*Family Name*_____

*Given Name* _____

*Student ID*_____

# Curtin University
# Department of Computing

# Software Engineering Testing
# Semester 2, Sample Test Paper

## Mid-Semester Test

**Instructions to Students:**

- This mid-test counts for **X%** of the final marks in SET
- This is a closed book Test.
- Total Marks allocated is 50.
- This paper contains 10 pages including the cover page and one rough work page.
- Answer all 4 questions.

***Write all answers in the spaces provided.***

## Question One (Total 18 marks)

(I) Consider the `countNegative ( )` method below. Answer the following questions based on the given method. **(4 X 2 mark each = Total 8 marks)**

```
public int countNegative (int[] x){
// Effects: If x is null throw NullPointException
//Else return the number of negative elements in x.
int count = 0;
for (int i=0; i<x.length-1; i++)
{
      if (x[i] <0)
      {
            count ++;
      }
}
return count;
}
```

a) Identify the **fault** in the given program.
**Answer:**

_____

_____

b) Identify a **test case input** (`array x[]`) that **does not** execute the fault.
**Answer:**

_____

c) Identify a **test case input** (`array x[]`) that executes the fault, but **does not** result in a **failure**.
**Answer:**

_____

d) Identify a **test case input** (`array x[]`) that results in a **failure**.
**Answer:**

_____

(II) State whether True OR False. **(5 X 1 mark each = Total 5 marks)**

a) Complete graph coverage is possible for all graphs. **Answer: _____**

b) If test set $T_1$ achieves a higher coverage level than $T_2$ on a set of test requirements TR, then $T_1$ will detect more defects than $T_2$. **Answer: _____**

c) Any path can be composed by concatenating prime paths. **Answer: _____**

d) Dead code makes it impossible to achieve node coverage. **Answer: _____**

e) The edge-pair coverage criterion was originally defined for finite state machines and is also called transition-pair and two-trip. **Answer: _____**

(III) Consider the terms below: **(10 X 0.5 mark each = Total 5 marks)**

- Coverage Criterion
- Debugging
- Software Error
- Software Failure
- Software Fault
- Test Requirement
- Test Set
- Testing
- Software Controllability
- Software Observability
- Validation
- Verification

For each definition below, identify the appropriate term (listed above):

a) The process of finding a fault given a failure.
**Answer: _____**
b) Evaluating software for compliance with intended usage.
**Answer: _____**
c) Externally visible incorrect behavior.
**Answer: _____**
d) A rule or collection of rules that impose test requirements on test sets.
**Answer: _____**
e) An incorrect internal state.
**Answer: _____**
f) A set of test cases.
**Answer: _____**
g) Determining whether a given artifact satisfies the requirements set by some preceding artifact.
**Answer: _____**
h) A static defect in the software.
**Answer: _____**
i) How easy it is to provide a program with the needed inputs, in terms of values, operations, and behaviors.
**Answer: _____**
j) A specific element of a software artifact that a test case must satisfy or cover.
**Answer: _____**

## Question Two (Total 10 marks)

Consider the graph given below:

```
N = {1, 2, 3, 4, 5, 6}
N0 = {1}
Nf = {6}
E = {(1, 2), (2, 3), (3, 4), (3, 5), (4, 5), (5, 2), (2, 6)}
def(x) = {1,4}
use(x) = {3,5,6}

Test Paths:
t1 = [1,2,3,5,2,6]
t2 = [1,2,3,4,5,2,6]
```

a) Draw the control flow graph. ***(3 Marks)***

**Answer:**

b) List all of the du-paths with respect to x. (Note: Include all du-paths, even those that are subpaths of some other du-path). ***(3 Marks)***

**Answer:**

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)***

**Answer:**

| | **Direct touring** | **With/Sidetrip** |
|---|---|---|
| $t_1$ | | |
| $t_1$ | | |

d) List a minimal test set that satisfies *all-defs* coverage with respect to x. (Direct tours only.) Use the given test paths. ***(1 Mark)***

**Answer:**

## Question Three (Total 12 marks)

Use the method `printPrimes( )` [Shown in Figure 1 on next page] for answering questions (a) – (d) .

```
/** ********************************************************
 * Finds and prints n prime integers
 * Jeff Offutt, Spring 2003
********************************************************** */
private static void printPrimes (int n)
{
   int curPrime;           // Value currently considered for primeness
   int numPrimes;          // Number of primes found so far.
   boolean isPrime;        // Is curPrime prime?
   int [] primes = new int [MAXPRIMES]; // The list of prime numbers.

   // Initialize 2 into the list of primes.
   primes [0] = 2;
   numPrimes = 1;
   curPrime  = 2;
   while (numPrimes < n)
   {
      curPrime++;  // next number to consider ...
      isPrime = true;
      for (int i = 0; i <= numPrimes-1; i++)
       {  // for each previous prime.
         if (isDivisible (primes[i], curPrime))
         {  // Found a divisor, curPrime is not prime.
            isPrime = false;
            break; // out of loop through primes.
         }
      }
      if (isPrime)
      {  // save it!
         primes[numPrimes] = curPrime;
         numPrimes++;
      }
   }  // End while

   // Print all the primes out.
   for (int i = 0; i <= numPrimes-1; i++)
   {
      System.out.println ("Prime: " + primes[i]);
   }
}  // end printPrimes
```

**Figure 1.** Method `printPrimes( )`

a) Draw the control flow graph for the `printPrimes( )` method. ***(4 Marks)***
**Answer:**

b) Consider test cases $t_1$ = (n = 3) and $t_2$ = (n = 5). Although these tour the same prime paths in `printPrimes( )`, they do not necessarily find the same faults. Design a simple fault that $t_2$ would be more likely to discover than $t_1$ would. ***(2 Marks)***

**Answer:**

c) For `printPrimes( ),` find a test case such that the corresponding test path visits the edge that connects the beginning of the **while** statement to the **for** statement **without** going through the body of the while loop. ***(2 Marks)***

**Answer:**

d) Enumerate the test requirements for edge coverage, and prime path coverage (list any 8 TRs) for the graph for `printPrimes( ).` ***(4 Marks)***

**Answer:**

## Question Four (5 X 2 mark each= Total 10 marks)

Consider the given code and test cases to answer following questions:

```
public void test(int a, int b)
{
        if(a%b==2 || a>b)
                system.out.println("a is valid")
        else
                system.out.println("invalid")
}
```

```
Test case t1: (a=12,b=10)
Test case t2: (a=2,b=4)
Test case t3: (a=3,b=1)
Test case t4: (a=6,b=6)
```

Use the above given test cases to answer following questions. Identify the minimal test set for, 100%

a) Predicate Coverage

**Answer:**

_____

_____

b) Clause Coverage

**Answer:**

_____

_____

c) Combinatorial Coverage

**Answer:**

_____

_____

d) General Active Clause Coverage

**Answer:**

_____

_____

e) Restricted Active Clause Coverage

**Answer:**

_____

_____

**END OF TEST PAPER**

**Please restrict rough work below this line:**

**********