*//Write the following a functional interface and implement it using lambda:*

*//*

*//(1) First number is greater than second number or not Parameter (int ,int ) Return boolean*

*//(2) Increment the number by 1 and return incremented value Parameter (int) Return int*

*//(3) Concatination of 2 string Parameter (String , String ) Return (String)*

*//(4) Convert a string to uppercase and return . Parameter (String) Return (String)*

**package** Java8InterfaceExercise;

@FunctionalInterface

**interface** CompareInterface{

**boolean** iCompare(Integer i1, Integer i2);

}

@FunctionalInterface

**interface** IncrementInterface{

Integer increament(Integer i1);

}

@FunctionalInterface

**interface** ConcatInterface{

String concat(String s1, String s2);

}

@FunctionalInterface

**interface** ToUpperCaseInterface{

String toUpperCase(String s1);

}

**public class** Ques1 {

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

*//(1) First number is greater than second number or not Parameter (int ,int ) Return boolean*

CompareInterface compareInterface = (i1,i2)->{

**if**(i1>i2)

**return true**;

**else**

**return false**;

};

System.***out***.println(compareInterface.iCompare(10,20));

*//(2) Increment the number by 1 and return incremented value Parameter (int) Return int*

IncrementInterface incrementInterface =(i1)->{

**return** i1+=1;

};

System.***out***.println(**"Incremented Value : "**+incrementInterface.increament( 10));

*//(3) Concatination of 2 string Parameter (String , String ) Return (String)*

ConcatInterface concatInterface = (String::concat);

System.***out***.println(**"Concatinated String : "**+ concatInterface.concat(**"Helo "**, **"World"**));

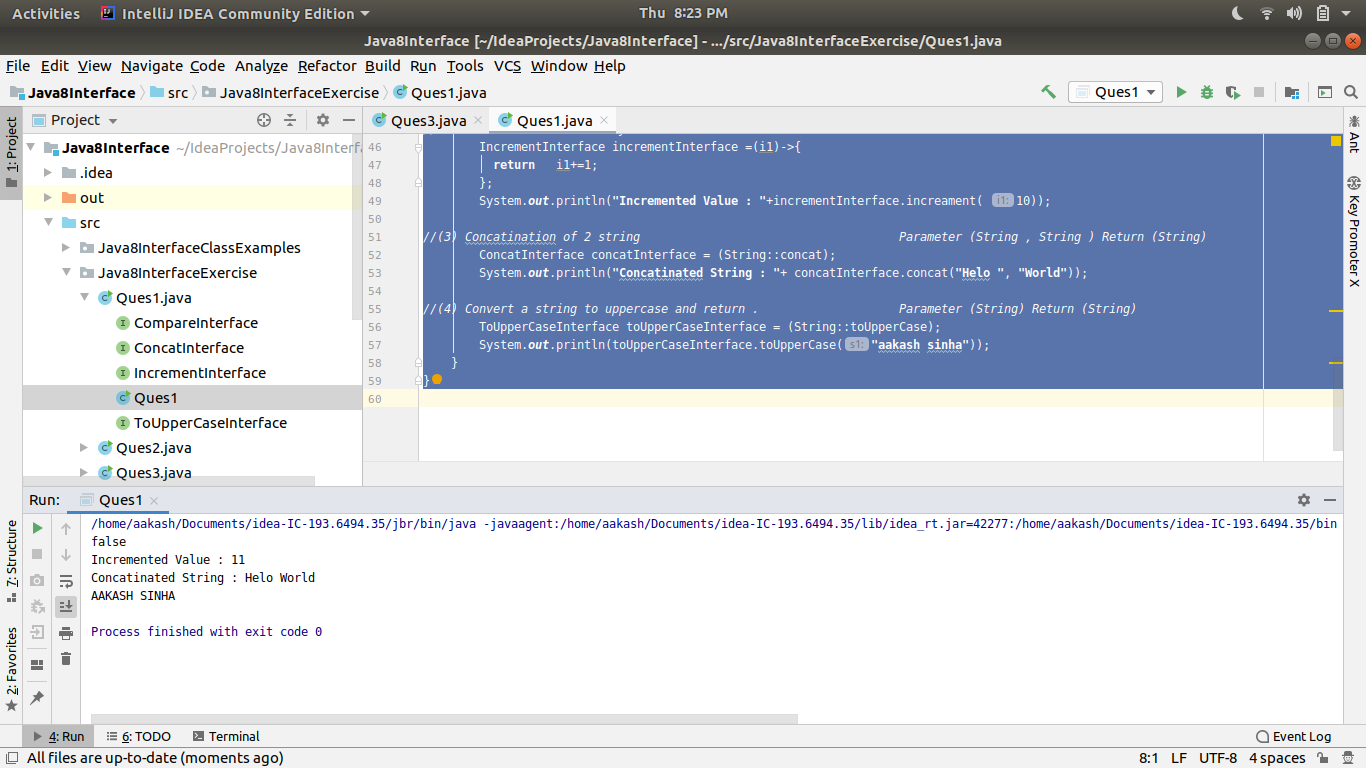
*//(4) Convert a string to uppercase and return . Parameter (String) Return (String)*

ToUpperCaseInterface toUpperCaseInterface = (String::toUpperCase);

System.***out***.println(toUpperCaseInterface.toUpperCase(**"aakash sinha"**));

