Lista de Exercícios INF1316 - Sistemas Operacionais

Questão 1

Para os seguintes processos e aplicando separadamente as politicas de escalonamento FCFS, SJF preemtivo e Round-Robin (Quantum = 3) , desenhe os diagramas de Gantt correspondente e calcule:

- Qual é o tempo médio de execução dos processos?
- Qual é o tempo médio de espera do processos?

Processo	Tempo de chegada	Rajada de CPU
P_0	0	2
P_1	1	7
P_2	5	6
P_3	7	3

Obs: Com relação ao RoundRobin note que caso um processo conclua seu quantum de tempo ao mesmo tempo em que outro processo chega no sistema, o processo que acaba de concluir seu quantum é colocado na fila à frente do processo que chega.

Questão 2

O que são chamadas de sistema? O que diferencia uma chamada de sistema de uma chamada de sub-rotina (função ou procedimento) de um programa em modo usuário?

Questão 3

Porque é importante um escalonador distinguir programas voltados a E/S (I/O bound) dos programas voltados a CPU (CPU bound)?

Questão 4

Considere o fragmento de código abaixo executado, inicialmente, por um processo:

```
for (i=0; i<3; i++) p[i]=fork();
```

Quantos processos filhos são criados ao total? Justifique sua resposta.

Questão 4

Qual, ou quais, entre os algoritmos básicos de escalonamento listados a seguir, podem resultar em inanição (starvation)?

(i) First Come, First Served (FCFS);
(ii) Shortest Job First (SJF);
(iii) Round Robin; e

(iv) Por prioridade fixa.

Para cada algoritmo que POSSA gerar inanição, EXPLIQUE como, SE possível, essa situação pode ser evitada.

Questão 5

Cinco tarefas estão esperando para serem executadas. Seus tempos de execução esperados são 9, 6, 3, 5 e X. Em qual ordem elas devem ser executadas para minimizar o tempo de resposta médio? (Sua resposta dependerá de X.)

Questão 6

Escalonadores RoundRobin em geral mantêm uma lista de todos os processos executáveis, com cada processo ocorrendo exatamente uma vez na lista. O que aconteceria se um processo ocorresse duas vezes? Você consegue pensar em qualquer razão para permitir isso?

Questão 7

Um analista de sistemas foi encarregado de desenvolver um sistema de reserva de passagens aéreas. Os voos e a ocupação dos lugares estão em um banco de dados que é acessado de forma concorrente por processos através de duas primitivas básicas:

```
int GetVacantPlace (int FlightNumberCode, struct DayMonthYear);

void Reservation (int Place, int FlightNumberCode, struct
DayMonthYear);
```

A função GetVacantPlace retorna um código de poltrona livre em um determinado voo e a função Reservation torna indisponível (ocupada) a poltrona identificada por Place.
Esse analista deve se preocupar com problemas de exclusão mútua? EXPLIQUE. Se for o caso, diga como esse problema pode ser resolvido apresentando e explicando sua solução com um pseudo-código.

Questão 8

É possível determinar se um processo é propenso a se tornar limitado pela CPU ou limitado pela E/S analisando o código fonte? Como isso pode ser determinado no tempo de execução?

Questão 9

Para se implementar semáforos, é necessário o mesmo ser capaz de desabilitar e habilitar interrupções? Justifique.

Questão 10

No código abaixo, as mensagens impressas pelos threads são intercaladas ao acaso. Existe alguma maneira de se forçar que a ordem seja estritamente:

```
thread 1 criado,
thread 1 imprime mensagem,
thread 1 sai,
thread 2 criado,
thread 2 imprime mensagem,
thread 2 sai
```

etc

Se a resposta for afirmativa, como? Se não, por que não?

```
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>

#define NUMBER_OF_THREADS    10

void *print_hello_world(void *tid)
{
      /* Esta funcao imprime o identificador do thread e sai. */
      printf("Ola mundo. Boas vindas do thread %d\n", tid);
      pthread_exit(NULL);
}

int main(int argc, char *argv[])
{
      /* O programa principal cria 10 threads e sai. */
      pthread_t threads[NUMBER_OF_THREADS];
      int status, i;

      for(i=0; i < NUMBER_OF_THREADS; i++) {
            printf("Metodo Main. Criando thread %d\n", i);
            status = pthread_create(&threads[i], NULL, print_hello_world, (void *)i);

            if (status != 0) {
                  printf("Oops. pthread_create returned error code %d\n", status);
                  exit(-1);
            }
      }
      exit(NULL);
}
```

Questão 11

Em um sistema com threads, há uma pilha por thread ou uma pilha por processo quando threads em modo usuário são usados? E quando threads de núcleo são usados? Explique.

Questão 12

Seja um servidor web multithread. Se a única maneira de ler de um arquivo html é a chamada de sistema read com bloqueio normal, você acredita que threads de usuário ou threads de núcleo estão sendo usados para o servidor web? Por quê?

Questão 13

Como funciona o exemplo escalonamento por múltiplas filas com realimentação? Qual a relação entre o quantum e o nível de prioridade de cada fila nesse esquema?

Questão 14

Em um sistema operacional, o escalonador de curto prazo está organizado como duas filas, a fila A contém os processos do pessoal do CPD e a fila B contém os processos dos alunos. O algorítmo entre filas é round-robin. De cada 11 unidades de tempo de cpu, 7 são fornecidas para os processos da fila A e 4 para os processos da fila B.
O tempo de cada fila é dividido entre os processos também por round-robin, com fatias de tempo de 2 unidades para todos. A tabela abaixo mostra o conteúdo das duas filas no instante zero. Considere que está iniciando um ciclo de 11 unidades, e agora a fila A vai receber as suas 7 unidades de tempo. Mostre a sequência de execução dos processos, com os momentos em que é feita a troca (diagrama de Gantt).

OBS: Se terminar a fatia de tempo da fila X no meio da fatia de tempo de um dos processos, a cpu passa para a outra fila. Entretanto, este processo permanece como primeiro da fila X, até que toda sua fatia de tempo seja consumida.

Fila	Processo	Duração do próximo ciclo de CPU
A	P1	6
A	P2	5
A	P3	7
B	P4	3
B	P5	8
B	P6	4