Function

Q. What is function?

Function is a block of code which is used for writing some logics

Q. Why use function or what is benefit of function?

If we think about function we have some benefits

1. Reusability code: Reusability means we can define function only once and we can reuse it more than one time.

Example: Suppose consider we are working on two projects

a. Parole management system: the purpose of this software is manage the salary of employee

b. Debar System for engineering college: the purpose of this application is to debar the student from exam on basis attendance criteria

So if we think about above two projects we have one common logic i.e attendance means if we give salary to employee we required to maintain attendance record as well as for debar we required attendance so we write attendance in single function and we can reuse that function in all other projects so it is called as reusability

2. Modularity of Code: Modularity means we can divide the large code in to sub code and we can integrate them called as modularity

If we want to work with any function in java we need to know the some important points.

---

1. Define function: function definition means write actual logical code in function called as function definition

Syntax:
return type function name(datatype variablename)
{ write here your logics
}
Note: here return type decide what kind of result can return from function or avoid returning result from a function
if we give void as return type then we cannot return value from function

Example:
void calAdd(int x,int y)
{ //write here your logics
}
2. Call function: if we want to reuse function we need to call it when we call function then function jumps on its definition.
Syntax:
functionname(variable or values);
Example: calAdd(100,200);
Note: in the case of call we not need to use data type for parameter just we can pass variable or values as parameter.

Example: WAP to create function name as calAdd(int,int) and pass two parameter in it and calculate its addition.

```
public class AddFunApp
{
    public static void main(String x[])
    {
        calAdd(100,200); //calling of function
    }
    public static void calAdd(int x,int y) //definition of function
    {                                  100  200
        int z=x+y; //100+200

        System.out.printf("Addition is %d\n",z);
    }
}
```

Output:
Addition is 300

Example: WAP to create function name as void table(int x):this function can accept integer as parameter and print its table

```
import java.util.*;
public class TabFunApp
{
    public static void main(String x[])
    {  Scanner xyz  = new Scanner(System.in);
       int no;
       System.out.println("Enter number");
       no = xyz.nextInt();//10

       table(no); //calling of function
    }
    public static void table(int n)//definition
    {                                  10
       for(int i=1; i<=10; i++)
       {
          int tab = i*n;

          System.out.printf("%d\n",tab);
       }
    }
}
```

Example: WAP to create function name as void power (int base,int index) and calculate power of number and display it.

```
import java.util.*;
public class PowerApp
{   public static void main(String x[ ])
    {  Scanner xyz  = new Scanner(System.in);
       int b,ind;
       System.out.println("Enter base and index");
        b    =xyz.nextInt(); //5
        ind=xyz.nextInt();   //4

        power(b    ,ind);  //calling

    }
   public static void power(int base, int index) //definition
   {   int p=1;                          5     4
      for(int i=1; i<=index; i++)
      {
            p = p  * base;
      }
   System.out.printf("Power is %d\n",p);  //625
   }
}
```

Example: WAP to create function name as void fact(int): you have to input number and calculate its factorial and display it.

```java
import java.util.*;
public class FactFunApp
{
   public static void main(String x[])
   {
      Scanner xyz  = new Scanner(System.in);
      System.out.println("Enter number");
        int num=xyz.nextInt(); //5

        fact(num); //calling
   }
   public static void fact(int no) //definition
   {
        int f=1;
       for(int i=1; i<=no;i++)
       { f=f*i;
       }
       System.out.printf("Factorial is %d\n",f);
   }
}
```

5

How to return value from a function?

If you want to return value from a function then your return type of function should not void

So we need to give return type of function which kind of value we want to return and you have to use return keyword in function definition at the time of returning value.When we return value from function definition then value can catch at function calling point and for that we have to use variable at left hand side at the time of function calling according to return type.

Example: WAP to create function name as int getCube(int x) : this function can accept integer as parameter and calculate its cube and return it.

```java
import java.util.*;
public class FunCubeApp
{
    public static void main(String x[])
    { Scanner xyz  = new Scanner(System.in);
       int no;
      System.out.println("Enter number");
      no=xyz.nextInt();  //5
          125
      int result=getCube(no);

      System.out.printf("Cube is %d\n",result);
    }
    public static int getCube(int x)
    {       125
         int r=x*x*x;

         return r;//return value at function calling point.
    }
}
```

5

125

Example: WAP to create function name as float getArea(float) this function can accept radius as input and calculate its area and return it.