}

}



*//Create a functional interface whose method takes 2 integers and return one integer.*

**package** Java8InterfaceExercise;

@FunctionalInterface

**interface** FunctionalInterfaceEXample{

Integer func(Integer i1, Integer i2);

}

**public class** Ques2 {

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

FunctionalInterfaceEXample interfaceObject = (i1, i2)->{

**if**(i1>i2)

**return** i1;

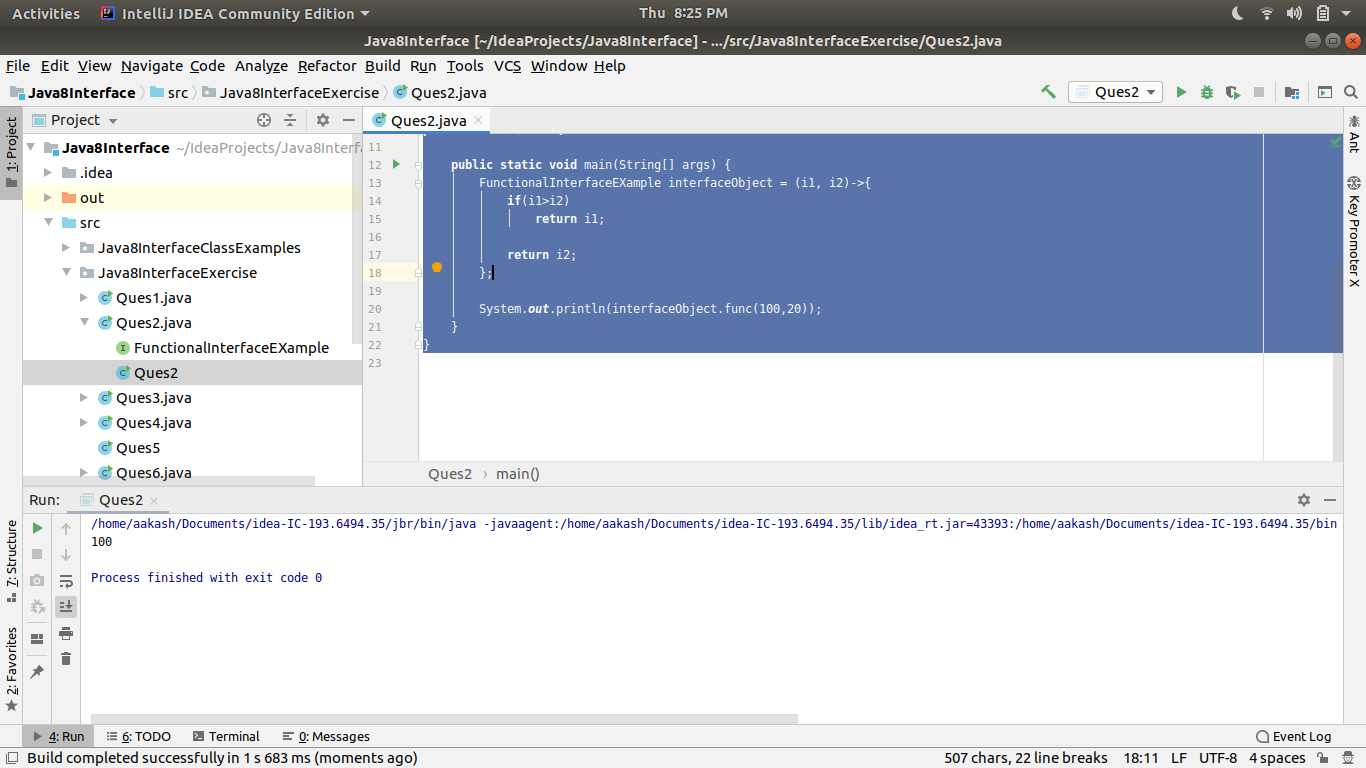
**return** i2;

};

System.***out***.println(interfaceObject.func(100,20));

}

}



*//Using (instance) Method reference create and apply add and subtract method*

*// and using (Static) Method reference create and apply multiplication method for the functional interface created.*

**package** Java8InterfaceExercise;

**interface** Ques3Interface{

**public void** something(**int** x , **int** y);

}

**public class** Ques3 {

**public static void** multiply(**int** x , **int** y)

{

System.***out***.println(**"The multiplication is :"** + (x\*y));

}

**public void** addsub(**int** x, **int** y)

{

System.***out***.println(**"Additon is :"**+ (x+y));

System.***out***.println(**"Subtraction is :"**+ (x-y));

}

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

Ques3Interface operate1 = Ques3::*multiply*; *//(Static) Method reference*

operate1.something(5,6);

