SEMESTER 1 SOLUTIONS 2022/2023

MODULE: EE402 - OOP with Embedded Systems

PROGRAMME(S):

ECSAO	Study Abroad (Engineering & Computing)
MECE	MEng Electronic & Computer Engineering
MCTY	MSc Electronic and Computer Technology
ECE	BEng Electronic & Computer Engineering
ECEI	BEng Electronic & Computer Engineering

YEAR OF STUDY: 1, 4, C, O

EXAMINER(S):

Prof. Derek Molloy	(Internal)	(Ext:5355)
Prof. Roberto Verdone	(External)	External
Prof. Martin Glavin	(External)	External

TIME ALLOWED: 3 Hours

INSTRUCTIONS: Answer Question 1 and any three questions from Question 2, 3, 4 and 5. All questions carry equal marks and the total marks available is 100.

Please type your DCU Examination ID here: Click here to enter text.

PLEASE DO NOT TURN OVER THIS PAGE UNTIL YOU ARE INSTRUCTED TO DO SO.

This is a computer-based examination. This is your exam paper and your examination script.

The use of programmable or text storing calculators is expressly forbidden.
Please note that where a candidate answers more than the required number of questions, the examiner will mark all questions attempted and then select the highest scoring ones.

This is a computer-based examination. Save any working files, including this document, in the directory `C:\Temp\` as all other locations are lost on a power cycle of the PC. Save this document regularly.

At the end of the examination you must upload this completed document as a PDF to the Loop Exam site. You are not permitted to submit any other files. Do not re-order or edit the questions.

There are no additional requirements for this paper.

QUESTION 1 (Compulsory) *[TOTAL MARKS: 25]*

Q 1 Answer the following questions. Please keep any text answers concise.

a) There are no global variables or functions in Java. What can we use in their place? Give an example of where this usage occurs in the core API language. (2 marks)

Replace this text with your answer to Q1(a)

Use static functions as part of a class. Math.random() etc. will suffice as examples

b) Which of the following are default return types for a function in C++? Select all that apply. (2 marks)

☐char
☐void
☐double
☐float
☒int

c) Compare the use of finalize() in Java to a C++ destructor. (2 marks)

Replace this text with your answer to Q1(c)

Finalize provides clean up function on the destruction of an object, just like a destructor but because the way that memory is managed there is no guarantee that it will be called before a program exits.

d) Which of the following statements are true in relation to *has-a* and *is-a* relationships? (2 marks)

☐ inheritance represents a *has-a* relationship
☐ interfaces must be used in Java when creating *has-a* relationships
☒ instance variables can be used when creating *has-a* relationships
☒ inheritance represents an *is-a* relationship
☐ unlike C++, inheritance in Java is limited to a single *has-a* relationship

e) In C++ what is the rationale for methods being non-virtual by default? Why is this the case and when is it considered good practice to use non-virtual functions in C++? (2 marks)

Replace this text with your answer to Q1(e)

Performance. Virtual functions require additional code to be introduced by the compiler for RTTI. If they are used in a class that will not have any child classes.

f) Given only the following segment of Java code, complete X, Y and Z: (3 marks)

```
interface Reloadable {
   public void reload();
}
class Test {
   public void edit() {/* add code here */}
}
interface Displayable X Y {
   Z
}
```

X, **Y** and **Z** can be replaced by (select each from each row):

X = ☒ extends ☐ implements ☐ public
Y = ☐ Display ☒ Reloadable ☐ Test
Z = ☐ main() ☐ public void display() {/*code*/} ☒ public void display();

g) Write a small segment of code in C++ to demonstrate how a function can receive an object that has a different dynamic type that the static type of the function parameter: (3 marks)

Replace this text with code to answer Q1(g)

The class must have an inheritance relationship

The parameter must be a pointer of the parent type.

h) Given the following segment of Java code, select all of the statements that follow which are true: (3 marks)

```
class X {
     static void alpha()  { /* code here */ }
     void beta()   { /* code here */ }
}
```

☐ X.beta() is a valid invocation of beta()
☒ X.alpha() is a valid invocation of alpha()
☒ The beta() method can directly call method alpha()
☐ The alpha() method can directly call method beta()

i) Is it possible to overload a destructor in C++? Give reasons for your answer and example code to demonstrate your answer. (2 marks)

Replace this text with your answer to Q1(c)

No. A destructor cannot have any parameters. Code example should just show a standard destructor with no parameters.

