Sistemas Operacionais

Orientações para a lista de exercícios

- Data de entrega: **10/09/2019 -** Por email **OU** presencialmente durante a aula
    - Email: **airtonbjunior@gmail.com**
    - Caso o envio seja feito por email, o arquivo deverá estar em formato PDF
    - Não serão aceitas entregas após a data estabelecida
    - Caso os códigos-fonte sejam disponibilizados em suas respectivas contas do github, será atribuído um **ponto extra** para a lista de exercício. Lembrando que o ponto é extra, sendo um complemento para a nota final do bimestre.

Questões teóricas sobre o conteúdo visto até o momento

1. O que é um Sistema Operacional?
2. Quais os principais tipos de sistemas operacionais?
3. Como podemos classificar os sistemas multitarefas?
4. O que são algoritmos de escalonamento de processos? Cite e explique, de forma sucinta, pelo menos 3.
5. O que são escalonadores preemptivos? E não-preemptivos?
6. O que é um Processo?
7. Quais os estados que um processo pode ocupar em um sistema baseado no modelo de processos?
8. Quais as transições possíveis entre os estados de um processo?
9. Explique como funciona a prioridade no Sistema Operacional GNU/Linux. Quais as prioridades disponíveis? Como alterá-las?
10. Explique como funciona a prioridade no Sistema Operacional Windows. Quais as prioridades disponíveis? Como alterá-las?
11. Explique o conceito de *starvation*.
12. O que são *threads*?
13. Quais as principais diferenças entre processos e *threads*?
14. Cite ao menos 3 benefícios de utilização de *threads*.
15. O que são condições de corrida?
16. Explique a hierarquia de memória.

Questões práticas

As questões práticas poderão ser codificadas na linguagem de maior preferência/afinidade. Importante salientar que há um **ponto extra** caso os códigos sejam disponibilizados no github. Caso tiverem dúvidas de como criar um repositório e fazer o commit dos códigos no github, façam a leitura da documentação oficial da ferramenta ou pesquisas em fóruns na Internet.

1. Criar um programa que utilize o conceito de *threads*. Rode, pelo menos, duas *threads* simultâneas.
2. Criar a seguinte função que simula o gerenciamento de memória do Sistema Operacional

- A função recebe como parâmetro a quantidade de espaços de memória que um processo precisa para executar;
- A função retorna a primeira posição livre que pode ser utilizadas pelo processo;
- A memória a ser considerada será disponibilizada no arquivo **memoria.csv**. O valor 1 representa uma célula de memória ocupada e o valor 0 representa célula livre. Os valores serão separados por vírgulas.
    - Vocês podem representar esses valores diretamente em alguma estrutura de dados (array, por exemplo) no código, ou fazer a leitura do arquivo texto para carregar essa estrutura;
- Tome como exemplo a Imagem 1 abaixo, que representa a organização de uma memória, onde os espaços em azul estão sendo utilizados por outros processos
    - Como exemplo de representação da memória abaixo, teríamos os seguintes valores em nossa estrutura de dados: 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1
    - Caso seja passado como parâmetro para a função o valor 3, que representa a necessidade de ocupar 3 espaços na memória, deve ser retornado o valor 7, que é o endereço de início do espaço que pode suportar esse processo;
    - Caso seja passado como parâmetro um valor maior que o disponível na memória atual (valor 6, que representa a necessidade de 6 espaços, por exemplo), a função deve retornar uma mensagem de que não há espaço disponível para o processo OU retornar o valor -1.

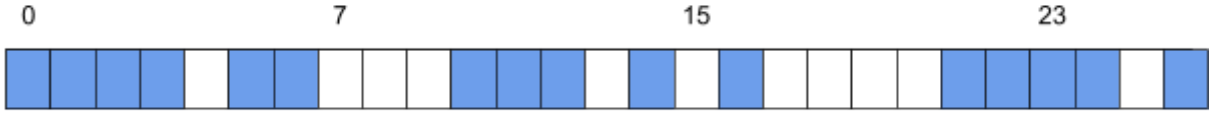

Imagem 1

Observações importantes:

- Os programas desenvolvidos para a solução das questões práticas serão discutidos na aula do dia 10/09/2019. Se possível, levem seus computadores pessoais ou o programa em um pendrive/HD externo para apresentação.
- Dúvidas enviar email para **airtonbjunior@gmail.com** e poderão ser sanadas, também, na aula do dia 05/09/2019.