Duration: 3hrs **[Max Marks:80]**

N.B. : (1) Question No 1 is Compulsory.
(2) Attempt any three questions out of the remaining five.
(3) All questions carry equal marks.
(4) Assume suitable data, if required and state it clearly.

1 Attempt any FOUR **[20]**

a Define each software testing terminology:
i) Failure, ii) Defect, iii)Error, iv)Testware and v)Test oracle.

b What is Mutation testing? Differentiate between primary and secondary mutants.

c What criteria you will consider for selection of test tools for automation Testing.

d Explain structure of testing Group.

e Discuss Six Sigma.

2 a Consider a project with the following distribution of data and calculate its defect spoilage. **[10]**

SDLC Phase	No. of Defects	Defect Age
Requirement Specs.	34	2
HLD	25	3
LLD	17	7
Coding	10	8

b Explain Agile Testing Life Cycle and its challenges. **[10]**

3 a A program reads three numbers A, B and C, within the range [1,100] and prints the minimum number. Design test cases for this program using BVC and Robust testing methods. **[10]**

b What is the need of software measurement? Discuss the various types of software metrics. **[10]**

4 a What is the need of automation testing activities? Differentiate between static and dynamic tools? **[10]**

b Consider following C code. **[10]**

```
main()
{
    int number, index;
  1.  printf("Enter a number");
  2.  scanf("%d",&number);
  3.  index=2;
  4.  while(index<=number-1)
  5.  {
  6.      if(number%index==0)
  7.      {
  8.          printf("Not a prime number");
  9.          break;
  10.     }
  11.    index++;
  12. }
  13. if(index==number)
  14.    printf("prime number");
  15. } // end main
```

Draw DD graph, Calculate cyclomatic complexity, List out independent paths and design test cases.

5 a What are the components of a test plan? Illustrate test plan hierarchy with a neat diagram. **[10]**

b Explain McCall's Quality factors and Criteria. **[10]**

6 a Explain a bug life cycle with a neat diagram in detail. List down the states of a bug. **[10]**

B Differentiate between Effective Software Testing and Exhaustive Software Testing. **[10]**
