A report submitted for

Operating System For

FIRST COME FIRST SERVE simulation

BY

|  |  |
| --- | --- |
| **Name** | **Roll Number** |
|  |  |
|  |  |
|  |  |
| Pankaj Sheokand | 102203410 |

Submitted to:

Shashank Sheshar Singh



Computer Science and Engineering Department Thapar Institute of Engineering & Technology, Patiala Summer Semsester 2024

Table of Contents

[BACKGROUND 3](#_bookmark0)

[INTRODUCTION. 4](#_bookmark1)

[MECHANISMANDWORKING: 5](#_TOC_250000)

[FUNCTIONALITIES: 12](#_bookmark2)

[REMAININGCODEMODULES,API’SANDPLATFORMS. 13](#_bookmark3)

[FUTUREWORK: 14](#_bookmark4)

[CONCLUSION. 15](#_bookmark5)

[REFERENCES. 16](#_bookmark6)

# BACKGROUND:

First Come First Serve (FCFS) is one of the simplest and most intuitive CPU scheduling algorithms. In FCFS scheduling, processes are executed in the order they arrive in the ready queue. It's a non-preemptive algorithm, meaning that once a process starts its execution, it runs to completion. This simplicity makes it easy to implement but can lead to issues like the convoy effect, where short processes get stuck waiting behind long ones.

# INTRODUCTION:

CPU scheduling is a fundamental concept in operating systems, determining the order in which processes access the CPU. Effective scheduling maximizes CPU utilization and system throughput while minimizing wait times and response times. The FCFS algorithm, due to its straightforward nature, serves as a foundational scheduling technique upon which more advanced methods are built. Despite its drawbacks, understanding FCFS is crucial for grasping the basics of process management and scheduling in operating systems.

# MECHANISM AND WORKING:

FCFS scheduling, processes are handled in the order they arrive:

1. **Arrival**: When a process arrives in the ready queue, it is appended to the end of the queue.
2. **Execution**: The CPU picks the first process in the queue and executes it until completion.
3. **Completion**: After a process finishes execution, it is removed from the queue, and the next process in line is picked for execution.

## Menu Code:

#!/bin/bash

# First Come First Serve (FCFS) CPU Scheduling Script

# Function to display the process details

display\_processes() {

echo "Processes:"

echo "PID Arrival Time Burst Time"

for ((i=0; i<${#pids[@]}; i++));

do

echo "${pids[$i]} ${arrival\_times[$i]} ${burst\_times[$i]}"

done

echo ""

}

# Function to initialize processes

initialize\_processes()

{

pids=()

arrival\_times=()

burst\_times=()

# Add processes with arrival times and burst times here

pids=(1 2 3 4 5) # Process IDs

arrival\_times=(0 1 2 3 4) # Corresponding arrival times

burst\_times=(10 4 5 3 2) # Corresponding burst times

}

# Function to perform FCFS scheduling and generate Gantt chart

fcfs\_scheduling() {

local time=0

local gantt=()

echo "Starting FCFS Scheduling"

echo ""

for ((i=0; i<${#pids[@]}; i++));

do

local pid=${pids[$i]

}

local arrival=${arrival\_times[$i]

}

local burst=${burst\_times[$i]

}

if [[ $time -lt $arrival ]];

then

time=$arrival

fi

gantt+=("P${pid}($burst)")

time=$((time + burst))

echo "Process $pid executed from $((time - burst)) to $time"

done

echo ""

echo "All processes completed. Total time: $time units"

echo ""

echo "Gantt Chart:"

echo "${gantt[\*]}"

}

# Main script execution

initialize\_processes

display\_processes

# Perform FCFS scheduling

fcfs\_scheduling

## Output Screenshots:

## 

## 



# FUNCTIONALITIES:

The main functionalities of an FCFS scheduling algorithm include:

1. **Process Arrival Management**: Handling the arrival of new processes and adding them to the ready queue.
2. **CPU Scheduling**: Selecting the next process to run based on the FCFS policy.
3. **Execution Management**: Running the selected process to completion.
4. **Queue Management**: Maintaining the order of processes in the ready queue.
5. **Gantt Chart Generation**: Visual representation of process execution over time.

# REMAINING CODE MODULES, API’S AND PLATFORMS:

**Code Modules**

1. **Process Management Module**: Handles the creation and initialization of processes.
2. **Scheduling Module**: Implements the FCFS scheduling logic.
3. **Queue Module**: Manages the ready queue and process order.
4. **Gantt Chart Module**: Generates and displays the Gantt chart.

# FUTURE WORK:

Future improvements to the FCFS scheduling algorithm can address its inherent limitations:

1. **Minimizing the Convoy Effect**: Investigate hybrid approaches that combine FCFS with other scheduling algorithms.
2. **Dynamic Time Quantum**: Explore the use of dynamic time quantum adjustments to optimize wait and response times.
3. **Integration with Advanced Schedulers**: Combine FCFS with more sophisticated algorithms like Shortest Job First (SJF) or Round Robin (RR) for better performance under varied workloads.

# CONCLUSION:

First Come First Serve (FCFS) scheduling algorithm is a foundational concept in operating systems. While simple and easy to implement, it is not without drawbacks, such as the convoy effect. Understanding FCFS is crucial for anyone studying CPU scheduling as it provides the basis for more advanced algorithms. By managing processes in a straightforward manner, FCFS exemplifies the trade-offs between simplicity and performance in scheduling.

# REFERENCES:

* + Idea from stackoverflow.com Submitted by Hai Vau , on 10 April 2011 [https://stackoverflow.com/questions/5613283/bash-script-on-linux-to-create-atm-](https://stackoverflow.com/questions/5613283/bash-script-on-linux-to-create-atm-machine-system) [machine-system](https://stackoverflow.com/questions/5613283/bash-script-on-linux-to-create-atm-machine-system)
  + Use of stat command answer by <<https://linuxhint.com/linux_stat_command/>>

~~~~~~\*\*/**THE END**/\*\*~~~~~~