**Java-8 – Two argument bi functional interfaces introduction**

* **BI Functional Interfaces:**

**BiPredicate:**

Normal Predicate can take only one argument, if you want to take two argument and perform a conditional check, we have to use BiPredicate.

**BiFunction:**

Normal function can take only one input argument, if you want to take two input argument and do some operation, we have to use BiFunction.

**BiConsumer:**

Same as others.

* **BiPredicate:**

interface BiPredicate<T, U> {

public boolean test(T t, U u);

}

Example:

import java.util.function.BiPredicate;

class Test{

public static void main(String[] args){

BiPredicate<Integer, Integer> = (a, b) -> (a+b) % 2 ==0;

System.out.println(p.test(10, 20));

System.out.println(p.test(15, 20));

}

}

Output: true, false

Note: Even this BiPredicate interface contains all the default methods of Predicate (and(), or(), negate());

* **BiFunction:**

interface BiFunction<T, U, R> {

public R apply(T t, U u);

}

T & U – Input type.

R – Output / return type.

Example:

import java.util.function.BiFunction;

class BiFunctionDemo{

public static void main(String[] args){

BiFunction<Integer, Integer, Integer> f = (a,b) -> a \* b;

System.out.println(f.apply(10, 20));

System.out.println(f.apply(100, 200));

}

}

* **Creation of student object by taking name and roll number using BiFunction:**

class Student{

String name;

int rollNo;

Studen(String name, int rollNo){

this.name = name;

this.rollNo = rollNo;

}

}

import java.util.List;

import java.util.ArrayList;

import java.util.function.BiFunction;

class BiFunctionDemo{

public static void main(String[] args){

List<Student> l = new ArrayList<>();

BiFunction<String, String, Student> f = (name, rollNo) -> new Student(name, rollNo);

l.add(f.apply(“Saravana”, 100));

l.add(f.apply(“Rithika”, 200));

l.add(f.apply(“Gokul”, 300));

l.add(f.apply(“Devi”, 400));

l.add(f.apply(“Deva”, 500));

}

for(Student s: l){

System.out.println(“Student Name:”+s.name);

System.out.println(“Student Rollno:”+s.rollNo);

System.out.println();

}

}

* **Calculate monthly salary with employee and timesheet objects as input by BiFunction:**

class Employee {

int eNo;

String name;

double dailyWage;

Employee(int eNo, String name, double dailyWage){

this.eNo = eNo;

this.name = name;

this.dailyWage = dailyWage;

}

}

class TimeSheet{

int eNo;

int days;

TimeSheet(int eNo, int days){

this.eNo = eNo;

this.days = days;

}

}

class Test{

public static void main(String[] args){

BiFunction<Employee, TimeSheet, Double> f = (e, t) -> e.dailyWage \* t.days;

Employee e = new Employee(101, “Saravana”, 1500);

TimeSheet t = new TimeSheet(101, 25);

System.out.println(“Employee Monthly Salary:”+f.apply(e,t));

}

}

* **This document contains contents of 58 to 62**