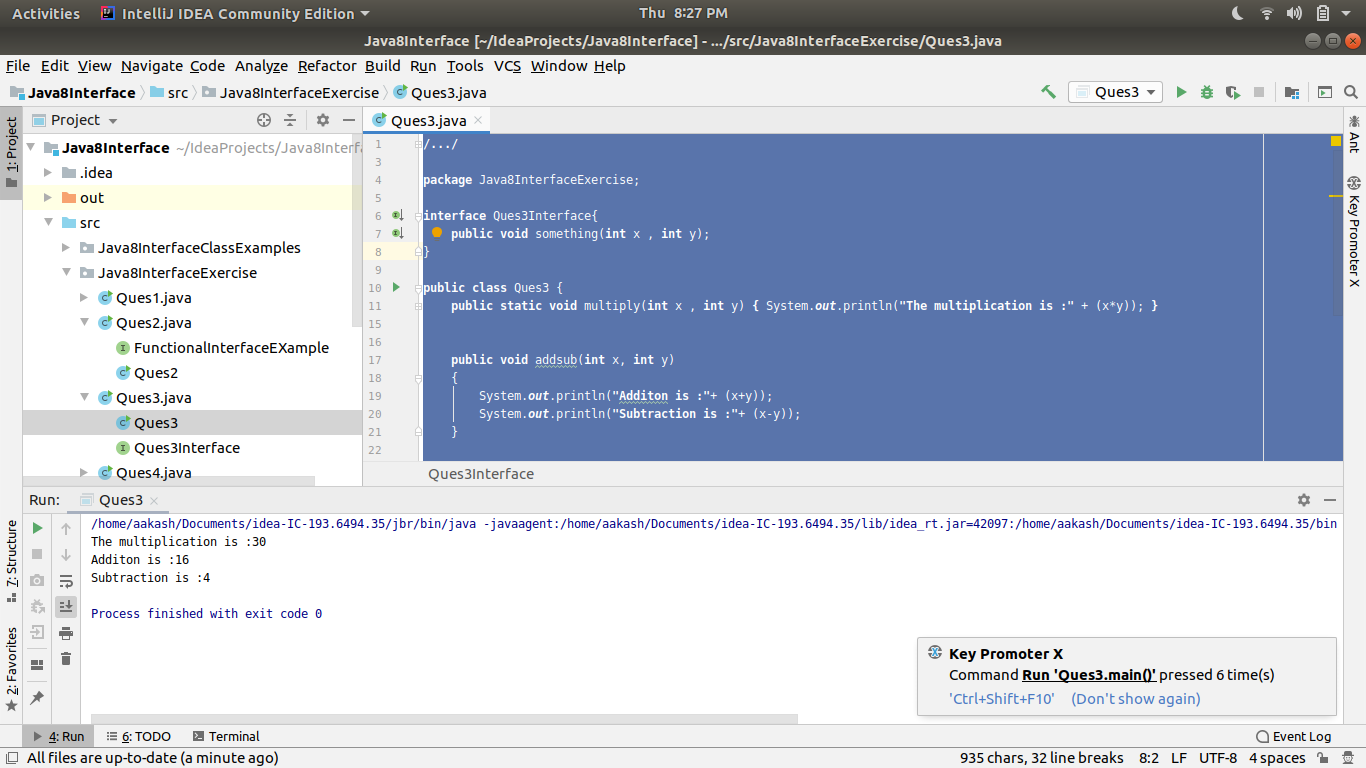
Ques3 methodReference = **new** Ques3(); *// (instance) Method reference*

Ques3Interface operate2 = methodReference::addsub;

operate2.something(10,6);

}

}



*//Create an Employee Class with instance variables (String) name, (Integer)age, (String)city*

*// and get the instance of the Class using constructor reference*

**package** Java8InterfaceExercise;

**interface** Ques4Interface{

**void** createEmployee(String name, Integer age, String city);

}

**public class** Ques4 {

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

Ques4Interface interfaceObject = Employee::**new**;

interfaceObject.createEmployee(**"Aakash "**,23, **"New Delhi"**);

}

}

**class** Employee{

String **name**;

Integer **age**;

String **city**;

