A black background with pink text

Description automatically generated

**FACULTY OF ENGINEERING, DESIGN AND TECHNOLOGY**

DEPARTMENT OF COMPUTING AND TECHNOLOGY

ADVENT 2024 SEMESTER

DESIGN AND ANALYSIS OF ALGORITHMS COURSEWORK PROJECT REPORT

PROGRAM: BSC 2:1

COURSE: DESIGN AND ANALYSIS OF ALGORITHMS

COURSE LECTURER: JOHN HABARE

PROJECT TITLE:

**PERSONAL SCHEDULING ASSISTANT**

*Submitted by*

|  |  |  |  |
| --- | --- | --- | --- |
| S/N | Reg Number | Access.No | Name |
| 1. | M23B23/006 | B20233 | GENO OWOR JOSHUA |
| 2. | S23B23/056 | B20726 | REBECCA ALINDA |
| 3. | S23B23/023 | B23568 | KASINGYE LEON MATABI |

# Table of contents

Contents

[Table of contents 2](#_Toc182320023)

[1.0 Abstract 3](#_Toc182320024)

[2.0 Introduction, Problem Statement, and Project Objectives 3](#_Toc182320025)

[Introduction 3](#_Toc182320026)

[Problem Statement 3](#_Toc182320027)

[Objectives 4](#_Toc182320028)

# [3.0 System Architecture……………………………………...](#_Toc182320029)......5

[Classes](#_Toc182320030) 6

[Features](#_Toc182320031) 7

[4.0 Algorithms and Techniques](#_Toc182320032) 8

[5.0 Work Flow](#_Toc182320033) 9

[6.0 Conclusion & Future Recommendations](#_Toc182320034) 10

# 

# Abstract

The Personal Scheduling Assistant is a Python-based task management system designed to enhance productivity by enabling users to efficiently organize and prioritize their personal and academic responsibilities. The application integrates advanced algorithms such as dynamic programming, merge sort, and binary search to provide an optimized task scheduling solution that respects deadlines and priority levels. Key features include task addition, deadline tracking, priority-based scheduling, task density analysis, and Gantt chart visualization. By offering insights into workload distribution and presenting schedules visually, the system empowers users to manage their time effectively. The project demonstrates the utility of algorithmic approaches in solving real-world scheduling challenges and sets the stage for potential enhancements, such as calendar integrations and AI-driven recommendations, to further streamline task management.

# 2.0 Introduction, Problem Statement, and Project Objectives

## **Introduction**

The *Personal Scheduling Assistant* project is a comprehensive Python-based application designed to streamline task management for individuals. It accommodates both personal and academic responsibilities, helping users effectively organize their time, meet deadlines, and achieve their priorities. By utilizing advanced algorithms such as dynamic programming, merge sort, and binary search, the system ensures optimal task scheduling while providing meaningful insights through visual and analytical tools.

## **Problem Statement**

Managing time effectively is a significant challenge in today’s fast-paced world, where individuals must balance multiple personal and academic responsibilities. Traditional task management methods often fail to prioritize tasks efficiently, meet deadlines, or adapt to time constraints, leading to missed deadlines, increased stress, and reduced productivity. The Personal Scheduling Assistant addresses this gap by providing a dynamic, user-friendly system that leverages advanced algorithms such as dynamic programming, merge sort, and binary search to optimize task prioritization and scheduling. Key features include task addition, deadline tracking, priority-based scheduling, workload density analysis, and Gantt chart visualization, offering users actionable insights into workload distribution and clear, intuitive representations of their schedules. This project demonstrates the potential of algorithmic approaches in solving real-world scheduling challenges and sets the foundation for future enhancements, such as calendar integrations and AI-driven task recommendations, to further streamline and improve time management.

## **Objectives**

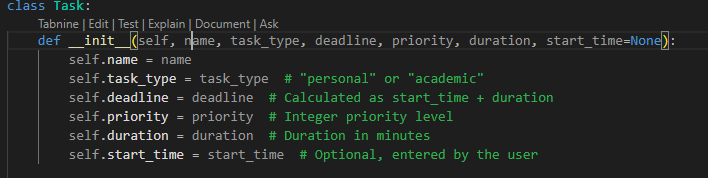
1. Provide a user-friendly interface for managing tasks.
2. Enable the addition, scheduling, and retrieval of tasks based on user priorities and deadlines.
3. Offer an optimized scheduling solution to fit tasks within time constraints effectively.
4. Analyze and visualize task density over specified time intervals.
5. Present task schedules in a clear Gantt chart format for better comprehension.

