CS264 Laboratory Session 8

John McDonald

08 December 2015

1 Introduction & Lab Objectives

In this lab you will gain experience with the Strategy design pattern by using it to implement a Container class for manipulating collections of strings. You will be given a single cpp file which has much of the required code implemented, but which you will need to alter and extend to complete the assignment. The code is heavily commented to help you understand how it works. A subset of the comments provide stepwise instructions for completing the assignment (broken down into 6 steps).

To submit the assignent you will receive a hackerrank link which will allow you to upload your work in sections. The hackerrank link will provide you with access to a test consisting of 3 code submission sections. For each section you will be asked to cut-and-paste a specific set of lines from your solution (corresponding to a subset of the 6 steps).

Each section will explain the aspect of your solution that it will evaluate, on a line by line basis. As such, marks will be awarded for each individual element of the assignment (e.g. correctly inheritting from a class, correctly defining an abstract base class, etc.).

Deadline & submitting your solution
To complete the assignment you must follow the hackerrank link during today's lab (i.e. 4-6pm on Dec 8th). You have 13 days from the point at which you log in to complete the assignment. So following the link today (08/12/2015) **at e.g. 4.30pm, will mean that you have until 4.29pm on Tuesday** 21^{st} **December 2015 to submit your solution. Note that you do not have to stay logged in i.e. you may return to the test at different points over the course of the 13 days. However it is probably more sensible to download the code and work on it locally on your machine, and then login and upload your solutions to hackerrank we completed. If you are unable to login to the hackerrank test today you should contact the course lecture immediately (Dr. John McDonald - johnmcd@cs.nuim.ie).**

2 Code Orientation:

To start the assignment you should first create a `Lab8` directory in your `Labs` folder, into which you should download the `Strategy.cpp` file under the Lab 8 section of the CS264 moodle page.

Note: The sample code uses some C++11 features (i.e. lambda expressions), and so you should ensure that g++ has C++11 compilation enabled when compiling the code. To do this you should use one of the following:

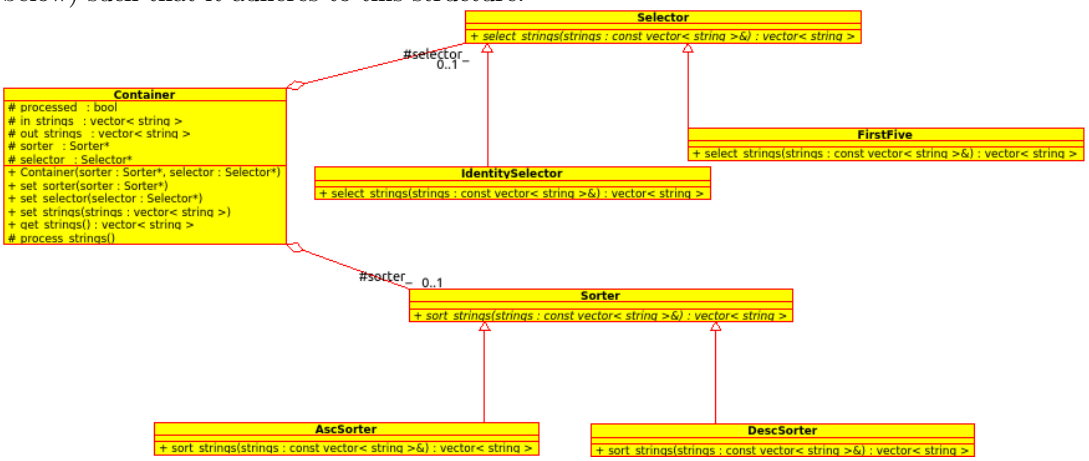
```
g++ --std=c++11 -o Strategy Strategy.cpp
```

or

```
g++ --std=c++0x -o Strategy Strategy.cpp
```

If you are using a different compiler you should refer to your compiler's documentation to ensure that it is using C++11 extensions.

The code file partially defines a number of classes to partially form the structure shown in the UML class diagram below. Your task is to complete the code (as specified below) such that it adheres to this structure.



The principal class in the hierarchy is the `Container` class whose function is to take a vector of strings as input which it sorts and selects based on the specific `Sorter` and `Selector` associated with that container. Setting the input strings to the class is done via the `set_strings` method. Getting the sorted and selected list is achieved via `get_strings`.

As you will see from the code, `get_strings` first calls `process_strings`, which in turn calls `sort_strings` and `select_strings`. Both of these calls are handled by the `Sorter` and `Selector` associated with that container. This is the *Strategy Pattern*. `Container` class also provides methods for switching the current `Sorter` and `Selector`.

The `main` routine, as it is written in the downloadable code, creates a `Container` object, called `container`, which it sets to use the `AscSorter` and `IdentitySelector`. The result is a container that will sort the input string list in ascending order and then select all elements in the resulting list.

Exercise 1.1: Your task is to complete the implementation provided in `Strategy.cpp` by following the 6 steps specified in the comments, each of which is surrounded by `/*** STEP <N>: ... ***/`.