# Functions

- Functions are exactly same as predicates except that functions can return any type of result but function should (can) return only one value and that value can be any type as per our requirement.
- To implement functions oracle people introduced Function interface in 1.8version.
- Function interface present in *Java.util.function* package.
- Functional interface contains only one method i.e., apply()

```
interface function(T,R) {
        public R apply(T t);
}
```

**Assignment:** Write a function to find length of given input string.

**Ex:**

```
import Java.util.function.*;
class Test {
        public static void main(String[] args) {
                Function<String, Integer> f = s ->s.length();
                System.out.println(f.apply("Durga"));
                System.out.println(f.apply("Soft"));
        }
}
```

**Note:** Function is a functional interface and hence it can refer lambda expression.

# Differences between predicate and function

| Predicate | Function |
|---|---|
| To implement conditional checks We should go for predicate | To perform certain operation And to return some result we Should go for function. |
| Predicate can take one type Parameter which represents Input argument type.<br>Predicate<T> | Function can take 2 type Parameters. First one represent Input argument type and Second one represent return Type.<br>Function<T,R> |
| Predicate interface defines only one method called test() | Function interface defines only one Method called apply(). |
| public boolean test(T t) | public R apply(T t) |
| Predicate can return only boolean value. | Function can return any type of value |

**Note:** Predicate is a boolean valued function and(), or(), negate() are default methods present inside Predicate interface.

## Program to remove spaces present in the given String by using Function:

```java
import java.util.function.*;
class Test
{
   public static void main(String[] args)
   {
      String s="durga  software  solutions hyderabad";
      Function<String,String> f= s1->s1.replaceAll(" ","");
      System.out.println(f.apply(s));
   }
}
```

**Output:** durgasoftwaresolutionshyderabad

## Program to find Number of spaces present in the given String by using Function:

```java
import java.util.function.*;
class Test
{
   public static void main(String[] args)
   {
      String s="durga  software  solutions  hyderabad ";
      Function<String,Integer> f= s1->s1.length() - s1.replaceAll(" ","").length();
      System.out.println(f.apply(s));
   }
}
```

**Output:** 3

## Program to find Student Grade by using Function:

```java
import java.util.function.*;
import java.util.*;
class Student
{
   String name;
   int marks;
   Student(String name,int marks)
   {
      this.name=name;
      this.marks=marks;
   }
}
```

```java
class Test
{
    public static void main(String[] args)
    {
        ArrayList<Student> l= new ArrayList<Student>();
        populate(l);
        Function<Student,String> f=s->{
            int marks=s.marks;
            if(marks>=80)
            {
                return "A[Dictinction]";
            }
            else if(marks>=60)
            {
                return "B[First Class]";
            }
            else if(marks>=50)
            {
                return "C[Second Class]";
            }
            else if(marks>=35)
            {
                return "D[Third Class]";
            }
            else
            {
                return "E[Failed]";
            }
        };
        for(Student s : l)
        {
            System.out.println("Student Name:"+s.name);
            System.out.println("Student Marks:"+s.marks);
            System.out.println("Student Grade:"+f.apply(s));
            System.out.println();
        }
    }
    public static void populate(ArrayList<Student> l)
    {
        l.add(new Student("Sunny",100));
        l.add(new Student("Bunny",65));
        l.add(new Student("Chinny",55));
        l.add(new Student("Vinny",45));
        l.add(new Student("Pinny",25));
    }
}
```

**Output:**

```
D:\durgaclasses>java Test
Student Name:Sunny
Student Marks:100
Student Grade:A[Dictinction]

Student Name:Bunny
Student Marks:65
Student Grade:B[First Class]

Student Name:Chinny
Student Marks:55
Student Grade:C[Second Class]

Student Name:Vinny
Student Marks:45
Student Grade:D[Third Class]

Student Name:Pinny
Student Marks:25
Student Grade:E[Failed]
```

