MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

VII SEMESTER B.TECH. (COMPUTER SCIENCE AND ENGINEERING)

MAKE-UP EXAMINATIONS, DEC 2019

SUBJECT: SOFTWARE TESTING AND ANALYSIS [CSE 4020]

REVISED CREDIT SYSTEM
(29/12//2019)

Time: 3 Hours MAX. MARKS: 50

Instructions to Candidates:

- Answer **ALL** the questions.
- Missing data may be suitable assumed.

1A. Differentiate between Alpha, Beta and Acceptance Testing **3M**

1B. A marketing company wishes to construct a decision table to decide how to treat clients according to three characteristics: Gender, City Dweller, and age group: A (under 30), B (between 30 and 60), and C (over 60). The company has four products (W, X, Y and Z) to test market. Product W will appeal to female city dwellers. Product X will appeal to young females. Product Y will appeal to Male middle aged shoppers who do not live in cities. Product Z will appeal to all but older females. Clearly show all the steps involved for the construction of Decision Table. **4M**

1C. What are the limitations of BVA testing? Why do we go for Equivalence Class Testing? Give an example where Equivalence class testing is better than BVA. **3M**

2A. Consider a program that takes three inputs: gender (Boolean), age([18-55]), salary ([0-10000]) and output the total mortgage for one person **3M**
Mortgage = salary * factor, where factor is given in table Q2A.

Category	Male	Female
Young	(18-35 years) 75	(18-30 years) 70
Middle	(36-45 years) 55	(31-40 years) 50
Old	(46-55 years) 30	(41-50 years) 35

Table Q2A.

Design the BVA and robust test cases for the given problem.

5M

2B. Consider following code

```
1. Maxsum(int maxint, int value)
2. int result=0, value=0;
3. if(value<0)
4.  then value= - value;
5. while((i<=value) AND (result <=maxint))
6. DO i=i+1;
7.      result=result+i;
8. OD;
```

[P.T.O]

```
9.   if (result<=maxint)
10. then output(result);
11. else output("too large");
12. end
```

Draw the control flow graph using basic blocks. Write efficient test cases for statement, branch and condition testing.

2C. Explain the TWO types of decision table. **2M**

3A. What is Linear Code Sequence and Jump (LCSAJ)? Find all linear code sequence and jump (LCSAJ) for the following program in the Fig. 3A. Generate test set T containing two test cases so that T is adequate with respect to decision coverage but not with respect to LCSAJ coverage criteria. Generate an additional test case to achieve 100% LCSAJ coverage. **5M**

```
1    begin
2      int x, y, p;
3      input (x, y);
4      p= g(x);
5      if(x<0)
6          p= g(y);
7      if(p<0)
8         q= g(x);
9      else
10        q= g(x*y);
11   end
```

Fig. 3A.

3B. For the program given in Fig. 3B.1, identify the different basic blocks and draw the control flow graph. Check whether the given set of 3 test cases in Fig 3B.2 is adequate w.r.t block coverage criterion. **3M**

```
1    begin
2      int x, y;
3      int z;
4    input (x, y);z=0;
5      if( x<0 and y<0)
6        z=x*x;
7     if (y≥0) z=z+1;
8      }
9      else
10       z=x*x*x;
11   output(z);
12   end
```

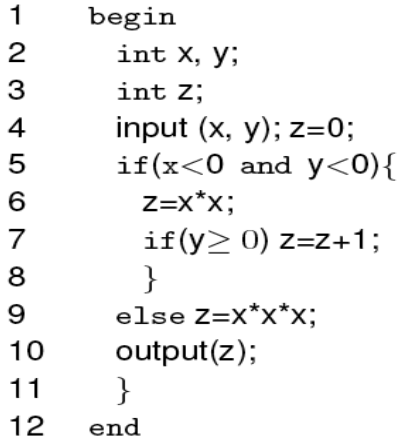
Fig. 3B.1

$$T_2 = \left\{ \begin{array}{lll} t_1: & <x=-1 & y=-1> \\ t_2: & <x=-3 & y=-1> \\ t_3: & <x=-1 & y=-3> \end{array} \right\}$$

Fig. 3B.2

3C. Explain multiple condition coverage with the help of an example. **2M**

4A. Define c-use and p-use of a variable with suitable examples. Explain the 3 steps to construct a data flow graph for the given program. Identify basic blocks and hence construct dataflow graph for the following program in Fig. 4A. specifying **def**, **c-use** and **p-use** for each node. **5M**

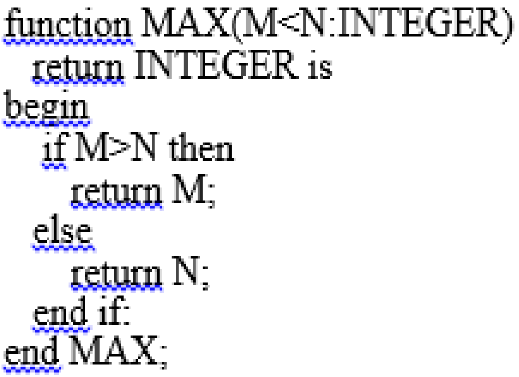


```
begin
  int x, y;
  int z;
  input (x, y); z=0;
  if(x<0 and y<0){
    z=x*x;
    if(y≥ 0) z=z+1;
    }
  else z=x*x*x;
  output(z);
  }
end
```

Fig. 4A.

4B. Define distinguished mutant and live mutant. Explain 3 conditions for distinguishing a mutant. **3M**

4C. Consider the program in Fig. 4C. that computes maximum of two integers. **2M**



```
function MAX(M<N:INTEGER)
  return INTEGER is
begin
   if M>N then
     return M;
  else
     return N;
  end if:
end MAX;
```

Fig. 4C.

Consider five mutants of the above program by replacing ">" operator in if statement by (<, ≤, ≥, =, or ≠). Generate 2 test cases to achieve 100% mutation score adequacy.

5A. With a table, explain two main differences between integration testing and unit testing. **2M**

5B. What is regression testing? Define two types of regression testing. With a neat figure, explain various steps in regression testing process. **5M**

5C. .Explain important six advantages of unit testing. **3M**