2019-05-27, Fábio Silva e Nuno Bernardes, Laboratório de Programação

RELATÓRIO 2º TRABALHO

1. INTRODUÇÃO

Neste trabalho foi pedido para implementar um simulador de caixas de um supermercado tendo em conta quatro variáveis, afluência, apetência, número de caixas disponíveis e a quantidade de ciclos para resolver o processo. Tendo estes quatro fatores preenchidos o simulador resolve um modelo ideal de gestão dos clientes desse supermercado pelas caixas existentes e devolve o resultado da simulação.

2. ESTRUTURA DO PROGRAMA E COMO FOI ABORDADO A IMPLEMENTAÇÃO PARA AS EXTENSÕES FORNECIDAS

O programa está dividido em dois diretórios, "Simulador com filas de prioridade" e "Simulador sem filas de prioridade". Ambos os diretórios partilham os mesmos ficheiros à exceção que o "Simulador com filas de prioridade" tem dois ficheiros extra ("*QueueP.c*" e "*QueueP.h*") que fornece as funções extra para que a fila seja guardada numa ordem específica). Foi fornecido um ficheiro escrito na linguagem *Python* com a implementação necessária para o problema enunciado, então traduzimos o programa para *C* e dividimos o código em quatro grupos, Cliente, Caixa, Queue e Simulador. Também está presente em ambos os diretórios um ficheiro Makefile para facilidade na compilação dos restantes ficheiros.

A. Cliente

Incluí os ficheiros "*Cliente.c*" e "*Cliente.h*". Informação do número de produtos e tempo de entrada de um determinado cliente. *Getters* para retorno das informações.

B. Caixa

Incluí os ficheiros "*Caixa.c*" e "*Caixa.h*". Informação da fila (com prioridade para simulação com fila de prioridade), tempo estimado de atendimento, número de clientes, produtos, tempo de espera, número da caixa e velocidade de atendimento. *Getters* para retorno das informações acima descritas e *Setters* para atualização dos valores.

C. Queue

Incluí os ficheiros "*Queue.c*", "*Queue.h*", "*QueueP.c*" e "*QueueP.h*". Os dois primeiros ficheiros mencionados estão presentes em ambos os diretórios pois a fila de prioridade ("*QueueP*") usa a implementação normal de filas, acrescentando as funções que irão tratar da organização da fila pelo seu grau de urgência. Este tipo de queue de prioridade usa duas filas, uma fila urgência e uma normal. Só são tratados os clientes da fila normal quando a urgente estiver vazia. Para além da criação das filas, estes ficheiros manipulam a fila, adicionado e removendo valores, verificar se está vazia, verificar se está cheia, libertar a queue e escrevê-la.

D. Simulador

Incluí os ficheiros "*Simulador.c*" e "*Simulador.h*". Este grupo trata de todo o processo de simulação, desde obter as informações do primeiro cliente na fila e tratar do mesmo, identificar quais as caixas que estão cheias e não permitir que entrem mais clientes até abrirem vagas, mostrar as caixas existentes e processar os resultados obtidos de cada ciclo da simulação. Todos os valores que são expostos à simulação são randomizados (usando a biblioteca *rand* do *C*), exceto os quatro fatores apresentados na introdução, esses são inseridos pelo utilizador.

3. FILAS DE PRIORIDADE: SUPORTE VETORIAL VS LISTAS LIGADAS

Apesar de na nossa implementação usarmos suporte vetorial nas filas de prioridade, entendemos as desvantagens do seu uso face às listas ligadas.

FILAS DE PRIORIDADE	SUPORTE VETORIAL	LISTAS LIGADAS
Memória	Estática (tamanho máximo tem de ser declarado antes do uso do vetor).	Cresce consoante o espaço que é necessário.
Enqueue	Sabendo qual o índice em que é preciso adicionar o valor da fila, torna-se mais rápido. Tendo um suporte vetorial circular, a adição ainda é mais rápida.	Necessário percorrer a lista toda para adicionar.
Dequeue	Tempo idêntico pois remove-se o primeiro valor (Filas FIFO (*First In, First Out*))	Tempo idêntico pois remove-se o primeiro valor (Filas FIFO (*First In, First Out*))