CMSC 312 – Final Project
Operating System Simulator
Jason Ball

DESCRIPTION

This Python project simulates an operating system with a round-robin scheduler, dispatcher, basic I/O, memory, and critical sections. Program files tell the simulator what to do and how long to do it.

EXAMPLE PROGRAM FILE

```
Name: Calculator
Total runtime: 137
Memory: 45

CALCULATE 25
CALCULATE 79
CRITICAL BEGIN
YIELD
CALCULATE 2
CRITICAL END
I/O 57
I/O 4
YIELD
CRITICAL BEGIN
CALCULATE 23
CALCULATE 8
CRITICAL END
OUT
I/O 8
EXE
```

Name of the program
Total runtime of the program
Memory Necessary for execution

CALCULATE – Takes n CPU cycles lasting 10 milliseconds each.
I/O – Causes the process to wait for a random number of seconds before running again.
YIELD – Yields control back to the scheduler. The scheduler then starts another process
CRITICAL BEGIN and END – Marks the beginning and end of a critical section.
EXE – Marks the end of the program file

INSTRUCTIONS FOR RUNNING

This project was built using Python 3.7. Run the following commands to start the simulator:

```
git clone https://github.com/jason-ball/JASON_BALL_CMSC312_2019.git
cd JASON_BALL_CMSC312_2019
python3 main.py
```

Note for macOS: The version of Python and Tkinter supplied by Apple is not compatible with the process GUI. You can download the latest version here.

Note for Windows: You may have to run python instead of python3.

PROJECT REQUIREMENTS

- Process Implementation and PCB:
  - Process.py
  - PCB.py
  - commands.py
  - tables.py: 7-10
  - Program.py
  - Instruction.py
  - main.py: 16-87
- Critical section within each Process:
  - CriticalSection.py
  - Process.py: 39-64
  - main.py: 40-54, 61-67
- Critical section resolving scheme:
  - CriticalSection.py
  - Process.py: 39-64
- Scheduler:
  - Scheduler.py
  - Dispatcher.py
  - main.py: 90-100
- Basic memory and operations on it:
  - tables.py: 12
  - main.py: 27
  - Scheduler.py: 36-37
  - Dispatcher.py: 21
- I/O interrupts and handlers:
  - commands.py: 14-25
  - Dispatcher.py: 40-44
- Loading external processes and generating new ones on user request
  - main.py: 15-87