# 3.0 System Architecture

## **Classes:**

**Task Class:**

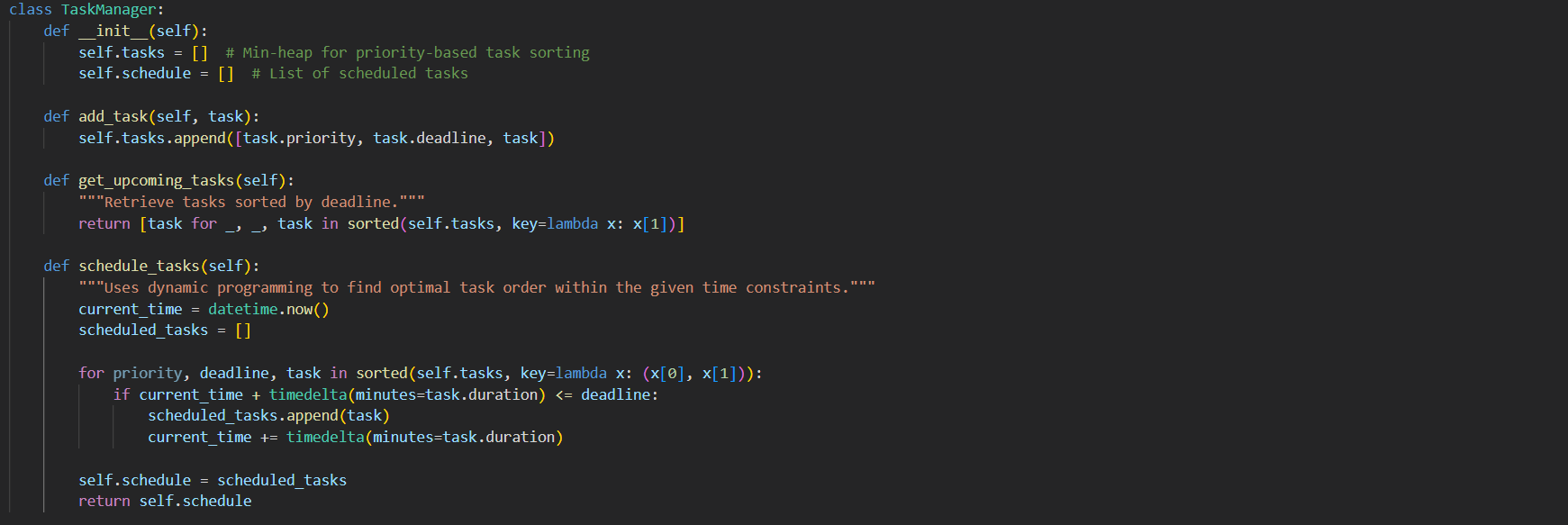
* + Attributes include:
    - name: Name of the task.
    - task\_type: Categorizes the task as "academic" or "personal".
    - deadline: Automatically computed as start\_time + duration.
    - priority: User-defined integer, with 1 being the highest priority.
    - duration: Task duration in minutes.
    - start\_time: Optional user-defined task start time.
  + Example Representation:



**Task Manager Class:**

* + Central management of tasks, implemented using:
* **Dynamic Programming for Scheduling**: Optimally fits tasks into available time slots.
* **Sorting and Binary Search**: Retrieves tasks efficiently based on specific attributes.

Example



**User Interaction Module:**

* Provides a menu-driven interface for interaction. Options include:
  + - Adding tasks.
    - Viewing upcoming tasks.
    - Scheduling tasks.
    - Task density analysis.
    - Program exit.

## Features

## **Task Addition**

* Users can input details like:
  + Name: Descriptive title of the task.
  + Type: Academic or personal.
  + Start time: Optional; defaults to system time.
  + Duration: Length of the task in minutes.
  + Priority: Integer for prioritization.