j) Select which of the following statements are true in relation to friend functions in C++. Choose all that apply. (2 marks)

☐ Friend functions can be members of the class to which friendship is granted.
☐ Friendship relationships are automatically inherited.
☒ Friendship relationships are not transitive.
☐ Friend functions can access the private, but not the protected states of a class.
☐ Friend functions must be virtual to allow for dynamic binding.

k) The following is an example of const modified methods in a class. Review the code example and select which of the statements that follow are true. (2 marks)

```
class Number{
private:
    int value;
public:
    Number(int value): value(value) {};
    void a() const {};
    void b() {};
    void c() const;
};

void Number::c(){}
```

☒ Const and non-const functions can be called by non-const modified objects.
☐ Non-const functions cannot be called by const modified objects if the non-const function does not modify the object.
☒ Non-const functions cannot be called by any const modified objects.
☐ The implementation of `c()` is missing a `const` keyword before `Number::c(){}`
☒ The implementation of `c()` is missing a `const` keyword after `Number::c(){}`

[End of Question 1]

QUESTION 2 *[TOTAL MARKS: 25]*

Q 2(a) **[10 Marks]**

Using OOP best practice, write all of the code necessary to ensure that the following block of code:

```
using EE402::Point2D;
using EE402::Vector2D;

int main() {
	Point2D  a(2, 3.5), b(5, 6.5);
	Vector2D v(1.1, 2.5);
	cout << "Point a has values " << a << endl;
	cout << "Point b has values " << b << endl;
	cout << "The midpoint of a and b is " << a.midPoint(b) << endl;
	cout << "Vector v has values " << v << endl;
	Point2D c = a + v;
	cout << "Point c = a + v has values " << c << endl;
	Vector2D w = b - a;
	cout << "The vector w = b - a is " << w << endl;
	cout << "The magnitude of w is " << w.getMagnitude() << endl;
	return 0;
}
```

results in the following output:

```
Point a has values (2, 3.5)
Point b has values (5, 6.5)
The midpoint of a and b is (3.5, 5)
Vector v has values [1.1, 2.5]
Point c = a + v has values (3.1, 6)
The vector w = b - a is [3, 3]
The magnitude of w is 4.24264
```

HINT**:** The output stream operator, where `T` is a type, has the general form:

`ostream& operator << (ostream&, const T&)`

and the magnitude of a 2D vector $|\vec{v}| = \sqrt{{v_x}^2 + {v_y}^2})$

Replace this text with your solution. You may cut and paste the code above into your main() function and please paste your output into this box.

Example Solution:

```
#include <iostream>
#include <cmath>
using namespace std;

namespace EE402{

class Vector2D{
private:
	double x,y;
public:
	Vector2D(double x, double y);
```

```
        virtual double getX() { return x; }
        virtual double getY() { return y; }
        friend ostream& operator << (ostream&, const Vector2D&);
        virtual double getMagnitude();
};
Vector2D::Vector2D(double x, double y): x(x), y(y) {}

double Vector2D::getMagnitude(){
        return sqrt((x*x)+(y*y));
}

class Point2D{
private:
        double x,y;
public:
        Point2D(double x, double y);
        Point2D midPoint(Point2D);
        friend ostream& operator << (ostream&, const Point2D&);
        virtual Point2D operator + (Vector2D v);
        virtual Vector2D operator - (Point2D p);
};
Point2D::Point2D(double x, double y): x(x), y(y) {}
Point2D Point2D::midPoint(Point2D p){
        return Point2D( (x+p.x)/2, (y+p.y)/2 );
}

Point2D Point2D::operator + (Vector2D v){
        return Point2D(x+v.getX(), y+v.getY());
}

Vector2D Point2D::operator - (Point2D p){
        return Vector2D(x - p.x, y - p.y);
}

ostream& operator << (ostream& os, const Vector2D& v){
        os << "[" << v.x << ", " << v.y << "]";
        return os;
}

ostream& operator << (ostream& os, const Point2D& a){
        os << "(" << a.x << ", " << a.y << ")";
        return os;
}

}
```

Q 2(b) [9 Marks]

