**Sistemas Operativos**

**Trabalho 2**

**Restaurante**

107283 – Nuno Vieira

XXXXXX – Airton Moreira

Índice

Introdução

Foi-nos proposto um problema de um restaurante onde cada processo deve ser sincronizado com a ajuda de semáforos de acesso a memória partilhada entre estes processos. No decorrer deste programa, destacam-se quatro processos concorrentes que disputam acesso à memória partilhada. São estes:

* Chef: Recebe os pedidos que os grupos fazem ao waiter, sendo responsável pela sua confeção, e chama o waiter assim que a comida esteja pronta para que este a leve para as mesas.
* Waiter: É responsável por receber os pedidos de cada grupo, levá-los ao chef para que este os prepare, e, por fim, encarrega-se de levar a comida às mesas.
* Receptionist: Encarrega-se de esperar pela chegada dos grupos, procedendo à atribuição de uma mesa para o respetivo grupo ou, no caso de não haver mesas disponíveis, à colocação do grupo na sala de espera. Por fim, trata de receber os pagamentos.
* Group: Clientes do restaurante que chegam em grupos, pedem mesa, fazem o pedido da comida, esperam pela mesma, comem e pagam.

Chef

O life cycle deste processo consiste em aguardar pelo pedido da parte do group e do waiter e confecionar o pedido.

Funções

waitForOrder

Nesta função, que serve para que o chef aguarde por um pedido da parte do waiter, inicialmente é feito um decremento do semáforo waitOrder, para que o chefe, assim que exista um pedido (um incremento neste semáforo), prossiga para a sua confeção.

De seguida, para conseguir acesso à região crítica de memória partilhada, o processo decrementa o semáforo mutex e atualiza o estado do chef para COOK, atualizando também a flag foodOrder para 0, sendo que o pedido acabou de ser recebido por ele. Por fim, é incrementado o semáforo mutex para sair da região crítica e é também sinalizado o semáforo orderReceived, para indicar ao waiter que deve esperar pela confeção do pedido.

processOrder

Depois de receber o pedido, é simulada uma quantidade de tempo necessária para a preparação do mesmo e, após este intervalo, o processo volta a esperar por acesso à memória partilhada para atualizar o estado do chef para REST, significando que o pedido já foi confecionado. Saíndo da região crítica, resta apenas sinalizar o waiter através de um decremento no semáforo waiterRequestPossible seguido de um incremento no semáforo waiterRequest.