Reg. No.										

MANIPAL INSTITUTE OF TECHNOLOGY

MANIPAL

A Constituent Institution of Manipal University

VII SEMESTER B.TECH. (COMPUTER SCIENCE AND ENGINEERING) END SEMESTER EXAMINATIONS, NOV/DEC 2016

SUBJECT: SOFTWARE TESTING AND ANALYSIS [CSE 421]

REVISED CREDIT SYSTEM
(28/11/2016)

Time: 3 Hours MAX. MARKS: 50

Instructions to Candidates:

- Answer **ANY FIVE FULL** questions.
- Missing data may be suitable assumed.

1A. Consider a program that takes three numbers as input and prints the values of these numbers in descending order. Its input is a triple of positive integers (say x, y and z) and values are from interval [300,700]. Generate boundary value and robust test cases. **4M**

1B. Why do we undertake robustness testing? What are the additional benefits? Explain with an example. **3M**

1C. A supermarket has a loyalty scheme that is offered to all customers. Loyalty card holders enjoy the benefits of either additional discounts on all purchases or the acquisition of loyalty points, which can be converted into vouchers for the supermarket or to equivalent points in schemes run by partners. Customer without a loyalty card receive an additional discount only if they spend more than $100 on any one visit to the store, otherwise only the special offers offered to all customers apply.
Design the test cases for the above scenario using Decision Table Based Testing. **3M**

2A. For the code given below, draw the control flow graph by considering node granularity as 1 and design the test cases to achieve 100% statement and branch coverage.

```
    int main() {
1. int a[30], ele, num, i;
2  printf("\nEnter no of elements :");
3 scanf("%d", &num);
4 printf("\nEnter the values :");
5 for (i = 0; i < num; i++) {
6 scanf("%d", &a[i]); }
7printf ("\nEnter the elements to be searched :");
 8 scanf("%d", &ele);
 9 i = 0;
 10while (i < num && ele != a[i]) {
 11 i++; }
 12 if (i < num) {
 13 printf("Number found at the location = %d", i + 1); }
 14  else { printf("Number not found"); }
 15  return (0); }
```

4M

2B. For the program given in Q2A. List the Definitions, P-Use and C-Use of all the variables used and list the DU pairs. Also give test cases to cover All-Definitions. **4M**

2C. With an example code, show how condition testing is different from branch testing. **2M**

3A. Explain the various steps in Object Oriented Software testing. **5M**

3B. For the program given in Q2A. Perform independent path testing. **3M**

3C. What is the critical assumption made with BVA based on reliability theory? How test cases are selected based on this assumption? **2M**

4A Consider the requirements given for "*Make a Reservation"* functionality of Online Ticket Booking System. **5M**

Initially the user Provides information including departure and destination cities, dates, and times. A reservation agent uses that information to make a reservation. At that point, the Reservation is in the "*Made"* state. The system creates and starts a timer. If this timer expires before the reservation is paid for, the reservation is cancelled by the system.When money is paid, through initiation of the PayMoney action, the system goes into the "*Paid*" state.

Events may have parameters associated with them. For example, Pay Money may indicate Cash, Check, Debit Card, or Credit Card. When the ticket is printed, the system goes into the "*Ticketed*" State. Upon boarding the plane, the customer gives the boarding pass along with the ticket, which signals that the ticket has been Used. If the "*Reservation*" is not paid on time, the PayTimer expires and the Reservation is cancelled for non-payment. A reservation may become cancelled if the customer wishes to do so.

- ✓ This can happen prior to payment or after payment.
- ✓ If after payment, a refund needs to be generated.
- ✓ If the customer had the tickets with him, a refund cannot be given unless the printed ticket itself is returned to the agent.

a. Draw a state chart diagram for the above scenario.
b. Design test cases to achieve All State and All Transitions coverage.

4B. Explain the importance of Mutation Testing. Also give an example for Equivalent mutant. **3M**

4C. A test engineer generates 70 mutants of a program P and 150 test cases to test the program P. After the first iteration of mutation testing, the tester finds 58 dead mutants and 4 equivalent mutants. Calculate the mutation score for this test suite. Is the test case adequate for program P? Should the test engineer develop additional test cases? Justify your answer. **2M**

5A. Describe the scaffolding structure used during Top Down testing approach. Consider the following program: **5M**

```
main ( )
{
int a,b,c, sum, diff, mul;
scanf( "%d %d %d", &a, &b, &c);
sum=calsum(a,b,c); diff=caldiff(a,b,c); mul=calmul(a,b,c);
printf("%d %d %d", sum,diff,mul);
}
calsum(int x, int y, in z){
  int d;
  d=x+y+z; return(d);  }
```

a. In case the main ( ) module is not ready for the testing of calsum ( ) module. How do you test it?
b. Suppose modules caldiff() and calmul( ) are not ready when called in main(). How can we test them?

5B. Explain any TWO issues seen during Object Oriented Testing. **3M**

5C. A program takes an angle within the range [0,360] and determines in which quadrant the angle lies. Identify input domain based and output domain based equivalence classes. **2M**

6A. What are the different kinds of memory access failures? Briefly explain each of them and also explain how it can be prevented. **5M**

6B. Write a short note on the following with an example for each. **5M**

a. Y2K problem b. Sneak path and Trap door c. Def-Clear path and DU pair.