CSE 3311 Software Design
Report 4 Specification
Due: Thursday, November 17, 5:30 pm
Where: In class
If the class has begun your report is late

1 Main Points

Be sure to read and follow all the guidelines from the links on **reports** and **academic honesty** from the WWW home page for the course. The specification is the union of this document plus the program text you are given. Pattern examples can be downloaded from the course web page by following the link *Pattern studies*.

1.1 Learning objectives

- Extending systems – dealing with multiple inheritance and feature adaptation
- Design patterns – Observer and model-view-controller
- Design description and documentation

1.2 To hand in

Hand in, in class, a report containing the following items as a package in the given order. If the package is too big to staple as one unit, then please staple in multiple parts. Cover page plus the first part of your report is part 1 of N, the remaining are parts x of N. Include your name(s) on each part. The report is required for your work to be evaluated. The electronic submission is used to run your system but with no report, the electronic submission is ignored. For reports done in pairs, include an appendix describing the contributions of the two team members.

- Cover page – – download from the forum posting with the subject *Report 4 specification*.
- Design document
- Listings of the Dequeue system classes **stack_twl.e**, **queue_twl.e**, **dequeue_twl.e**.
- Listings of your *.e files for the MVC system; except for the files **application.e**, **observer.e**, and **subject.e**.

1.3 Electronic submission

Before the deadline submit your class files and ecf files for the indicated report 4 sub-directories.

```
submit 3311 r4 Dequeue MVC
```

No other files should be submitted (e.g. EIFGEN files etc). Files cannot be deleted – the submit command can only add or replace files – so be very careful to clean up your directory before any submission. You should use **`eclean`** before submitting to clear away the unnecessary files.

While you can develop your system on your personal computer, be sure your program will compile on Prism using **estudio15.01**.

1.4 To get started

Download the file **`3311_report4.tar.gz`** to a local directory. When you untar with the command **`tar xzf 3311_report4.tar.gz`**, you will get a directory called **`report4`** that contains the following directories,

- **Dequeue** directory – the dequeue system.
  - **dequeue.ecf** – Do not modify this file.
  - **start.e** – The starting point for testing the system. Do not modify this file.
  - **stack_twl.e** – The class STACK_TWL that adapts Eiffel's two-way list structure to implement a stack ADT.
  - **stack_twl_test.e** – The test class for STACK_TWL. Do not modify this file.
  - **queue_twl.e** – The class QUEUE_TWL that adapts STACK_TWL to implement a queue ADT.
  - **queue_twl_test.e** – The test class for QUEUE_TWL. Do not modify this file.
  - **dequeue.e** – The class DEQUEUE that adapts QUEUE_TWL to implement a double-ended queue.
  - **dequeue_test.e** – The test class for DEQUEUE. Do not modify this file.

- **MVC** directory – the model-view-controller system.
  - **application.e** – Entry point – do not modify this file
  - **main_controller.e** – Defines the system view
  - **martian_temperature_controller.e** – Defines the Martian temperature view
  - **mcv.ecf** – Eiffel configuration file – do not change this file
  - **observer directory –** do not change these files
  - **temperature_farenheit_controller.e** – Defines the Fahrenheit view

2 Tasks

There are two tasks you are to complete for Report 4. For each of them, keep track of the design modifications you make, as you need to describe them in your design document. I recommend that you begin with the design document to plan and understand what you are going to do.

2.1 Designing and implementing a double-ended queue

You are to use the adaptor pattern to adapt Eiffel's TWO_WAY_LIST class to implement a sequence of linear data structures consisting of a stack, a queue and a dequeue (pronounced as "deck", a double-ended queue). Each structure is in a different class that has been started for you with the desired inheritance structure.

The STACK_TWL class uses an instance of a TWO_WAY_LIST to represent a stack. The class gives the interface for all the features you are to implement.

The QUEUE_TWL class inherits from STACK_TWL. You are to adapt the features from STACK_TWL to implement a queue.