## Program to find Students Information including Grade by using Function whose marks are >=60:

```java
import java.util.function.*;
import java.util.*;
class Student
{
   String name;
   int marks;
   Student(String name,int marks)
   {
      this.name=name;
      this.marks=marks;
   }
}
class Test
{
   public static void main(String[] args)
   {
      ArrayList<Student> l= new ArrayList<Student>();
      populate(l);
      Function<Student,String> f=s->{
         int marks=s.marks;
         if(marks>=80)
```

```
            {
                return "A[Dictinction]";
            }
            else if(marks>=60)
            {
                return "B[First Class]";
            }
            else if(marks>=50)
            {
                return "C[Second Class]";
            }
            else if(marks>=35)
            {
                return "D[Third Class]";
            }
            else
            {
                return "E[Failed]";
            }
        };
        Predicate<Student> p=s->s.marks>=60;

        for(Student s : l)
        {
            if(p.test(s))
            {
                System.out.println("Student Name:"+s.name);
                System.out.println("Student Marks:"+s.marks);
                System.out.println("Student Grade:"+f.apply(s));
                System.out.println();
            }
        }
    }
    public static void populate(ArrayList<Student> l)
    {
        l.add(new Student("Sunny",100));
        l.add(new Student("Bunny",65));
        l.add(new Student("Chinny",55));
        l.add(new Student("Vinny",45));
        l.add(new Student("Pinny",25));
    }
}
```

**Output:**

```
D:\durgaclasses>java Test

Student Name:Sunny
```

```
Student Marks:100
Student Grade:A[Dictinction]

Student Name:Bunny
Student Marks:65
Student Grade:B[First Class]
```

## Progarm to find Total Monthly Salary of All Employees by using Function:

```
import java.util.*;
import java.util.function.*;
class Employee
{
   String name;
   double salary;
   Employee(String name,double salary)
   {
      this.name=name;
      this.salary=salary;
   }
   public String toString()
   {
      return name+":"+salary;
   }
}
class Test
{
   public static void main(String[] args)
   {
      ArrayList<Employee> l= new ArrayList<Employee>();
      populate(l);
      System.out.println(l);
      Function<ArrayList<Employee>,Double> f= l1 ->{
         double total=0;
         for(Employee e: l1)
         {
            total=total+e.salary;
         }
         return total;
      };
      System.out.println("The total salary of this month:"+f.apply(l));
   }

   public static void populate(ArrayList<Employee> l)
   {
      l.add(new Employee("Sunny",1000));
```

```
        l.add(new Employee("Bunny",2000));
        l.add(new Employee("Chinny",3000));
        l.add(new Employee("Pinny",4000));
        l.add(new Employee("Vinny",5000));
    }
}
```

**Output:**

```
D:\durgaclasses>java Test
[Sunny:1000.0, Bunny:2000.0, Chinny:3000.0, Pinny:4000.0, Vinny:5000.0]
The total salary of this month:15000.0
```

## Progarm to perform Salary Increment for Employees by using Predicate & Function:

```
import java.util.*;
import java.util.function.*;
class Employee
{
   String name;
   double salary;
   Employee(String name,double salary)
   {
      this.name=name;
      this.salary=salary;
   }
   public String toString()
   {
      return name+":"+salary;
   }
}
class Test
{
   public static void main(String[] args)
   {
      ArrayList<Employee> l= new ArrayList<Employee>();
      populate(l);

      System.out.println("Before Increment:");
      System.out.println(l);

      Predicate<Employee> p=e->e.salary<3500;
      Function<Employee,Employee> f=e->{
          e.salary=e.salary+477;
          return e;
      };

```

```
        System.out.println("After Increment:");
        ArrayList<Employee> l2= new ArrayList<Employee>();
        for(Employee e: l)
        {
          if(p.test(e))
          {
            l2.add(f.apply(e));
          }
        }
        System.out.println(l);
        System.out.println("Employees with  incremented salary:");
        System.out.println(l2);
    }
    public static void populate(ArrayList<Employee> l)
    {
        l.add(new Employee("Sunny",1000));
        l.add(new Employee("Bunny",2000));
        l.add(new Employee("Chinny",3000));
        l.add(new Employee("Pinny",4000));
        l.add(new Employee("Vinny",5000));
        l.add(new Employee("Durga",10000));
    }
}
```

