



SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE SERGIPE
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
DEPARTAMENTO DE COMPUTAÇÃO – DCOMP



Página 1/2

SISTEMAS OPERACIONAIS
PROF. CALEBE CONCEIÇÃO

Prova 1

*Instrumento avaliativo referente à primeira unidade da disciplina, conforme previsto no plano. Peso: 70% da unidade 1. Responda com atenção. **Anote seu nome também na folha avulsa**, e use-a para rascunho e respostas. Use-a com moderação. Priorize respostas curtas e diretas, gerencie o tempo. Desejo sucesso!*

Aluna(o): _____

Matrícula: _____

Questão 1) (0,5 ponto) Sobre o histórico de desenvolvimento dos Sistemas Operacionais (SOs), preencha os parênteses com numerais ordinais de maneira a refletir a ordem relativa de surgimento de cada dos marcos listados.

- () Sistemas de tempo compartilhado
- () Sistemas multiprogramados
- () Sistemas Operacionais de Código aberto.
- () Monitor residente
- () Sistemas em Batch

Questão 2) (0,5 ponto) As interrupções de hardware e a memória de disco magnético, em oposição às fitas, foram duas inovações em hardware importantes para a criação dos sistemas operacionais multiprogramados. Explique os motivos deles viabilizarem esse avanço.

Questão 3) (1 ponto) Avalie as assertivas abaixo e então marque a alternativa correta sobre a estrutura dos SOs.

- I. A interface de usuário é um dos muitos serviços que os Sistemas Operacionais fornecem, sendo GUI, CLI e Batch os tipos geralmente fornecidos pelos Sistemas Operacionais.
- II. O Bootstrap é um programa armazenado em uma ROM (ou EPROM) que é capaz de localizar o kernel, carregá-lo na memória e dar início à sua inicialização.
- III. Uma forma dos programas passarem parâmetros para uma chamada de sistema é por meio de registradores. A vantagem desse método é que ele não limita a quantidade de parâmetros que podem ser passados.

Quais estão corretas?

- a) Somente I
- b) Somente II
- c) Somente III
- d) Somente I e II

e) Somente II e III

Questão 4) (1 ponto) Sobre os estados dos processos, avalie em V ou F as assertivas que seguem.

- () O primeiro estado de um processo, imediatamente após o comando que dispara a sua execução, é o estado de pronto.
- () Um processo no estado executando volta para o estado de pronto no momento em que executa uma operação de entrada e saída.
- () Quando uma interrupção é gerada no sistema, o processo que está executando passa do estado esperando até que a interrupção seja atendida.
- () O despachante do escalonador é responsável por colocar o processo no estado executando.
- () Ao encerrar sua execução, o processo vai para o estado terminado até que seja desalocado da memória.

Questão 5) (1 ponto) Avalie as assertivas que seguem.

- I. O escalonador de curto prazo, assim chamado por ser executado apenas esporadicamente, seleciona quais processos devem ser carregados na memória, na fila de prontos,
- II. O escalonador de longo prazo, também conhecido como escalonador de CPU, seleciona qual processo deve ser executado e o aloca na CPU.
- III. A técnica de envelhecimento (*Aging*) é aplicada em algoritmos de escalonamento com preempção para evitar o problema de *starvation*.

Quais estão corretas?

- a) Somente I
- b) Somente II
- c) Somente III
- d) Somente I e II
- e) Somente II e III

Questão 6) (1 ponto) O Bloco de Controle do Processo

(BCP) armazena informações associadas a cada processo. Liste 5 (cinco) informações presentes no BCP.

Questão 7) (1 ponto) Sabemos que a decisão de escalonar pode ser demandada quando o processo: 1) troca do estado rodando (running) para o estado esperando (waiting); 2) troca do estado rodando (running) para o estado pronto (ready); 3) troca do estado esperando (waiting) para o estado pronto (ready); 4) termina. Quais desses casos são preemptivos? Explique.

Questão 8) (Total: 1 ponto) Considere o algoritmo *Shortest Job First* (SJF) com preempção (também conhecido como *Shortest Remaining Time First*). **(0,5 ponto)** Desenhe o diagrama de Gantt que representa o resultado da execução do algoritmo para a seguinte configuração de processos, com suas respectivos tempos de chegada na fila de pontos e tempos de pico de CPU, conforme segue. **(0,5 ponto)** Apresente o cálculo do tempo médio de espera nesse caso.

Processo	Tempo de Chegada	Duração do pico
P ₁	0	8
P ₂	1	4
P ₃	2	9
P ₄	3	5
P ₅	18	3

Questão 9) (1 ponto) O que são thread pools e quais as vantagens do uso desse mecanismo?

Questão 10) (1 ponto) Considere um processo de tamanho 115785 bytes, e um tamanho de página de 4 KB. Calcule a quantidade de páginas necessária **(0,5 ponto)** e a fragmentação interna resultante dessa alocação **(0,5 ponto)**. Apresente as contas organizadas.

Questão 11) (Total: 1 ponto) O código ao lado contém uma implementação do problema Produtor/Consumidor usando pthreads apresentado em aula. Sublinhe a linha

que contém a seção crítica **(0,3 pontos)**, descreva o que é condição de corrida **(0,4 pontos)** e porque ela acontece nesse exemplo **(0,3 pontos)**.

```

1  #include <pthread.h>
2  #include <stdint.h>
3  #include <stdio.h>
4  #define N
5  void * consumidor ( void * p ) {
6      int32_t * v = ( int32_t *) ( p ) ;
7      for ( uint32_t i = 0; i < N ; i ++ ) {
8          while ( * v <= 0 ) sched_yield () ;
9          printf("consumidor = %i \n",
10              (*v>0) ? (--(*v)) : (*v) ) ;
11      }
12  }
13  void * produtor ( void * p ) {
14      int32_t * v = ( int32_t *) ( p ) ;
15      for ( uint32_t i = 0; i < N ; i ++ ) {
16          while ( * v >= N ) sched_yield () ;
17          printf ( "produtor = %i \n",
18              (*v<N) ? (++(*v)) : (*v) ) ;
19      }
20  }
21  int main() {
22      int32_t counter = 0;
23      pthread_t consumers [N] , producers [N];
24      for ( uint32_t i = 0; i < N ; i ++ ) {
25          pthread_create( &consumers[i] ,
26              NULL , consumidor ,
27              (void*) (&counter) ) ;
28          pthread_create( &producers[i] ,
29              NULL , produtor ,
30              (void*) (&counter) ) ;
31      }
32      for ( uint32_t i = 0; i < N ; i ++ ) {
33          pthread_join ( consumers[i],NULL);
34          pthread_join ( producers[i],NULL);
35      }
36      return 0;
37  }

```