CLASS EXERCISES

Today's exercises will help you practice using structures to define custom data types and the STL containers to manage collections of data efficiently. They also provide opportunities to work on tasks commonly encountered in software development projects.

Exercise 1: Student Database

Create a program that manages a database of student records using structures and the STL containers like vector or map. Each student record should contain fields such as name, roll number, age, and grade. Implement functionalities to add new student records, display all records, search for a student by roll number, and delete a student record.

Hint

Start by defining a structure to represent a student record. This structure should contain fields such as name, roll number, age, and grade. For example:

```
struct Student {
    std::string name;
    int rollNumber;
    int age;
    double grade;
};
```

Decide on the appropriate STL container to store the student records. For a simple student database, a vector might be a suitable choice. Include the necessary headers at the beginning of your program:

```
#include <vector>
```

Create functions to perform various operations on the student database, such as adding new students, displaying all records, searching for a student by roll number, and deleting a student record.

Write the main function to interact with the user and call the appropriate functions based on their input. Provide a menu-driven interface to perform operations on the student database.

Test the program by adding sample student records, displaying them, searching for students, and deleting records. Ensure that the program behaves as expected and handles edge cases gracefully (e.g., handling invalid input).

Exercise 2: Employee Management System

Design a program to manage employee data using structures and STL containers. Each employee record should include attributes like employee ID, name, department, position, and salary. Implement functions to add new employees, display employee details, search for an

employee by ID or name, update employee information, and remove employees from the database.

Hint

Start by defining a structure to represent an employee record. This structure should contain fields such as employee ID, name, department, position, and salary. For example:

```
struct Employee {
    int employeeID;
    std::string name;
    std::string department;
    std::string position;
    double salary;
};
```

Exercise 3: Task Scheduler

Create a task scheduler program that allows users to add, update, and delete tasks using structures and STL containers. Each task should have attributes such as task ID, description, priority, deadline, and status. Implement functionalities to add new tasks, display all tasks, update task details, mark tasks as completed, and remove tasks from the scheduler.

Exercise 4: Library Management System

Develop a library management system that manages books, borrowers, and transactions using structures and STL containers. Each book record should contain details like ISBN, title, author, genre, and availability status. Implement functionalities to add new books, add borrowers, check out books to borrowers, return books, display book details, search for books by title or author, and generate reports.

Exercise 5: Student Gradebook

Design a program to maintain a gradebook for students using structures and STL containers. Each student's gradebook should include their name, roll number, and grades for multiple subjects. Implement functionalities to add new students, enter grades for subjects, calculate average grades for each student, display individual student grades, and generate a class report with statistics like average, highest, and lowest grades.