



ALUNO: _____

- 1) (2,0 Pontos) Sobre conceitos básicos de sistemas operacionais, responda:
- Como dois ou mais processos podem executar em paralelo em uma máquina que possua um processador com apenas um núcleo?
 - Porque, atualmente, a maioria dos sistemas operacionais **exige** que as aplicações utilizem *system calls* para executar as funções desejadas?

- 2) (2,0 Pontos) Sobre processos e threads, responda:
- Qual a principal diferença entre threads de núcleo e threads de usuário?
 - Suponha que você precisa desenvolver uma aplicação que execute duas tarefas independentes e que estas duas tarefas requerem muito uso de processamento. A execução paralela das tarefas não é um requisito da aplicação. Em que cenário(s) não seria vantajoso implementar estas tarefas como threads? Justifique.

- 3) (2,0 Pontos) Suponha que os seguintes processos chegaram para execução nos tempos indicados. Cada processo rodará a quantidade de tempo listada na tabela.

Processo	Tempo de Chegada (ms)	Tempo de Execução (ms)	Prioridade
A	62	5	1
B	36	10	2
C	19	12	2
D	26	14	2
E	89	30	1

Qual o tempo médio de espera para estes processos quando são utilizados os algoritmos de escalonamento abaixo. Considere que o sistema operacional gasta 1 ms para realizar a troca de contexto.

- PMTR (Próximo de Menor Tempo Restante).
- Escalonamento circular com prioridade estática e *timeslice* de 4 ms.

- 4) (2,0 Pontos) Sobre deadlock, responda:
- Por que, em um sistema que possua apenas um recurso de cada tipo, é melhor utilizar o método de detecção de deadlocks Grafos ao invés do método baseado em Matrizes?
 - Análise o sistema abaixo, considerando que existem N dispositivos do mesmo tipo, e determine o menor valor de N para garantir que não haverá deadlock. Mostre os cálculos realizados.

Processo	Dispositivos alocados	Máximo de requisições
1	3	8
2	3	9
3	4	9
4	2	4

- 5) (2,0 Pontos) Em uma aplicação que controla saldo bancário em contas-correntes, dois processos compartilham uma região de memória onde estão armazenados os saldos dos clientes A e B. Os códigos dos processos são mostrados na figura abaixo. Suponha que os valores dos saldos de A e B sejam, respectivamente, 500 e 900, antes de os processos executarem.

- Quais os valores finais dos saldos dos clientes se a sequência temporal de execução das operações for: 1a, 2a, 1b, 2b, 1c, 2c, 1d, 2d, 1e, 2e, 1f, 2f?
- Utilizando semáforos, proponha uma solução que garanta a integridade dos saldos e permita o maior compartilhamento possível de recursos entre os processos.

Processo 1 (Cliente A)

```
/* saque em A */
1a. x := saldo_do_cliente_A;
1b. x := x - 200;
1c. saldo_do_cliente_A := x;

/* deposito em B */
1d. x := saldo_do_cliente_B;
1e. x := x + 100;
1f. saldo_do_cliente_B := x;
```

Processo 2 (Cliente B)

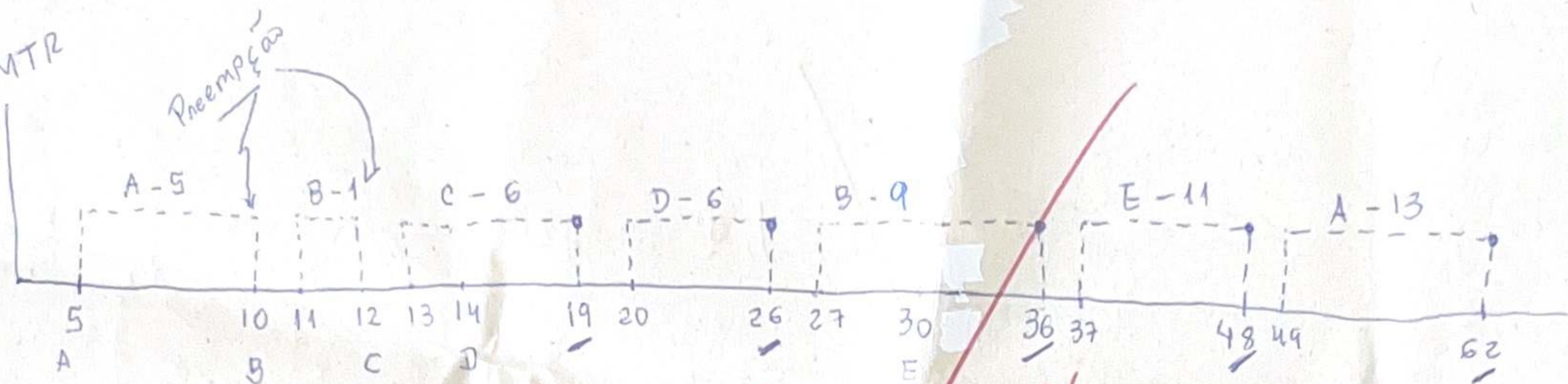
```
/* saque em A */
2a. y := saldo_do_cliente_A;
2b. y := y - 100;
2c. saldo_do_cliente_A := y;

/* deposito em B */
2d. y := saldo_do_cliente_B;
2e. y := y + 200;
2f. saldo_do_cliente_B := y;
```


