ENGENHARIA DA COMPUTAÇÃO
ESTRUTURAS DE DADOS
Professor: George Luiz Medeiros Teodoro

Projeto de Programação 01

O trabalho pode ser feito em duplas. Este trabalho exercita as primitivas básicas da Linguagem C e inicia a discussão e solução de problemas complexos.

Após diversas averiguações, o grupo de laboratórios Nibas concluiu que seus lucros estavam decrescendo devido aos clientes esperarem muito para serem atendidos, o que causava um elevado índice de insatisfação. Assim, a empresa não era recomendada para novos clientes, sendo que os antigos geralmente não retornavam.

Dessa forma, o laboratório decidiu contratar um desenvolver para criar um sistema de senhas e priorização de atendimento, que pudesse resolver o problema e fazer com o grupo Nibas voltasse a atrair novos clientes e aumentar seus lucros.

Cada laboratório Nibas possui um determinado número de guichês abertos diariamente. Esse valor é passado como parâmetro para o sistema. Entretanto, devido a segmentação dos atendimentos, cada um guichê atende um único tipo de serviço:

- Hemograma simples
- Pré-natal.
- Vacinação.
- Hemograma completo.
- Entrega de resultados.

Guichês diferentes podem atender ao mesmo serviços, mas para que todos os clientes possam ser atendidos é necessário que existam ao menos cinco atendentes/guichês, um para cada tipo de atendimento.

Um grande problema é decidir a ordem na qual os clientes serão chamados, pois isso afeta diretamente o grau de satisfação do atendido e representa a maior fonte de problemas do Nibas. Por obviedade e força da lei, pessoas idosas devem ser atendidas primeiro, bem como gestantes e pessoas com necessidades especiais. Indivíduos com a mesma prioridade devem ser atendidos conforme a ordem de chegada.

Cada cliente chegará em um instante diferente no laboratório e pedirá a senha de atendimento, quando informará sua idade, condição física e serviço desejado. Então, receberá um número que indica sua prioridade e será direcionado para uma das filas correspondente ao serviço requerido. Pessoas que chegarem posteriormente terão suas prioridades e tempos de chegada comparados com esses clientes e, baseado nisso, terão sua posição na fila escolhida. Caso a prioridade do indivíduo X seja maior que a do Y, ele ficará na frente de Y mesmo tendo chegado depois. Caso a prioridade seja igual, a alocação deve seguir ordem de chegada entre pessoas com a mesma prioridade.

Foram definidos os seguintes códigos para cada uma das possíveis situações dos clientes:

Serviços:

Hemograma simples – 0 (5 unidades de tempo de atendimento)
Pré-natal - 1 (10 unidades de tempo de atendimento)
Vacinação – 2 (8 unidades de tempo de atendimento)
Hemograma completo – 3 (7 unidades de tempo de atendimento)
Entrega de resultados – 4 (2 unidades de tempo de atendimento)

Condição física:

Gestante – 0
Deficiente – 1
Saudável – 2

Para efeitos de simplificação, considere que uma pessoa possa ser classificada apenas em uma das condições físicas acima.

PRIORIDADES

A prioridade do cliente será calculada como a soma de duas prioridades parciais, baseadas na idade e condição física. A prioridade baseada na idade X será tal que:

$X < 65$ *a prioridade é 1*
$65 \leq X < 80$ *a prioridade é 2*
$X \geq 80$ *a prioridade é 3*

A prioridade baseada na condição física é tal que se a pessoa é: *Gestante, a prioridade é*

2; Deficiente, a prioridade é 2; Saudável, a prioridade é 1.

ENTRADAS

O programa receberá como entrada dois arquivos, o primeiro deles com a configuração dos guichês e o segundo com a carga de trabalho que representa a chegada de clientes.

Arquivos de configuração dos guichês seguirá o formato seguinte: a primeira linha representa o número de guichês abertos e a demais linhas indicam o serviço realizado por cada guichê. Exemplo:

5
0
1
2
3
4

Nesse caso, 5 guichês estão funcionando, o primeiro com serviço 0 (*Hemograma simples), o segundo com serviço 1 etc.*

A carga de trabalho representando as chegadas de clientes será armazenada em um arquivo onde cada linha descreve a chegada de um cliente. Para um cliente que possui:

Tempo de chegada: 10 *Idade:* 28 *Serviço:* Hemograma simples *Condição física:* Gestante

O padrão de entrada deverá ser:

10 28 0 0

Os dados dos clientes devem ser impressos na saída utilizando a mesma ordem (tempo de chegada) da entrada.

O arquivo de entrada deverá ser gerado por um programa à parte.

A execução do simulador de senhas deverá se dar da seguinte forma:

./sim_senhas <arquivo_configuracao> <arquivo_carga_trabalho> <arquivo_saída>

SAÍDAS

O programa deverá retornar um relatório a respeito do funcionamento do programa, com

dados como o tempo de espera médio dos clientes nas filas (float com duas casas de precisão) e quantidade média de clientes atendidos por unidade de tempo na primeira linha (float com duas casas de precisão).

A seguir, cada linha da saída terá um relatório de cada cliente atendido, informando qual a sua prioridade, qual foi o tempo de espera por atendimento, em qual guichê foi atendido e o serviço. Para um cliente que possui os seguintes dados:

Guichê: 2, *Prioridade:* 4, *Tempo de espera:* 19 unidades de tempo, *Serviço:* Hemograma simples.

O padrão de saída por cliente deverá ser:

2 4 19 0

Da entrega do trabalho

A entrega do trabalho deverá ser feita via moodle (aprender) por meio de um único arquivo zip contendo a documentação (incluindo instruções de execução e compilação e código fonte) e o código fonte.

Avaliação

O documento a parte "roteiro de documentação" disponível no aprender contém detalhes sobre como fazer uma boa documentação de um trabalho.

Detalhes da organização do programa

- Projete todo seu simulador, incluindo as estruturas de dados a serem utilizadas.
- O programa deve ser implementado em ao menos 3 módulos. O primeiro módulo deve tratar dos procedimentos de entrada e saída. O segundo módulo contém a lógica do simulador. O terceiro módulo contém o programa principal. Deve ainda ser definido um arquivo de protótipos e definições contendo as estruturas de dados e protótipos das funções dos vários módulos. A compilação do programa deve utilizar o utilitário Make.
- Todas as estruturas de dados devem ser alocadas dinamicamente, assim como devem ser desalocadas após a execução.
- Faz parte do trabalho gerar cargas realistas, que, por exemplo, tenham uma

distribuição de frequência de chegada de clientes, por exemplo, desigual indicando horários de pico.

Realize várias execuções do seu simulador, mostrando a correção do seu funcionamento.

Deverão ser entregues:

- Descrição do problema
- Abordagem da solução
- Descrição breve dos algoritmos e das estruturas de dados utilizadas;
- Listagem das rotinas;
- análise da complexidade das rotinas;
- entradas de dados utilizadas para avaliar o algoritmo;
- análise dos resultados obtidos (diversas cargas de trabalho devem ser construídas, exercitando o simulador com quantidades variadas de guichês e intensidades distintas de chegada de clientes).

Distribuição dos pontos:

- execução
  execução correta: 45%
- estilo de programação
  código bem estruturado: 5%
- documentação
  comentários explicativos: 5%
  análise de complexidade: 10%
  análise de resultados e apresentação da solução 35%