**public** String getName() {

**return name**;

}

**public** Integer getAge() {

**return age**;

}

**public** String getCity() {

**return city**;

}

**public** Employee(String name, Integer age, String city){

**this**.**name** = name;

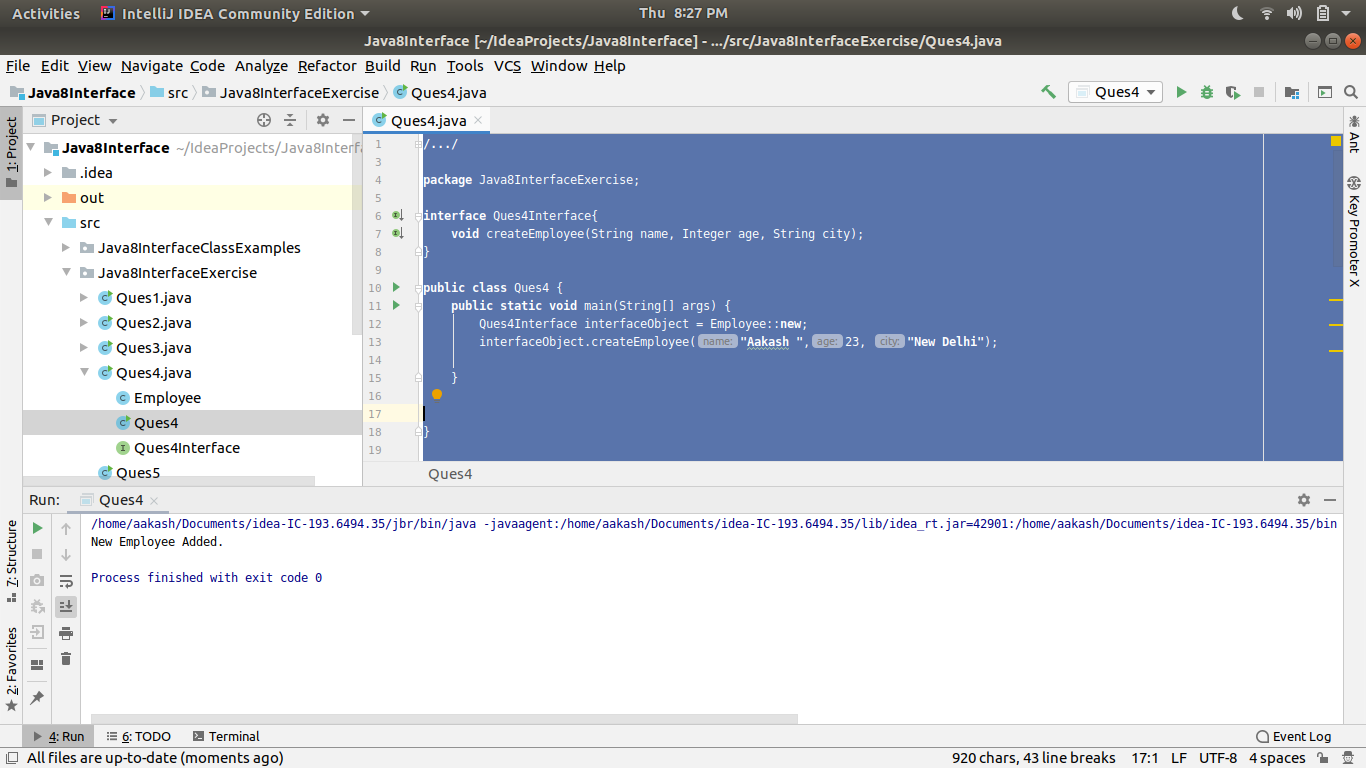
**this**.**age**= age;

**this**.**city**= city;

System.***out***.println(**"New Employee Added."**);

}

}



*//Implement following functional interfaces from java.util.function using lambdas:*

*//*

*// (1) Consumer*

*//*

*// (2) Supplier*

*//*

*// (3) Predicate*

*//*

*// (4) Function*

**package** Java8InterfaceExercise;

**import** java.util.function.Consumer;

**import** java.util.function.Function;

**import** java.util.function.Predicate;

**import** java.util.function.Supplier;

**public class** Ques5 {

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

*// Consumer Functional Interface*

Consumer<Integer> consumerObject =(e1)->{

System.***out***.println(**"consumed : "**+ e1);

};

*// Supplier Functional Interface*

Supplier<Integer> supplierObject = ()->{

System.***out***.println(**"Supplier :"**);

**return** 100;

};

*// Predicate Function Interface*

Predicate<Integer> predicateRefObject = (e1)->{

**if**(e1%2==0)

**return true**;

**return false**;

};

*// Function Functional Interface*

Function<Integer,Integer> functionObject=(e1)->{

**return** e1\*e1;

};

System.***out***.println(functionObject.apply(10));

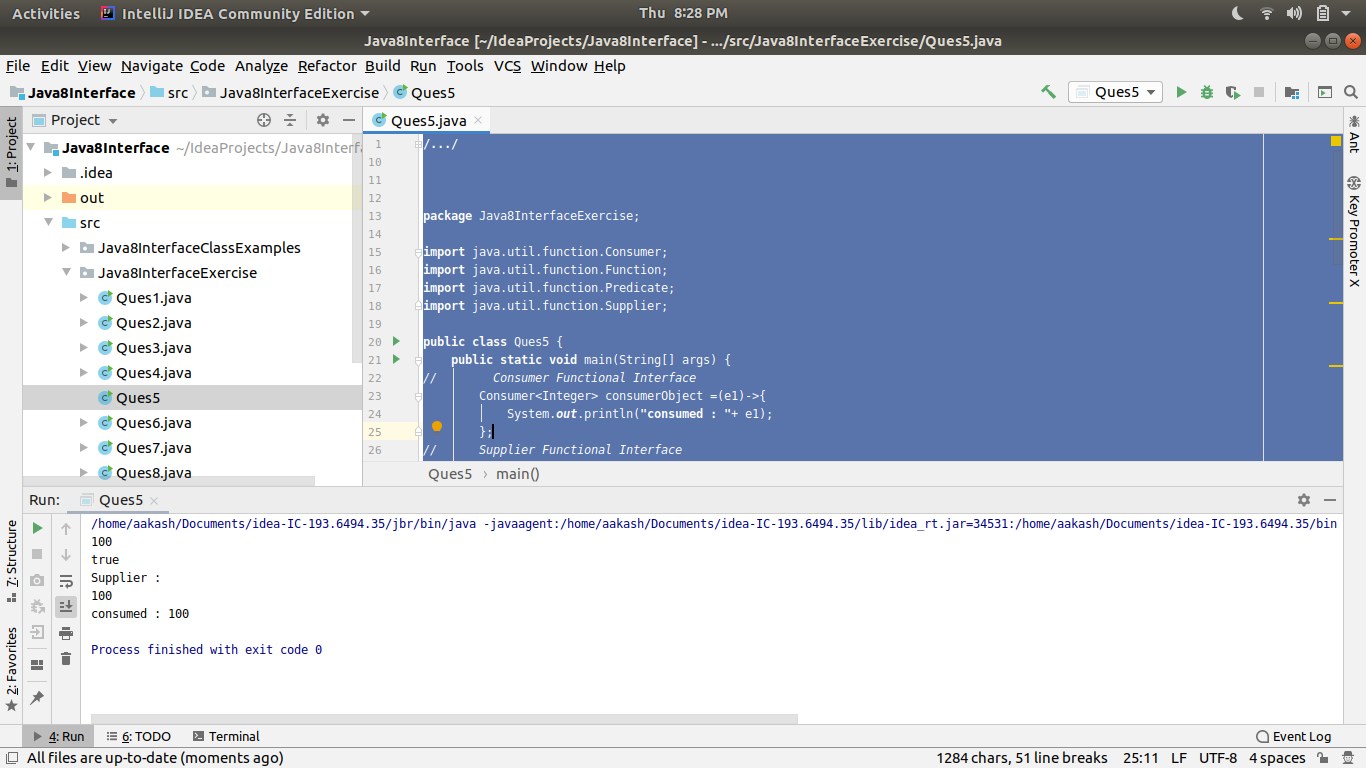
System.***out***.println(predicateRefObject.test(100));