Write the code necessary to sort a C++ Vector of Vector2D objects from Q 2(a) by magnitude. Demonstrate that your code works correctly.

```
class Vector2D{
private:
        double x,y;
public:
        Vector2D();
        Vector2D(double x, double y);
        …
        virtual bool operator < (const Vector2D&) const;
        virtual bool operator == (const Vector2D&) const;
        virtual double getMagnitude() const;
};
Vector2D::Vector2D(): x(0), y(0) {}
Vector2D::Vector2D(double x, double y): x(x), y(y) {}

double Vector2D::getMagnitude() const{
        return sqrt((x*x)+(y*y));
}

bool Vector2D::operator < (const Vector2D& v) const{
        return (this->getMagnitude() < v.getMagnitude());
}

bool Vector2D::operator == (const Vector2D& v) const{
        return (getMagnitude() == v.getMagnitude());
}
```

The solution requires that const values are added along with the < and == operators in order to use the std::sort() algorithm.

```
void outputFn(Vector2D v){
        cout << v << endl;
}

int main() {
        …

        vector<Vector2D> vectors;
        vectors.push_back(Vector2D(1,1));
        vectors.push_back(Vector2D(2,2));
        vectors.push_back(Vector2D(5,5));
        vectors.push_back(Vector2D(3,3));
        vectors.push_back(Vector2D(4,4));
        std::sort(vectors.begin(), vectors.end());
        for_each(vectors.begin(), vectors.end(), outputFn);
        return 0;
}
```

This addition to main() tests the code and will result in the sorted output:

```
[1, 1]
[2, 2]
[3, 3]
[4, 4]
[5, 5]
```

Q 2(c) **[6 Marks]**

Write a non-member operator * that allows a Vector2D to be multiplied by a double as illustrated in the following code segment:

```
        Vector2D v(2.0, 5.0);
        v = v * 2.0;
        v = 2.0 * v;
        cout << "Vector v has the value " << v << endl;
```

To give the output:

```
Vector v has the value [8, 20]
```

Replace this text with your solution.

```
class Vector2D{
private:
        double x,y;
public:
        Vector2D(double x, double y);
        virtual double getX() const { return x; }
        virtual double getY() const { return y; }
        virtual bool operator < (const Vector2D&) const;
        virtual bool operator == (const Vector2D&) const;
        virtual double getMagnitude() const;
        friend ostream& operator << (ostream&, const Vector2D&);
        virtual ~Vector2D() {}
};
Vector2D::Vector2D(double x, double y): x(x), y(y) {}

double Vector2D::getMagnitude() const{
        return sqrt((x*x)+(y*y));
}

…

Vector2D operator * (const Vector2D & left, const double d){
        return Vector2D( (double) left.getX() * d, (double) left.getY() * d);
}

Vector2D operator * (const double d, const Vector2D & right){
        return (right*d);
}
}
```

[End of Question 2]

QUESTION 3 (Optional) ***[TOTAL MARKS: 25]***

Q 3(a) **[13 Marks]**

Write a template class called `Numbers` that can store an arbitrary number of elements, sum, add, swap, remove and display all elements in the store. The swap functionality should not permit out of bounds operations. Once complete, the following segment of code should give the exact output that follows:

```
int main() {
   int test[] = {11, 22, 33, 44, 55};
   Numbers<int, 10> intStore;
   Numbers<double, 20> doubleStore;
   for (int i=0; i<5; i++){
        intStore.addElement(test[i]);
        doubleStore.addElement(0.3333 * (double) test[i]);
   }
   doubleStore.removeLastElement();
   cout << "The size of the intStore is " << intStore.size() << endl;
   cout << "The sum of the doubleStore is " << doubleStore.sum() << endl;
   if (intStore.swap(6,-1) == false){ cout << "First swap was not allowed" << endl; }
   if (intStore.swap(0,1) == false){ cout << "Second swap was not allowed" << endl; }
   cout << "The first element in the intStore is now " << intStore[0] << endl;
   cout << "The elements in the doubleStore are " << doubleStore.display() << endl;
   return 0;
}
```

```
The size of the intStore is 5
The sum of the doubleStore is 36.663
First swap was not allowed
The first element in the intStore is now 22
The elements in the intStore are [3.6663][7.3326][10.9989][14.6652]
```

HINT: The stringstream class from <sstream> might be useful to you for this question. You can use it as an output stream and convert it to a string using its str() method.

```
//Replace this text with your solution.

template<class T, int maxsize>
class Numbers{
  T values[maxsize];
  int count;
  T total;

  public:
    Numbers(){
      total = 0;
      count = 0;
    }

    T& operator [](int index){
      return values[index];
    }

    T sum(){
        total = 0;
        for(int i=0; i<count; i++){
                total+=values[i];
        }
        return total;
    }

    int size(){
        return count;
    }

    T addElement(T element){
        values[count] = element;
        count++;
        return element;
    }

    T removeLastElement(){
        count--;
        return values[count];
    }

    bool swap(int i, int j){
        if (i < 0 || j < 0 ) return false;
        if (i >= count || j >= count ) return false;
        T temp = values[i];
        values[i] = values[j];
        values[j] = temp;
        return true;
     }

    string display(){
        stringstream temp;
        for(int i=0; i<count; i++){
            temp << "[" << values[i] << "]";
        }
        return temp.str();
    }
};
```

Q 3(b) [6 Marks]

Explain the OOP concepts of upcasting and downcasting using a simple code example of each. In each example, indicate the static and dynamic types at all times.

