Capítulo 1 - Introdução

Problema 1:
Quais as principais razões (motivações) para se usar programação concorrente?

Problema 2:
Quais os benefícios que programas concorrentes oferecem sobre programas sequenciais?

Problema 3:
Quais são os estados de um processo? Explique cada um deles.

Problema 4:
Quais as diferenças entre um processo e um thread?

Problema 5:
Quais as principais propriedades de programação concorrente?

Problema 6:
Comente a relação entre as propriedades de segurança (security) e vivacidade (liveness) em um sistema concorrente

Problema 7:
Como você pode verificar manualmente uma propriedade de segurança?

Problema 8:
Quais são as principais falhas de vivacidade? Explique.

Capítulo 2 - Revisão de Sistemas Operacionais

Problema 9:
Cite cinco serviços oferecidos por um Sistema Operacional.

Problema 10:
O que são System Calls? Cite alguns exemplos.

Problema 11:
Descreva as principais arquiteturas de sistemas operacionais.

Problema 12:
Quais são as ações tomadoas pelo kernel do suistema operacional para realizar a troca de contexto entre processos?

Problema 13:
Explique a solução de Peterson.

Problema 14:
O que são Semáforos? Mostre um exemplo de acesso à região crítica usando semáforo.

Problema 15:
Mostre uma solução de produtor-consumidor com buffer limitado usando semáforo.

Problema 16:
Quais os principais algoritmos de escalonamento? Explique cada um deles. Quais poderão causar 'starvation'?

Capítulo 3 - Simultaneidade e Paralelismo

Problema 17:
Qual é a diferença entre simultaneidade e paralelismo?

Problema 18:
Quais as formas de se implementar o paralelismo de processos? Explique.

Problema 19:
Quais as maneiras que processos concorrentes podem se comunicar e sincronizar?

Problema 20:
Quais são as três técnicas de sincronização? Explique e dê um exemplo.

Problema 21:
Para que serve a Espera Ocupada? Explique seu funcionamento. Exemplifique um exemplo de uso?

Problema 22:
Para que serve o semáforo? Explique o seu funcionamento. Qual as vantagens do semáforo comparado com espera ocupada ?

Problema 23:
Para que serve o monitor? Explique o seu funcionamento.

Problema 24:
Como você pode implementar um semáforo com monitores?

Problema 25:
Como você implementar monitores usando semáforos?

Problema 26:
Explique as formas de passar mensagem síncrona e assíncrona.

Capítulo 4 - Modelagem e Threads

Problema 27:
O que são FSP (Finit State Process) e LTS (Labelled Transition System)? Qual a diferença?

Problema 28:
Como podemos decidir se um sistema precisa de máquinas separadas, processadores, processos ou threads? Quais são os trade-offs (custos versus benefícios) de cada tipo de implementação?

Problema 29:
Como você pode especificar um FSP que repetidamente imprime 'Olá', mas pode parar a qualquer momento quando aperta Esc no teclado?

Problema 30:
Faça um programa em pseudocódigo que troque o conteúdo de duas posições de memória. Defina a função Ex(a,b) que realize uma função indivisíveis das instruções de atribuição a seguir:

```
temp: = a;
a: = b;
b: = temp;
```

O programa deve usar exclusão mútua.

(a) Discutir a segurança (ausência de bloqueio), da solução com exclusão mútua.

(b) Generalize para n processos.

Capítulo 5 - Sincronização

Problema 31:
O que é uma condição de corrida (race condition)?

Problema 32:
O que é uma operação atômica? Como podemos transforma um conjunto de operações críticas em uma operação atômica?

Problema 33:
Como podemos implementar sincronização entre vários processos para proteger uma região crítica (ou recurso compartilhado)?

Problema 34:
O que é uma Espera Ocupada? Quais suas vantagens e desvantagens?

Problema 35:
O que é Starvation? Mostre um caso onde isso acontece.

Problema 36:
O que é Deadlock? Mostre um caso onde isso acontece.

Capítulo 6 - Vivacidade e Métodos Protegidos

Problema 37:
Que tipos de problemas de vivacidade (liveness) pode ocorrer em programas concorrentes?

Problema 38:
Qual é a diferença entre a fome (starvation) e impasse (deadlock)?

