

Universidade Federal de Viçosa - Campus Florestal

Disciplina: Sistemas Operacionais

Professor: Daniel Mendes Barbosa

**Trabalho Prático 3**

Eduardo Vinicius – 3498

Pablo Ferreira – 3480

Samuel Sena – 3494

Florestal – MG

2020

**SUMÁRIO**

1. Introdução........................................................................................................3

2. Decisões do projeto........................................................................................4

3. Implementação................................................................................................5

4. Testes de execução.........................................................................................7

5. Referência Bibliográfica.................................................................................13

**1. Introdução**

O trabalho consiste em duas tarefas A e B, que possuem o objetivo de simular uma memória principal em forma de vetor e simular o funcionamento de memória virtual, respectivamente. Ambas tarefas realizadas tendo como base o trabalho prático 2 da disciplina.

Para compilar o programa, criamos um arquivo *makefile*, portanto basta iniciar uma instância de um terminal navegado até a pasta de uma das tarefas (em algum sistema operacional Linux) e compilar o programa com o uso do comando *make*. Em seguida, para executar o programa, é possível com o uso do comando *make run* ou simplesmente: *./EXEC* .

O código se encontra com diversos comentários que auxiliam no entendimento do funcionamento e fluxo de execução.

**2. Decisões do Projeto**

Para a implementação do trabalho decidimos fazer uma modificação profunda no código do trabalho anterior. Novas estruturas de dados necessitavam ser criadas, além de modificações nas estruturas já existentes.

1. **Implementação**

A implementação da tarefa A do trabalho consistiu na adaptação de um vetor estático de tamanho pré definido de tamanho “*MAXMEM”*. Devido a isso, em locais no código onde antes os valores inteiros pertencentes a processos eram simplesmente alocados dinamicamente, definidos como 0 (instrução D) ou ainda operações aritmeticas (instruções A, S), tiveram a implementação modificada e foi chamada a função de alocação no vetor estático de memória. Função esta dividida em duas opções de alocação: FirstFit e NextFit. Para alternar entre os algoritmos de alocação de memória basta ajudar o valor do define FIRSTFIT, quando o valor é definido para 0 é executado o algoritmo NextFit, e quando o valor do define é definido para 1 é executado. Caso um processo tente alocar suas variaveis ou um processo tente criar um processo filho e não haja espaço para o mesmo criar sua variável, o mesmo será bloqueado e o seu contador de programa não será incrementado (permitindo que a mesma instrução execute na próxima oportunidade de execução). Caso um processo termine, pela execução da instrução T ou simplesmente as instruções acabem, a memória alocada por aquele processo será liberada e um processo anteriormente bloqueado será desbloqueado. Além disso, uma implementação que busca simular um disco (assim como recomendado pelo monitor no fórum de dúvidas) foi implementado e sempre que um processo se encontra bloqueado, tem seu conteudo da memória movido para o vetor de memória em disco. E por fim, novas funções de impressão (para imprimir o vetor de memória) e calculo de metricas do uso de memória foram implementadas. As figuras abaixo ilustram detalhes citados na implementação:

**4. Testes de execução**

Os testes de execução foram feitos levando em consideração os arquivos de entrada: Controle.txt (arquivo de instruções lido pelo controle), Programa.txt (arquivo de programa 1), Programa2.txt (arquivo de programa 2), Programa3.txt (arquivo de programa 3). As figuras a seguir ilustram os testes de execução nas respectivas tarefas:

**5. Referência Bibliográfica**

TANENBAUM, Andrew S. **Sistemas Operacionais Modernos**. 3. ed. São Paulo: Pearson, 2010. 653 p.