M.M.

University of Calgary
Schulich School of Engineering

Department of Electrical and Computer Engineering

ENSF 480: Principles of Software Design

Final Exam

Exam Duration: 3 Hours
Time: 12:00 – 3:00 PM
Date: Dec 12, 2018
Location: Aux Gym

Instructor: Mahmood Moussavi

First Letter of Your Last Name:
Your Last Name:
Your First Name:
Your Student Id:

Section I	/ 14
Section II	/23
Section III	/50
Section III	/20
Section IV	/20
Total	**/127**

Please read these Instructions first:

- You may **not** use any electronic devices during the test.
- The test is **closed book.** You may **not** refer to books or notes during the test.
- To reduce distraction to other students, you are not allowed to leave during the last **15 minutes** of the test.
- Write all answers on the question paper and hand in the question paper when you are done.
- Please print your answers legibly. What cannot be read cannot be marked.
- If you write anything that you do not want to be marked, put a large **X** through it and write "rough work" beside it.
- The official University of Calgary examination regulations are also printed on page 2.

EXAMINATION RULES AND REGULATIONS

STUDENT IDENTIFICATION

Each candidate must sign the Seating List confirming presence at the examination. All candidates for final examinations are required to place their University of Calgary I.D. cards on their desks for the duration of the examination. (Students writing mid-term tests can also be asked to provide identity proof.) Students without an I.D. card who can produce an acceptable alternative I.D., e.g., one with a printed name and photograph, are allowed to write the examination.

A student without acceptable I.D. will be required to complete an Identification Form. The form indicates that there is no guarantee that the examination paper will be graded if any discrepancies in identification are discovered after verification with the student's file. A Student who refuses to produce identification or who refuses to complete and sign the Identification Form is not permitted to write the examination.

EXAMINATION RULES

(1) Students late in arriving will not normally be admitted after one-half hour of the examination time has passed.

(2) No candidate will be permitted to leave the examination room until one-half hour has elapsed after the opening of the examination, nor during the last 15 minutes of the examination. All candidates remaining during the last 15 minutes of the examination period must remain at their desks until their papers have been collected by an invigilator.

(3) All inquiries and requests must be addressed to supervisors only.

(4) Candidates are strictly cautioned against:

(a) speaking to other candidates or communicating with them under any circumstances whatsoever;

(b) bringing into the examination room any textbook, notebook or memoranda not authorized by the examiner;

(c) making use of calculators and/or portable computing machines not authorized by the instructor;

(d) leaving answer papers exposed to view;

(e) attempting to read other student's examination papers.

The penalty for violation of these rules is suspension or expulsion or such other penalty as may be determined.

(5) Candidates are requested to write on both sides of the page, unless the examiner has asked that the left hand page be reserved for rough drafts or calculations.

(6) Discarded matter is to be struck out and not removed by mutilation of the examination answer book.

(7) Candidates are cautioned against writing in their answer book any matter extraneous to the actual answering of the question set.

(8) The candidate is to write his/her name on each answer book as directed and is to number each book.

(9) A candidate must report to a supervisor before leaving the examination room.

(10) Answer books must be handed to the supervisor-in-charge promptly when the signal is given. Failure to comply with this regulation will be cause for rejection of an answer paper.

(11) If during the course of an examination a student becomes ill or receives word of a domestic affliction, the student should report at once to the supervisor, hand in the unfinished paper and request that it be cancelled. If physical and/or emotional ill health is the cause, the student must report at once to a physician/counsellor so that subsequent application for a deferred examination is supported by a completed Physician/Counsellor Statement form. Students can consult professionals at University Health Services or University Counselling Services during normal working hours or consult their physician/counsellor in the community.

Should a student write an examination, hand in the paper for marking, and later report extenuating circumstances to support a request for cancellation of the paper and for another examination, such a request will be denied.

(12) Smoking during examinations is strictly prohibited.

RO 94-01

Section I – Multiple Choice Questions (1 mark each):

Please select the **BEST** answer.