System.***out***.println(supplierObject.get());

consumerObject.accept(100);

}

}



*//Create and access default and static method of an interface.*

**package** Java8InterfaceExercise;

**interface** Interface{

**default void** defaultMethod(){

System.***out***.println(**"this is a default method in Interface"**);

}

**static void** staticMethod(){

System.***out***.println(**"This is a static method in Interface "**);

}

}

**public class** Ques6 **implements** Interface{

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

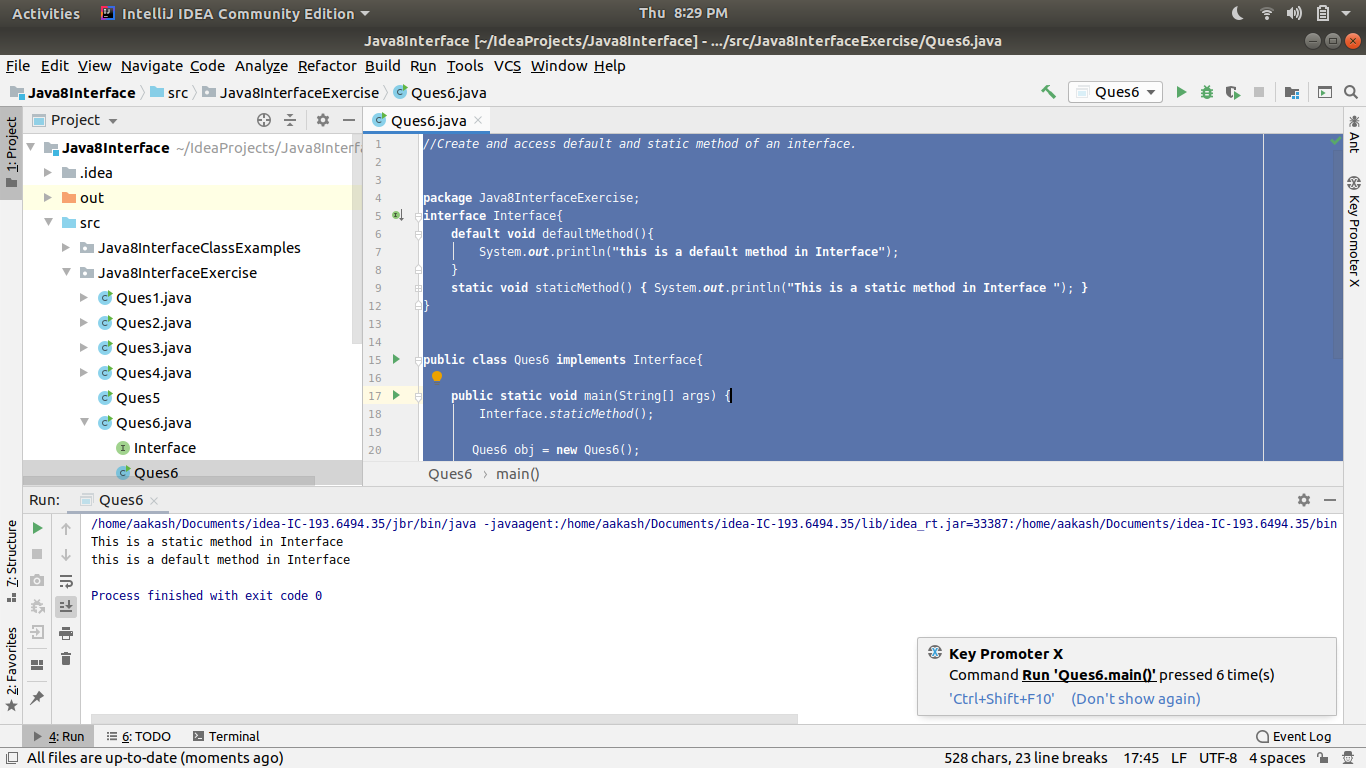
Interface.*staticMethod*();

Ques6 obj = **new** Ques6();

obj.defaultMethod();

