import os

import time

import random

from enum import Enum

# Define process states using Enum

class State(Enum):

READY = "Ready"

RUNNING = "Running"

WAITING = "Waiting"

TERMINATED = "Terminated"

# Process class to hold process attributes

class Process:

def \_\_init\_\_(self, pid, arrival\_time, burst\_time):

self.pid = pid

self.state = State.READY

self.arrival\_time = arrival\_time

self.burst\_time = burst\_time

self.remaining\_time = burst\_time

# Function to clear the console (cross-platform)

def clear\_console():

os.system('cls' if os.name == 'nt' else 'clear')

# Function to display process states

def display\_processes(processes, current\_time):

clear\_console()

print(f"\nTime: {current\_time}")

print("PID\tState\t\tArrival\tBurst\tRemaining")

print("--------------------------------------------")

for p in processes:

print(f"{p.pid}\t{p.state.value:<12}\t{p.arrival\_time}\t{p.burst\_time}\t{p.remaining\_time}")

print()

# Simulate process execution

def simulate\_processes(processes):

current\_time = 0

completed = 0

num\_processes = len(processes)

time\_quantum = 1

while completed < num\_processes:

for p in processes:

if p.arrival\_time <= current\_time and p.state != State.TERMINATED:

if p.state == State.READY:

p.state = State.RUNNING # Move to Running

if p.state == State.RUNNING:

p.remaining\_time -= time\_quantum

if p.remaining\_time <= 0:

p.state = State.TERMINATED # Process completed

completed += 1

elif random.random() < 0.2: # 20% chance to move to Waiting

p.state = State.WAITING

elif p.state == State.WAITING:

if random.random() < 0.3: # 30% chance to move back to Ready

p.state = State.READY

display\_processes(processes, current\_time)

time.sleep(1) # Pause for 1 second to simulate time

current\_time += time\_quantum

def main():

# Seed random number generator

random.seed()

# Initialize sample processes (PID, Arrival Time, Burst Time)

processes = [

Process(1, 0, 5),

Process(2, 1, 3),

Process(3, 2, 4),

Process(4, 3, 2),

Process(5, 4, 6)

]

print("Starting Process Visualization Tool...")

time.sleep(2) # Pause before starting

simulate\_processes(processes)

print("All processes completed!")

if \_\_name\_\_ == "\_\_main\_\_":

main()