Describe which C++ style explicit casts relate to upcasting/downcasting, and why it is important they are used?

Replace this text with your solution.

Upcasting involves casting a derived class object pointer/reference to a base class object pointer/reference. Downcasting involves casting a base class object pointer/reference to a derived class object pointer/reference. [2 marks]

```
int main() {

   Cat myCat;

   Animal *myAnimal = myCat;  // Upcasting

} [1 mark]

int main() {

   Animal myAnimal;

   Cat* myCat = dynamic_cast<Cat*>(myAnimal);  // Downcasting

} [1 mark]
```

static_cast: Used for safe and well-defined conversions between related types (upcasting).

dynamic_cast: Used for runtime type checking and downcasting in situations where the dynamic type is unknown.

[2 marks]

Q 3(c) **[6 Marks]**

Explain the following concepts related to Smart Pointers with short text answers:

- Explain the difference between a shared pointer and a weak pointer, and describe the problem a weak pointer solves. (2 marks)

Replace this text with your solution.

Shared pointers (std::shared_ptr) manage ownership of dynamically allocated objects. Weak pointers (std::weak_ptr) do not participate in the ownership of the object. They are created from a shared pointer.

The primary problem solved by weak pointers is the prevention of cyclic dependencies and the associated memory leaks.

- Why is a reference count is available on a shared pointer? (2 marks)

Replace this text with your solution.

To manage the shared ownership of a dynamically allocated object. Multiple shared pointers can point to the same dynamically allocated object. The reference count keeps track of how many shared pointers are currently owning (referencing) the object. The reference count allows the system to determine when the object is no longer in use by any part of the code.

- How are custom deleters are different to destructors? (2 marks)

Replace this text with your solution.

A destructor is a special member function of a class that is automatically called when an object of that class goes out of scope or is explicitly deleted. A custom deleter is a function or callable object that is specified when creating a smart pointer to define the cleanup action for the associated resource. They are particularly useful when dealing with resources that are not automatically managed, such as dynamically allocated arrays, file handles, database connections or external libraries.

[End of Question 3]

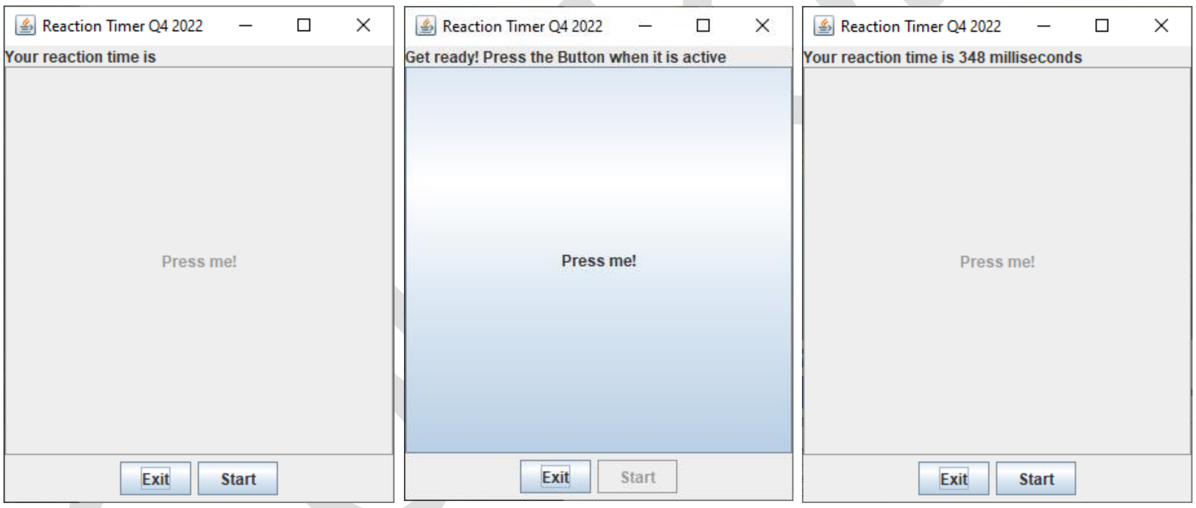
QUESTION 4 ***[TOTAL MARKS: 25]***

Q4(a) **[20 Marks]**

Write the Java Swing reaction timer application exactly as illustrated in Figure 4. The application should measure the reaction time of a user in pressing a “Press me!” button once it is enabled after a short random delay time. The reaction time should be displayed in milliseconds. The following features should be present:

