Priority Scheduling of Processes

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING)

By

Pavan

Registration number: 12218283

Supervisor

Mr. Waseem Ud DinWani

School of Computer Science and Engineering

Lovely Professional University

Phagwara, Punjab (India)

Declaration:

I, Pavan, solemnly declare that this project report, titled "Priority Scheduling of Processes," is the result of my independent work. I confirm that I have not plagiarized any material from any other source.

Certificate:

This is to certify that the project report titled "Priority Scheduling of Processes," submitted by Pavan in fulfilment of Bachelor of Technology at Lovely Professional University, is an authentic work. The project has been examined and approved.

Signature of Supervisor

(Name of Supervisor)
Date:

Acknowledgment:

I extend my heartfelt thanks to my professor, **Mr.Waseem Ud DinWani** and for his invaluable support, guidance, and encouragement throughout this academic year.

Table of Contents:

Abstract:

The abstract is a concise summary of the Priority Scheduling of Processes system project. In this project, we have created a Java-based Priority Scheduling of Processes that allows users to schedule and calculate the priority of a process . This system is designed to help individuals and organizations manage their priorities efficiently.

Introduction:

The introduction provides an overview of the project, explaining its purpose and significance. In today's busy world, Priority scheduling is a widely used algorithm in computer operating systems to manage the execution of processes. It is designed to allocate the CPU (Central Processing Unit) time to different tasks or processes based on their relative priorities. This approach ensures that higher-priority processes are given preference over lower-priority ones, which can be crucial for the efficient functioning of the system and achieving specific goals or service level agreements.

Objective of the Project:

The project has two primary objectives:

The primary objectives of priority scheduling in the context of operating systems and process management are as follows:

Optimizing System Resource Utilization: Priority scheduling aims to maximize the utilization of system resources, such as the CPU and memory, by ensuring that higher-priority processes are allocated resources ahead of lower-priority processes. This helps in efficiently using available hardware resources.

Meeting System Goals and Objectives: Priority scheduling allows system administrators to set priorities based on the system's goals and requirements. For example, a real-time system may prioritize tasks that require immediate responses, while a general-purpose system may focus on maximizing throughput or minimizing response time.

Description of the Project:

The project components encompass:

The project components encompass the following key elements:

1. User Interface: The project features a user-friendly interface that allows users to interact with the appointment scheduling system. This interface is designed to be intuitive and menu-driven, making it easy for users to schedule and manage appointments. It provides clear prompts and options for users to navigate through the system effortlessly.

2. Appointment Class: The core data structure in the project is the Appointment class. This class encapsulates the essential attributes of an appointment, including a title, date, and time. It provides methods for creating and representing appointments, making it a fundamental building block of the system.

3. Data Storage with a Queue: Appointments are stored and managed using a queue data structure. The queue ensures a first-in-first-out (FIFO) order for appointments. New appointments are added to the end of the queue, and upcoming appointments are displayed in the order in which they were scheduled.

4. Scheduling Appointments: One of the primary functionalities of the project is the ability to schedule appointments. Users can input the title, date, and time of the appointment. The system then creates an Appointment object and adds it to the queue. This process ensures that appointments are organized based on their scheduled time.

5. Viewing Upcoming Appointments: The project allows users to view their upcoming appointments. If the queue is not empty, the system displays a list of appointments in the order they are scheduled, helping users keep track of their commitments.

6. Exit Process: The project includes an option for users to gracefully exit the system. This ensures that the application can be closed without any issues, and it releases any resources used during the program's execution.

Source Code:

Below is a simplified Java code snippet illustrating the Priority Scheduler

```
import java.util.PriorityQueue;
import java.util.Scanner;

class Process implements Comparable<Process> {
    private int pid;
    private String name;
    private int priority;

    public Process(int pid, String name, int priority) {
        this.pid = pid;
        this.name = name;
        this.priority = priority;
    }

    public int getPid() {
        return pid;
    }

    public String getName() {
        return name;
    }

    public int getPriority() {
        return priority;
    }

    @Override
    public int compareTo(Process other) {
        return Integer.compare(other.getPriority(), this.getPriority());
    }
```

```
    @Override
    public String toString() {
        return "Process ID: " + pid + ", Name: " + name + ", Priority: " +
priority;
    }
}

public class PriorityQueueApp {
    public static void main(String[] args) {
        PriorityQueue<Process> priorityQueue = new PriorityQueue<>();
        int counter = 0;

        Scanner scanner = new Scanner(System.in);

        while (true) {
            System.out.println("\nOptions:");
            System.out.println("1. Add a process");
            System.out.println("2. Remove a process");
            System.out.println("3. View processes");
            System.out.println("4. Exit");

            System.out.print("Enter your choice: ");
            String choice = scanner.nextLine();

            if (choice.equals("1")) {
                System.out.print("Enter Process ID: ");
                int pid = Integer.parseInt(scanner.nextLine());
                System.out.print("Enter Process Name: ");
                String name = scanner.nextLine();
                System.out.print("Enter Priority: ");
                int priority = Integer.parseInt(scanner.nextLine());
                Process process = new Process(pid, name, priority);
                priorityQueue.add(process);
                System.out.println("Process " + name + " added to the
queue.");
            } else if (choice.equals("2")) {
                Process process = priorityQueue.poll();
```

```
            if (process != null) {
                System.out.println("Process " + process.getName() + "
removed from the queue.");
            } else {
                System.out.println("Queue is empty.");
            }
        } else if (choice.equals("3")) {
            for (Process process : priorityQueue) {
                System.out.println(process);
            }
        } else if (choice.equals("4")) {
            break;
        } else {
            System.out.println("Invalid choice. Please try again.");
        }
    }
}
```

Input/Output:

In this section, we showcase examples of input and output for the appointment scheduling system. We demonstrate how users can input appointment details and how the system displays upcoming appointments.

Scope of the Project:

The scope of this project is to provide a basic priority scheduling system. It covers essential features for scheduling and viewing appointments, making it suitable for personal use or as a foundation for a more comprehensive scheduling system. However, it does not include advanced features like user authentication, notifications, or integration with external calendars.

Development:

The development section outlines the tools and technologies used to create the project. It discusses the software development environment, libraries, and the step-by-step process of developing the priority scheduling system.

Conclusion:

In conclusion, this project offers a simple yet effective solution for priority scheduling. It can be a valuable tool for individuals or small organizations looking for an uncomplicated way to manage their appointments. While it provides a basic framework, further enhancements and features can be added to make it more robust and adaptable to specific needs. This project demonstrates how Java programming can be applied to practical tasks, and it serves as a foundation for more advanced priority scheduling systems.