Venue _____

Student Number |__|__|__|__|__|__|__|__|

Family Name _____

First Name _____

End of Semester 2, 2017
CMPE3008-1 Software Engineering Testing

Curtin University

Department of Computing

EXAMINATION

End of Semester 2, 2017

CMPE3008-1 Software Engineering Testing

This paper is for Sri Lanka Inst Info Tech students

This is a CLOSED BOOK examination

Examination paper IS NOT to be released to student

Examination Duration 2 hours

Reading Time 10 minutes

Notes in the margins of exam paper may be written by Students during reading time

Total Marks 100

Supplied by the University

None

Supplied by the Student

Materials

None

Calculator

No calculators are permitted in this exam

Instructions to Students

Students to write their answers on the examination paper in the space provided below each question.

For Examiner Use Only

Q	Mark
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	
18	

Total _____

Examination Cover Sheet

Question One (Total 30 marks)

(I) Answer the following questions based the code fragment given below.
(Total 8 marks for (I))

```
public class firsthwk
{
public static int nameCheck (char[] names)
{
// Effects: Checks a list of names and counts how many blanks
   int blankCount = 0;
   for (int i = names.length; i > 1; i--)
   {
      if (names[i-1] == ' ')
      {
         blankCount++;
      }
   }
   return blankCount;
}
public static void main (String args[])
{
   // Test driver -- method calls to nameCheck()
}
}
```

(a) This program contains a fault. What is it? Does executing the program necessarily result in either incorrect output or in failure? ***(2 marks)***

(b) Find a test case that results in failure. ***(2 marks)***

(c) Identify another test case that does not result in failure. ***(2 marks)***

(d) Find tests to execute every statement in the method nameCheck( ). Find the minimal set, that is, turn in a set of tests such that if any one test was removed, the remaining tests would no longer execute every statement. ***(2 marks)***

(II) Answer the following questions based the code fragment given below. **(Total 22 marks for (II))**

```
/*Expected behavior of the following method is to print the indexes of the
occurrences of the given number (x), in the given two dimensional array
(arr) ) */
public void Test (int[][] arr, int x){
    boolean found = false;
    for (int i = arr.length-1; i>0; i--){
      for (int j = arr[i].length-1; j>0; j--){
            if (arr[i][j] == x){
                  found = true;
                System.out.println(i + " " + j);
            }
      }
  }
  if (!found)
      System.out.println("END");
 }
```

(a) Identify the **fault(s)** in the given code fragment. ***(2 marks)***

Use the given following test case format to answer parts b and c.

(b) Identify a test case that executes the fault, but does **not** result in a **failure**. ***(4 marks)***

TC #	Input (arr),x	Expected Output	Actual Output

(c) Identify a test case that results in a **failure**. ***(4 marks)***

TC #	Input (arr),x	Expected Output	Actual Output

(d) Fix the fault(s) and draw the control flow graph for the corrected code. **Hint: Label the nodes as given in the code.** ***(6 marks)***

Control Flow Graph:

Identify the **Test Requirement** set for:

i. Edge Pair Coverage ***(2 marks)***

ii. Prime Path Coverage (any 8 paths) ***(4 marks)***

Question Two (Total 24 marks)

(I) State whether True OR False. **(Write in the space provided - 10 X 1 mark each = Total 10 marks for (I))**

(a) An unreachable program fault can be detected using testing.
Answer: _____

(b) Formal coverage criteria are used to decide which test inputs to use during testing process.
Answer: _____

(c) A path that starts at an initial node and ends at a final node is known as a Test Path.
Answer: _____

(d) Scaffolding is extra software components that are created to support integration and testing.
Answer: _____

(e) A sidetrip is always a simple path.
Answer: _____

(f) Test criteria are also called metrics.
Answer: _____

(g) Integration testing is the testing of incompatibilities and interfaces between otherwise correctly working components.
Answer: _____

(h) There is no need to automate regression test suits.
Answer: _____

(i) It's possible to satisfy General Active Clause Coverage without satisfying predicate coverage.
Answer: _____

(j) If test path p tours subpath q with sidetrips, then p also tours q with detours.
Answer: _____

(II) Consider the graph: **(Total 14 marks for (II))**

```
N  = { 1, 2, 3, 4, 5, 6}

N0 = { 1 }

Nf = { 6 }

E =  { (1, 2), (2, 3), (2, 6), (3, 4), (3, 5), (4, 5), (5, 2) }

def(x) = {1, 5}

use(x) = {5, 6} // Assume the use of x in 5 precedes the def
```

