Ollscoil Mhá Nuad
Maynooth University

SEMESTER 2
2023-2024

CS608
Software Testing

Prof. O. Conlan, Prof. R.J. Farrell, Dr. S. Brown

Time allowed: 3 hours

Answer at least ***three*** questions

Your mark will be based on your best ***three*** answers

All questions carry equal marks

Instructions

	Yes	No	N/A
Formulae and Tables book allowed *(i.e. available on request)*		X	
Formulae and Tables book required *(i.e. distributed prior to exam commencing)*		X	
Statistics Tables and Formulae allowed *(i.e. available on request)*		X	
Statistics Tables and Formulae required *(i.e. distributed prior to exam commencing)*		X	
Dictionary allowed *(supplied by the student)*		X	
Non-programmable calculator allowed		X	
Students required to write in and return the exam question paper		X	

[25 marks]

Q1 (a) Explain, with reference to the method **maxv(x,y,z)** defined below, why *exhaustive testing* is infeasible. Consider the test design time and the test execution time in your answer. [5 marks]

Reminder: a long is 64-bits long and can contain 2^{64} different values.

The method **long maxv(long x, long y, long z)** returns the maximum value of the three parameters x, y, and z.

(b) A CPU cooling fan controller includes a method **fanSetting()** as defined below to decide what speed the fan should run at. [15 marks]

Analyse this method in preparation for Combinational Testing using Decision Tables:

- Develop a value line for **t** only
- Define boolean expressions for the causes (parameters **t** & **isOn**)
- Define boolean expressions for the effects (**return value**)
- List all the candidate combinations of causes in a table
- Clearly cross out the infeasible combinations
- Develop a decision table from the table of feasible combinations
- Confirm your decision table by interpreting each rule in turn

Level is defined as follows:

```
public enum Level {NONE, LOW, HIGH};
```

The method fanSetting() is defined as follows:

boilersetting

```
public static Level fanSetting(int temp, boolean isOn)

A cpu fan controller has 3 settings speed: off, low, and
high. This method determines the setting required based on
the temperature and the on/off switch. The CPU may not be
used below 0 degrees C.

Parameters:
  temp – the current temperature (in degrees C)
  isOn – whether the heating system is switched on

Returns:
  If the fan is switched off, always return NONE
  Otherwise, return:

    • HIGH – temp is at or above 90 degrees
    • LOW – temp is at or above 40 degrees and below 90
      degrees
    • NONE – temp is below 40 degrees
```

(c) Based on your answer to Part (b), develop combinational tests using a Decision Table (DT) for the method fanSetting(). Include three tables in your answer: the Test Coverage Items (TCI), selected data values (pick sensible, representative values for low, intermediate, and high temperatures), and Test Cases. Review your design: complete the TCI table, show that each test case is covered. [5 marks]

[25 marks]

Q2 (a) Compare black-box and white-box testing as follows: [5 marks]

i. What types of errors they are likely to find?
ii. What impact do source code changes have on existing tests?
iii. Can tests be written before the code itself?
iv. What role does the specification play in developing the tests?
v. Is it difficult to measure the coverage that the tests achieve?

(b) The method **Replacer.upload()** decides whether to upload a file or not to a music server. [15 marks]

If the file already exists, and the user selects to replace it, then the file is to be uploaded. Also, if the file does not exist, and the user does not select to replace it, then the file is to be uploaded. But if the user selects not to replace a file that already exists, or selects to replace a file that does not exist, the file is not to be uploaded.

The method has already been tested using EP Black-Box test techniques.

```
// EP test data
private static Object[][] testData1 = new Object[][] {
    // test, in.ex, in.rep, expected
    {  "T1",  true,   true,     true },
    {  "T2", false,  false,     true },
};
```

The White-Box coverage (JaCoCo) achieved by the tests is shown below. Lines 19 and 23 are yellow. Lines 22 and 25 are red. Lines 18, 20, 27 are green.

```
16.   public static boolean upload(boolean ex, boolean rep) {
17.
18.       if (ex)
19.           if (rep)
20.               return true;
21.           else
22.               return false;
23.       else
24.           if (rep)
25.               return false;
26.           else
27.               return true;
28.
29.   }
```

Develop the **additional** tests required to provide full statement coverage (SC) for this method. In your answer make sure to: clearly identify the unexecuted statements, identify and explain the conditions for the inputs required to execute each unexecuted statement, and provide both a completed Test Coverage Items table, and a Test Cases table.