1. UML use case diagram is used to:
    a. Model non-functional requirements of the system
    b. Show the order of activities in a software
    c. Show temporal behaviour of the system or a system's components
    d. Model functional requirements of the system
    e. Both b and c are correct answers
    f. None of the above is a correct statement for a use case diagram

2. In a use case diagram the line between an actor and a use case:
    a. Indicates a communication between an actor and a use case
    b. Indicates that system may require some sort of software or hardware interface
    c. May appear as a boundary class in the class diagram of system
    d. Both statements (a) and (b) are correct.
    e. All of the above statements are correct.
    f. None of the above is correct

3. A UML state transition diagram is normally used to:
    a. Specify the flow of data in system.
    b. Specify the flow of control and sequence of actions in a system
    c. Specify the time-based behaviour of a system or a system's components
    d. All of the above are correct statements

4. Strong coupling leads to a program that:
    a. Is easier to maintain.
    b. Is more efficient.
    c. Is not easier to maintain but it is more efficient.
    d. None of the above

5. A C++ or Java class defines the common characteristics of a group of objects by its:
    a. Identity, attributes, and operations
    b. Attributes, and operations
    c. Identity, behaviour and state
    d. Attributes, operations and relationships
    e. Relationships, operations and multiplicity

Consider the **partial** definition of the following C++ classes, and the given `main` function, to answer the following two questions.

```
class A{
  char * s1;
public:
  A(int n){
    s1= new char[n];
  }
  ~A() { delete [] s1;}
};
```

```
class B : public A {
  char * s2;
public:
  B(int n):A(n) {
     s2=new char[n];
  }
  ~B() { delete [] s2;}
};
```

```
void main(void)  {
  A *p1 = new B(5);
  B *p2 = new B(6);
  p1 = p2;
  p2 = p1;
  delete p1;
  return 0;
}
```

6. The above C++ program has compilation error because:
    a. Destructor of class A is not properly defined
    b. Constructor of class B is not properly defined
    c. `p1` and `p2` in the main function are not properly declared
    d. None of the above

7. If there is any compilation error in the given C++ and its fixed. The program may still face a runtime error because:
    a. Constructor of class B is not properly defined
    b. `p1` and `p2` are not properly used
    c. Destructor of class A is not properly defined
    d. There is no compilation error in this program
    e. None of the above

8. Object-oriented software development supports two types of hierarchy which includes:
    a. Class hierarchy and entity hierarchy
    b. Whole-part hierarchy and Generalization-Specialization hierarchy
    c. Class hierarchy and structure chart
    d. Class hierarchy and process hierarchy
    e. None of the above

9. A boxes on the top of the sequence diagrams **always** represent:
    a. A use case
    b. A process
    c. A state
    d. An object
    e. None of the above

10. A derived class in C++ inherits all member functions except:
    a. Constructors
    b. Copy constructor
    c. Assignment operators
    d. All of the above
    e. None of the above

Consider partial definition of the following C++ classes called `Base` and `Child1` and answer the following two questions:

```
class Base {
public:
    void foo() {cout << "Base foo\n";};
    virtual void bar(){cout << "Base bar\n";}
};
class Child1: virtual public Base {
public:
   void foo() {cout << "Child1 foo\n";};
   void bar() {cout << "Child1 bar\n";}
};
```

11. The `virtual` keyword in front of `public Base` in class `Child1` means:
    a. Member functions of class `Child1` will be inherited to its subclasses
    a. Member functions derived from `Base` are all `virtual` by default
    b. Both statements (a) and (b) are correct.
    c. None of the above statements about `virtual` keyword in front of `public Base` are correct

12. The `virtual` keyword in front of `bar` member function in class `Base` allows:
    a. Member function to be inherited to the subclasses
    b. Dynamic (runtime) binding of an object to a member function
    c. Static binding of an object to a member function
    d. None of the above

13. Which of the following statement(s) is a basic principle for many design patterns
    a. Separation of changeable behaviours from steady behaviours
    b. Programming to interface not to implementation
    c. Programming to implementation not to interface
    d. Both statements (a) and (b) are principle statements for many design pattern
    e. None of the above

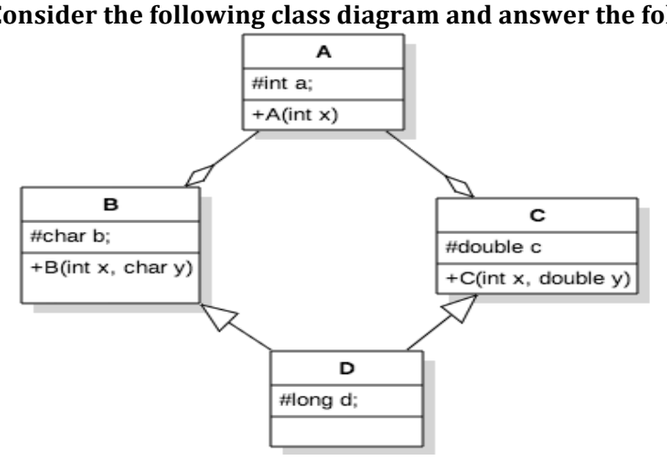
14. Java I/O is a real word example of:
    a. Strategy pattern
    b. Decorator pattern
    c. Bridge pattern
    d. Adaptor pattern
    e. None of the above

