**Software Engineering**

**Dr. Yasser Fouad**

**Sheet Testing**

1. Use **“Equivalence Partitioning”** and **“Boundary Value Analysis”** techniques to test the system below

*Consider a component, generate\_grading, with the following specification: The component is passed an exam mark (out of 75) and a coursework (c/w) mark (out of 25), from which it generates a grade for the course in the range 'A' to 'D'. The grade is calculated from the overall mark which is calculated as the sum of the exam and c/w marks, as follows:*

*Greater than or equal to 70 - 'A'*

*Greater than or equal to 50, but less than 70 - 'B'*

*Greater than or equal to 30, but less than 50 - 'C'*

*Less than 30 - 'D'*

*Where a mark is outside its expected range then a fault message ('FM') is generated. All inputs are passed as integers.*

1. **Now consider the following program fragment:**

1 a := b \* c

2 d := e + f

3 f := a - d

4 if f > 0 then

5 f := b \* c

6 else

7 from

8 g := 1

9 until a \* g > 10 loop

10 a := a \* f

11 g := g + 1

12 end

13 end

14 b := a + b \* c

Draw the control flow graph of the program fragment and label each elementary block and DU paths

1. **Consider the following program fragment and draw the control flow graph of the program fragment.**

x = 10

**while** x > 0 **do**

y = 2 \* y

**if** y > 10 **do**

x = x -1

**else**

y = x + 2

**end**

x = x -1

**end**

1. Using the program factorial given below, describe with the aid of two examples of how Strong Mutation software testing is carried out. Ensure to include examples of test cases and test data to strengthen your argument.

int factorial(int n)

{

int result=1;

for (i=2; i<=n; i++)

{

result = result \* i;

}

return result;

}

**8.**

int factorial(int n) {

if (n==0) return (1);

else

return (n \* factorial (n-1));

}

}

Give example of test cases for black box testing and an example for white box based testing for the above program.

Q9:

Consider the program given below

void main() {

int i,j,k;

readln (i,j,k);

if( (i < j) || ( i > k) ) {

writeln("then part");

if (j < k) writeln ("j less then k");

else writeln ( " j not less then k"); }

else writeln( "else Part"); }

1. Draw the flow graph.
2. Determine DU test for i, j and k

Q10:

**For the following code fragment, describe 3 different test cases, and for each, describe the class of test cases it represents.**

*char \* triangle (int x, y, z) {*

*/\** ***requires****: The parameters are in ascending order (i.e. x <= y <= z)*

***effects****: If x, y and z are the lengths of the sides of a triangle,*

*this function classifies the triangle using one of the three*

*strings, "scalene", "isosceles" or "equilateral". If x, y, and z*

*do not form a triangle, the empty string is returned.*

*\*/*

*char \*r;*

*r="scalene";*

*if (x==y || y==z)*

*r="equilateral";*

*if (x==z)*

*r="isosceles";*

*if (x <= 0 || (x+y) <= z)*

*r="";*

*return (r); }*