University of Calgary
Department of Electrical and Computer Engineering
Principles of Software Design - ENSF480
Lab 4 – October 5, 2018

M. Moussavi, Ph.D. P.Eng

This is a Group Assignment:

In this lab you can work with a partner. Only groups of two are allowed. If you are working with a partner, you must only submit one copy of the lab report, with both names. Please do not submit two copies.

Objective:

This lab is different from previous labs, as there is no exercise that needs writing C++ or Java code. The purpose of this lab is to review and understand some of the basic modelling concepts, using in UML 2.5, and to get familiar with the modeling tool that we will use in this course for drawing UML diagram, called StarUML.
Exercises in this lab help you to learn:

- Class notation details
- Association and multiplicity details
- Class relationship details

And, helps you to get familiar with the process of reverse engineering of two programs one in C++ and the other one in Java codes

Marking Scheme: (40 marks total)

- Exercise A: 4 marks
- Exercise B: 6 marks
- Exercise C: 15 marks
- Exercise D: 15 marks

Method of Submission and Due Dates:

- You should submit your lab report (in PDF format), and associated files into the **Dropbox** created for lab-1 on the D2L.
- Due date is Friday October 12 at 2:00 PM

Exercise A – UML Class Notation (4 marks)

Consider the definition of the following classes in **Java** and **C++:**

```
class Point {
    private: double x, double y;
    public:
    Point (double a, double b): x(a), y(b) {}
    double getx() const {return x;}
    double gety()const {return y;}
};
```

```
class Node< D >implements Cloneable{
       Integer keyM;
       D itemM;
       Node <D>nextM;
       public Node(){
              keyM = itmeM = nextM = null;
       }

       public Node(D itemA, Integer keyA, Node < D> nextA){
              itemM= itemA ;
              keyM = keyA;
              nextM = nextA;
       }
}


class Shape{
public:
  Shape(double x_origin, double y_origin, char* name);
  ~Shape();
  Shape(const Shape& source);
  Point getOrigin();
  char* getName();
  static double distance (Shape& the_shape,  Shape& other);
  virtual double area();
protected:
  Point origin;
  char* shapeName;
};
```

Now use UML 2.5 class notations for each class, using either StatUML which is installed in our lab and you can download from Internet. You should show all attributes, behaviours and their accessibilities, and their relationships. You don't need to show the constructor destructor or assignment operator, getters and setters for any of the classes.

What to Submit:

Submit your diagram as part of your lab-report in (PDF format).

Exercise B - Association and Multiplicity/Cardinality (6 marks)

What to Do:

Consider the following simple problem statements and draw the association relationships and the cardinalities among the classes expressed in each statement. Don't worry about relationships such as aggregation, composition, realization, or inheritance, or attributes and operations in each class. Just draw simple association relationships. The focus of the question is on multiplicity/cardinality. Here are the statements:

- The Department must use one or more Generic Vector to store different type of data.
- Each course may have prerequisites.
- Each Department has many Professors and one of the Professors assumes the role of Department head.
- Each Course may have one or more Sessions and may have up to a maximum of 4 pre-requisites. Each Session can have maximum up to 80 Students, but if the number of Students enrolled in each Session is less than 8 the lecture session will be cancelled.
- Each Professor must teach one or more Sections of one or more Courses

What to Submit:

Submit your diagram as part of your lab-report in (PDF format).

Exercise C - Reverse Engineering a Java Program (15 marks)

What to Do:

In this exercise your job is to reverse engineer a Java program that I had designed for another course a few years ago. You should use one of the available tools StarUML, and draw a design-class diagram for the program that is posted on the D2L under the folder ExC. Your diagram should include all details that can be extracted from given files, such as:

- Relationship among classes (association, aggregation or composition, inheritance, realization, etc.). For associations make sure to use appropriate label and navigability.
- Multiplicity or cardinality
- Only important operations (no obviously needed functions such as: constructor, clone method, no getters and setters)
- Attributes

Note:

- Make sure to use correct notations for attributes and behaviours.
- Whenever applies use appropriate stereotypes.

What to Submit:

Submit your diagram as part of your lab-report in (PDF format).

Exercise D - Reverse Engineering a C++ Program (15 marks)

In this exercise you will reverse engineer a C++ program. You should use StarUML, and draw a class diagram for the program that is posted on the D2L under the folder ExD. Your diagram should include all details that can be extracted from given files, such as:

- Relationship among classes (association, aggregation or composition, inheritance, realization, etc.). For associations make sure to use appropriate label and navigability.
- Multiplicity or cardinality
- Only important operations (no obviously needed functions such as: constructor, copy constructor, assignment operator, no getters and setters)
- Attributes

Note:

- Make sure to use correct notations for attributes and behaviours.
- Whenever applies use appropriate stereotypes

What to Submit:

Submit your diagram as part of your lab-report in (PDF format).