Section II - Short Answer Questions:

Question 1: Consider the following statements and draw simple class diagram (using UML 2.5) with simple **association** relations that show the cardinalities among classes **(3 marks)**:

- Every Invoice must have a Date.
- Each Computer may use a Printer
- Each Invoice may have many Items

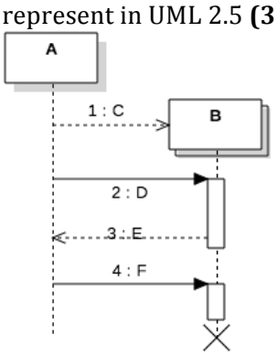
Consider the following class diagram and answer the following two questions:



Question 2: in the following space write the implementation of a constructor for class C **(2 marks)**.

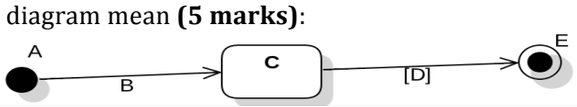
Question 3: in the following space write the implementation of a constructor for class D **(2 marks)**.

Question 4 - Consider the following diagram and in the following table indicate what does each notation represent in UML 2.5 **(3 marks)**:



Label element	The label or component is a:
A	
B	
C	
D	
E	
F	

Question 5 – Consider the following diagram and in the following table indicate what does each notation of this diagram mean **(5 marks)**:



Component/label	What each component means
A	
B	
C	
D	
E	

Question 6: Consider the following program **(8 marks)**:

```
class Animal{
public:
    virtual void sleep() = 0;
    void walk(){cout << "Animal walks." << endl;}
    virtual void eat()= 0;
    virtual void talk(){cout << "Animal talks." <<
endl;}
};

class Cat: public Animal{
public:
    void sleep(){cout << "Cat sleeps." << endl;}
    void walk(){cout << "Cat walks." << endl;}
    virtual void eat() {cout << "Cat eats." << endl;}
    void talk(){cout << "Cat talks." << endl;}
};

class Tiger: public Cat{
public:
    void sleep(){cout << "Tiger sleeps." << endl;}
    void walk(){cout << "Tiger walks." << endl;}
    void eat(){cout << "Tiger eats." << endl;}
    void talk(){cout << "Tiger talks." << endl;}
};
```

```
class Lion: public Tiger{
public:
    void walk(){cout << "Lion walks." << endl;}
    void talk(){cout << "Lion talks." << endl;}
};

int main() {
    Lion lion;
    Animal *animal = & lion;
    animal -> walk();
    animal -> eat();
    animal -> talk();
    animal -> sleep();
    return 0;
}
```

What is the output of this program?

Section III – UML Modelling

Consider the following C++ classes:

```
class Tax{
private:
  int amount;
  static int stock;
public:
  Tax(int x):amount(x){}
};

int Tax::stock;
```

```
class Account{
private:
   int id;
public:
   Account(int
x) :id(x){}
};
```

```
class Stock{
private:
   int id;
public:
   Stock(int x):id(x){}
   void print(Tax* ben){}
};
```

```
class Customer{
private:
  Account acc;
  Stock *sto;
public:
  Customer(int x, Stock& s):acc(x){
     sto = &s;
   }
};
```

Question 1 - In the following space draw a UML class diagram that shows attributes, and relationships among classes (Don't worry about behaviours). If there is a whole/part relationship use aggregation or composition notations, whichever is more appropriate: **(5 marks)**.

Question 2 - Consider the following C++ classes:

```
class A: public E{
 protected:
   int id:
 public:
   A(int x);

 void fun1();
 void fun2 ();

};
```

```
class B:public A{
 protected:
  int id;
 public:
  B(int x, int y);
  // more code
 };
```

```
class C: public A{
 protected:
  int id;
 public:
  C(int x, int y);
    // more code
 };
```

```
class E {
 public:
  virtual void fun1() =0;
  virtual void fun2() =0;
 // No data members – serves as an interface
};
```

```
class D:public B,public C{
 private:
  int id;
 public:
  D(int x,int y,int z);
 // more code
};
```

In the following space draw a class diagram that shows the attributes, and relationships such as inheritance and realization among them. Don't worry about behaviours. **(6 marks**):

Question 3: In the following space write your comments to explain, whether anything should be done in the above code to prevent any possible implementation ambiguity for accessing the data members or functions of these classes **(2 marks)**.

