Universidade Federal de Minas Gerais

TRABALHO PRÁTICO 01
ALGORITMOS E ESTRUTURAS DE DADOS II

Henrique Cheik Freire Cabral

BELO HORIZONTE

2017

1- INTRODUÇÃO

Todos os dias uma grande quantidade de pessoas almoçam no restaurante da cantina do ICEX. Com isso, grandes filas são formadas para que os usuários possam comprar a ficha no caixa, esperar por uma bandeja e se servir.

O seguinte projeto tem como objetivo resolver o problema das filas no restaurante da cantina do ICEX. O projeto simulará o ambiente de almoço no restaurante durante um período de 4 horas, buscando a melhor solução para amenizar o problema das grandes filas.

O caso inicial contemplará os seguintes parâmetros:

- 1 fila de tamanho indefinido para o caixa e 1 caixa
- 1 fila para pegar bandejas
- 1 pilha de bandejas com no máximo 30 bandejas
- Reposição de 10 bandejas a cada 12 minutos

O usuário gasta 1 minuto para comprar sua ficha no caixa, 1 minuto para pegar uma bandeja da pilha e 4 minutos para se servir de alimentos.

Para resolver o problema, foram desenvolvidos TADs (tipos abstratos de dados), que fazem a representação do problema no ambiente computacional e, através das funções contidas dentro dessas estruturas de dados, é feita uma simulação das 4 horas de almoço, retornando dados importantes como o tempo médio de cada usuário para se servir e quantos usuários são atendidos naquele espaço de tempo.

2- DESENVOLVIMENTO

O projeto tem a implementação de TADs (tipos abstratos de dados) para representar o ambiente do restaurante da cantina. Os seguintes TADs foram desenvolvidos para a simulação do problema:

1. **TAD para os usuários:**

Foi implementado um TAD que faz a representação da posição dos usuários no restaurante. Esse TAD armazena o número de chegada do usuário, seu tempo de chegada e seu tempo total para se servir. Foram desenvolvidas as seguintes funções que fazem operações sobre esse TAD:

- **create_user:** Aloca memória para uma estrutura de usuário declarada no main.

- **remove_user:** Libera a memória para uma estrutura de usuário declarada no main.
- **set_arrival_time:** Define dentro da estrutura de um usuário selecionado seu tempo de chegada.
- **set_total_time:** Define dentro da estrutura de um usuário selecionado seu tempo total até se servir. O tempo total é tido pela diferença do tempo de saída e o tempo de chegada.
- **get_arrival_time:** Retorna o valor do tempo de chegada do usuário.
- **get_total_time:** Retorna o valor do tempo total que o usuário gasta para se servir.
- **print_total_time:** Imprime no console a seguinte mensagem **"O aluno %d gastou %d minutos para pegar sua refeição. Entrada: %d\n"**, sendo os valores dos %d respectivamente, o número de ordem de chegada do usuário, tempo total da rotina e, por último, o minuto em que o usuário entrou na fila para comprar uma ficha

2. **TAD para as filas dinâmicas:**

Foi implementado um TAD que faz a representação das filas presentes no problema, que são a fila do caixa e a fila da pilha de bandejas. As filas podem crescer indefinidamente. Nas filas cada elemento é composto por uma estrutura que armazena as informações do usuário e um apontador para o próximo elemento da fila. Foram desenvolvidas as seguintes funções que fazem operações sobre esse TAD:

- **cria_fila:** Alocação de memória para uma estrutura do tipo fila declarada no main.
- **destroi_fila:** Libera a memória alocada em uma estrutura do tipo fila declarada no main.
- **tamanho_fila:** Retorna quantas pessoas existem na fila selecionada.
- **fila_vazia:** Verifica se a fila está vazia. Retorna 1 caso esteja vazia e 0 caso possua algum elemento.
- **insere_usuario_fila:** Aloca memória para um elemento na fila e insere os dados de um usuário específico nesse elemento na última posição livre da fila selecionada.
- **desenfileira:** Retira o primeiro elemento de uma determinada fila e libera a memória alocada.
- **transfere_usuario:** Transfere os dados do usuário presente na primeira posição de uma determinada fila para uma estrutura alvo que pode armazenar esses dados. Essa função é utilizada para transferir os dados do primeiro usuário da fila do caixa para o

caixa e os dados do primeiro usuário da fila da pilha de bandejas para a pilha de bandejas.

3. **TAD para as pilhas estáticas de bandejas:**

Foi implementado um TAD para representar as pilhas de bandejas presentes no problema. As pilhas tem um número máximo de bandejas e as pilhas são repostas em determinados intervalos de tempo, observando o limite máximo possível de bandejas. A pilha é formada por uma estrutura que possui informações sobre o número máximo de bandejas possível, o número atual de bandejas e o número de bandejas que são repostas. Foram desenvolvidas as seguintes funções que fazem operações sobre esse TAD:

- **cria_pilha:** Aloca memória para uma estrutura do tipo pilha declarada no main e, através dos parâmetros recebidos na função, define o número máximo de bandejas possível, o número atual de bandejas e o número de bandejas que são repostas.
- **destroi_pilha:** Libera a memória alocada em uma estrutura do tipo pilha declarada no main.
- **pilha_vazia:** Verifica se a pilha selecionada está vazia. A função retorna 1 para pilha vazia e 0 caso contrário.
- **repoe_pilha:** Repõe a pilha com o número definido para a reposição de bandejas.
- **retira_bandeja:** Retira uma bandeja da pilha.

