AEDS II

Exercício Prático I
Registro de Atendimentos

Entrega: 21/Abr/2016 via Moodle

1 Enunciado e contexto do problema

João é dono de uma lanchonete e notou nas últimas semanas o aumento da fila de atendimento, além de perceber em alguns momentos que clientes desistem de comprar após certo tempo na fila, perdendo assim faturamento. Preocupado com o aumento de pessoas desistindo de comprar e com as reclamações de seus clientes relacionadas ao tempo de espera em fila, João contratou uma empresa para desenvolver um sistema que seja capaz de fornecer a ele, em tempo real, algumas estatísticas da fila de atendimento. Você, como desenvolvedor dessa empresa, ficou responsável pela implementação de um sistema que faz o registro dessas informações e fornece as estatísticas de interesse ao João.

2 Entrada

A entrada deverá ser lida de um arquivo. A primeira linha do arquivo contém a quantidade máxima (n) de clientes que serão registrados, seguida de uma quantidade indefinida de linhas contendo informações de eventos no sistema. Eventos podem ser de 6 tipos diferentes, cujos formatos são descritos a seguir:

- Chegada de cliente

  `1 <hora> <numero_cliente> <gasto_estimado>`

- Atendimento de cliente

  `2 <hora> <numero_cliente>`

- Saída após ser atendido

  `3 <hora> <numero_cliente>`

- Desistência

  `4 <hora> <numero_cliente>`

- Imprimir relatório

  `5`

- Sair do sistema

  `6`

Em todos os formatos, o primeiro valor indica o código do evento (de 1 a 6). Os demais valores seguem o seguinte formato:

- `<hora>`: string no formato `HH:MM` (e.g., `10:15`).
- `<numero_cliente>`: número inteiro no intervalo $[1, n]$ (e.g., `3`).
- `<gasto_estimado>`: número de ponto flutuante (e.g., `21.10`).

3 Saída

A saída deverá ser impressa em um arquivo. Seu sistema irá fornecer saída apenas quando requisitado pelo código `5` (Imprimir relatório) na entrada. Tal relatório deve obrigatoriamente seguir o padrão:

```
Consulta numero: resp
Quantidade de clientes que entraram: resp
Quantidade de clientes atendidos: resp
Quantidade de clientes desistentes: resp
Tempo medio em fila geral: resp
Tempo medio em fila antes de desistir: resp
Tempo medio de atendimento: resp
Valor vendido estimado: resp
Valor perdido estimado: resp
```

Notas:

- O número da consulta começa em 1 e é acrescido em 1 a cada relatório gerado.
- Nos cálculos das médias de tempo, desconsiderar os segundos. Exemplo: tempo médio de 2 minutos e 30 segundos deverá se escrito como 00:02.
- Vários eventos podem ocorrer no mesmo horário.

4 Exemplo de entrada e saída

Conforme explicitado nas seções anteriores, tanto a entrada quanto a saída do seu programa serão feitas utilizando arquivos. Seu programa deverá executar através da seguinte linha de comando:

```
./main <entrada> <saida>
```

onde `<entrada>` e `<saida>` indicam os nomes dos arquivos de entrada e saída, respectivamente, os quais devem ser obtidos utilizando a variável `argv` na linguagem C.

4.1 Exemplo de entrada

```
20
5
1 13:20 1 10.00
2 13:21 1
1 13:22 2 25.00
4 13:26 2
3 13:27 1
5
6
```

4.2 Exemplo de saída

```
Consulta numero: 1
Quantidade de clientes que entraram: 0
Quantidade de clientes atendidos: 0
Quantidade de clientes desistentes: 0
Tempo medio em fila geral: 00:00
Tempo medio em fila antes de desistir: 00:00
Tempo medio de atendimento: 00:00
Valor vendido estimado: 0.00
Valor perdido estimado: 0.00
Consulta numero: 2
Quantidade de clientes que entraram: 2
Quantidade de clientes atendidos: 1
Quantidade de clientes desistentes: 1
Tempo medio em fila geral: 00:02
Tempo medio em fila antes de desistir: 00:04
Tempo medio de atendimento: 00:06
Valor vendido estimado: 10.00
Valor perdido estimado: 25.00
```

5 Restrições

1. A implementação deve ser feita obrigatoriamente em C.
2. A implementação deverá compilar sem *warnings* em ambiente Linux (Ubuntu 14.04.4 LTS), utilizando a seguinte linha de comando:

   ```
   gcc -Wall *.c -o main
   ```

3. Deve ser criado um tipo abstrato de dados para representar as informações de eventos de cada cliente.
4. O vetor desse tipo abstrato de dados deve ser alocado dinamicamente de acordo com o valor n informado na entrada inicialmente.
5. A saída deve obedecer rigorosamente ao padrão especificado (nota: a saída não contém acentuações).

6 Entregáveis

Crie um diretório (nomeie o diretório com seu número de matrícula) contendo seu código fonte (todos os arquivos `.c` e `.h`). Comprima o diretório com o seguinte comando:

```
tar zcvf <diretorio>.tar.gz <diretorio>
```

onde `<diretorio>` indica o nome do diretório criado. Submeta o arquivo `.tar.gz` via Moodle.

7 Critérios de Correção

A pontuação do trabalho se divide nos seguintes critérios:

- (10%) Adequação às normas de envio.
- (20%) 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).
- (70%) Saída correta do programa.