}

}



*//Override the default method of the interface.*

**package** Java8InterfaceExercise;

**interface** SampleInterface{

**default void** func1(){

System.***out***.println(**"this is a Default method of Interface- Sample Interface"**);

}

}

**public class** Ques7 **implements** SampleInterface{

@Override

**public void** func1(){

System.***out***.println(**"This is overloading default method of Interface."**);

}

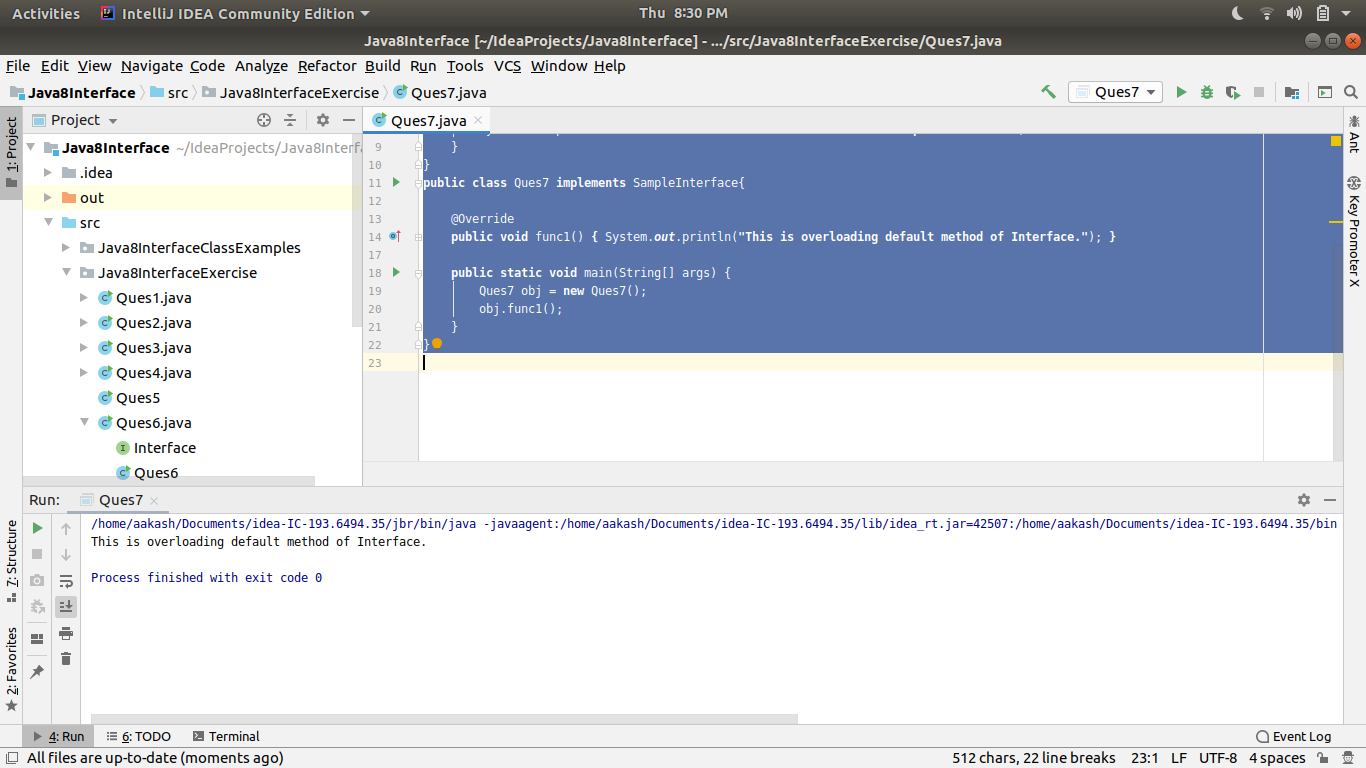
**public static void** main(String[] args) {

Ques7 obj = **new** Ques7();

obj.func1();

}

}



*//Implement multiple inheritance with default method inside interface.*

**package** Java8InterfaceExercise;

**interface** Interface1{

**default void** show(){

System.***out***.println(**"This is Interface 1"**);

}

}

**interface** Interface2{

**default void** show(){

System.***out***.println(**"This is Interface 2"**);

}

}

**public class** Ques8 **implements** Interface1, Interface2{

@Override

**public void** show() {

System.***out***.println(**"This is Class Ques 8"**);

