Trabalho 2

Simulação de Ponte Aérea

Objectivo

Compreensão dos mecanismos associados à execução e sincronização de processos e *threads*.

Guião

Admita que é necessário transportar N passageiros entre as cidades *Origin* e *Target* usando uma ponte aérea de 1 avião. Os passageiros chegam ao aeroporto em tempos aleatórios (distribuição uniforme com máximo conhecido) e as regras são que o avião deve descolar de *Origin* sempre que está cheio, ou já tem o número mínimo de passageiros e a fila está vazia, ou ainda quando todos os N passageiros já embarcaram. O avião pode, naturalmente, ter de fazer várias viagens para transportar todos os passageiros. A controlar o embarque existe uma hospedeira que trata da verificação de identidade dos passageiros, dando autorização a cada passageiro para sair da fila de espera e entrar no avião. O piloto informa a hospedeira sempre que o avião está pronto para iniciar o processo de embarque, é informado por esta sempre que o *boarding* está completo e dá permissão aos passageiros para sairem do avião em *Target*. O piloto apenas inicia o voo de regresso quando o último passageiro sai do avião.

Tomando como ponto de partida o código fonte em **`semaphore_airlift.tgz`**. Desenvolva uma aplicação em C que simule a ponte aérea. Os passageiros, a hospedeira e o piloto serão processos independentes, sendo a sua sincronização realizada através de semáforos e memória partilhada. Todos os processos são criados a partir do programa **`probSemSharedMemAirLift`** e estão em execução a partir dessa altura. Assume-se que os passageiros demoram algum tempo a chegar ao aeroporto e que esse tempo de chegada pode ser determinado através de uma distribuição de probabilidade uniforme com tempo máximo. Os processos devem estar ativos apenas quando for necessário, devendo bloquear sempre que têm de esperar por algum evento. Assuma que no início da simulação o avião se encontra em trânsito para *Origin*.

Para visualizar o resultado da execução de todos os processos numa versão pré-compilada, executar, dentro da diretoria **`src`** o comando **`make all_bin`** e depois dentro da diretoria **`run`** o comando **`./probSemSharedMemAirLift`**.

Apenas devem ser alterados os ficheiros **`semSharedMemPilot.c`**, **`semSharedMemPassenger.c`**, **`semSharedMemHostess.c`**. Dentro destes ficheiros estão assinaladas as zonas a alterar.

O trabalho será realizado em grupos de 2 alunos. Durante a execução do trabalho deve ser respeitado um exigente código de ética que impede o plágio, sob qualquer forma, bem como a execução do trabalho por elementos externos ao grupo ou a partilha de código entre grupos distintos.

A entrega do trabalho será realizada através do **`elearning.ua.pt`** e deverá incluir o código fonte da solução encontrada e um relatório que descreve qual a abordagem usada para resolver o problema e os testes realizados para validar a solução.

Data de entrega do trabalho: 1 de fevereiro de 2022