- The user can press “Exit” or X on the window to quit the application at any stage. On exit the program should return the reaction time in milliseconds to the operating system, or 0 if it has not been determined.
- Swing must be used.
- On pressing “Start” there should be a short random delay of approximately 1 to 10 seconds to allow the user to ready themselves. See Figure 4(a).
- After this random delay the “Press me!” button should be enabled for the user to press. See Figure 4(b).
- The reaction time should then be displayed as in Figure 4(c).

Hint: See `java.lang.System.currentTimeMillis()` for assistance.



Your reaction time is

Get ready! Press the Button when it is active

Your reaction time is 348 milliseconds

(a) (b) (c)

Figure 4 (a) application on start-up (b) application after the Start button has been pressed and a random delay has elapsed; and, (c) the reaction time being calculated and displayed after the “Press me!” button was pressed by the user.

Replace this text with your solution. Describe the architecture, but do not attempt to include a diagram.

The example solution is below. The final solution need not be as refined and please keep in mind when reviewing this code that Eclipse is available to students during the exam and it fills in code automatically -- e.g., interface implementation stubs etc. provided that students have practiced these features.

```java
package EE402;
```

```java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class ReactionTimer extends JFrame implements ActionListener, WindowListener{

        private static final long serialVersionUID = 1L;
        private long reactionTime;
        private JButton start, exit, react;
        private JLabel message;
        private TimerThread timer;

        public ReactionTimer() {
                super("Reaction Timer Q4 2022");
                this.setLayout(new BorderLayout());

                message = new JLabel("Your reaction time is");
                this.add(message, BorderLayout.NORTH);
                react = new JButton("Press me!");
                react.addActionListener(this);
                react.setEnabled(false);
                react.setPreferredSize(new Dimension(300,300));
                this.add(react, BorderLayout.CENTER);

                Panel bottom = new Panel(new FlowLayout());
                exit = new JButton("Exit");
                bottom.add(exit);
                exit.addActionListener(this);
                start = new JButton("Start");
                bottom.add(start);
                start.addActionListener(this);
                this.add(bottom,BorderLayout.SOUTH);

                this.pack();
                this.setVisible(true);
        }

        public void updateTime(long l) {
                this.reactionTime = l;
                this.message.setText("Your reaction time is " + l + " milliseconds");
                this.start.setEnabled(true);
        }

        public void enableReactionButton() {
                this.react.setEnabled(true);
        }

        public static void main(String[] args) {
                new ReactionTimer();
        }

        public void windowOpened(WindowEvent e) {}
        public void windowClosing(WindowEvent e) {
                System.exit((int)reactionTime);
        }
        public void windowClosed(WindowEvent e) {}
        public void windowIconified(WindowEvent e) {}
        public void windowDeiconified(WindowEvent e) {}
        public void windowActivated(WindowEvent e) {}
        public void windowDeactivated(WindowEvent e) {}

        public void actionPerformed(ActionEvent e) {
                if (e.getSource().equals(exit)) {
                        System.exit((int)reactionTime);
                }
                else if (e.getSource().equals(start)) {
                        this.message.setText("Get ready! Press the Button when it is active");
                        this.start.setEnabled(false);

                        int delay = (int)(Math.random()*5);
                        timer = new TimerThread(this, delay);
                        timer.start();
                }
                else if (e.getSource().equals(react)) {
                        timer.stopTimer();
                        this.react.setEnabled(false);
                }

        }

}


package EE402;

public class TimerThread extends Thread {

        private boolean running = true;
        private int delay;
        private ReactionTimer owner;
        private long startMillis, endMillis;


        public TimerThread(ReactionTimer owner, int delay){
                this.owner = owner;
                this.delay = delay;
        }

        public void stopTimer(){
                this.running = false;
        }

        public void run(){
                try {
                        System.out.println("Sleeping for " + delay + " seconds");
                        Thread.sleep(delay*1000);
                }
                catch (InterruptedException e) {
                    System.out.println("Failed to start with random delay");
                }
                owner.enableReactionButton();
                this.startMillis = System.currentTimeMillis();
                while(running)
                {
                        try {
                                Thread.sleep(1);
                        }
                        catch (InterruptedException e) {
                            System.out.println(e.toString());
                        }
                }
                this.endMillis = System.currentTimeMillis();
                owner.updateTime(this.endMillis - this.startMillis);
        }
}
```

