

Sistemas Operativos

André Ribeiro - 39686

José Costa - 39428

~~Diogo Nave - 34519~~

**Modelo de Relatório do Projeto Simulador de Gestão de Processos Escalonamento e Gestão de Memória**

**URL do projeto no GitHub -** [**https://github.com/VanSheep7/SO-Projeto.git**](https://github.com/VanSheep7/SO-Projeto.git)

**O PROJETO:**

Temos vindo a desenvolver um simulador cujo código fonte está escrito na linguagem Python.

O simulador tem a capacidade de aceder e ler os ficheiros de texto progenitor e filho e interagir com os mesmos.

Faz escalonamento e também tem implementadas funções de gestão de memória.

Recebemos a notícia que um membro do nosso grupo por motivos próprios já não trabalha neste projeto. O projeto desenvolvido é da autoria de André Ribeiro e José Costa.

**URL do projeto no GitLab -** [**https://gitlab.com/vansheep/so/-/tree/master**](https://gitlab.com/vansheep/so/-/tree/master)

O programa consiste num simulador que interage com ficheiros, gere memória e executa processos de forma organizada por meio de operações de escalonamento, criação de processos, o seu término e permuta.

São exemplos das suas funcionalidades:

. Leitura de ficheiros de texto

. Acompanhamento e Alteração de estado dos processos

. Organização de processos e o seu controle (PCB)

. Gestão de memória e tempo

Conteúdo (resumido) de cada file

simulador1.py – Process Control Block (PCB);First Come First Served (FCFS)

simulador2.py – Priority Establishment

simulador3.py - Shortest Job First (SJF)

simulador4.py – Gerenciamento de memória

Observações e Informação adicional

O programa foi escrito em Python3.

Os ficheiros de texto utilizados estão presentes nas plataformas github e gitlab.

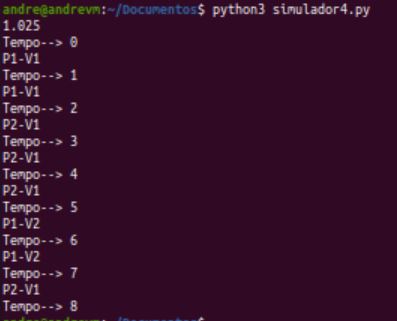
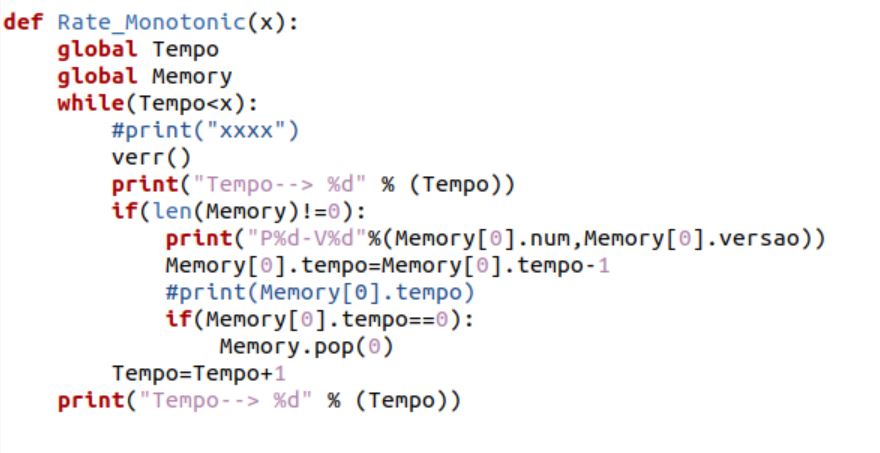
No código estão presentes alguns testes que ajudaram no tweaking do projeto, são irrelevantes ao funcionamento.

Algoritmos Implementados

Rate-monotonicscheduling (RMS)

First Come First Served (FCFS)