3- IMPLEMENTAÇÃO

O programa principal é composto pelas rotinas de declaração das variáveis e estruturas a serem utilizadas no projeto, pelo loop de simulação, o cálculo da média, a impressão dos resultados na tela e a rotina de liberação de memória que fora alocada para as estruturas.

O loop de simulação segue o seguinte padrão em cada iteração:

- 2 alunos entram na fila do caixa
- As bandejas são respostas na pilha (ocorre a cada 12 minutos)
- Verifica se existe alguém na posição do caixa e o transfere para a fila de bandejas
- Verifica se existe alguém na posição do caixa e, caso esteja vazio, o transfere da fila para o caixa

- Verifica se existe alguém na posição onde a pilha de bandejas está e, em caso positivo e caso a pilha de bandejas não esteja vazia, retira uma bandeja da pilha, serve o usuário (adicionando 4 espaços de tempo ao seu tempo total de estadia) gravando seu tempo total até ser servido e adiciona esse tempo gravado ao tempo total gasto por todos os usuários que foram servidos para que a média seja calculada, e por fim o remove da simulação.
- Verifica se existe alguém na posição da pilha de bandejas e, caso esteja vazia, o transfere da fila das bandejas para o lugar onde a pilha está localizada.

4- ANÁLISE

Como o algoritmo utiliza em sua maioria funções que fazem atribuição ou retornam valores, pode-se dizer que o algoritmo chama funções que tem complexidade O(1) n vezes, onde n será o espaço de tempo que o usuário do programa quiser analisar. O algoritmo apresenta então uma complexidade O(n).

5- RESULTADOS

Todos os casos contemplam um período de tempo de 4 horas, ou 240 minutos, e que a cada minuto dois usuários entram na fila. A simulação para quando se passam os 240 minutos, independente se existe algum aluno sem se servir. Isso é feito para se analisar e obter a melhor configuração para atender mais usuários naquele espaço de tempo.

- **Caso inicial: 1 fila para caixa, 1 caixa, 1 fila para a pilha de bandejas, 1 pilha de bandejas com no máximo 30 bandejas e reposição de 10 bandejas a cada 12 minutos.**

Resultados: Foram atendidos 220 alunos. Média de tempo gasto para ser atendido: 64

- **Caso 2: 1 fila para caixa, 1 caixa, 1 fila para a pilha de bandejas, 1 pilha de bandejas com no máximo 30 bandejas e reposição de 10 bandejas a cada 10 minutos.**

Resultados: Foram atendidos 238 alunos. Média de tempo gasto para ser atendido: 65

- **Caso 3: 1 fila para caixa, 1 caixa, 1 fila para a pilha de bandejas, 1 pilha de bandejas com no máximo 30 bandejas e reposição de 15 bandejas a cada 12 minutos.**

Resultados: Foram atendidos 238 alunos. Média de tempo gasto para ser atendido: 65

- **Caso 4: 2 filas para caixa, 2 caixas, 1 fila para a pilha de bandejas, 1 pilha de bandejas com no máximo 30 bandejas e reposição de 10 bandejas a cada 12 minutos.**

Resultados: Foram atendidos 220 alunos. Média de tempo gasto para ser atendido: 64

- **Caso 5: 2 filas para caixa, 2 caixas, 2 filas para a pilha de bandejas, 2 pilhas de bandejas com no máximo 30 bandejas e reposição de 10 bandejas a cada 12 minutos.**

Resultados: Foram atendidos 440 alunos. Média de tempo gasto para ser atendido: 9

- **Caso 6: 2 filas para caixa, 2 caixas, 2 filas para a pilha de bandejas, 2 pilhas de bandejas com no máximo 30 bandejas e reposição de 10 bandejas a cada 10 minutos.**

Resultados: Foram atendidos 476 alunos. Média de tempo gasto para ser atendido: 6

6- CONCLUSÃO

No caso inicial, além dos usuários terem que esperar na fila que o usuário da frente seja atendido, ocorre em alguns momentos a falta de bandejas na pilha, reduzindo o número máximo de usuários atendidos. Nos casos 2 e 3 não ocorre a falta de bandejas na pilha, mas ainda sim os usuários devem esperar a vez de serem atendidos, ou seja, não importa se há um número máximo de bandejas maior ou que mais bandejas sejam repostas por vez, pois haverá engarrafamento nas filas.

O caso 4, apesar de mais usuários serem atendidos por vez no caixa, eles ainda caem na mesma fila para pegar uma bandeja, ou seja, o engarrafamento de usuários na fila apenas deslocou da fila do caixa para a fila para a pilha de bandejas.

Os casos 5 e 6 demonstram uma situação em que não há engarrafamento em nenhuma das filas. No caso 5 há uma pequena diminuição no número de usuários atendidos, pois em alguns momentos a pilha de bandejas fica vazia devido ao tempo de reposição. No caso 6 não há nenhum momento que a pilha de bandejas fique vazia, portanto todos os usuários que entram acabam sendo atendidos.

Concluindo o projeto, analisamos que a situação ideal é aquela que cada usuário que entra tem uma trajetória ideal e filas e pilhas dedicadas para o mesmo, não havendo falta de bandejas na pilha em nenhum momento e nenhum engarrafamento nas filas. Porém, adicionar novos caixas, pilhas e filas dedicadas para cada usuário pode acarretar em grande custo operacional para o proprietário ou responsável pelo funcionamento do restaurante da cantina do ICEX, pois envolve a contratação de funcionários para atender os usuários, talvez a compra de mais bandejas, etc. Sendo assim, fica a critério do responsável se vale ou não a pena investir no restaurante para atingir o cenário ideal.