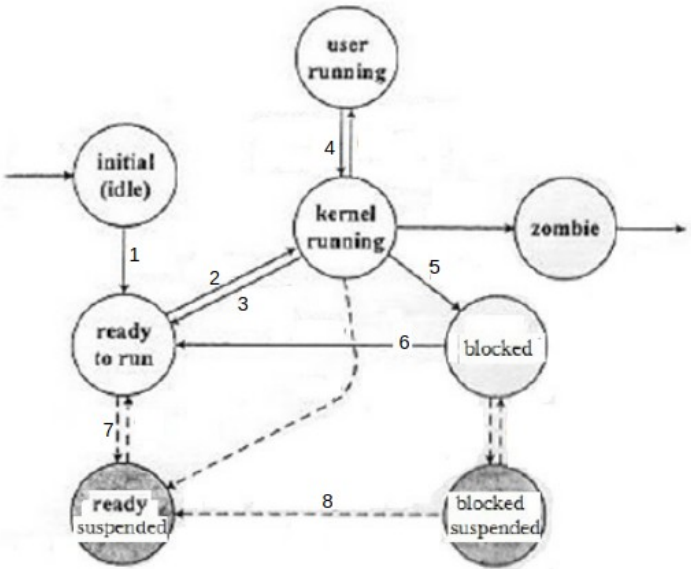
UFES – Departamento de Informática

1ª. Prova de Sistemas Operacionais – 2018/2 – Profª. Roberta L. Gomes

1) **(2,0)** Dado o diagrama de estados do Unix ilustrado abaixo, escreva em que situações (e por que) ocorrem cada uma das transições enumeradas (1 a 8).



2) **(3.0)** Suponha que um sistema operacional deva gerenciar a execução de diferentes tipos de processos, atendendo aos seguintes requisitos:

(i) O número de processos interativos é relativamente pequeno. Eles deverão ter alta prioridade, mas são aceitáveis pequenos atrasos.

(ii) O número de processos de tempo-real é muito pequeno, mas além de ter alta prioridade, precisam de garantias de tempo de resposta.

(iii) Processos podem ser I/O bound em um momento de sua execução, e se tornarem CPU bound em outro momento de sua execução. Enquanto forem I/O bound, devem ter prioridade alta. Enquanto forem CPU bound, devem ser tratados conforme o requisito.

(iv) Quando os processos CPU bound são os únicos que estão prontos, a sobrecarga do sistema operacional deve ser minimizada.

(v) Nenhum processos deve sofrer *Starvation*.

Defina uma política de escalonamento descrevendo os tipos de estrutura de dados e algoritmos que seu escalonador deve usar para garantir os requisitos listados acima. Explique como ele garantirá cada um dos requisitos. Caso necessário, você também pode fazer considerações sobre outras características do Sistema Operacional para poder garantir algum(s) desses requisitos.

3) **(2,0)** No UNIX, um processo pode encontrar-se no estado *Kernel Running* (CPU em Kernel Mode) enquanto o sistema pode apresentar dois contextos de execução, *Process Context* e *System Context*. Explique a diferença entre esses dois contextos de execução.

4) **(3,0)** Considere o trecho de código a seguir:

```
1       int c = 5;
2       pid_t pid = fork();
3       if (pid == 0) {
4          c += 5;
5       } else {
6          pid = fork();
7          c += 10;
8          if (pid)
9             c += 10;
10      }
11      fork();
12      printf("%d\n", c);
```

a) Desenhe uma árvore representado a hierarquia de processos criados ao executar esse programa (lembre-se de incluir o primeiro processo criado).

b) Escreva uma saída possível para esse programa.

Boa prova!!!