Exemplo da Execução do programa e verificação

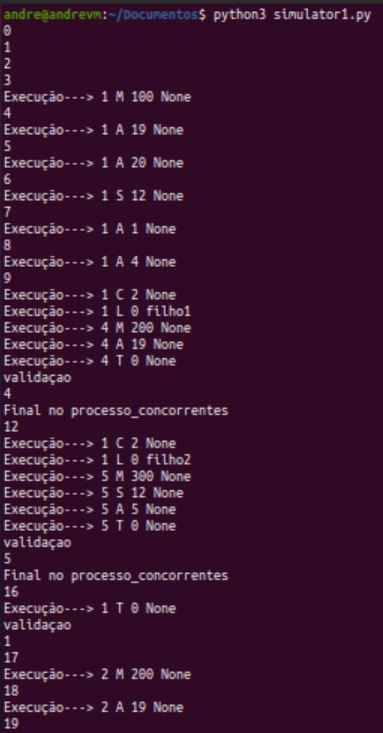
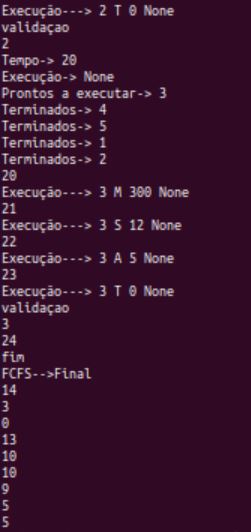


O ficheiro simulador4.py tem as funcionalidades de memória. Este programa por meio do Bloco de controlo de processos armazena informações dos processos e também os divide em classes.

Está também neste ficheiro contido o algoritmo que escolhemos implementar, o Rate-monotonic scheduling (RMS), um processo com menor duração de funcionamento terá uma prioridade mais alta e será priorizado.

Excerto do código do simulador4.py

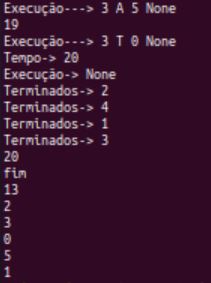
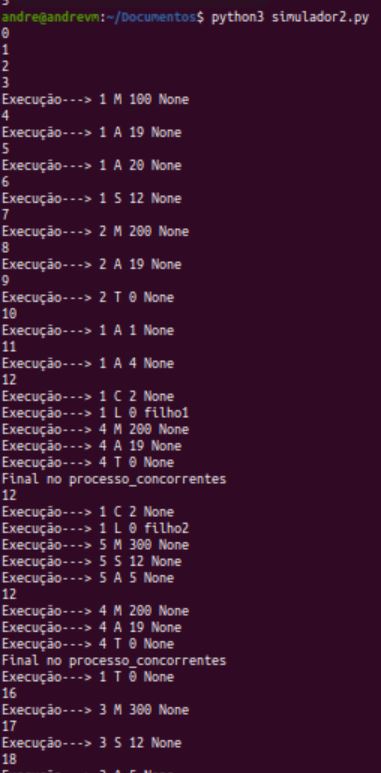
output do simulador4.py



OUTPUTS OBTIDOS

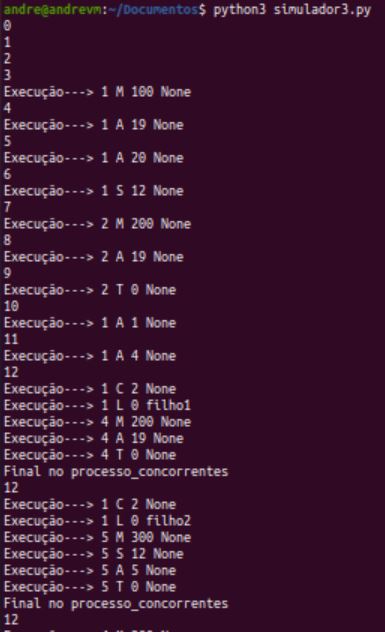
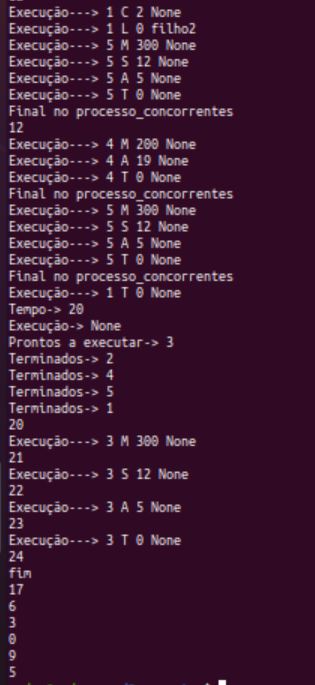
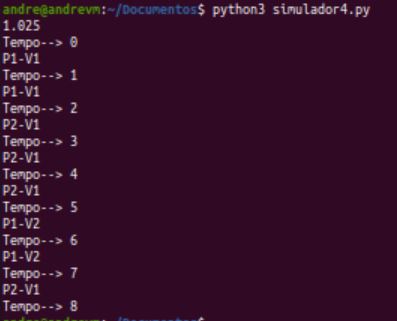
NA

EXECUÇÃO DOS FICHEIROS



simulador2.py

simulator1.py



simulador4.py

simulador3.py