Advanced Topics Final Presentation

By: Vishwesswaran Gopal

Introduction

- Fundamentals of Operating Systems
- Course Content
  - Virtualization
  - Concurrency
  - Persistence
  - Security
- Course Goals: To understand how operating systems work.
- *No* certification

Key Concepts

CPU Scheduling

- How can multiple processes (running programs) seemingly run on a single CPU, which can only perform one task at a time?
- *FIFO* (First In, First Out)
  - Strengths
  - Weaknesses

Virtualization of Memory

- How is physical memory split amongst many processes?
- ***Base & Bounds***
  - Strengths
  - Weaknesses

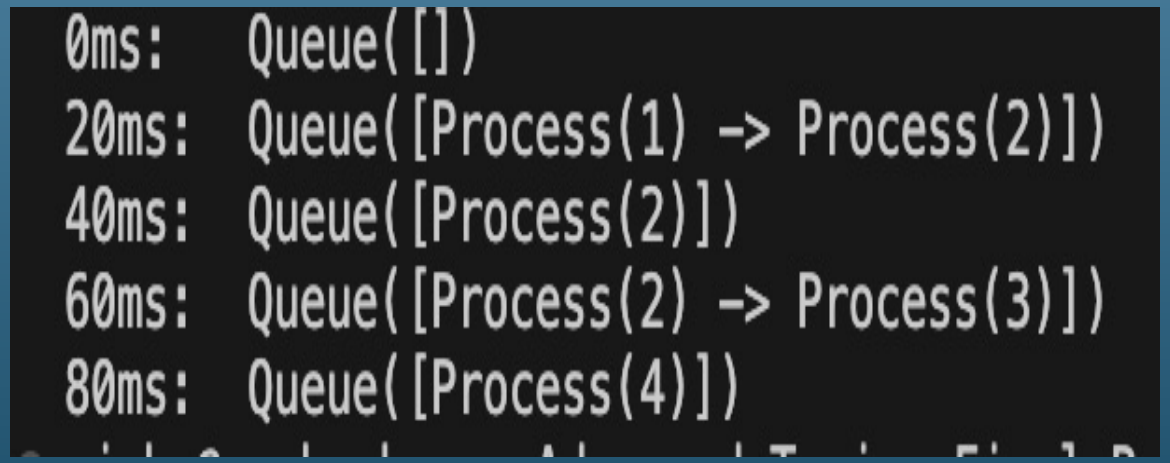
Code for FIFO Scheduler

```python
class Node:
    def __init__(self: Self, value: int, next: "Node" = None) -> None:
        self.value = value
        self._next = next

    def __str__(self: Self) -> str:
        return f"Node({self.value})"


class Queue:
    def __init__(self: Self, node: "Node" = None) -> None:
        self.first: "Node" = node
        self.last: "Node" = node
        self.add(node)

    def add(self: Self, node: "Node") -> None:
        if node is None:
            return
        if not isinstance(node, Node):
            raise ValueError(
                "You passed in the wrong type for node, it should be an object of Node."
            )
        elif self.first is None:
            self.first = node
            self.last = node
            return
        prev: "Node" = self.last
        self.last = node
        prev._next = node

    def pop(self: Self) -> "Node":
        if self.first is None:
            return
        first: "Node" = self.first
        self.first = first._next
        return first

    def __str__(self: Self) -> str:
        current = self.first
        nodes = []
        while current:
            nodes.append(str(current))
            current = current._next
        return f"Queue([{' -> '.join(nodes)}])"
```

Implementation Code

Testing Code

```
53    q = Queue()
54    q.add(Node(1))
55    q.add(Node(2))
56    print(q)
57
58    q.pop()
59    print(q)
60
61    q.add(Node(3))
62    print(q)
63
64    q.pop()
65    q.pop()
66    q.add(Node(4))
67    print(q)
```



Terminal Output

The End!