ESC101 : Fundamental of computing

I Mid-semester Exam

29 August, 2008 **Duration : 9:30 - 10:30 AM**
max. marks = 30

Name :

Roll No. :

Section :

Instructions : 1.There are total three double sided pages including this cover page. Please contact the invigilator if there are fewer pages.
2. There are total **five** questions. Write answer for each question only in the space provided within or after the question itself.
3. You will be given an extra sheet to be used only for **rough work**.
4. Write your name and roll number carefully.

	Question 1	Question 2	Quesstion 3	Question 4	Question 5
Marks					

Total Marks :

1. **(marks=6)**
   Factorial of a positive integer n is defined $n \times (n-1) \times (n-2) \times ... \times 1$. The following program is supposed to compute the factorial of a positive integer `num`. Fill in the blanks appropriately. There are total six blanks. **(each blank carries one mark)**.

```
class factorial
{
  public static void main(String args[])
  {
    int num;
    //num is assigned some positive integer value here

    int i = num ;

    long factorial = 1;

    while(i > 1 )
    {

        factorial = factorial * i;

                i = i-1;

    }

    System.out.println(‘‘factorial of ’’+num+‘‘ is ’’+factorial);
  }
}
```

2. **(marks=6)**
Print the output of the following program.

```
class strange
{
   public static void main(String args[])
   {
     int num=10;

     int i=1;

     while(i<=12)
     {

         if(i%2==0)
         {

             num = num + 2;

             i = i+5;
         }
         else
         {

             num = num-2;

             i = i-1;

         }
         System.out.println(‘‘value of num and i are  : ’’+num+‘‘,’’+i);
      }
   }
}
```

Answer :

value of num and i are : 8 , 0

value of num and i are : 10 , 5

value of num and i are : 8 , 4

value of num and i are : 10 , 9

value of num and i are : 8 , 8

value of num and i are : 10 , 13

Each correct print statement carries 1 mark.

3. **(marks=5)**
After each of the following code fragments, print the value of variable `x`. If there is any compilation error, mention the cause of the error. **Each correct answer carries one mark.**

(a)
```
float f = 364L;

int x = 4/5 + 6/7  + (int)(f/10) +  8/9;
```

Answer :

x=36

(b)
```
byte b1 = 3; byte b2 = 4;

short x = b1*b2;
```

Answer :
syntax error since type of b1*b2 is int

(c)
```
float f = 23.2F;

float x = (int)((23.1+4)/3*5/2);
```

Answer :

x=22.0

(d)
```
long l = 124;

float x = (int)((float)l/3 + 0.8);
```

Answer :

x=42.0

(e)
```
double d = 6.6F;

long x = (long)(d*2+9/10);
```

Answer :

x=13

4. (**marks=6**)
A single step of cyclic rotation applied on an integer moves the least significant digit to the MOST significant digit. For example a single cyclic rotation applied on number 3965 would give us number 5396, applied on 3690 would give us 369. Two integers are said to be cyclic numbers if we can obtain one number from another by one or more steps of cyclic rotations of digits. The following program is supposed to determine if two 6-digit numbers are cyclic numbers. Fill in the blanks (total number of blanks=9). **each blank carries 2/3 marks.**

```
class cyclic
{
   public static void main(String args[])
   {
     int n,m;   // n and m are assigned values in range [100000, 999999].

     int digit = 0;

     int temp = m;

     int i = 1;

     int power_of_ten = 100000;

     boolean flag = false;

     while( i<=6 && flag == false)
     {

            digit = temp%10;

            temp = temp/10;

            temp  = temp + power_of_ten*digit ;

            if(temp==n)

               flag=true;

            i = i+1;
     }
     if(flag==true)
          System.out.println(n+'' and ''+m+''are cyclic numbers'');
     else
          System.out.println(n+'' and ''+m+''are NOT cyclic numbers'');
   }
}
```

5. (**marks=7**)
We want to write a program which prints all combinations of 1 or more digits of a given integer `num`. For example, for `num`=321, one possible output could be

1
2
12
3
13
23
123

Note that the relative order among digits does not matter in the combination. For example 12, 21 represent same combination and so only one of them should appear in the output.

The following program prints all combinations of one or more digits of a number `num` which has **exactly six digits**. All the digits of num are distinct. The program based on the fact that there is a one to one and onto mapping between the set of binary numbers from 1 to $2^6 - 1$ and the set of combinations of one or more digits of num. For example if `num = 876543`, then 000001 is mapped to 3, 000010 is mapped to 4, 010101 is mapped to 753. Fill in the blanks appropriately (there are total 9 blanks). Each blank appearing in the condition of **while** and **if** carries 0.5 marks. Every other blank carries 1 mark.

```
class superset
{  public static void main(String args[])
   {
     int num;     //num is assigned some six digit positive number.

     for(int i=1; i<=63; i=i+1)
     {
         int binary_temp = i;

         int decimal_temp = num;

         while(binary_temp > 0)
         {
             int binary_digit  = binary_temp % 2;

             binary_temp  = binary_temp / 2;

             int decimal_digit = decimal_temp % 10;

             decimal_temp  = decimal_temp / 10;

             if(binary_digit!=0)  System.out.print(decimal_digit);
         }
         System.out.println();
      }
   }
}
```