****

**Lab Assignment on Memory Management**

**Submitted To**

**Faculty Name :** **Prof. Dr. Md. Motaharul Islam,**

**Professor, Department of CSE**

**Submitted By**

**Name : Sheikh Sarafat Hossain**

**Id No : 2022-3-60-109**

**Course Title :** Operating System

**Course Code :** CSE325

**Section :** 07

**Semester :** Fall 24

**Date of Submission:** 18 January 2025

**Memory Management Techniques**

**Introduction**

This report discusses four different memory management techniques implemented in C programming. These techniques include:

1. **First Come First Serve (FCFS) Scheduling**
2. **Shortest Job Next (SJN) Scheduling**
3. **Round Robin Scheduling**
4. **Paging Technique**

The implementations aim to demonstrate how these algorithms handle tasks or processes in a computer system, manage memory, and improve system performance.

**Code 1: First Come First Serve (FCFS) Scheduling**

**Output:   
A screen shot of a computer program

Description automatically generated**

**Description**

The FCFS scheduling algorithm executes processes in the order they arrive in the ready queue. It is non-preemptive, meaning once a process starts execution, it cannot be stopped until it finishes.

**Key Features**

* Input includes the number of processes, their arrival times, and burst times.
* Calculates and displays waiting time and turnaround time for each process.
* Computes average waiting time and average turnaround time.

**Strengths**

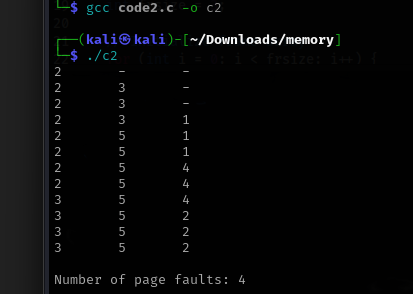
* Simple and easy to implement.
* Works well when processes have similar burst times.

**Weaknesses**

* Can lead to the **convoy effect**, where longer processes delay shorter ones.
* Poor performance for a mix of short and long processes.

**Code 2: Shortest Job Next (SJN) Scheduling**

**Output:**

****

**Description**

The SJN scheduling algorithm selects the process with the smallest burst time from the ready queue for execution. This algorithm is also non-preemptive.

**Key Features**

* Requires burst time as input for each process.
* Sorts processes based on burst time.
* Calculates and displays waiting time and turnaround time for each process.

**Strengths**

* Minimizes average waiting time in ideal scenarios.
* Suitable for systems where process burst times are predictable.

**Weaknesses**

* Starvation is possible for processes with longer burst times.
* Not practical in real-time systems where burst time is not known in advance.

**Code 3: Round Robin Scheduling**

**Output:**

**A computer screen with numbers and text

Description automatically generated**

**Description**

Round Robin scheduling is a preemptive algorithm where each process is assigned a fixed time quantum. If a process does not complete within its time quantum, it is moved to the back of the ready queue.

**Key Features**

* Requires time quantum as an additional input.
* Ensures that all processes get a fair share of CPU time.
* Calculates and displays waiting time and turnaround time for each process.

**Strengths**

* Prevents starvation as all processes get equal priority.
* Suitable for time-sharing systems.

**Weaknesses**

* Performance depends heavily on the choice of time quantum.
* Overhead due to frequent context switching.

**Code 4: Paging Technique for Memory Management**

**Output:**

**A computer screen shot of a black screen

Description automatically generated**

**Description**

Paging is a memory management technique that eliminates the need for contiguous allocation of physical memory. Logical memory is divided into pages of fixed size, which are mapped to frames in physical memory.

**Key Features**

* Divides physical and logical memory into fixed-size blocks.
* Implements page and frame tables for mapping.
* Displays page table and frame table.

**Strengths**

* Reduces fragmentation by allowing non-contiguous memory allocation.
* Simplifies memory allocation and deallocation.

**Weaknesses**

* Requires additional overhead for managing page tables.
* Increases access time due to address translation.

**Comparative Analysis**

| **Feature/Technique** | **FCFS Scheduling** | **SJN Scheduling** | **Round Robin Scheduling** | **Paging** |
| --- | --- | --- | --- | --- |
| Type | Non-preemptive | Non-preemptive | Preemptive | Memory Management |
| Input | Arrival, Burst | Burst | Arrival, Burst, Quantum | Memory Details |
| Strengths | Simple | Minimizes Waiting Time | Fairness | Reduces Fragmentation |
| Weaknesses | Convoy Effect | Starvation | Overhead | Translation Overhead |
| Use Case | Batch Systems | Predictable Systems | Time-Sharing Systems | Modern OS |

**Conclusion**

Each technique addresses specific challenges in process and memory management. While scheduling algorithms like FCFS, SJN, and Round Robin focus on process execution, the paging technique is vital for efficient memory utilization. Choosing the right approach depends on the system requirements and workload characteristics.

These C implementations serve as foundational examples for understanding these techniques and their real-world implications in operating systems.