}

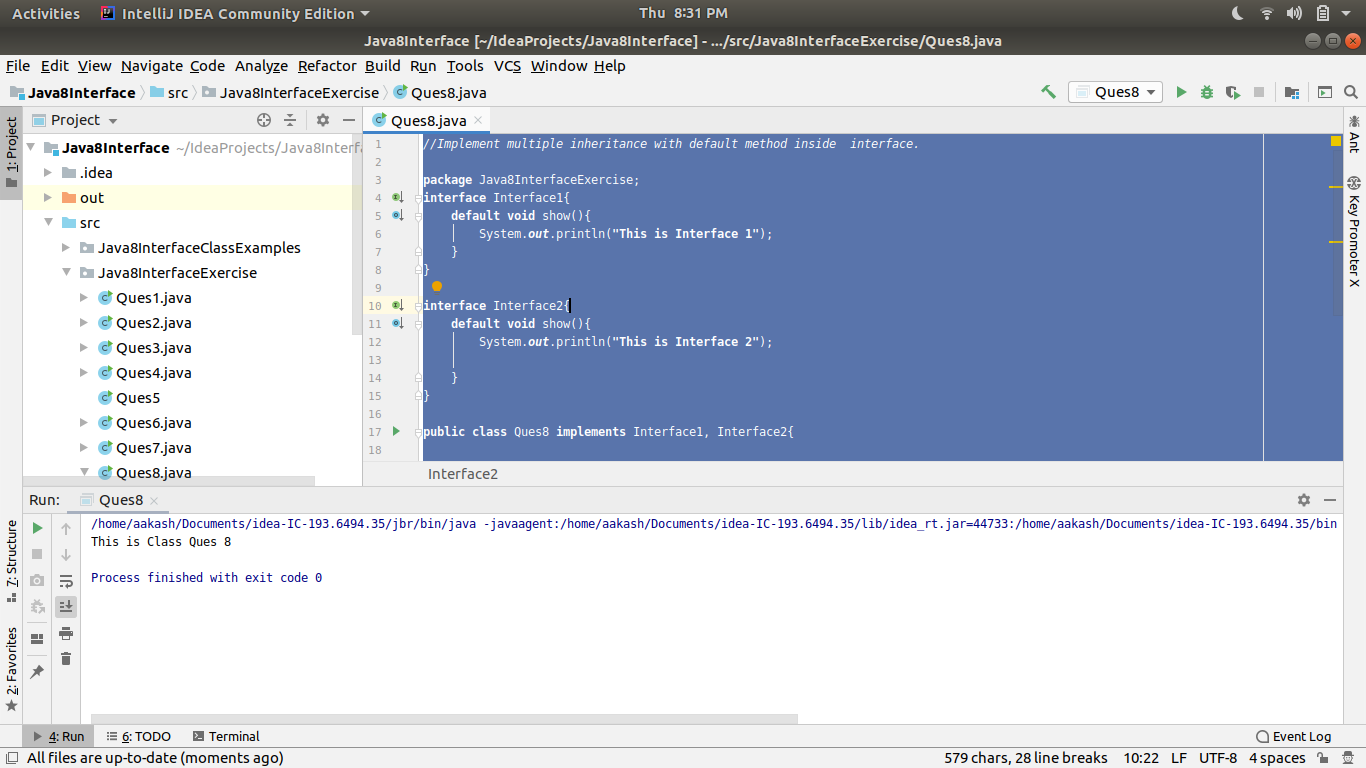
**public static void** main(String[] args) {

Ques8 obj = **new** Ques8();

obj.show();

}

}



*//Collect all the even numbers from an integer list.*

**package** Java8InterfaceExercise;

**import** java.util.Arrays;

**import** java.util.stream.Collector;

**import** java.util.stream.Collectors;

**public class** Ques9 {

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

*// System.out.println(*

Arrays.*asList*(1,2,3,4,5,6,7)

.stream()

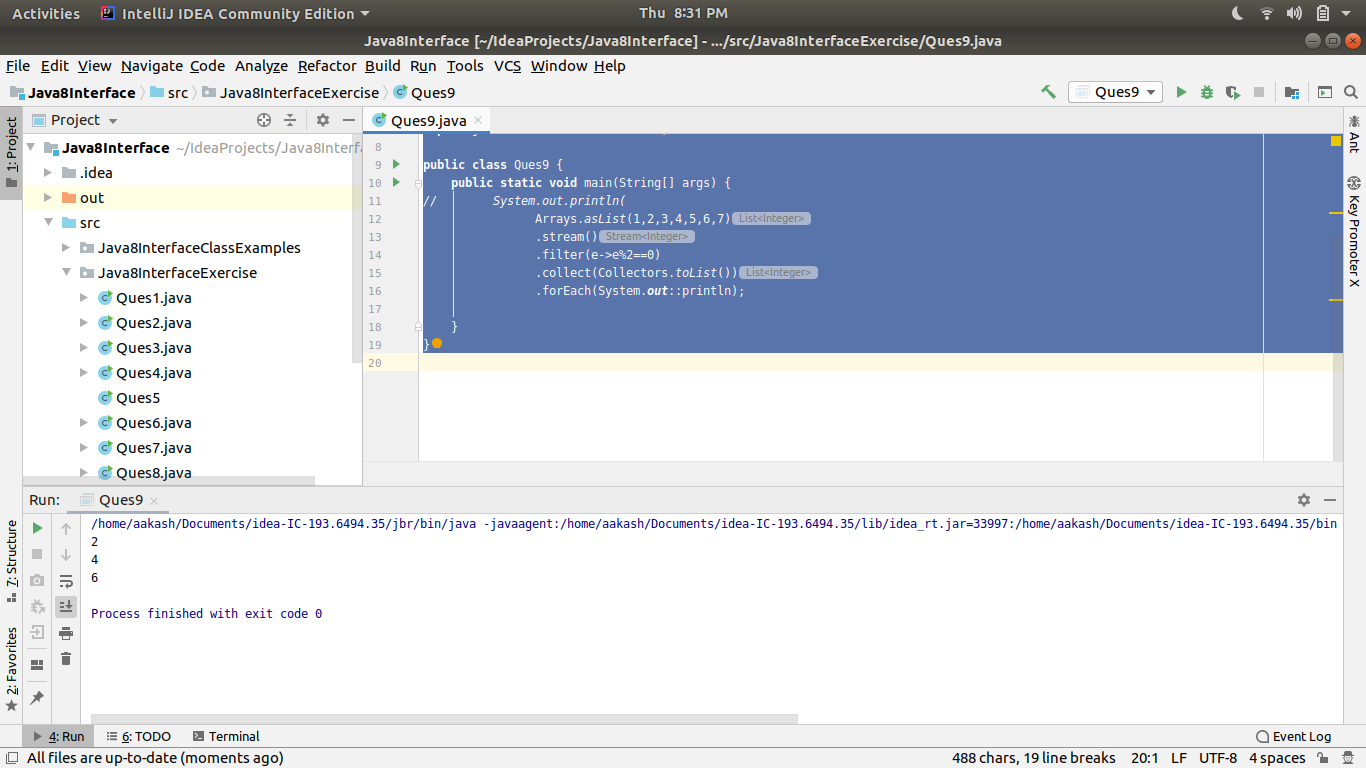
.filter(e->e%2==0)

