Reg. No.										

MANIPAL INSTITUTE OF TECHNOLOGY

MANIPAL

A Constituent Institution of Manipal University

VII SEMESTER B.TECH. (COMPUTERSCIENCE AND ENGINEERING)
END SEMESTER EXAMINATIONS, NOV 2017
SUBJECT: SOFTWARE TESTING AND ANALYSIS [CSE 4020]
REVISED CREDIT SYSTEM
(23/11/2017)

Time: 3 Hours MAX. MARKS:50

Instructions to Candidates:

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

1A. Explain the dynamic quality attributes with an example. **4M**

1B. Consider the specification(S) of the program under test: The program should prompt the user for a positive integer in the range 1 to 20 and then for a string of that length. The program then prompts for a character and returns the position in the string at which the character was first found or a message indicating that the character was not present in the string.

Original program(P) is given below:

```
1.  found := FALSE;
2.  i := 1;
3.  while(not(found)) and (i <= x) do begin // x is the length
4.  if a[i] = c then
5.  found := TRUE
6.  else
7.  i := i + 1
8.  end
9.  if (found)
10. print("Character %c appears at position %i");
11. else
12. print("Character is not present in the string");
13. end
```

Fig Q.1.B

Perform mutation testing for the code given in **fig Q.1.B** with following as mutants:

a. Replace found := FALSE; by found:= TRUE in line no:1
b. Replace i:=1; by x:=1; in line no:2
c. Replace i:=i+1; by i:=i+2; in line no:7

Test cases are t1:<x=1,arr='d',c='d'>;t2=<x=1,arr='d',c='a'> **4M**

1C. Find mutation score of the test suite given in **Question 1B**. Improve the test suite if required **2M**

2A. Explain factors that effect integration strategy. **4M**

2B.

```
1. read (x, y);
2. z = x + 2;
3. if (z < y)
4 w = x + 1;
else
5. y = y + 1;
6. print (x, y, w, z);
```

List definition, c use and p use for every variable. **2M**

2C. Write efficient test cases for all DU pair testing for the program given in **fig Q.2.C**.

```
1.  procedure insert(a, b, n, x);
2.  begin bool found:= false;
3.  for i:= 1 to n do
4.  if a[i] = x
5.  then found:= true; goto leave endif
    enddo;
    leave:
6.  if found
7.  else n:= n+1;
8.  a[n]:= x;
9.  b[n]:= 1
10. endif
11. end insert;
```

Fig. Q.2.C **4M**

3A. A function named compute-electricity-bill was written to compute the electricity bill by an electricity distribution company. This function takes two parameters, the number of units consumed by a customer and the corresponding customer type. The customer type is an integer value in the range 1 to 5 indicating whether the customer is domestic, industrial, commercial establishment, etc. The tariff depends not only the customer type, but also on the number of units consumed. The slabs for different charges based on the units consumed are 0 to 100 units, 100 to 200 units, 200 to 500 units, and 500 units and above. Identify the input domain based equivalence classes. Write the test cases for weak normal and robust equivalence class testing for the function compute-electricity-bill. **4M**

3B. EasyBuy.com is an online shopping website. When a customer is ready to check-out, the amount which he/she has to pay is computed as follows:

1. The transaction will be invalid if (1) the things you buy are out of stock, (2) the credit card information provided is incorrect, and (3) the balance on your card is insufficient.
2. The two main cancellation policies include:
   - cancellation within the day of the order → Full refund
   - cancellation after the day of the order→ Full refund less a $20 USD administrative fee
3. The three shipping policies include:
   - For an order over $100
   – In-state shipping fee --> Free
   – Interstate shipping fee --> 50% off
   - For an order over $200
   – Domestic shipping fee --> Free
   – International shipping fee --> 50% off
   - For an order over $500
   – Domestic shipping fee --> Free
   – International shipping fee --> Free
4. Special offers:
   - Buy-one-get-one on Labor Day, Thanksgiving, and Christmas
   - 20% off on Valentine's Day, Easter, and Memorial Day

Draw decision table for the EasyBuy.com shopping website. **4M**

3C. Based on the decision table obtained in Question 3B, write efficient test cases for testing. **2M**

4A. The New Telephone Company has the following rate structure for long distance calls:

a. Any call started at or after 6:00 p.m. (1800 hours) but before 8:00 a.m. (0800 hours) is discounted 50%.
b. Any call started at or after 8:00 a.m. (0800 hours) but before 6:00 p.m. (1800 hours)

is charged full price.

c. All calls are subject to a 4% Federal tax.
d. The regular rate for a call is $0.40 per minute.
e. Any call longer than 60 minutes receives a 15% discount on its cost (after any other discount is subtracted but before tax is added).

A computer program reads the start time for a call based on a 24-hour clock and the length of the call. The gross cost (before any discounts or tax) is printed followed by the net cost (after discounts are deducted and tax is added). The program will assume only whole number values are input, that the duration is non-negative and the start time represents a real clock time. Results are rounded to the nearest cent.

Write a complete set of boundary value analysis test cases for testing of the program which solves the problem above. **4M**

4B. We have 3 modules Sign-in, home-page, and user-database module. Sign-in module is ready and needs to be tested. Sign-in module is called from home-page module (which is not ready) and sign-in module calls user-database module(which is not ready) to authenticate user credentials. To test the Sign-in module, write stub and driver for home-page and user-database module. **4M**

4C. Write the differences between stub and mock objects. **2M**

5A.

```
1.  for i=0 to n
2.  input(value[i]);
3.  i=1;
4.  total.input = total.valid = 0;
5.  Sum=0;
6.  Do while value[i]<> -999 and total.input <= 100
7.  increment total.input by 1;
8.  IF value[i] >= minimum and value[i] <=maximum
9.  THEN increment total.valid by 1;
10. sum = sum + value[I];

11. ENDIF
12. increment i by 1;
13. Enddo
14. IF value[i] >0
15. THEN average = sum / total.valid;
16. ELSE average = -999;
17. ENDIF
```

Fig Q.5.A

Draw control flow graph for the code in f**ig Q.5.A** and write efficient separate test cases to achieve 100% coverage for statement, branch and condition testing. **3M**

5B. For f**ig Q.5.A**, write efficient test cases to achieve 100% coverage for path testing. **3M**

5C. Explain test revalidation, selection, minimization and prioritization with an example. **4M**