Example Input

Task Name: Research Paper

Task Type: Academic

Start Time: 2024-11-24 10:00

Duration: 120 minutes

Priority: 1

**View Upcoming Tasks**

* Displays all tasks sorted by deadlines using the heap structure for efficiency.

Example Output:

**1. Research Paper (academic) - Priority: 1, Start: 2024-11-24 10:00, Due: 2024-11-24 12:00, Duration: 120 min**

**2. Grocery Shopping (personal) - Priority: 2, Start: 2024-11-24 12:30, Due: 2024-11-24 13:15, Duration: 45 min**

**Task Scheduling**

* Implements a dynamic scheduling algorithm that ensures:
  + Tasks are fit into available slots without exceeding deadlines.
  + Higher-priority tasks are scheduled first.
* Produces a Gantt chart for visualization.

**Task Density Analysis**

* Analyzes task density in user-defined time intervals (e.g., hourly).
* Outputs intervals with the highest number of overlapping tasks.

Example Output:

**Gantt Chart Visualization**

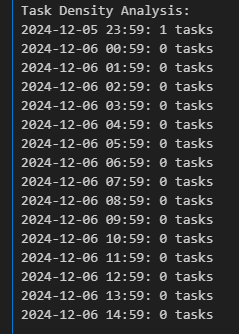
* Plots a Gantt chart categorizing tasks into academic (blue) and personal (orange).

Example Visualization:

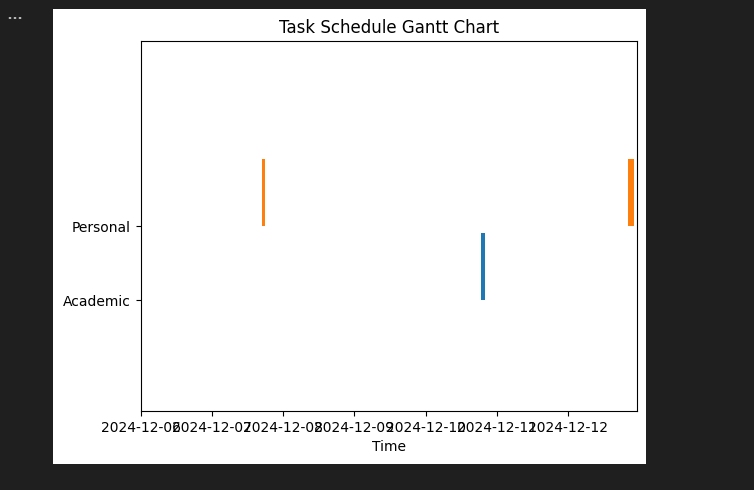
# 4.0 Algorithms and Techniques

1. **Dynamic Programming**:
   * Schedules tasks within deadlines by minimizing time conflicts and maximizing priority fulfillment.
2. **Merge Sort**:
   * Sorts tasks based on attributes like deadline or priority.
3. **Binary Search**:
   * Efficiently locates tasks with specific or closest deadlines.
4. **Task Density Analysis**:

Iterates over tasks to count those falling within specified time intervals



# 5.0 Work Flow

1. **Input Tasks**:
   * The user enters task details interactively via a menu.
2. **Process Tasks**:
   * Tasks are stored in a heap and processed for scheduling or density analysis.
3. **Output Results**:
   * Displays upcoming tasks, schedules them optimally, or generates analytical data.
4. **Visualization**:
   * Outputs a Gantt chart for enhanced clarity of the task timeline.

# 6.0 Conclusion & Future Recommendations

The Personal Scheduling Assistant successfully demonstrates how algorithmic techniques can be applied to optimize task management, enabling users to prioritize, schedule, and visualize their personal and academic responsibilities efficiently. By providing features such as dynamic task scheduling, workload density analysis, and Gantt chart visualization, the system enhances productivity and time management. Future enhancements could include integration with calendar APIs like Google Calendar for synchronization, AI-driven task recommendations to suggest optimal scheduling based on user habits, notifications and reminders for upcoming tasks, and support for custom task categories. These improvements would further refine the tool’s capabilities, making it even more versatile and effective in addressing the complexities of modern task management.