Indian Institute of Technology, Kharagpur

Department of Computer Science and Engineering

Class Test 1, Autumn 2013-14

Programming and Data Structures (CS 11001)

Full marks: 50 **Date**: 02-Sep-13 **Time**: 1 hour

Name	Roll No.	Section	Marks Obtained

1. Answer ALL questions.
2. Answer all questions in the space provided in this question paper itself. Use the designated spaces and the last sheet for rough work.
3. Marks for every question is shown with the question.

---

1. Consider the following program:
   (a) Write the output of the program. **[4 marks]**
   (b) Write the output of the program if the line `x=25; y=35; z=45;` is replaced by `x=45; y=35; z=40;`. **[4 marks]**

```
#include <stdio.h>
void main() {
    int x, y, z, t;
    x=25; y=35; z=45;
    printf("x=%d, y=%d, z=%d\n", x, y, z);
    if (x < y) {t=y; y=x; x=t;}
    printf("x=%d, y=%d, z=%d\n", x, y, z);
    if (x < z) {t=z; z=x; x=t;}
    printf("x=%d, y=%d, z=%d\n", x, y, z);
    if (y > z) {t=z; z=y; y=t;}
    else if (x < z) {t=z; z=x; x=t;}
    printf("x=%d, y=%d, z=%d\n", x, y, z);
}
```

Answer (a):

```
x=25, y=35, z=45
x=35, y=25, z=45
x=45, y=25, z=35
x=45, y=25, z=35
```

Answer (b):

```
x=45, y=35, z=40
x=45, y=35, z=40
x=45, y=35, z=40
x=45, y=35, z=40
```

2. Write C programs for the following problems.

   (a) A special disaster fund contribution is calculated as 10% of the total salary subject to a minimum of Rs. 1000.00 and a maximum of Rs. 10000.00. Read the total salary of a person as a positive integer and print the special disaster fund contribution as computed. **[6 marks]**

   Note: While reading the salary you may assume that input will be correctly given as a positive integer. There is no need to check and repeat for correct input.

   Note*: Multiple solutions are possible. We show a sample here. All correct solutions will earn full credit.*

Answer:

```
#include <stdio.h>

void main() {
    unsigned int salary;
    float contribution;

    printf("Input total salary\n");
    scanf("%d", &salary);

    contribution = salary * 10.0 / 100.0;

    if (contribution < 1000)
        contribution = 1000;
    else
        if (contribution > 10000)
            contribution = 10000;

    printf("Special Disaster Fund Contribution = %8.2f\n", contribution);

    return;
}
```

Roll No:

(b) Read in a positive integer n. Then read in n (> 0) positive integers. Print the largest odd number among the positive numbers you have read. If there is no odd number in the list print that such a number has not been found. **DO NOT USE ARRAYs**. **[10 marks]**

Note: While reading the numbers you may assume that inputs will be correctly given as positive integers. There is no need to check and repeat for correct input.

Note*: Multiple solutions are possible. We show a sample here. All correct solutions will earn full credit.*

Answer:

```
#include <stdio.h>

void main() {
    unsigned int n, i, x, largest = 0;

    // Read number of numbers n
    // No check is done if n is positive
    printf("Input number of numbers: ");
    scanf("%d", &n);
    printf("\n");

    // Read n numbers and compute largest
    // No check is done if the input number is not positive
    for(i = 0; i < n; ++i) {
        printf("Input %2d-th number: ", i);
        scanf("%d", &x);
        printf("\n");

        if (x % 2 == 1) { // Case of odd
            if (largest < x)
                largest = x;
        }
    }

    // Print the largest odd number
    if (largest == 0) // Check if we at all got an odd number
        printf("No positive odd number in the list\n");
    else
        printf("Largest positive odd number = %d\n", largest);

    return;
}
```

3. Write the output of the following program [**10 marks**]

```
#include <stdio.h>

void main() {
    int i;
    int k;
    int n;

    n = 6;

    printf("PART A \n");
    for(i = 1, k = 1; i < n; i++) {
        k = k + (n - i) * i;
        printf("i= %d, k= %d\n", i, k);
    }

    printf("PART B \n");
    i = 1;
    k = 1;
    while (i < n) {
        if (i < k)
            i = i + 2;
        else
            k = k + 3;
        printf("i= %d, k= %d\n", i, k);
    }
}
```

```
Answer:

PART A
i= 1, k= 6
i= 2, k= 14
i= 3, k= 23
i= 4, k= 31
i= 5, k= 36
PART B
i= 1, k= 4
i= 3, k= 4
i= 5, k= 4
i= 5, k= 7
i= 7, k= 7
```

4. Write the output of the following program [**9 marks**]

```
#include<stdio.h>

void main()
{
    int p;
    int q;
    int sum;
    int term;

    sum = 0;
    term = 0;

    for(p = 1; p < 10; p++)
    {
        printf("%d: sum = %d, term = %d\n",
            p, sum, term);

        sum = sum + term;
        term = p;

        for (q = 1; q < 4; q++) {
            term = term + p*q;
        }
    }
}
```

```
Answer:

1: sum = 0, term = 0
2: sum = 0, term = 7
3: sum = 7, term = 14
4: sum = 21, term = 21
5: sum = 42, term = 28
6: sum = 70, term = 35
7: sum = 105, term = 42
8: sum = 147, term = 49
9: sum = 196, term = 56
```

Roll No:

5. Consider the C program below (with gaps to be filled in) to perform the following task:

   Reads a positive integer k ($0 < k < 5$) and then prints all permutations (with repetitions allowed) of length k using the digits from 1 to 9. Note that since repetitions are allowed, there will be 9^k permutations for a given k. You are required to fill up the missing lines in the program. **[7 marks]**

   Note: *Interchanging* `i2`, `i3`, *and* `i4` *as loop control is okay as long as the correct permutations are printed.*

Answer:

```
void main()
{
    int k, count = 1, i1, i2, i3, i4;
    scanf("%d", &k);
    printf("k = %d\n", k);
    for (i1 = 1; i1 < 10; ++i1)
        if (k == 1) {
            printf("Permutation No %d = ", count++);
            printf("%d\n", i1);
        }
        else
            for (i2 = 1; i2 < 10; ++i2)                                    <-----
                -----------------------
                if (k == 2) {
                    printf("Permutation No %d = ", count++);
                    printf("%d, %d\n", i1, i2);                            <-----
                          --------------------
                }
                else
                    for (i3 = 1; i3 < 10; ++i3)                            <-----
                        -----------------------
                        if (k==3) {
                            printf("Permutation No %d = ", count++);
                            printf("%d, %d, %d\n", i1, i2, i3);            <-----
                                  ----------------------------
                        }
                        else
                            for (i4 = 1; i4 < 10; ++i4)                    <-----
                                ------------------------
                                if (k==4) {                                <-----
                                   ------
                                    printf("Permutation No %d = ", count++);
                                    printf("%d, %d, %d, %d\n", i1, i2, i3, i4); <-----
                                          -------------------------------------
                                }
}
```