Question 4 - Use case diagram details

In an online shopping system for Canadian customers the following use cases are needed: *make payment, pay GST, pay shipping cost (minimum shipping cost applies to every item -normally take couple of weeks), pay for fast shipping (optional for customers who want to pay extra cost; but they receive their parcel within 48 hours)*. In the following space draw a use case diagram that shows customer makes payment. Try to use: `<<include>>` or `<<extend>>` to better express this scenario **(5 marks)**.

Question 5 – In this question you should draw a state transition diagram for an object of Java ArrayList. For your information: *In Java when ArrayList is originally created with certain capacity, it is empty and its size is zero. After adding N data its size becomes equal to the capacity and at that point its capacity will be increased (based on certain memory management policy) to allow adding a new data. To understand this process read carefully the following Java program and its given output. Don't worry about Java policy of increasing the capacity and about the implementation of method `getCapacity` that returns the capacity of the list* **(10 marks).**

```
import java.util.ArrayList;
public class Test {
 public static void main(String[] args) throws Exception {
   ArrayList<Integer> list;
   list = new ArrayList<Integer>(3);
   System.out.format("Size: %2d, Capacity: %2d%n", list.size(), getCapacity(list));
   for (int i = 0; i < 10; i++) {
    list.add(i);
    System.out.format("Size: %2d, Capacity: %2d%n", list.size(), getCapacity(list));
   } // end of for loop
   list.clear();
   System.out.format("Size: %2d, Capacity: %2d%n", list.size(), getCapacity(list));
 }  // end of main
}
```

Here is the program output:

```
Size:  0, Capacity:  3
Size:  1, Capacity:  3
Size:  2, Capacity:  3
Size:  3, Capacity:  3
Size:  4, Capacity:  4
Size:  5, Capacity:  6
Size:  6, Capacity:  6
Size:  7, Capacity:  9
Size:  8, Capacity:  9
Size:  9, Capacity:  9
Size: 10, Capacity: 13
Size:  0, Capacity: 13
```

In the following space draw a state transition diagram that expresses the state of an ArrayList object:

Question 6: In the space after this program draw a **sequence diagram** that models the behaviour of the program from the point that function `start` is called. Make sure to specify the boundary and control class for this program. Please consider an actor called **User** to show that he starts the game **(10 marks).**

```
class Ship {
  int id;
  public Ship(int i) {id = i;}
  public void draw(int x, int y){
    System.out.println("ship displayed");
  }
  public void move(int x, int y){
    System.out.print("ship moving");
  }
}

class Cannon {
  int id;
  public Cannon(int i) {id = i;}
  public void draw(int x, int y) {
   System.out.print("cannon displayed");
  }
  public void shoot() {
   System.out.print("cannon shooting");
  }
}

class BattleFieldViewer{
  ManageBattle mb;
  public BattleFieldViewer(ManageBattle mb){
    this.mb = mb;
  }
  public void displayfield() {mb.fi.draw(); }
  public void displayShip(int x,int y){
    mb.sh.draw(x,y);
  }
  public void displayCannon(int x, int y){
  mb.ca.draw(x, y);}
}
```

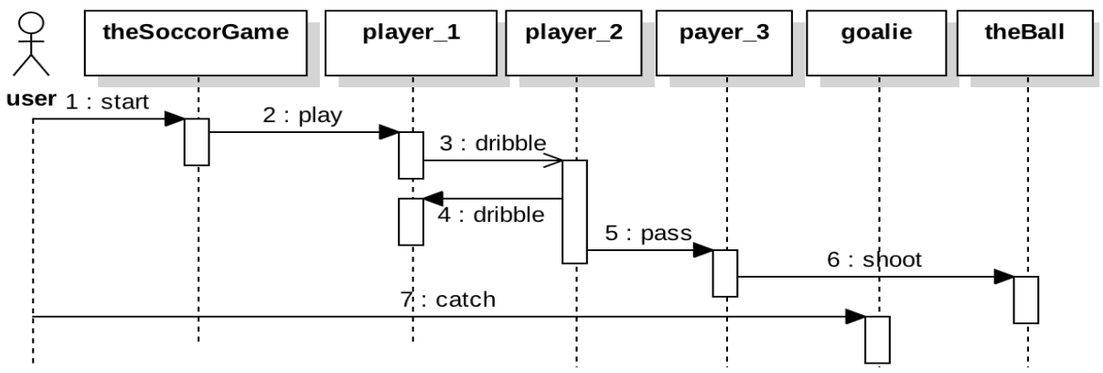
```
class Field {;
  public void draw(){
    System.out.print("field displyed.");
  }
}

class ManageBattle{
  BattleFieldViewer bf;
  Ship sh;
  Cannon ca;
  Field fi;
  public void start(){
      sh = new Ship(1);
      ca = new Cannon(2);
      fi = new Field();
      fi.draw();
      sh.draw(10, 10);
      ca.draw(20, 20);
      ca.shoot();
      sh.move(200, 200);
  }
}

public class Game {
 public static void main(String[] arg){
   ManageBattle bf = new ManageBattle();
   bf.start();
 }
}
```