QUESTION 2 CONTINUES ON THE NEXT PAGE

QUESTION 2 CONTINUED

(c) Work out the required input parameter values for `x` and `larger` that ensure execution of lines 7 and 9 in the method inRange() shown below. Clearly identify the conditions required to execute each of the lines, and then show how you determine what input parameter values are required to meet these conditions. [5 marks]

```
 1 public boolean inRange(int x, boolean larger) {
 2    int lower=0; int upper=1000;
 3    if (larger) upper = 2000;
 4    x = x - lower;
 5    upper = upper - lower;
 6    if (x>=0 && x<=upper)
 7       return true;
 8    else
 9       return false;
10 }
```

[25 marks]

Q3 (a) Compare conventional testing of the static method isZero(int x) with testing 'in class context' of the instance method isZero() in class Numerical. [5 marks]

```
class Numerical {
  private int x;
  private boolean result;
  public void setX(int value);
  public boolean getResult();
  public void isZero();                    // sets result=(this.x==0)
  public static Boolean isZero(int x); // returns true if x==0
}
```

(b) Develop EP tests 'in class context' for the method Heating.decide(). [20 marks]

A home heating system can be set to three power settings: 0 is off, 1 is low power, and 2 is full power.

If override is true, then the power setting is always set to full power. Otherwise:

- if temperature is less than 12 degrees, full power is required
- if temperature is between 12 and 24 degrees low power is required
- if temperature is above 24 degrees, no power is required

In your analysis, identify the accessor methods, develop a value line for temperature, and identify the input and output Equivalence Partitions. Also, include three tables in your answer for: (a) the Test Coverage Items, (b) selected data values for temperature, and (c) the Test Cases. Your Test Cases must show the exact sequence of method calls and expected return values. Complete the TCI table, and review your Test Cases.

```
public class Heating {

   private int power;
   private boolean override;

   // Get the power setting
   public int getPower() {
      return power;
   }

   // Set override
   public void setOverride(boolean value) {
      override = value;
   }

   // Calculate power (set the power attribute)
   public void decide(int temperature) {
      // code not required to answer the question
   }
}
```

[25 marks]

Q4 (a) You are providing random testing for method **boolean: isOne(int x)** which returns true if x==1. [6 marks]

Describe the three key problems in fully automated random testing, as listed below, and demonstrate each problem using **isOne ()** as an example:

(i) the test data problem
(ii) the test oracle problem
(iii) the test completion problem

(b) Develop tests for **Num.isInRange(int x)** using random data selection. [19 marks]

You are given the Equivalence Partition Test Coverage Items Table.

In your answer:
(i) Complete the provided Equivalence Value Criteria Table for selecting random values
(ii) Complete the provided Random EP Test Cases Table
(iii) Provide an outline structure of the automated test code
(iii) Describe how this resolves the 3 random test problems

isInRange returns true if x is in the range 0..100, otherwise false.

Equivalence Partition Test Coverage Items Table

TCI	**Inputs**	**Expected Results**
EP1	x	Integer.MIN_VALUE..-1
EP2		0..100
EP3		101..Integer.MAX_VALUE
EP4	Return Value	true
EP5		false

Equivalence Value Criteria Table (to be completed)

Parameter	**EP**	**Criteria**
x	Integer.MIN_VALUE..-1	
	0..100	
	111..Integer.MAX_VALUE	

Random EP Test Cases Table (to be completed)

ID	**TCI Covered**	**Inputs**	**Expected Results**
		x	
T1	EP1,5	rand(Integer.MIN_VALUE,-1)	
T2	EP2,4		
T3	EP3,5		