Assignment 4: Advanced Data Structures

Instructions:

You are required to implement a min-heap in C++ using Object-Oriented Programming (OOP) principles. The min-heap will be used to manage a priority queue for task scheduling based on task priority. Your program should:

- Implement a min-heap in C++ using a vector to manage a priority queue for task scheduling.
- Each task should have a priority (integer) and a description (string). The heap should always allow extraction of the task with the highest priority (lowest integer value).
- Your implementation must support insertion of new tasks, extraction (pop) of the highest-priority task, and heapify operations to maintain the heap property.
- Write a main program that demonstrates scheduling tasks by priority, showing the order in which tasks are executed.
- Include comments in your code to explain key sections and logic.
- Provide at least two test cases that show your heap correctly schedules and executes tasks based on priority.

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 are applied.
- Ensure your code is well-structured and follows best practices.

A 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;
}
```

B 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
```

C Inserting Figures

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

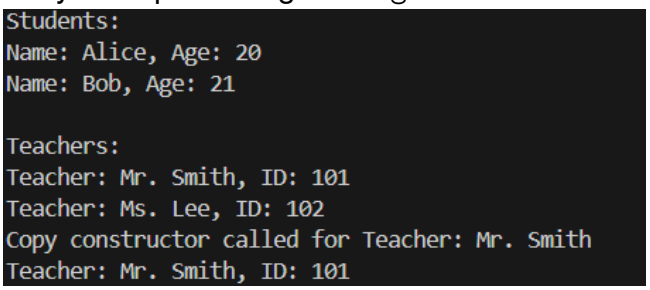

Figure 1: Sample Image