Question 7: Draw a class diagram for the following C++ class code and show all the possible details **(4 marks)**.

```
template <class D>
class Utility{
    int x;
public
    int sort(int key);
};
```

```
template <class D>
class Collection{
protected:
     void sort(Utility <D> utilities[]);
private:
    int y;
};
```

Question 8: Consider the following sequence diagram and draw a class diagram the show their relationship and their behaviours. Please don't consider user as class and don't worry about attributes of each class (**8 marks).**



SECTION III – Application Level Analysis & Design - Process Oriented (20 marks):

A traffic system on an intersection consists of several devices such as traffic lights, pedestrian lights, pedestrian buttons, remote emergency vehicle sensors, and finally a camera with a motion sensor that its images can be used to keep track of number of cars stopped on each side of the intersection when the traffic light is red. Normally traffic lights operate on a timely fashion (3 minutes green, 3 seconds yellow, 3 minutes red), unless that one of the followings events happens:

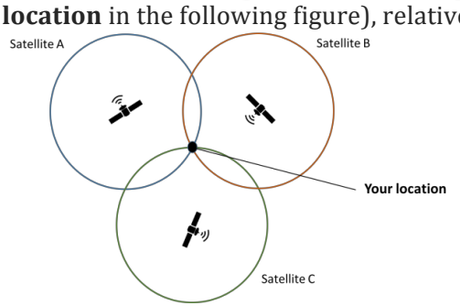
- **The pedestrian-button is pushed:** If this event is not within the rush hour, the timing of the street that the pedestrians need to cross will change to: 1.5 minute green, 3 second yellow, 3 minutes red.
- **An emergency vehicle such as police arrives and the remote emergency vehicle sensor goes off:** In this case the traffic lights on all sides will change to red. Once the vehicle is passed it resets the lights back to normal
- **The number of stopped cars on any side exceeds certain threshold:** In this case the traffic light on the busier side of the intersection will follow a different rule. Note: threshold can be set by a city-traffic-center via Internet connection.

Question 1: In the following space draw a context diagram for this system (**6 marks**)

Question 2 (14 marks): In the following space draw a DFD diagram (The level below context diagram).

SECTION IV – Application Level Analysis & Design – Object Oriented (20 marks):

In this section you produce a UML design artefacts for a GPS system in car. For your information, the process of calculating a car location is based on ***trilateration*** method that uses overlapping regions to find a common, intersecting area. On a 2-D space consider the location of a receiver on a map (shown as **your-location** in the following figure), relative to three satellites A, B, and C:



Now if you draw a circle that satellite A is the center and it passes '**your location'** the line that represents the perimeter of the circle also represents all the possible places that have the same distance to the satellite A. If you draw another circle to represent your distance from Satellite B, the number of pointes that can have the same distance to both satellites will reduce to two. And finally, by drawing a third circle, there will be only one point where all three circles intersect and that is your location. The distance between your-location and each satellite is calculated by timing a signal's journey from satellite to receiver. In other words distance is calculated from the time a radio signal travels between satellite and receiver, where the radio signals' travel with speed of light (300,000 km/sec).

Here are some essential required functionalities of this system:

a. An available regional-map (Example: North America) should be loaded from a built disk.
b. System should be able to read the signals received from a built-in device called, satellite-receiver.
c. When a radio signal is received, the system should also receive the exact satellite-time (which is the time when the signal was sent), and should be able to covert it to the corresponding local time of GPS location.
d. In concurrence with above item (item c), when a radio signal is received, the system should immediately read the local time.
e. System should be able to calculate its distance from three satellite based on real-time information received in (c), and (d).
f. System should be able to display the current location on the GPS monitor.
g. System should able to provide trip guidance to the user's destination via a built-in speaker and on the GPS monitor

Question 1 – in the following space draw a use case diagram for this system **(6 marks)**.

Question 2 – in the following space draw an activity diagram that expresses the major activities from the moment that user turns on the GPS, followed by, getting each satellite times, finding the current position of the user. Please consider three swim lanes in your diagram: User , Satellite, and GPS **(14 marks).**