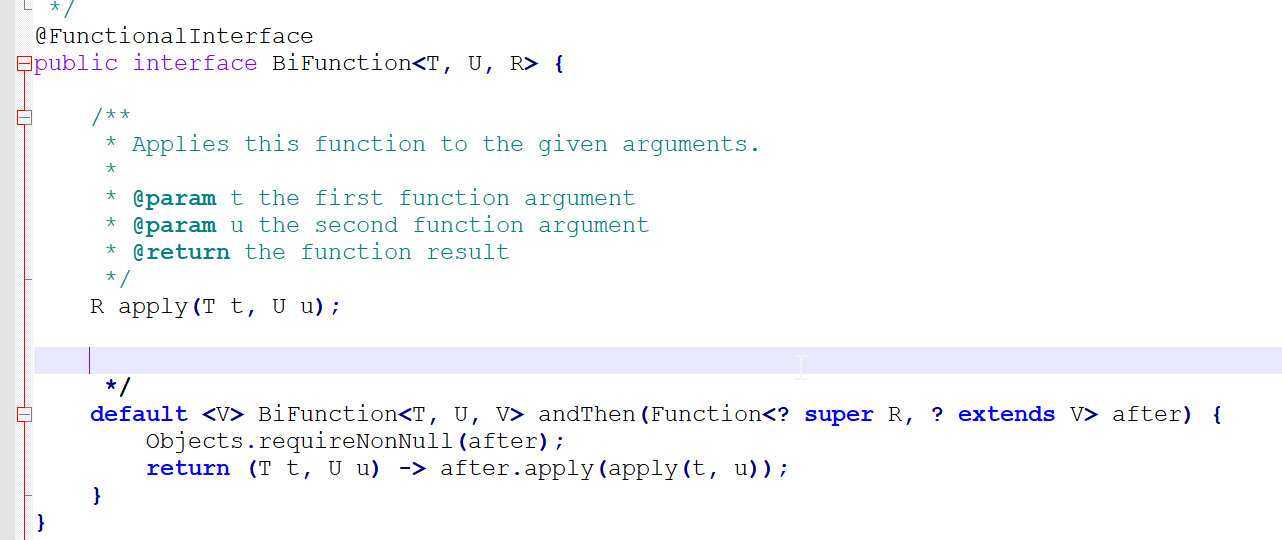
Lecture 58 Introduction:

|  |  |  |
| --- | --- | --- |
| SI | One Argument Interfaces | Two Argument Interfaces |
|  |  |  |
| 1 | Predicate | BI Predicate |
| 2 | Function | BI Function |
| 3 | Consumer | BI Consuer |
| 4 | Supplier | Not Available:  Supplier does not have any argument it just returns a value. So Bi Predicate does not exist. |

Lecture 59: BI Predicate:



**package** com.durgaSoft.section7.lecture60;

**import** java.util.function.BiFunction;

**public** **class** Example1 {

**public** **static** **void** main(String[] args) {

// Program to calculate product of two numbers

BiFunction<Integer, Integer, Integer> productOfTwoNumbers = (num1,num2) -> num1\* num2;

System.***out***.println(productOfTwoNumbers.apply(2, 3));

System.***out***.println(productOfTwoNumbers.apply(4, 4));

}

}

Lecture 61:

Program to create a student object by taking name and rollnumber as input

**package** com.durgaSoft.section7.lecture61;

**import** java.util.function.BiFunction;

**public** **class** Example1 {

**public** **static** **void** main(String[] args) {

BiFunction<String, Integer, Student> student = (name, rollnumber) -> **new** Student(name, rollnumber);

System.***out***.println(**new** Student("Student1", 1));

System.***out***.println(student.apply("Student2", 2));

}

}

**class** Student{

String name;

Integer rollNumber;

**public** Student(String name, Integer rollNumber) {

**super**();

**this**.name = name;

**this**.rollNumber = rollNumber;

}

@Override

**public** String toString() {

**return** "Name=" + name + ", RollNumber=" + rollNumber;

}

}

Lecture 62:

Calculate the monthly Salary of Employee and TimeSheet by using Bi-Function.

