Venue _____

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

Family Name _____

First Name _____

Curtin Univers

Department of Computing

EXAMINATION

End of Semester 2, 2016

CMPE4001-1 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 both margins and reverse of exam paper may be written by Students during reading time

Total Marks 100

Supplied by the University

none

Supplied by the Student

none

No calculators are permitted in this exam

Instructions to Students

Students are to write their answers in 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	

Examination Cover Sheet

Total _____

Question 1. **Worth 14 Marks**

Answer the following questions based the code fragment given below.

```
class Vehicle implements Cloneable
{
   private int x;
   public Vehicle (int y) { x = y;}
   public Object clone()
   {
      Object result = new Vehicle (this.x);
      // Location "A"
      return result;
   }
   // other methods omitted
}
class Truck extends Vehicle
{
   private int y;
   public Truck (int z) { super (z); y = z;}
   public Object clone()
   {
      Object result = super.clone();
      // Location "B"
      ((Truck) result).y = this.y;  // throws ClassCastException
      return result;
   }
   // other methods omitted
}
// Test: Truck suv = new Truck (4); Truck co = suv.clone()
//       Expected: suv.x = co.x; suv.getClass() = co.getClass()
```

(a) Explain what is wrong with the given code. Describe the fault precisely by proposing a modification to the code. ***(4 marks)***

(b) If possible, give a test case that does not execute the fault. If not, briefly explain why not. ***(2 marks)***

(c) If possible, give a test case that executes the fault, but does not result in an error state. If not, briefly explain why not. ***(4 marks)***

(d) If possible give a test case that results in an error, but not a failure. If not, briefly explain why not. Hint: Don't forget about the program counter. ***(1 marks)***

(e) In the given code, describe the first error state. Be sure to describe the complete state. ***(3 marks)***

Question 2. **Worth 38 Marks**

(I) Use the following `class vendingMachine` for the questions below:
(Total 14 marks for (I))

```
// A Java implementation of VendingMachine.
importjava.util.*;
import java.io.*;

public class vendingMachine
{
private int credit;
private LinkedList stock;

// Maximum size of vendingMachine
private static final int MAX = 10;

//************************************************
// Constructor
// vendingmachine starts empty.
//************************************************
vendingMachine()
{
   credit = 0;
stock  = new LinkedList(); // Empty stock.
}

//************************************************
// A coin is given to the vendingMachine.
// Must be a dime, quarter or dollar.
// Ignores invalid input
//************************************************
public void coin (int coin)
{
   if (coin != 10 && coin != 25 && coin != 100)
      return;
   if (credit >= 90)
      return;
   credit = credit + coin;
   return;
}

//************************************************
// User asks for a chocolate.
// Returns the change and the sets the
// parameter StringBuffer variable Choc.
// If not enough money or no chocolates,
// returns 0 and a blank string.
//************************************************
public intgetChoc (StringBuffer choc)
{
int change;

   if (credit < 90 || stock.size() <= 0)
   {
      change = 0;
choc.replace (0, choc.length(), "");
      return (change);
   }
   change = credit - 90;
   credit = 0;

choc.replace (0, choc.length(), (String) stock.removeFirst());
```

```
   return (change);
}

//************************************************
// Adds one new piece of chocolate to the machine
// If machine is full, nothing happens
//************************************************
public void addChoc (String choc)
{
   if (stock.size() >= MAX)
      return;
stock.add (choc);
   return;
}

} // End class vendingMachine
```

(a) Draw the appropriate Control Flow Graph (CFG). ***(4 marks)***

(b) Enumerate the test requirements for Node Coverage, Edge Coverage, Edge-Pair Coverage and Prime Path Coverage for the graph. If test requirements for some criteria are the same as for others, you do not need to copy them, just write "same as X". ***(10 marks)***

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

$N = \{ 1, 2, 3, 4, 5, 6\}$

$N_0 = \{ 1 \}$

$N_f = \{ 6 \}$

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

$def(1) = def(4) = \{x\}$

$use(3) = use(5) = use(6) = \{x\}$

Also consider test paths t_1 and t_2 given below:

$t_1 = [1, 2, 3, 5, 2, 6]$

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

(a) Draw the appropriate graph. ***(2 marks)***

(b) Identify the 6 du-paths with respect to x. Please list in sorted order. ***(3 marks)***

(c) Fill in the following table with respect to direct du-tours and du-tours with sidetrips. If a test path tours a du-path directly, you do not need to analyze whether it also tours the du-path with a sidetrip. ***(3 marks)***

