isec Engenharia

Sistemas Operativos

2022/23

DEIS – Engenharia Informática

Prova escrita
Parte I
Época especial
27 / janeiro / 2023

Duração 30 minutos

Parte 1

1. (10%) Considere que existe um ficheiro que não lhe pertence e não lhe dá permissões de abertura, mas que quer mesmo abrir e ver. O problema é que as funções de sistema internamente verificam a sua identificação e recusam-lhe a abertura.
   Não é o root do sistema, mas sabe programar muito bem e decide implementar de raiz, para correr dentro do seu processo, um conjunto de funções para chegar diretamente ao disco rígido e ao conteúdo do ficheiro, contornando o sistema.
   Explique se existe alguma coisa que o impeça, indicando os detalhes técnicos envolvidos caso haja mesmo algo.

2. (10%) Explique de que forma é que o endereçamento virtual é utilizado para garantir o isolamento entre processos. Deve dar os detalhes acerca de como funciona. Nota: meramente debitar slides decorados não terá qualquer valor.

3. (20%) Considere um programa para traduzir código morse para ascii. O programa tem duas threads: **traduz** e ***imprime***.

   A thread ***traduz*** recorre a uma função **recebeProxSimb** (já implementada) que obtém o próximo símbolo morse (. ou -), acumulando-os num buffer interno estático (preserva os valores entre chamadas). Quando os símbolos morse armazenados no tal buffer interno estático já formam uma letra, então a função devolve essa letra, limpando o buffer, e colocando-a na variável partilhada **letra**; caso contrário a função devolve '\0' e continua a acumular símbolos.

   A thread ***imprime*** verifica se o caracter presente na variável letra é diferente de '\0'. Se for, está-se perante uma nova letra ascii que deve ser impressa, e nesse caso o conteúdo de letra é reposto a '\0'.

   Existe o seguinte código:

   traduz

```
int aux;
while (...) {
   aux = recebeProxSimb();
   if (aux != '\0')
      letra = aux;
}
```

   imprime

```
while (...) {
   while (letra == '\0');
   printf("nova letra %c, letra);
   letra = '\0';
}
```

   a) Verifique se existe algum problema com a estratégia e implementação, explicando o que encontrar e solucionando-os. Independentemente do que decidir fazer, tem que se manter as duas threads e não pode mexer na função já implementada recebeProxSimb().

   b) Explique se se poderia duplicar a thread recebeProxSimb (por exemplo: ter duas threads traduz e uma imprime). Assuma que o código é o mesmo que foi descrito, sem qualquer interferência da alínea a).

isec Engenharia

Sistemas Operativos

2022/23

DEIS – Engenharia Informática

Prova escrita
Parte II
Época especial
27 / janeiro / 2023

Duração **1h30 + 15 minutos**

Com consulta indicada

Parte 2

1. (5%) Utilizando apenas uma linha de comandos de Unix, apresente no ecrã o tamanho e o nome do 2º, 3º e 4º maiores ficheiros, do tipo pipe, existentes na pasta /tmp. A sua solução deve funcionar a partir de qualquer ponto do sistema de ficheiros, independentemente de onde é executada.

2. (5%) Utilizando apenas uma linha de comandos de Unix, acrescente ao ficheiro ultimos.txt o nome completo dos três últimos utilizadores que foram adicionados ao sistema, cujo login começa por uma vogal. Os nomes devem estar por ordem alfabética. A sua solução deve funcionar a partir de qualquer ponto do sistema de ficheiros, independentemente de onde é executada.

3. (25%) Considere um sistema informático para Unix para gerar música sintética. Já existem três instrumentos musicais, piano, guitarra e bateria, cada um deles implementado sob a forma de uma aplicação de linha de comandos, cada uma com o nome do instrumento respectivo, que geram notas musicais no formato ***LN***, onde ***L*** corresponde a uma nota musical/acorde de A a F, e ***N*** a um número entre 1 a 9 (por exemplo: C1, D3, etc.), e as envia para a consola ("ecrã"). As notas geradas por cada aplicação e o intervalo entre elas são aleatórias.