The DEQUEUE class inherits twice from QUEUE_TWL. You are to adapt the features from QUEUE_TWL to implement a dequeue.

There are multiple ways in which the features and feature adaptation can be implemented to develop the classes STACK_TWL, QUEUE_TWL and DEQUEUE; as a consequence, you are given a set of test classes that constrain the implementation. Your report should include a discussion of possible alternatives.

An additional consequence of being given test classes is that the system, as given, will not compile until a significant part is developed, which is not a good way to develop a system. As a consequence, you need to reflect on how you can incrementally develop the system. If you type a class name in the class field of eStudio, the class will display and be editable (if you own it).

2.2 MVC pattern and Observer pattern task

Construct a model-view-controller example for the observer pattern temperature example. Figure 1 right shows the view when the completed system is loaded. Pressing the increase and decrease buttons will change the Martian temperature by 100 Martees (the Martian temperature scale), whereupon the three observers on Earth will display the corresponding temperature.

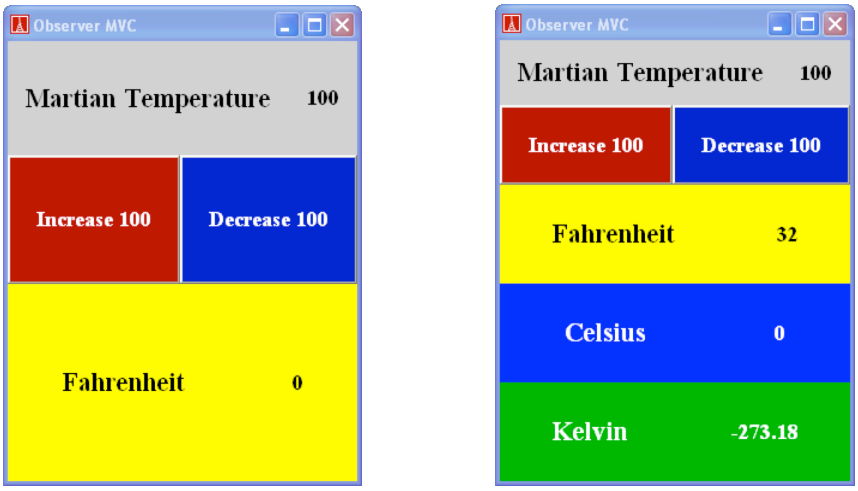

Figure 1: Left – The view in the given system.
Right – The starting view for the final system.

The given system will compile and produce the view seen in Figure 1 left. Your task is to modify the appropriate classes and create new classes to correctly implement the MVC pattern and the Observer pattern so the system works as described in opening paragraph for this section. To simplify programming, make the appropriate controllers be the subject and observers.

Secret[1] archeological studies on Mars have shown that there was intelligent life on Mars and the Martians engaged in scientific studies. Their temperature scale, called the Martee, was also defined in terms of the freezing point and boiling point of water as follows.

- Freezing point of water is 100 Martees
- Boiling point of water is 500 Martees

3 Design document

The design document has the following parts.

Part 1 is a description of what you did[2] for the Dequeue task.

Part 2 is a description of what you did[2] for the MVC and Observer pattern task.

Include detailed descriptions of all the changes you did to the distributed program text and explain the reasons for the changes. This includes introduction of new classes, moving features between classes, changing the inheritance hierarchy, redefining features etc.

[1] Secret because security services around the world did not want the public to become alarmed, as many paranoid people will believe there may be an imminent Martian attack.

[2] This is not a blow-by-blow lab report but a summary of the starting point, the finish point, how you got from start to finish with justification for major decisions made, and alternatives rejected.

4 Grading scheme

The grade for the report is partitioned into the following parts. It is expected that your systems will at least compile and execute without failing, and that for each part your program text and design document correspond.

- Dequeue system – 50%
  - 25% design document
  - 25% program text
- MVC Observer pattern task – 50%
  - 25% design document
  - 25% program text