Test Path	Du-Paths Toured Directly	Du-Paths Toured With Sidetrips
t_1		
t_2		

(d) Which du-path is not toured, either directly or indirectly? ***(2 marks)***

_____

(e) Using the given test paths, give a minimal test set that satisfy All-Defs (Direct Tours Only). ***(2 marks)***

_____

(f) Does the test set $\{t_1, t_2\}$ satisfy All-Uses (Direct Tours Only)? ***(2 marks)***

_____

_____

(III) State whether True OR False (write in the space provided).
(10 X 1 mark each = Total 10 marks for (III))

(a) Unit testing involves only those characteristics that are vital to the performance of the unit under test. _____
(b) There is a need to automate regression test suits. _____
(c) The notion of class integration and test order is associated with unit testing. _____
(d) If test set T2 achieves a higher coverage level than T3 on a set of test requirements (TR), then T2will detect more defects than T3. _____
(e) Any path can be composed by concatenating prime paths. _____
(f) Dead code makes it impossible to achieve node coverage. _____
(g) Projects that begin test activities after implementation is complete, produce reliable software. _____
(h) Integration testing is the testing of incompatibilities and interfaces between otherwise correctly working components. _____

(i) Scaffolding is extra software components that are created to support integration and testing. _____
(j) The key to mutation testing is the design of the mutation operators. _____

Question 3. **Worth 22 Marks**

(I) Answer the following questions for the program fragments below:

(Total 16 marks for (I))

```
fragment P:
   if (A || B || C)
   {
      m();
   }
   return;
```

```
fragment Q:
   if (A)
   {
      m();
      return;
   }
   if (B)
   {
      m();
      return;
   }
   if (C)
   {
      m();
   }
```

(a) Give General active clause coverage (GACC) test set for fragment P. GACC, Correlated active clause coverage (CACC), and Restrictive active clause coverage (RACC) yieldidentical test sets for this example? If not, specify test sets which are not identical. ***(4 marks)***

(b) Does the GACC test set for fragment P satisfy Edge Coverage on fragment Q? ***(2 marks)***

(c) Write down an Edge Coverage test set for fragment Q. Make your test set include asfew tests from the GACC test set as possible. ***(10 marks)***

(II) Consider the following deterministic finite state machine:

(Total 6 marks for (II))

Current State	*Condition*	*Next State*
Idle	$a \vee b$	*Active*
Idle	$\neg a \wedge \neg b$	*Idle*
Active	$a \wedge b$	*Idle*
Active	$\neg b$	*WindDown*
Active	$\neg a \wedge b$	*Active*
WindDown	a	*Idle*
WindDown	$\neg a$	*WindDown*

(a) Draw the finite state machine. ***(3 marks)***

(b) Find Correlated Active Clause Coverage (CACC) tests for each transition from the Active state. ***(3 marks)***

Question 4. **Worth 26 Marks**

(I) Use the following characteristics and blocks for the questions below:
(Total 13 marks for (I))

Characteristics	Block 1	Block 2	Block 3	Block 4
Value 1	< 0	0	> 0	
Value 2	< 0	0	> 0	
Operation	$+$	$-$	$\times$	$\div$

(a) Give tests to satisfy the *Each Choice* criterion.***(4 marks)***

(b) Give tests to satisfy the *Base Choice* criterion. Assume base choices are Value 1 = > 0,Value 2 = > 0, and Operation = +.***(6 marks)***

(c) How many tests are needed to satisfy the *All Combinations* criterion? (Do not list all the tests!) ***(3 marks)***

(II) Answer questions (a) and (b) for the following BNF grammar:

(Total 7 marks for (II))

```
phoneNumber   ::= exchangePart dash numberPart
exchangePart  ::= special zeroOrSpecial ordinary
numberPart    ::= ordinary^4
ordinary      ::= zero | special | other
zeroOrSpecial ::= zero | special
zero          ::= "0"
special       ::= "1" | "2"
other         ::= "3" | "4" | "5" | "6" | "7" | "8" | "9"
dash          ::= "-"
```

(a) Classify the following as either phoneNumbersin the grammar or not. For numbers not inthe grammar, state why not. ***(5 marks)***

- 123-4567
- 012-3456
- 109-1212
- 246-9900
- 113-1111

(b) Consider the following mutation of the grammar: ***(2 marks)***

```
exchangePart ::= special  ordinary  other
```

If possible, give a string that appears in the mutated grammar but not in the original grammar, another string that is in the original but not the mutated, and a third string that is in both.

(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 6 marks for (III))**

(a) Give a brief argument in **support** of the fundamental mutation premise.***(3 marks)***

(b) Give a brief argument **against** the fundamental mutation premise. ***(3 marks)***

END OF EXAMINATION