Also consider test paths t_1, t_2 and t_4 given below:

t_1 = [1, 2, 6]

t_2 = [1, 2, 3, 4, 5, 2, 3, 5, 2, 6]

t_3 = [1, 2, 3, 5, 2, 3, 4, 5, 2, 6]

(a) Draw the appropriate graph. ***(3 marks)***

(b) List all of the du-paths with respect to x. ***(2 marks)***

(c) For each test path, determine which du-paths that test path tours. For this part of the exercise, you should consider both direct touring and sidetrips. ***(3 marks)***

	Direct touring	**With/Sidetrip**
t_1		
t_2		
t_3		

(d) List a minimal test set that satisfies **all defs** coverage with respect to x. Use the given test paths. ***(2 marks)***

(e) List a minimal test set that satisfies **all uses** coverage with respect to x. Use the given test paths. ***(2 marks)***

(f) List a minimal test set that satisfies **all du-paths** coverage with respect to x. (Direct tours only.) Use the given test paths. ***(2 marks)***

Question Three (Total 18 marks)

(I) Consider the given code and answer the following questions. **(Total 10 marks for (I))**

```
public void test(int a, int b){
if ((a<10) && (b>20))
      System.out.println(a,b);
else
      System.out.println(b,a);
}
```

Test case t_1: (a=5,b=25)

Test case t_2: (a=5,b=15)

Test case t_3: (a=15,b=25)

Test case t_4: (a=15,b=15)

HINT: Use the above given test cases to answer following questions.

Identify the minimal test set for, 100%

(a) Predicate Coverage ***(2 mark)***

(b) Clause Coverage ***(2 mark)***

(c) Combinatorial Coverage ***(2 mark)***

(d) General Active Clause Coverage ***(2 marks)***

(e) Restricted Active Clause Coverage ***(2 marks)***

(II) Answer the following questions for the method `search( )` below: **(Total 8 marks for (II))**

```
public static int search (List list, Object element)
// Effects: if list or element is null throw NullPointerException
//   else if element is in the list, return an index
//   of element in the list; else return -1
//   for example, search ([3,3,1], 3) = either 0 or 1
//       search ([1,7,5], 2) = -1
```

Base your answer on the following characteristic partitioning:

```
Characteristic: Location of element in list
      Block 1: element is first entry in list
      Block 2: element is last entry in list
      Block 3: element is in some position other than first or last
```

(a) "Location of element in list" fails the disjointness property. Give an example that illustrates this. ***(2 marks)***

(b) "Location of element in list" fails the completeness property. Give an example that illustrates this. ***(2 marks)***

(c) Supply two new partitions that capture the intent of "Location of element in list" but do not suffer from completeness or disjointness problems. ***(4 marks)***

Question Four (28 marks)

(I) Provide reachability conditions, infection conditions, propagation conditions, and test case values to kill mutants 1, 2, 3, 4, 5, and 6 in method `Min ( )` below:

(6 X 2 mark each = Total 12 marks for (I))

Original Method		With Embedded Mutants
int Min (int A, int B)		int Min (int A, int B)
{		{
int minVal;		int minVal;
minVal = A;		minVal = A;
if (B < A)	Δ1	minVal = **B**;
{		if (B < A)
minVal = B;	Δ2	if (B > A)
}	Δ3	if (B < **minVal**)
return (minVal);		{
} // end Min		minVal = B;
	Δ4	**Bomb()**;
	Δ5	minVal = **A**;
	Δ6	minVal = **failOnZero (B)**;
		}
		return (minVal);
		} // end Min

(II) Consider the following BNF with start symbol A:

```
A ::= B"@"C"."B
B ::= BL | L
C ::= B | B"."B
L ::= "a" | "b" | "c" | ... | "y" | "z"
```

and the following six possible test cases:

```
t1 = a@a.a
t2 = aa.bb@cc.dd
t3 = mm@pp
t4 = aaa@bb.cc.dd
t5 = bill
t6 = @x.y
```

For each of the six tests, (1) identify the test sequence as either “in” the BNF, and give a derivation, or (2) identify the test sequence as “out” of the BNF, and give a mutant derivation that results in that test. (Use only one mutation per test, and use it only one time per test).

(6 X 2 mark each = Total 12 marks for (II))

(III) The fundamental premise of mutation was stated as: “In practice, if the software contains a fault, there will usually be a set of mutants that can be killed only by a test case that also detects that fault”. **(Total 4 marks for (III))**

(a) Give a brief argument in support of the fundamental mutation premise. ***(2 marks)***

_____

_____

_____

_____

(b) Give a brief argument against the fundamental mutation premise. ***(2 marks)***

_____

_____

_____

_____

END OF EXAMINATION