1 Introduction

This practical will involve two tasks. During this practical assignment you will be required to implement the Strategy and State design patterns. After successful completion of the practical assignment, you will:

• have an understanding of the Strategy design pattern
• have an understanding of the State design pattern
• appreciate the differences between the Strategy and State design patterns
• be able to draw UML State diagrams.

2 Constraints

1. You must complete this assignment individually.
2. Teaching assistants will be available to help you during the practical times. They will however not give you the solutions.
3. Students should attend the practical that they booked for in order to get their practical tasks marked.

3 Submission Instructions

You are required to upload all your source files and the UML state diagram image file as a single tar.gz archive to the CS website before the deadline. You are required to implement all makefiles, headers and source files yourself. You should create a Main.cpp to test your code.

4 Mark Allocation

<table>
<thead>
<tr>
<th>Task</th>
<th>Marks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strategy</td>
<td>20</td>
</tr>
<tr>
<td>State</td>
<td>25</td>
</tr>
<tr>
<td>TOTAL</td>
<td>45</td>
</tr>
</tbody>
</table>
5 Assignment Instructions

This practical will involve two tasks. Each task will focus on a specific design pattern.

Task 1: Strategy ................................................................. (20 marks)

The Strategy design pattern belongs to the behavioural family of patterns that deals with changing the behaviour of a class by changing the internal algorithm at runtime without modifying the class itself.

You are to implement the situation described below using the strategy design pattern. Design a program that incorporates Bubble Sort and Selection Sort as two strategies to sort an array of numbers.

The user should input a list of five numbers.

The strategy used to sort the numbers is determined by the sum of the five numbers.

- If the sum of the numbers is greater than 100 then Selection sort needs to be used.
- If the sum of the numbers is less than 100 then bubble sort needs to be used.

Note: The strategy used should not be determined in your main.cpp function. Your code should resemble the UML diagram below:

```
SortingContext
+ setSortingMethod(sum : Integer) : void
+ sortNumbers(numbers : int []) : void

<<Interface>>
SortingStrategy
+ sort(numbers : int []) : void

Main
+ main(args : String []) : void

SelectionSort
+ sort(numbers : int []) : void

InsertionSort
+ sort(numbers : int []) : void
```

Task 2: State ................................................................. (25 marks)

State pattern allows objects to behave in different ways depending on internal state. State is used when you need a class to behave differently, such as performing slightly different computations, based on some arguments passed through to the class.

A Mp3 player has two possible states when one repeatedly presses the “Play” button. These are:

- Playing, and
- Standby / Pause

Implement a program using the state design pattern where every time you type “Play” or “Press Play” the state of the Mp3 player changes and the state is displayed. For example:

```
>>Play
Mp3 is playing >>Play
Mp3 is paused...
```

Your code should resemble the UML diagram given on the following page.

Draw a UML State diagram showing all the states which a Mp3 player can be in and what event needs to place to effect a transition between the states. You may draw this diagram using Visual Paradigm after importing your code into Visual Paradigm.
State pattern allows objects to behave in different ways depending on internal state. State is used when you need a class to behave differently, such as performing slightly different computations, based on some arguments passed through to the class.

A MP3 player has two possible states when one repeatedly presses the "Play" button. These are:
- Playing,
- Standby / Pause

Implement a program using the state design pattern where every time you type "Play" or "Press Play" the state of the MP3 player changes and the state is displayed. E.g:

```

Mp3 is playing
Play
Mp3 is paused
```

Your code should resemble the UML diagram below:

Note: Make sure you understand the difference between State and Strategy design patterns.