no significado dos valores colocados, e na adição de um resumo geral ao seu fim:

- *Tempo de Saída*: Dado pelo tempo de chegada + tempo de fila + tempo de operação.
- *Nome do Cliente*: Igual à entrada.
- *Tipo de transação*: Igual à entrada.
- *Total em BRL*: A quantidade total de dinheiro no banco (em reais). Esse valor começa com 0 e é incrementado ou decrementado pelo valor associado à transação do cliente no momento que esse vai embora (*tempo-de-saída*).

- *Total em USD*: A quantidade total de dinheiro no banco (em dólares). Esse valor se mantém constante para qualquer operação exceto câmbio, caso em que ele se comporta analogamente ao total de BRL.
- *Tempo de Espera*: O tempo total gasto para atender ao cliente (tempo de fila + tempo de operação)

O resumo geral aparece ao final deste arquivo, e contempla os seguintes aspectos:

1. A variação cumulativa mínima e máxima de dinheiro em BRL
2. A variação cumulativa mínima e máxima de dinheiro em USD
3. O tempo de espera mínimo e máximo de algum cliente

O arquivo de saída referente ao exemplo dado na entrada **assumindo que o banco possui um caixa de atendimento manual e dois de atendimento automático (um delas com biometria)** seria:

```
1444771899,André Alves,SAQUE,-500.00,0,200
1444772280,Junia Jansen,DEPOSITO,-199.63,0,550
1444771885,Ricardo Romário,CÂMBIO,300.37,-125.44,150
1444771959,Giovana Galvão,SAQUE,200.37,-125.44,154
1444772430,Sandrei Sousa,SAQUE,-499.63,-125.44,629
1444772291,Fabio Feliciano,CAIXA,-1999.63,-125.44,480
1444772771,Larissa Lousano,CAIXA,-1499.63,-125.44,940
VARIACAO DE BRL: -1999.63 a 300.37
VARIACAO DE USD: -125.44 a 0
VARIACAO DE TEMPO: 150 a 940
```

Exemplificando os requisitos do trabalho, observamos no exemplo acima que André Alves entra e escolhe o caixa de autoatendimento sem biometria (devido ao critério de desempate), o qual é ocupado em seguida por Junia Jansen pelo tempo que ela precisou para pagar suas contas. Sandrei Souza quis fazer um saque de BRL 700, mas ele teve de esperar Junia Jansen (que estava usando a máquina com biometria) acabar, deixando a Giovana Galvão passar na frente dele quando a máquina que André Alves estava usando foi liberada (visto que esta não deixava que ele sacasse a quantidade desejada).

Note que as linhas antes do resumo geral podem ser impressas em qualquer ordem.

4 O que deve ser entregue?

Você deverá submeter um arquivo `matricula.zip`, contendo:

- seu código fonte (todos os arquivos .c e .h);
- sua documentação (pdf, máximo 10 páginas).

A documentação deve conter:

1. Introdução: descrição sucinta do problema a ser resolvido e visão geral sobre o funcionamento do programa.

2. Implementação: descrição sobre a implementação do programa. Devem ser detalhadas as estruturas de dados utilizadas (de preferência com diagramas ilustrativos) e o funcionamento das principais funções.

3. Casos de teste: descrição dos distintos casos de teste que foram usados.

4. Estudo de complexidade: estudo da complexidade do tempo de execução dos procedimentos implementados e do programa como um todo (notação O).

5. Conclusão: comentários gerais sobre o trabalho e as principais dificuldades encontradas em sua implementação.

6. Bibliografia: bibliografia utilizada para o desenvolvimento do trabalho, incluindo sites da Internet, se for o caso.

Seu programa será compilado no ambiente Linux com o seguinte comando:

```
gcc -Wall -std=c99 *.c -o tp1
```

O programa compilado será executado com o seguinte comando:

```
./tp1 entrada.csv saida.csv
```

em que `entrada.csv` e `saida.csv` são respectivamente os arquivos de entrada e saída, conforme já especificados.

Comentários gerais:

- Comece a fazer este trabalho logo, enquanto o problema está fresco na memória e o prazo para terminá-lo está tão longe quanto jamais poderá estar.
- Clareza, indentação e comentários no programa também vão valer pontos.
- Trabalhos copiados serão penalizados com a nota zero.
- Penalização por atraso: $(2^d - 1)$ pontos, em que d é o número de dias de atraso.

5 Critério de Correção

A pontuação do trabalho se divide nos seguintes critérios:

- *(10%)* Adequação às normas de envio.
- *(20%)* Documentação
- *(70%)* Funcionamento e saída correta do programa. Será avaliado o funcionamento correto do programa no *valgrind*, ausência de erros de execução (e.g.: *segmentation fault*), validação da entrada (e.g.: checagem se o número de argumentos passados está correto).

Serão avaliados também aspectos relacionados à qualidade do código, como clareza, qualidade dos comentários (os quais devem ser suficientes para acompanhar o fluxo do código, e **nada mais** – evite comentários como *"loop que itera de 0 a N"*, os quais não adicionam nada no entendimento do código), e reuso (isto é, evite copiar e colar o código dentro do programa, como entre os diferentes tipos de caixa, optando em vez disso por abstrair essas funções em TADs ou métodos genéricos). Lembre-se: menos código implica em menos bugs e uma clareza maior.

6 Ponto Extra (até + 2.5 pontos)

Crie um gerador de instâncias de entradas e analise o tempo de espera dos clientes no *Banco do Brasil* da Praça de Serviços da UFMG (se for necessário para sua análise, altere o número de caixas fixados no enunciado durante este experimento). Faça suposições razoáveis sobre o fluxo de pessoas e responda: É provável alguém esperar mais que 30 minutos? Que 1 hora?