03)

a)

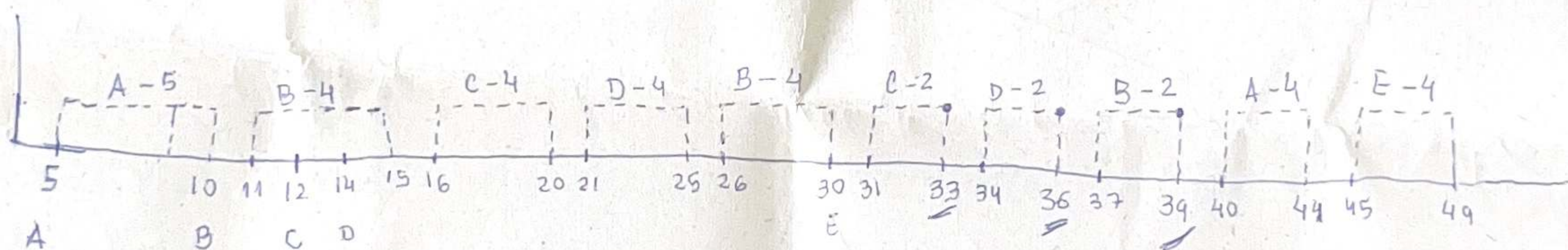
PMTR



$$TME = \frac{(62 - 5) + (36 - 10) + (19 - 12) + (26 - 14) + (48 - 30)}{5} = \frac{57 + 26 + 7 + 12 + 18}{5} = \frac{120}{5}$$

$$TME = 24 \text{ ms}$$

b)



$$TME = \frac{(70 - 5) + (39 - 10) + (33 - 12) + (36 - 14) + (68 - 30)}{5}$$

$$TME = \frac{65 + 29 + 21 + 22 + 38}{5} = 35 \text{ ms}$$

Ana Beatriz Araújo Leite

01).

a) Com o conceito de timesless é possível "fatiar" o tempo que um processo tem de acesso à CPU. Ou seja, ao invés de ~~ficar~~ esperar um processo finalizar, para iniciar outro, os processos ficam ~~alternando~~ o que resulta na diminuição no tempo de espera de cada processo, e, consequentemente, cria a sensação de "paralelismo".

b) ~~Porque~~ Porque as system calls são responsáveis pela integração do programa com o hardware, proporcionando que o ~~programa~~ usuário não precise requisitar recursos ou serviços diretamente ao hardware.

~~02). a) As threads de núcleo são mais "pesadas", pois múltiplas threads podem trabalhar de forma concorrente, enquanto as threads de usuário são mais "leves", possuindo limitações nesse sentido.~~

~~b) Caso essas aplicações não envolvam cálculos matemáticos muito complexos que exijam grande processamento de dados (CPU-Bound), por exemplo, um programa de ordenação precisa fazer constantes cálculos e comparações, exigindo constante acesso à CPU.~~

~~Caso essas aplicações não envolvam constantes requisições de entrada ou saída de dados (I/O Bound), por exemplo um programa em que o usuário necessita constantemente fazer a inserção de novos dados.~~

Ana Beatriz Araújo Leite

04)

a) Maior precisão de resultados. Apesar de o método das matrizes poder ser utilizado em qualquer situação e ser considerado ~~mais simples~~ ^{Q, 5}, o método dos grafos possui a vantagem da precisão. → Como Assam?

b) Utilizando o método baseado em matrizes:

$$C = \begin{bmatrix} 1 & 3 \\ 2 & 3 \\ 3 & 4 \\ 4 & 2 \end{bmatrix}$$

$$R = \begin{bmatrix} 5 & 1 \\ 6 & 2 \\ 5 & 3 \\ 2 & 4 \end{bmatrix}$$

$$E = [N]$$

$$A = [N - 12]$$

$$T = [12]$$

O valor de N precisa ser, no mínimo 14, pois o processo que exige menos requisições é o processo 4 (P4), então podemos calcular o valor de n da seguinte forma:

$$N - 12 = 2$$

$$\boxed{N = 14}$$

Dessa forma, podemos garantir que ~~o sistema~~ não haverá deadlock no sistema.

05).

a)

ANTES DE OS PROCESSOS EXECUTAREM:

SALDO A = 500 | SALDO B = 900

SAQUE A \rightarrow 1A) $x = 500$

SAQUE A \rightarrow 2A) $y = 500$

SAQUE A \rightarrow 1B) $x = 300$

SAQUE A \rightarrow 2B) $y = 400$

SAQUE A \rightarrow 1C) SALDO-CLIENTE-A = 300

SAQUE A \rightarrow 2C) SALDO-CLIENTE-A = 400

SALDO FINAL DE A = 400

DEPÓSITO B \rightarrow 1D) $x = 900$

DEPÓSITO B \rightarrow 2D) $y = 900$

DEPÓSITO B \rightarrow 1E) $x = 1000$

DEPÓSITO B \rightarrow 2E) $y = 1100$

DEPÓSITO B \rightarrow 1F) SALDO-CLIENTE-B = 1000

DEPÓSITO B \rightarrow 2F) SALDO-CLIENTE-B = 1100

SALDO FINAL DE B = 1100