Example:

```
import java.util.*;
public class AreaFunApp
{
   public static void main(String x[])
        {  Scanner xyz  = new Scanner(System.in);
                float result,r;

                System.out.println("Enter radius of circle");

                r=xyz.nextFloat();

                result=getArea(r); //calling

                System.out.printf("Area of circle is %f\n",result);
        }
        public static float getArea(float r) //definition
        {
          float a=r*r*3.14f;

          return a; //return value at function calling point
        }
}
```

Assignments

Q1. WAP to create function name as

public static boolean isPrime(int no): this function can accept number and parameter and check number is prime or not if number is prime then return true otherwise return false.

Q2. WAP to create function name as
boolean isDuck(int no) : this function is used for accept number as parameter and check number is duck or not if number is duck return true otherwise return false

Q3. WAP to create function name as
int getRev(int no): this function can accept number as parameter and reverse it and return it.

Q4. WAP to create function name as boolean isArmstrong(int no) : this function can accept number as parameter and check number is Armstrong or not if Armstrong then return true otherwise return false.

Q5. WAP to create function name as Fibonacii
void fibo(int limit): accept the limit and print the fibonacii series as per limit

Q6.WAP to create function name as Palimidrome
boolean isPalim(int no) this function can accept number as parameter and check number is palindrome or not if number is palindrome return true otherwise return false.

Function Recursion

Function Recursion means a function call itself again and again called as recursion. Means if we call same function from his own definition called as recursion.
Generalize syntax of function recursion

```
return type functionname(datatype variablename)
{       if(condition)
         {
           write here your logics
             functionname(datatype); //recursive call
         }
}
```

Example: we want to print five time good morning message using recursion.

```
public class TestRec
{
    public static void main(String x[ ])
    {
        show(5); //initial call
    }
    public static void show(int x)
    {
        if(  x!=0  )
        {
            System.out.println("Good Morning");

            show(--x); //recursive call
        }
        else{
            System.out.println("END");
        }
    }
}
```

0

0

Good Morning'
'Good Morning'
'Good Morning'
'Good Morning'
'Good Morning'
END

Example: WAP to input number from keyboard and print its table using recursion.

```
import java.util.*;
public class RecTabApp
{   static int count=1;
    public static void main(String x[])
    {
        Scanner xyz  = new Scanner(System.in);
        int no;
        System.out.println("Enter number");
        no=xyz.nextInt();
        showTable(no); //initial call
    }
    public static void showTable(int x)  //definition
    {
            while( count<=10 )
            {
               System.out.printf("%d\n",count*x);

                count=count+1;

                showTable(x); //recursive call
            }
    }
```

```
C:\Program Files\Java\jdk1.8.0_291\bin>java RecTabApp
Enter number
6
6
12
18
24
30
36
42
48
54
60
```

Output

Example: WAP to input number and calculate its factorial using recursion.

```
import java.util.*;
public class FactApp
{   static  int f=1;
    public static void main(String x[])
    {
        Scanner xyz  = new Scanner(System.in);
        int no;
        System.out.println("Enter number");
        no=xyz.nextInt();//5
        fact(no); //initial call   4
    }
    public static void fact(int x)  //definition
    {
            if( x!=0)
            {
                f = f * x; // 1 *5 = 5
                                5*4 = 20
                fact(--x); //recursive
            }
            else{
                System.out.println("Factorial is "+f);
            }
    }
}
```

4

Example: WAP to input number and reverse it using recursion.

```java
import java.util.*;
public class RevApp
{  static  int r=0;
     public static void main(String x[])
     {   Scanner xyz  = new Scanner(System.in);
         int no;
         System.out.println("Enter number");
         no=xyz.nextInt();
         rev(no);
     }
     public static void rev(int x)
     {
                if(x!=0)
                {
                   int rem=x%10;
                   x=x/10;
                   r=r*10+rem;
                   rev(x);//recursive call
                }
                else
                {
                 System.out.printf("Reverse number is %d\n",r);
                }
     }
}
```

Output

```
C:\Program Files\Java\jdk1.8.0_291\bin>javac RevApp.java

C:\Program Files\Java\jdk1.8.0_291\bin>java RevApp
Enter number
123
Reverse number is 321
```

Example: WAP to input base and index from keyboard and calculate its power using recursion

Example: WAP to input number and check number is prime or not using recursion

Example: WAP to input number and check number is duck or not using recursion

Example: WAP to input number and check number is Armstrong or not using recursion

Example: WAP to input number and calculate sum of all digits of number using recursion.