Problema 39:
O que é uma escolha justa? Por que precisamos dela?

Problema 40:
Como você pode detectar o impasse? Como você pode evitá-lo?

Problema 41:
Como podemos avaliar a propriedade progresso em um sistema concorrente?

Problema 42:
Quais são as condições necessárias e suficientes para o impasse (deadlock)?

Problema 43:
Descreva uma técnica para resolver o problema de deadlock do jantar dos filósofos e garantir a propriedade de vivacidade?

Problema 44:
O que é um método protegido?

Problema 45:
Por que geralmente é melhor usar notifyAll() em vez de notify()? E quando é melhor usar notify()?

Capítulo 7 - Objeto Condição

Problema 46:
O que é um objeto condição?

Problema 47:
Mostre um exemplo de objeto condição em Java ?

Problema 48:
O que é o *problema do monitor aninhado*? Explique.

Problema 49:
Quais as técnicas para evitar o *problema do monitor aninhado*?

Problema 50:
O que é o objeto Permit? De um exemplo de aplicabilidade.

Capítulo 8 - Propriedades de Concorrência

Problema 51:
O que é um método concorrente?

Problema 52:
Quais as propriedade de métodos concorrentes? Descreva cada uma delas.

Problema 53:
Descreva as regras gerais de um problema de *Leitores e Escritores*. De um exemplo de aplicação prática.

Problema 54:
Qual problema pode surgir se definirmos uma política de dar prioridade maior para os escritores?

Problema 55:
O que acontece se for implementado uma política que permite a entrada de novos leitores mesmo quando houver escritores em espera? E se a política for o contrário, não permitir a entrada de novos leitores quando houver escritores em espera?

Problema 56:
Cite algumas políticas para implementar Leitores e Escritores corretamente?

Problema 57:
Escreva o pseudocódigo de Leitores e Escritores usando monitores e condições?

Capítulo 9 - Redes de Petri

Problema 58:
O que é uma rede de Petri e qual a sua utilidade? Quais são seus principais elementos?

Problema 59:
Como uma rede de Petri pode ser especificada formalmente?

Problema 60:
O que é um grafo de acessibilidade de uma rede de Petri? Qual sua utilidade?

Problema 61:
Desenhe o modelo de uma rede de Petri que simule o funcionamento de um elevador em um prédio de 4 pisos. O elevador tem apenas um comando interno que indica subir um andar ou descer um andar. O comando do elevador não pode permitir a ida para o subsolo nem para o quinto piso. Avalie visualmente se essa rede é limitada, viva e alcançável.

Problema 62:
Desenhe o modelo de uma rede de Petri que simule o funcionamento de um semáforo em um cruzamento. Os semáforos mudam a cor vermelho -> verde -> amarelo a partir de um comando manual localizado em cada via do cruzamento. A única restrição é que os dois semáforos nunca poderão ficar com as cores verde ou amarelo acesas simultaneamente (sob o risco de provocar um acidente). Avalie visualmente se essa rede é limitada, viva e alcançável.

Problema 63:
Desenhe o modelo de uma rede de Petri que simule o funcionamento de uma máquina automática de venda de chocolate. Essa máquina aceita apenas moedas de 5 e 10 centavos, e não dá troco (o comprador tem que colocar o valor exato). De acordo com o valor depositado na máquina o comprador pode escolher uma barra de chocolate que custa 15 centavos ou 20 centavos. Avalie visualmente se essa rede é limitada, viva e alcançável.

Capítulo 10 - Computação Paralela

Problema 64:
Qual a diferença entre programação concorrente e computação paralela?

Problema 65:
Quais são os tipos de paralelismo em computação paralela?

Problema 66:
Qual o fator principal para avaliar o aumento de desempenho de uma aplicação se processar em múltiplos processadores ou máquinas?

Problema 67:
O que diz a Lei de Amdahl?

Problema 68:
Qual a metodologia para transformar um programa sequencial em paralelo?

Problema 69:
Um programa leva 10s para executar em um sistema com um único processador. O tempo de execução em um sistema com 8 processadores é de 5s. Pergunta-se: qual é o fator de aceleração (speed-up) obtido e qual a eficiência de cada processador?