**Output:**

```
Before Increment:
[Sunny:1000.0, Bunny:2000.0, Chinny:3000.0, Pinny:4000.0, Vinny:5000.0, Durga:10000.0]
After Increment:
[Sunny:1477.0, Bunny:2477.0, Chinny:3477.0, Pinny:4000.0, Vinny:5000.0, Durga:10000.0]
Employees with  incremented salary:
[Sunny:1477.0, Bunny:2477.0, Chinny:3477.0]
```

# Function Chaining:

We can combine multiple functions together to form more complex functions. For this Function interface defines the following 2 default methods:

f1.andThen(f2): First f1 will be applied and then for the result f2 will be applied.
f1.compose(f2)===>First f2 will be applied and then for the result f1 will be applied.

## Demo Program-1 for Function Chaining:

```
import java.util.function.*;
class Test
{
   public static void main(String[] args)
   {

      Function<String,String> f1=s->s.toUpperCase();
      Function<String,String> f2= s->s.substring(0,9);

      System.out.println("The Result of f1:"+f1.apply("AishwaryaAbhi"));
      System.out.println("The Result of f2:"+f2.apply("AishwaryaAbhi"));
      System.out.println("The Result of f1.andThen(f2):"+f1.andThen(f2).apply("Aishwarya Abhi"));
      System.out.println("The Result of f1.compose(f2):"+f1.compose(f2).apply("Aishwarya Abhi"));
   }
}
```

**Output:**

```
The Result of f1:AISHWARYAABHI
The Result of f2:Aishwarya
The Result of f1.andThen(f2):AISHWARYA
The Result of f1.compose(f2):AISHWARYA
```

## Demo program to Demonstrate the difference between andThen() and compose():

```
import java.util.function.*;
class Test
{
   public static void main(String[] args)
   {
      Function<Integer,Integer> f1= i->i+i;
      Function<Integer,Integer> f2= i->i*i*i;
      System.out.println(f1.andThen(f2).apply(2));
```

```
        System.out.println(f1.compose(f2).apply(2));
    }
}
```

**Output:**

```
64
16
```

## Demo Program for Function Chaining:

```
import java.util.function.*;
import java.util.*;
class Test
{
    public static void main(String[] args)
    {

        Function<String,String> f1=s->s.toLowerCase();
        Function<String,String> f2= s->s.substring(0,5);

        Scanner sc = new Scanner(System.in);
        System.out.println("Enter User Name:");
        String username=sc.next();

        System.out.println("Enter Password:");
        String pwd=sc.next();

        if(f1.andThen(f2).apply(username).equals("durga") && pwd.equals("java"))
        {
            System.out.println("Valid User");
        }
        else
        {
            System.out.println("Invalid User");
        }
    }
}
```

**Output:**

```
D:\durgaclasses>java Test
Enter User Name:
durga
Enter Password:
java
Valid User

D:\durgaclasses>java Test
```

```
Enter User Name:
durgasoftwaresolutions
Enter Password:
java
Valid User

D:\durgaclasses>java Test
Enter User Name:
DURGATECHNOLOGIES
Enter Password:
java
Valid User

D:\durgaclasses>java Test
Enter User Name:
javajava
Enter Password:
java
Invalid User

D:\durgaclasses>java Test
Enter User Name:
durga
Enter Password:
Java
Invalid User
```

## Function interface Static Method : identity()

Function interface contains a static method.
static <T> Function<T,T> identity()
Returns a function that always returns its input argument.

**Eg:**

```java
import java.util.function.*;
class Test
{
   public static void main(String[] args)
   {
      Function<String,String> f1= Function.identity();
      String s2= f1.apply("durga");
      System.out.println(s2);
   }
}
```

**Output:** durga