Process Scheduler simulation

- Demiana sharl
- Tony mikhael
- Paula ayman
- Abanoub shafik
- Antonios emad

Purpose:

- Design and implement a simulator to model multiprocessor process scheduling.
- Examine the performance of various scheduling policies in a multiprocessor environment.
- Provide a visual representation of scheduling decisions for analysis and comparison.
- Enable user interaction to explore different scheduling scenarios and parameters.

Scope:

- Scheduling policies to implement:
  - FCFS (First Come First Served)
  - SJF (Shortest Job First)
  - STCF (Shortest Time-to-Completion First)
  - RR (Round Robin)
  - Priority Scheduling
  - MLFQ (Multi-Level Feedback Queue)
  - Stride Scheduling
- Performance metrics to collect:
  - Turnaround time
  - Waiting time
  - Response time
  - CPU utilization
- Visualization of scheduling decisions:
  - Gantt chart
- User interaction features:
  - Control over scheduling policy selection
  - Modification of process workload parameters
  - Ability to create custom process sets

Implementation Details:

1. System Design:

- Data structures:
  - Process representation (process ID, arrival time, CPU bursts, I/O bursts, priority, etc.)
  - Ready queue for each processor
  - Job queue for incoming processes
  - Data structures to track performance metrics
- Modules:
  - Process generator
  - Scheduler (centralized or distributed)
  - CPU simulator
  - I/O simulator
  - Visualization module
  - User interface

2. Scheduling Algorithm Implementation:

- FCFS: Enqueue processes in order of arrival.
- SJF: Enqueue processes based on estimated shortest execution time.
- STCF: Enqueue processes based on estimated shortest remaining time.
- RR: Employ a time quantum for process switching.
- Priority: Prioritize processes based on assigned priorities.
- MLFQ: Utilize multiple queues with different priorities and time quanta.
- Stride: Calculate strides and schedule processes with the highest strides first.

3. Performance Evaluation:

- Collect and calculate performance metrics for each policy.
- Generate reports and visualizations (e.g., Gantt charts, histograms) to compare results.

4. Visualization:

- Create a Gantt chart to visually represent scheduling decisions and process execution.

5. User Interface:

- Design a user-friendly interface to allow:
    - Selection of scheduling policies
    - Modification of process parameters
    - Creation of custom process sets
    - Visualization of scheduling results

Testing and Validation:

- Develop a test suite with diverse process workloads to evaluate the simulator's behavior and accuracy.
- Compare simulation results with theoretical expectations and benchmarks.

Documentation:

- Create comprehensive documentation covering:
    - Project overview
    - Design and implementation details
    - User guide
    - Testing and validation results

References:

- Include citations for relevant sources on multiprocessor scheduling and simulation techniques.

Additional Considerations:

- Processor heterogeneity: Consider how to model systems with processors of varying speeds or capabilities.
- Synchronization: Address synchronization issues that may arise in multiprocessor systems.