2022

Project Report(Operating System)

GROUP MEMBERS :-

20I-0409 SYED AMMAR HUSSAIN SHIRAZI

20I-0402 MUHAMMAD DANIYAL

Language Used: C++

Objectives:

- To create a properly functioning Operating system that is able to:
    - Manage various types of scheduling policies, i.e. First Come First Serve, Round Robin and Priority based.
    - Be compatible with various number of Processors
    - Handle different types of processes, whether CPU or I/O bound.

Methods employed:

OOP concepts were utilized to make the management of the Operating System (OS) more efficient. In such we created classes for key aspects of our OS.

That include:

1. CPU class: This contains the details about the processor(s), processes currently loaded, the scheduler, running time etc.
2. Process structure: This oversees the storing of information about each individual process, such as name of the process, its arrival time, remaining time, its type and priority (if required) etc.
3. Scheduler Class: This class in charge of storing all the queues involved in scheduling such as waiting Queue, ready Queue etc. and information about scheduling policy being used.

Function of scheduler:

- The scheduler is created so that it is able to adapt and function for various scheduling policies.
    - First Come First Serve (FCFS), it will determine the order a process is executed at by the order of arrival of processes.
    - Round Robin uses the concept of allotting customizable time slices to each process as that they can all get a fairer chance to run
    - Priority based, as the name suggest will determine which process will run next based on the priority values of the process, the higher the value the higher the priority.

Difficulties faced:

- We faced quite a few issues while trying to integrate Thread Functions as member Functions of the classes.
    - Therefore, in the end we had to make the thread functions as friend functions of their respective classes.
- Moreover, another complication we faced was the inevitable circular dependency between our CPU and Scheduler Class, as CPU required a Scheduler object as a member and vice versa.
    - Therefore, we did forward declaration of our classes and used a separate .cpp file to define our functions.