Assignment II: Task Scheduler Implementation

Student Name: _____
Student ID: _____

Instructions:

You are required to implement a Task Scheduler Implementation using a queue data structure in C++.

- Implement a **Queue** class that manages tasks.
- Each task should be represented by a simulated execution time (e.g., an integer or float).
- Provide methods to add tasks to the queue, remove tasks from the queue, and display the current queue of tasks.
- Demonstrate the scheduling and processing of tasks in your report. Simulate the execution of at least 5 tasks by printing their execution times.
- Explain how the queue data structure is used to manage task scheduling.

Submission Requirements:

- Include your well-commented source code in the report.
- Provide sample test cases and their outputs.
- Briefly explain your design choices and how OOP principles and data structures are applied.
- Ensure your code is well-structured and follows best practices.

A Introduction

This document illustrates how to use the provided LaTeX class and environments for your programming assignment submissions. Use the `codelisting` environment for your code, and the `testcase` environment for worked examples.

B Sample Code Listing

Below is an example of how to include C++ code in your report:

```cpp
// Example: Hello World in C++
#include <iostream>
using namespace std;

int main() {
    cout << "Hello, world!" << endl;
    return 0;
}
```

C Test Cases

Here are some sample test cases you can include in your report:

Test Case: Swapping Two Numbers in C++

```
int a = 5, b = 10;
swap(a, b);
cout << "a␣=␣" << a << ",␣b␣=␣" << b << endl;
\end{codelisting   }

Output:
\begin{codelisting}
a = 10, b = 5
```

D Inserting Figures

You can include figures in your report using the `figure` environment. Here is an example:

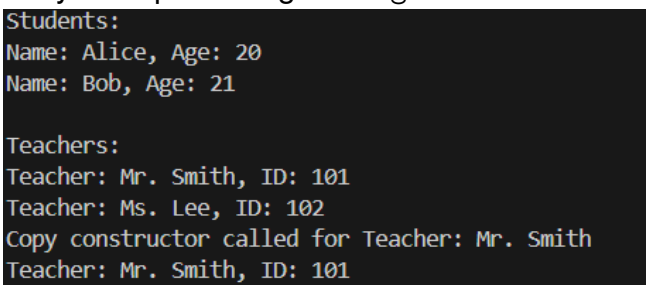

Figure 1: Sample Image