**import** java.util.function.BiFunction;

**public** **class** Example1 {

**public** **static** **void** main(String[] args) {

BiFunction<Employee, TimeSheet, Double> calculateWage = (employee,timeSheet) -> employee.dailyWage \* timeSheet.days;

System.***out***.println(calculateWage.apply(**new** Employee(01,"Emp1",800), **new** TimeSheet(01, 30)));

}

}

**class** TimeSheet{

**int** employeeNumber;

**int** days;

@Override

**public** String toString() {

**return** "EmployeeNumber=" + employeeNumber + ", Days=" + days;

}

**public** TimeSheet(**int** employeeNumber, **int** days) {

**super**();

**this**.employeeNumber = employeeNumber;

**this**.days = days;

}

}

**class** Employee{

**int** employeeNumber;

String name;

**double** dailyWage;

**public** Employee(**int** employeeNumber, String name, **double** dailyWage) {

**super**();

**this**.employeeNumber = employeeNumber;

**this**.name = name;

**this**.dailyWage = dailyWage;

}

@Override

**public** String toString() {

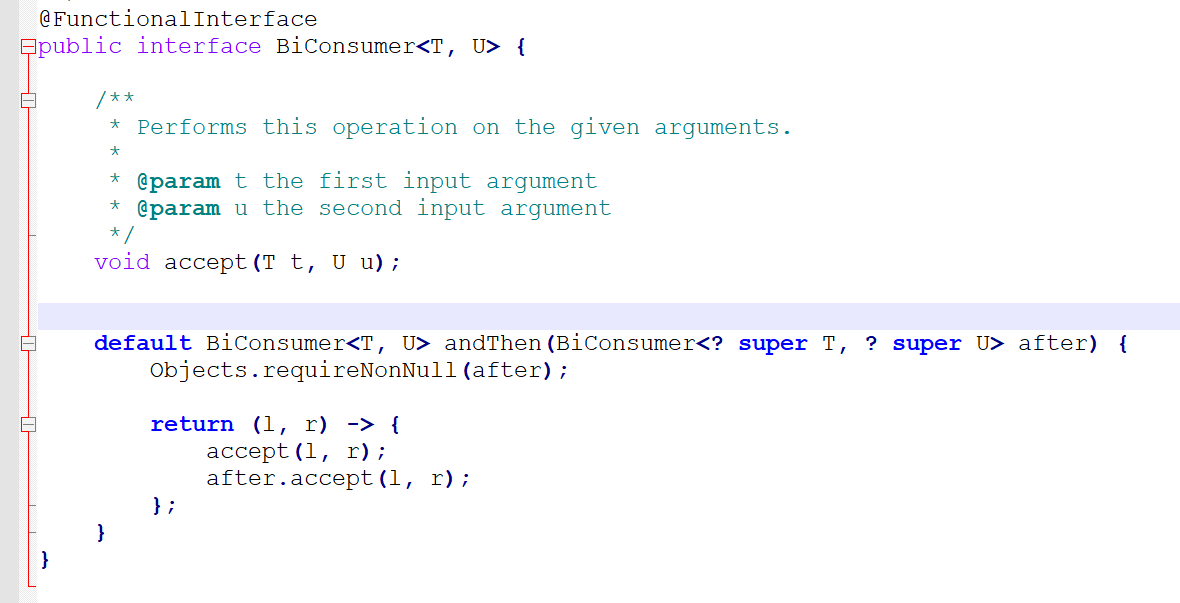
**return** "EmployeeNumber=" + employeeNumber + ", Name=" + name + ", DailyWage=" + dailyWage;

}

}

Lecture 63:

BI Consumer:



Lecture 64: To Increment Employee Salary by using BiFunction and Bi Consumer.

* Employee Creationed using BiFuntion and Increment is done using Consumer.

**public** **class** Example1 {

**public** **static** **void** main(String[] args) {

BiFunction<String, Double, Employee> employeeObject = (empName, salary) -> **new** Employee(empName, salary);

BiConsumer<Employee, Double> employeeIncrement = (employee, increment) -> employee.salary=employee.salary+increment;

ArrayList<Employee> employeeList=**new** ArrayList<Employee>();

employeeList.add(employeeObject.apply("Durga", 1000.0));

employeeList.add(employeeObject.apply("Sunny", 2000.0));

employeeList.add(employeeObject.apply("Bunny", 3000.0));

employeeList.add(employeeObject.apply("Chinny", 4000.0));

**for** (Employee employee : employeeList) {

System.***out***.println(employee);

}

**for** (Employee employee : employeeList) {

employeeIncrement.accept(employee, 100.0);

}

**for** (Employee employee : employeeList) {

System.***out***.println(employee);

}

}

}

**class** Employee{

String name;

**double** salary;

@Override

**public** String toString() {

**return** "Name=" + name + ", Salary=" + salary;

}

**public** Employee(String name, **double** salary) {

**super**();

**this**.name = name;

**this**.salary = salary;

}

}

Lecture 65:

|  |  |  |
| --- | --- | --- |
| Interfaces | One Argument Functional Interface | Two Argument Functional Interface |
|  |  |  |
| Predicate |  |  |
|  |  |  |
| Function |  |  |
|  |  |  |
| Consumer |  |  |
|  |  |  |
| Supplier |  | Not Available |