UNIOESTE – Universidade Estadual do Oeste do Paraná
Curso de Ciência da Computação
Disciplina de Sistemas Operacionais
Prof. Newton Spolaôr

Projeto do 1º bimestre – Processos – valor: 100 pontos

Data da Entrega: 13/10/2020, até as 23:59h por email (newton.spolaor@unioeste.br) ou chat privado com o professor no Microsoft Teams.

Observações:

- Em caso de identificação de plágio, a nota 0 será atribuída para todos os envolvidos.
- Cada dia de atraso na entrega implica na perda de 50 pontos.
- Nota: a entrega do trabalho vale como presença para as quatro aulas práticas assíncronas dos dias 19/09 e 03/10/2020.

Especificação do trabalho:

1. Equipes de até dois acadêmicos (preferencialmente dois acadêmicos) que irão desenvolver um grupo de algoritmos e códigos-fonte com documentação, bem como uma apresentação na forma de um seminário curto.
2. Como parte do trabalho, deve ser escrito um programa que simula a execução de dois algoritmos de escalonamento, sendo um preemptivo e um não preemptivo. Nesse programa, considere os seguintes pontos:
    - Ao executar o programa, deve-se ler um arquivo texto de entrada com as seguintes informações:
        - Quantidade de processos que se deseja escalonar (N);
        - Parâmetros do algoritmo de escalonamento, como tamanho do quantum para um algoritmo como *Round Robin*;
        - Informações inerentes a cada processo que será escalonado, como ID de processo.
    - As informações de cada processo devem ser modeladas no programa por meio de uma estrutura (*struct* ou *record*) denominada *Process Control Block* (PCB). Assim, dados N processos, deve existir uma lista de N instâncias de PCB. Além de conter as informações fornecidas no arquivo texto de entrada para um determinado processo i, o PCB do processo i deve registrar o estado do processo correspondente. Ao todo, o PCB deve conter pelo menos três campos citados em aula.
    - Cada algoritmo de escalonamento deve trabalhar sobre uma lista de processos prontos (estado “pronto”). A lista pode ser implementada como um vetor ou uma lista encadeada. Cada elemento da lista pode ser o PCB de um processo.
    - O processo selecionado para execução pelo algoritmo de escalonamento não precisa executar de fato em um computador. A ideia é que esse processo seja executado com o apoio de outro programa, como um terminal de comandos, que está fora do escopo do presente trabalho. No entanto, o estado do processo selecionado para execução deve ser “em execução”.

- Para cada algoritmo de escalonamento, gere um arquivo de saída que registre o ID de cada processo selecionado para execução. A cada processo selecionado para execução, apresente também o ID dos processos que ocupam a lista de processos prontos e, se necessário, informações específicas do algoritmo de escalonamento, como o tempo restante para execução do processo no algoritmo *Shortest Remaining Time Next*. Quando um processo encerrar, ele deve ser retirado da lista de processos prontos e o usuário deve ser notificado por meio do arquivo de saída. O arquivo de saída deve registrar informações desde o início da execução do primeiro processo até o final da execução do último processo. Após a conclusão do último processo, inclua nesse arquivo o valor de uma medida para avaliação do algoritmo de escalonamento. Uma das medidas que podem ser usadas nesse sentido consiste no tempo médio de resposta dos processos. A quantidade de trocas de contexto é outra medida alternativa.

3. Como parte do trabalho, apresente um algoritmo e a simulação correspondente para resolver um problema clássico de comunicação entre processos: leitores e escritores, barbeiro dorminhoco ou jantar dos filósofos. O algoritmo deve considerar um ou mais semáforos. Na sua simulação, devem ser usados no mínimo cinco processos. Exceto pelo problema do jantar dos filósofos, devem existir processos de cada papel. Por exemplo, no caso do problema de leitores e escritores, ao menos um processo deve ser leitor e ao menos um deve ser escritor. Essa simulação pode ser representada em uma animação ou documento eletrônico. Se preferir fazer a simulação em documento manuscrito, digitalize o mesmo e anexe uma imagem legível em documento eletrônico. Independente da modalidade escolhida, cada passo do algoritmo deve ser ilustrado, assim como o valor das variáveis e constantes consideradas nesses passos, incluindo o(s) semáforo(s).
4. Como parte do trabalho, apresente um exemplo de código Java ou C que utiliza a ideia de monitor. Caso alguma biblioteca adicional seja necessária, inclua ela no código. Esse código deve ser completo, de modo que possa ser compilado e executado em um computador com máquina virtual Java ou compilador C instalados.
5. A equipe deverá ainda documentar manualmente as principais funções dos códigos-fonte do projeto (itens 2 e 4). Essa documentação inclui comentários, pré e pós-condição de cada função. Um exemplo de documentação que pode servir como inspiração é verificado em: http://www.cplusplus.com/reference/cstdlib/abs/. O objetivo é que o leitor do código tenha uma ideia detalhada sobre o que cada uma dessas funções realiza com apoio desses comentários.
6. Até o final do prazo de entrega, a equipe deverá enviar ao professor um arquivo zipado contendo:
   - Todo o código-fonte correspondente aos itens 2 e 4, além dos passos do algoritmo e ilustração correspondente referentes ao item 3.
   - Arquivo WORD ou similar chamado README_2 e arquivos texto de entrada e de saída para uma execução do programa do item 2. A documentação deve possibilitar o preenchimento manual do arquivo de entrada para teste do programa pelo professor, a execução do programa, e a identificação do nome dos algoritmos de escalonamento implementados. Não esqueça de incluir no README_2 informações da plataforma

(sistema operacional, compilador/interpretador...) que deve ser usada para compilação e execução do programa.

- Arquivo WORD ou similar chamado README_4 referente ao programa do item 4. A documentação deve apresentar a importância do monitor no código-fonte correspondente e possibilitar a sua execução. Esse arquivo também deve relatar um problema que poderia ocorrer se o monitor não fosse usado no programa. Não esqueça de incluir no README_4 informações da plataforma que deve ser usada para compilação e execução.
- Arquivo com nomes dos alunos que fizeram o trabalho e todas referências utilizadas (livros, relatórios, artigos, sites).
- Um arquivo POWERPOINT ou similar a ser usado pela equipe em um seminário. Durante o seminário, a equipe deverá:
    - Informar o nome dos algoritmos de escalonamento selecionados pela equipe para o item 2, do problema clássico adotado no item 3 e da linguagem escolhida no item 4.
    - Demonstrar a execução do programa do item 2. **O software executado pela equipe no seminário deve ser o mesmo entregue dentro do prazo ao professor**. O seminário deve ser organizado para que toda a apresentação dure até 5 minutos.
- Após o encerramento do seminário, o professor pode fazer perguntas à equipe sobre os itens 2, 3 e ou 4. A equipe deverá ter disponível os códigos-fonte e demais arquivos de todo o projeto para esse momento.
- As apresentações dos seminários serão realizadas nos dias 14, 19 e 21/10/2020. **Quanto mais cedo a equipe entregar o trabalho completo, mais tarde fará a apresentação. Porém, uma equipe agendada para mais tarde pode apresentar mais cedo, se assim o desejar, desde que solicite ao professor com antecedência**.

7. Critérios de avaliação:
    - Conhecimento demonstrado na apresentação: 30 pontos;
    - Resolução dos itens 2, 3 e 4 conforme especificação do trabalho: 50 pontos;
    - Execução do simulador referente ao item 2: 10 pontos;
    - Comunicação utilizada na apresentação (clareza, objetividade, qualidade dos slides, respeito ao limite de tempo): 10 pontos;