**imagesUniversidade Federal Rural do Semi-Árido - UFERSA**

**Centro de Ciências Exatas e Naturais - CCEN**

**Departamento de Computação - DC**

**Ciência da Computação**

**Prof. Leiva Casemiro**

# 1.3– Tarefa de Sistemas Operacionais

**INSTRUÇÕES:** Essa atividade pode ser feita individual ou em dupla e os código e respostas correspondentes as questões devem ser enviados em um único arquivo (.zip) até a data estabelecida no SIGAA como parte da nota da Unidade em avaliação.

**Orientações gerais**

* A implementação deve ser feita no MARS versão 4.5, disponível em <http://courses.missouristate.edu/kenvollmar/mars/>
* A implementação corresponde de parte da infraestrutura de sistema operacional para realizar gerenciamento de processos.
* Para se ter acesso ao código do MARS, após baixar o arquivo .jar utilize uma ferramenta de descompactação (ex: Winrar) e extraia seu conteúdo. Em seguida crie um projeto em uma IDE para Java (ex: Eclipse) com os arquivos extraídos.
  + O “Apêndice A” do livro “PATTERSON, D. A. ; HENNESSY, J.L. Organização e projeto de computadores – a interface hardware software. 3. ed. Editora Campus, 2005” apresenta detalhes sobre o MIPS e chamadas de sistema.

**Parte 1 – Criação da PCB do gerenciador de Processos**

1. Crie seu próprio pacote “**mars.mips.SO”** onde estarão as implementações de cada gerenciador
   1. Todas as definições globais do seu “SO” devem fazer parte desse pacote
2. Dentro do seu pacote “SO” crie o pacote “ProcessManager” para criar as classes a seguir
3. Crie uma classe PCB (*Process Control Block*) que serve para armazenar todas as informações de contexto de um processo:
   1. Informações do hardware: o conteúdo de todos os registradores
      1. Observação: utilize as informações dos registradores contidas em: mars.mips.hardware.RegisterFile;
   2. Informações lógicas: endereço do início do programa (a partir de um *label* do código *assembly*), PID (número único como Identificador do processo), estado do processo (pronto, executando ou bloqueado)
      1. Nesse trabalho não será usado o estado “bloqueado” mas será usado no futuro, em outro trabalho
         1. A depender das atividades que serão escolhidas;
   3. Deverão existir métodos “get” e “set” para todas essas informações
   4. Incluir métodos
      1. Para copiar o conteúdo dos registradores físicos do hardware para a PCB (o conteúdo dos registradores é do tipo *int*, tentar copiar objetos *register* pode não funcionar)
         1. Observação: utilize as informações dos registradores contidas em: mars.mips.hardware.RegisterFile;
      2. E para copiar da PCB para os registradores físicos
4. A classe PCB deve estar preparada para ser incrementada, recebendo novos atributos e métodos para acessá-los nos próximos trabalhos
5. Crie uma classe Tabela de Processos que instancia objetos da classe PCB para cada novo processo
   1. Essa Tabela de Processo deve manter uma lista de processo “Prontos”
   2. E um processo em “Execução”
6. Crie uma classe Escalonador onde poderão ser implementados diferentes algoritmos para escolha dos processos que estão no estado de “pronto” na Tabela de Processos.
   1. A priori o algoritmo de escalonamento será uma fila, ou seja, sempre que a função “escalonar” for invocada o processo do início dessa fila é retirado. Novos processos “prontos” são incluídos sempre no fim dessa fila.