CARDIFF UNIVERSITY
EXAMINATION PAPER

Academic Year:	2017/2018
Examination Period:	Spring
Examination Paper Number:	CMT205
Examination Paper Title:	Object Oriented Development with Java
Duration:	2 hours

Do not turn this page over until instructed to do so by the Senior Invigilator.

Structure of Examination Paper:

There are 4 pages.
There are 4 questions in total.
There are no appendices.
The maximum mark for the examination paper is **60** and the mark obtainable for a question or part of a question is shown in brackets alongside the question.

Students to be provided with:

The following items of stationery are to be provided:
ONE answer book.

Instructions to Students:

Answer THREE questions.
Students are permitted to introduce to the examination any textbook, any printed / handwritten notes, and other similar materials. Use of annotations, highlighting and bookmarks is permitted.
The use of calculators is permitted in this examination.
Important note: if you answer more than the number of questions instructed, then answers will be marked in the order they appear only until the above instruction is met. Extra answers will be ignored. Clearly cancel any answers not intended for marking. Write clearly on the front of the answer book the numbers of the answers to be marked.

The use of translation dictionaries between English or Welsh and a foreign language bearing an appropriate departmental stamp is permitted in this examination.

Q1. (a) What are the values of the variables `res1`, `res2`, `res3` and `res4` after the following sequence of Java statements has been executed? [4]

```
int i = 31;
double d = 2.7;
double res1 = d * (5/3) + i;
double res2 = i%2 == 0 ? d : -d;
StringBuffer buffer = new StringBuffer("Test123");
buffer.replace(3, 5, "#");
String res3 = buffer.toString();
buffer.reverse();
buffer.insert(4, "+");
String res4 = buffer.toString();
```

(b) Write Java statements for the following tasks:

i. Given a `double` array `nums`, print the largest number and the position of this number in the array. The position starts from 0 and when multiple entries in the array have the largest number, the position of the first entry should be printed. [4]

ii. Given a class named `MyTask` that is a subclass of `Thread` and has a constructor that takes an integer (task ID) as input, Start 10 threads with the highest priority each running an instance of `MyTask` with task ID consecutively assigned from 1 to 10. Wait until all tasks complete. [5]

(c) Assume that you are writing a class `EventLogger` which records events in your program in a single log file. What design pattern will be useful in this scenario? Briefly describe one approach to implement this design pattern (up to three sentences). [3]

(d) Assume that a remote server implements the following interface

```
import java.rmi.*;
public interface CityMap extends Remote {
    public double getDist(String city1, String city2)
       throws RemoteException;
}
```

which retrieves the travel distance from one city to another specified by the parameters `city1` and `city2` respectively. The RMI service is registered with a URL of `rmi://rmi.cs.cf.ac.uk/citymap`

Complete the following code to print the total travel distance from **Cardiff** to **London** with a stopover at **Newport** using RMI. [4]

```
import java.rmi.*;
public class TravelDistClient {
    // main method
    public static void main( String[ ] args ) {
        // TODO: complete your code here
    }
}
```

Q2. (a) i. Briefly describe the Java Collections Framework and name TWO interfaces that are provided through this Framework. [4]

ii. Describe the classes that Java provides to allow primitive types to be stored in collections. Include a code example to show how a float primitive type can be stored in an ArrayList. [4]

iii. Code that uses generics has many benefits over non-generic code. Outline the concept of generics in Java and provide TWO advantages they offer. [4]

(b) The following code defines an interface in Java called `Pet`:

```
interface Pet
{
   public String eat();
   public String talk();
   public String toString();
}
```

Provide the code for a class called `Dog` that implements the interface `Pet`. Show how your class `Dog` could be re-designed to allow the creation of sub-classes to represent specific dog types (e.g. Boxer, Poodle). The main aim of your design should be to reduce code duplication across the sub-classes. [8]

Q3. Write a Java program `Find` that reads a text file, containing a list of different actors names (one on each line) and prints out all lines in that file, each on a separate line, that match a particular keyword provided by the user. Matching should be case insensitive. The user should provide TWO command line arguments, the FIRST, which specifies the text file name containing the list of actors names and the SECOND, which specifies the keyword to be searched. Finally, the number of lines that match the keyword should be printed, or if no matches exist, an appropriate message is printed.

Your program should include appropriate error handling. You can assume all the necessary classes from the Java standard library are imported. The following skeleton program is provided.

```
import java.io.*;
public class Find
{
    public static void main(String [] args)
    {
        if (args.length != 2)
        {
            System.err.println("Two arguments expected!");
            System.exit(1);
        }

        // TODO: Complete the program
    }
}
```

[20]

Q4. Complete the following Java program `CalcServer` which implements a TCP server listening to incoming connections at the port 6000. For simplicity only one connection needs to be dealt with at a time. Every time a new connection is made, the current value is set to 0.0 (a `double` number). While the connection keeps alive, the client sends a line at a time, containing a calculation operation in the following format:

```
operator operand
```

The operation (and associated operand) is then applied to the current value and the current value is set to the new result. The operator can be one of the following: + (addition), - (subtraction), * (multiplication) and / (division). The operand is a `double` number. For instance, if the current value is 3.0, and the received line is

```
* 2.3
```

this means 3.0×2.3 (where 3.0 is the current value) is to be calculated and the current value is updated to the result (6.9 in this case). The server sends back a line containing the updated current value every time a line is received from the client. When division by zero is attempted, instead of sending back the updated current value, the message "Error" should be returned for this and all subsequent calculation operations.

You may assume that the content received from the client is always in the correct format but communication exceptions need to be handled properly.

You can assume that relevant Java standard library classes have already been imported. The following code is provided and only the missing code needs to be completed:

```
import java.io.*;
import java.net.*;
import java.util.*;
public class CalcServer
{
    public static void main(String[] args)
    {
        // TODO: Complete the code here
    }
}
```

[20]