Q4(b) [5 Marks]

What are Java interfaces? Why and when are they used? How might you apply user-defined interfaces to Q4(a), where any application could use your code to determine the reaction time of a user?

Replace this text with your solution. Describe the architecture, but do not attempt to include a diagram.

Java Interfaces are classes where all methods are purely virtual. They are used to define a behaviour to be shared generically between classes. Classes in Java can implement many interfaces so behaviour can be shared without the need for multiple inheritance relationships

The callback functionality of the window class would need to be framed in an interface. For example, in the code above, updateTime() and enableReaction() could be captured as an interface called Reaction and the code would be changed so that ReactionTimer implements Reaction and the constructor of the TimerThread expects a Reaction object, rather than a ReactionTimer object.

[End of Question 4]

QUESTION 5 (Optional) ***[TOTAL MARKS: 25]***

Q 5(a) **[5 Marks]**

A distributed sensor system is required that uses many distributed embedded Linux sensors, each of which reads sensor data through its Analogue to Digital Converter (ADC) interfaces. A central server must also be developed that receives the data from the distributed sensors.

Describe a suitable client/server architecture for such as system and discuss the advantages and disadvantages of using embedded Linux on a board such as the BeagleBone Black or Raspberry Pi for such an application.

Replace this text with your solution. Describe the architecture, but do not attempt to include a diagram.

This exact problem is discussed in Week 12 of lectures. To get full marks the solution must include:

- A discussion on interfacing to ADC under Linux (e.g., mention availability on BBB, not on RPi).

- A discussion on client/server architectures and an example in Java, but calling out the problems with interfacing to ADCs from a VM. Could talk about C++ to act as the client or draw on other languages.

- Advantages include: software maintenance, available network standards (e.g., IPv6), available software services via Linux, remote SSH etc. GUI support.

- Disadvantages include: cost, complexity vs. simple embedded, maintaining a robust device, complexity of hardware design.

Q 5(b) **[20 Marks]**

It is decided that the Java programming language is to be used for the application in Q5(a) with the following specification:

- A class called `Sensor` must be written that contains a sensor name (`String`), a sensor type (`String`) (e.g., temperature, humidity etc.) and a single sensor reading (`Integer`). As the board has a 10-bit ADC, each reading will be a whole number between 0 and 1023. This range must be enforced.
- A Java client/server pair must be constructed, where the client application sends a `SensorReading` object that contains a vector (`java.util.Vector`) of `Sensor` objects and the time and date (`java.util.Date)` at which the readings took place. You can assume that each `Sensor` has a unique name and only one reading is sent from each sensor in each `SensorReading` object.

- The server will receive the `SensorReading` object and will average each sensor value over the last five values for each sensor in each `SensorReading` object. It will send back a new `SensorReading` object to the client that contains the moving averages for each sensor. If there are not yet 5 readings available (e.g., at start up) then the server need only use the available readings (e.g., the very first calculation will average over only the first `Sensor` object).
- If any sensor value is 50% higher or lower than its moving average then it should trigger an alert on the server side and a message should be displayed that includes the time and date of this potentially anomalous reading.
- You have been provided with three files to help you with your solution: `Client.java`, `Server.java` and `ConnectionHandler.java` on the Loop exam page. You must use these files as the structure of your solution.

Replace this text with your solution. Only include the classes that you have altered and clearly identify changes to the code that was provided -- e.g., by bolding the code. Where possible include output capture.

Code is provided for the client/server so marks are awarded for the ability to adapt this code to a new application and there are many different possible solutions. Code should be efficient and placed in the correct locations. (5 marks approx.)

The Sensor and SensorReading classes need to be designed as per the question specification (5 marks approx.)

The averaging and aggregation of sensor readings needs to be coded with the code properly integrated with the Sensor and SensorReading classes. (5 marks approx.)

The server needs to keep a run time moving average (5 marks approx.)

Output capture will help indicate success of the sections.

[End of Question 5]

[END OF EXAM]