Universidade de Aveiro

Departamento de Electrónica, Telecomunicações e Informática

Sistemas de Operação / Fundamentos de Sistemas Operativos

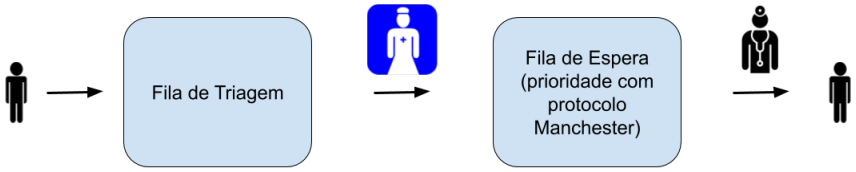
Exame prático NM (Ano lectivo de 2019/2020 – v2) 15 de Janeiro de 2020

Nome: _____ NMec: _____

1. A triagem de pessoas que se deslocam a urgências de hospitais segue um protocolo de cores designado por sistema de Manchester. Neste protocolo, existem 5 cores que ditam a urgência de atendimento de cada utente. Essas cores são, por ordem decrescente de prioridade: vermelho (red), laranja (orange), amarelo (yellow), verde (green) e azul (blue).

   Neste exame pretende-se desenvolver um simulador simples que implemente atendimentos em urgências seguindo este protocolo. Nesse sentido é fornecido código que implementa aspectos sequenciais da simulação, sendo o objectivo fazer uma implementação concorrente (com threads ou com processos).

   A figura seguinte mostra as fases de atendimento de um paciente na urgência.



   Tenha em atenção os seguintes aspectos:

   - É fornecida a implementação de um módulo para filas com prioridades (implementado com um array circular). Como a fila de triagem não tem prioridades (é apenas ordem de entrada), pode-se utilizar este módulo com uma prioridade única (exemplificado no código fornecido);
   - O ciclo de vida de um paciente, consiste simplesmente em entrar na urgência (`patient_goto_urgency`) e esperar pelo fim da consulta (`patient_wait_end_of_consultation`).
   - O ciclo de vida dos(as) enfermeiros(as), consiste em estar à espera que a fila de triagem deixe de estar vazia, retirar de lá o paciente mais antigo, atribuir-lhe o código Manchester de prioridade, colocá-lo na fila de espera de consulta, e voltar ao passo inicial.
   - O ciclo de vida dos(as) médicos(as), consiste em estar à espera que a fila de espera deixe de estar vazia, retirar de lá o paciente mais antigo dos mais prioritários, fazer a consulta (simulada com uma espera de um tempo aleatório), indicar ao paciente que a consulta está terminada, e voltar ao passo inicial.
   - Na implementação da fila com prioridade, fez-se com que de cada vez que sai um elemento da fila, a prioridade dos restantes elementos é aumentada em uma unidade (para garantir que, mais cedo ou mais tarde, todos serão atendidos). Note que este comportamento é irrelevante para o código a implementar.
   - O código fornecido contém a biblioteca de concorrência fornecida durante as aulas. Pode ver a documentação executando um browser sobre o ficheiro `doc/index.html` (ex: `firefox doc/index.html&`). Esta biblioteca resolve o problema da programação defensiva na utilização das bibliotecas nativas de threads, mutexes, variáveis de condição, processos, memória partilhada e semáforos.

- O código fornecido faz uso de asserções, expressas com as macros `require`, `ensure` e `check` que indicam as condições que se têm de verificar num código correcto.

Implemente esta simulação seguindo os seguintes passos:

(a) Altere o módulo da fila prioritária (pfifo) por forma a que passe a ser um módulo partilhado seguro. Nesse sentido, tenha em atenção que algumas asserções passam a ser pontos de sincronização condicional na utilização das funções (a regra geral é muito simples: isso tem de acontecer sempre que a condição puder depender de outra thread/processo).

(b) Implemente a entidade activa paciente (`patient`).

(c) Implemente a entidade activa enfermeiro(a) (`nurse`).

(d) Implemente a entidade activa médico(a) (`doctor`).

(e) Faça com que a operação de fechar a fila prioritária (`close_pfifo`) termine de forma segura a simulação. As filas devem ser fechadas quando não existam mais pacientes.