   Pretende-se apanhar as notas geradas pelos instrumentos e obter como resultado final uma cadeia de caracteres que identifica o **instrumento** (piano – "p", guitarra – "g" e bateria – "b") e a **nota** gerada (C1, D3, etc.). Exemplo: "pC2gD1pB3bF1".

   Utilizando mecanismos do sistema operativo, construa a aplicação "play" que coloca em execução cada uma das aplicações instrumento (todas em paralelo), recolhe a informação gerada por cada uma delas e constrói, de forma incremental, a cadeia de caracteres pretendida. Cada música terá uma duração fixa, em segundos, especificada através da variável de ambiente TEMPO, findo a qual todos devem terminar. As aplicações instrumento podem ser terminadas através do envio do sinal SIGINT. No final, a aplicação "play" deverá mostrar a música gerada no monitor, seguida do número de notas que a constituem.

   NOTA: Não é permitida utilização da chamada ao sistema SELECT para a resolução deste exercício.

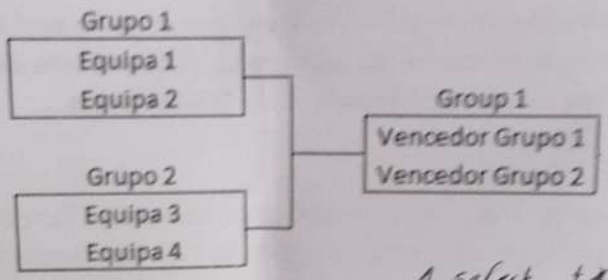
   EXEMPLO:

   ```
   $ ./play
   pC2 gD1 pB3 bF1
   4 notas/acordes
   ```

4. (25%) Pretende-se um sistema de gestão de jogos de basquete composto por um **programa gestor** que organiza campeonatos e **n programas equipa**.

O **programa gestor** fica inicialmente à espera de inscrições dos vários **programas da equipa** via named pipes. Após receber as **quatro** equipas, a prova é iniciada. Depois, se mais equipas tentarem entrar, o **programa gestor** deve avisar que já está a decorrer um campeonato, e o programa cliente terá que tentar, mais tarde, novo pedido de inscrição.

Um campeonato é realizado em duas rondas (ver figura), com eliminatória na primeira ronda (ou seja, "meia final" e "final"). O **programa gestor** distribui de forma aleatória, as quatro equipas pelos dois jogos e envia a distribuição (com quem joga cada um) aos quatro **programas equipa** inscritos naquele campeonato.



Além de distribuir as equipas, o programa gestor é responsável por dar ordens de início e fim de jogo (mensagens por named pipes cujo conteúdo deve propor e definir) e guardar o nome do jogador que mais pontos efetuou em todos o campeonatos (por cada ponto marcado a equipa envia o nome do jogador que marcou ao gestor). O fim do jogo é ordenado pelo gestor decorridos 40 minutos após o início ou por ordem do utilizador do gestor.

Os **programas equipas** contactam diretamente a equipa adversária (são responsáveis por toda a comunicação equipa - equipa), e todo o jogo é realizado entre os dois. No final de cada jogo, o **programa equipa** vencedor, deverá enviar uma mensagem com o resultado para o **programa gestor** (não existem empates).

Só existirá um campeonato em simultâneo, mas os jogos entre as 4 equipas decorrem ao mesmo tempo. Quando um campeonato terminar, o gestor repete todo o processo, ficando a aguardar por novas 4 equipas.

O **programa gestor** disponibilizará ao seu utilizador a hipótese de escrever comandos em qualquer instante durante todos os jogos. Os comandos são "**encerra**" – termina o campeonato imediatamente, avisando as equipas e "**ponto**" – apresenta o nome, número de pontos marcados do melhor marcador do campeonato até agora.

Implemente apenas o código do **programa gestor**;

- Toda a comunicação entre os programas gestor e equipa (nos dois sentidos) deverá ser realizada via named pipes;
- **Não pode utilizar threads neste exercício**;
- O nome do named pipe do programa gestor é obtido pela variável de ambiente GESTORNP;
- Deve propor e implementar (apenas no lado do gestor) uma forma do gestor saber o nome do named pipe de cada equipa

Nota: deve declarar todas as variáveis e estruturas de dados que utilizar. Não é necessário especificar #includes.