.collect(Collectors.*toList*())

.forEach(System.***out***::println);

}

}



*//Sum all the numbers greater than 5 in the integer list.*

**package** Java8InterfaceExercise;

**import** java.util.Arrays;

**public class** Ques10 {

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

System.***out***.println(

Arrays.*asList*(1,2,3,4,5,6,7,8)

.stream()

.filter(e->e>5)

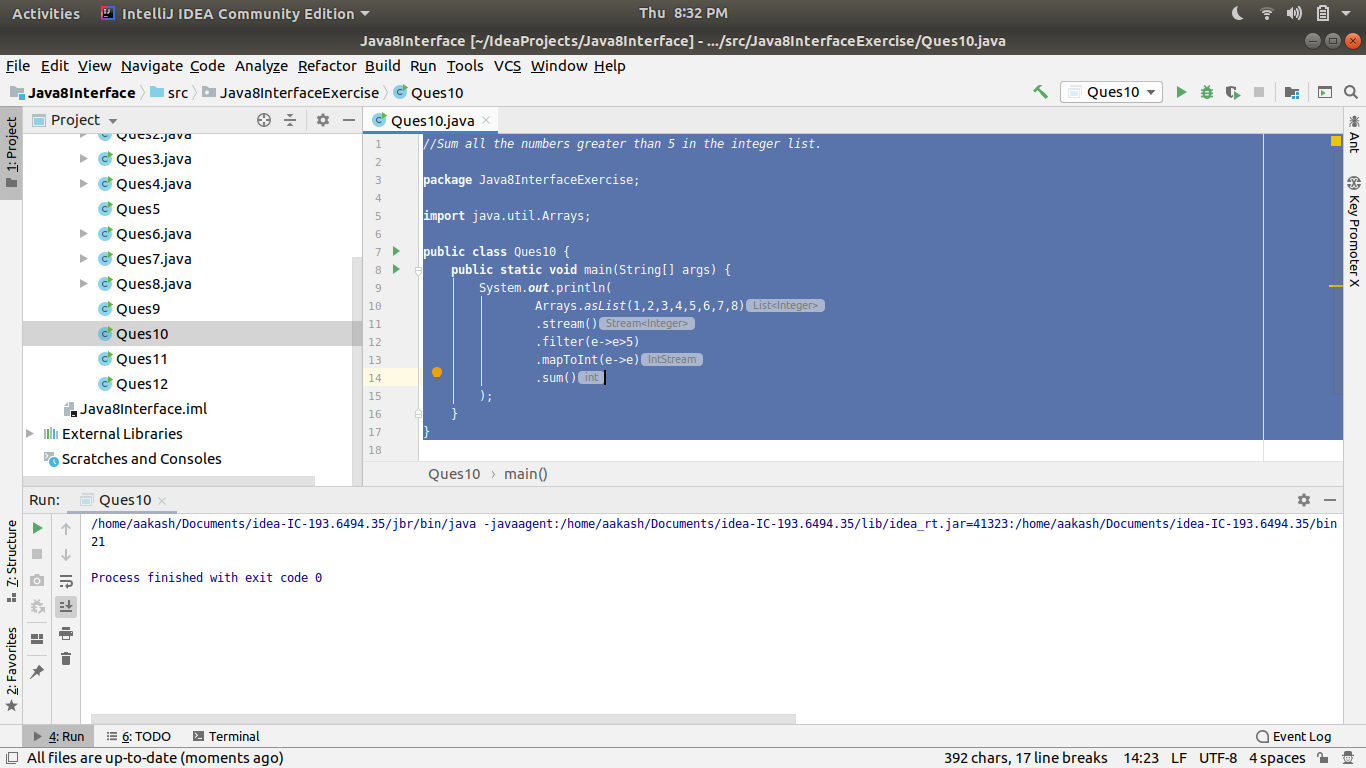
.mapToInt(e->e)

.sum()

);

}

}



*//Find average of the number inside integer list after doubling it.*

**package** Java8InterfaceExercise;

**import** java.util.Arrays;

**import** java.util.stream.Collectors;

**public class** Ques11 {

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

System.***out***.println(

Arrays.*asList*(1,4,2,4,5,6,3,6,2)

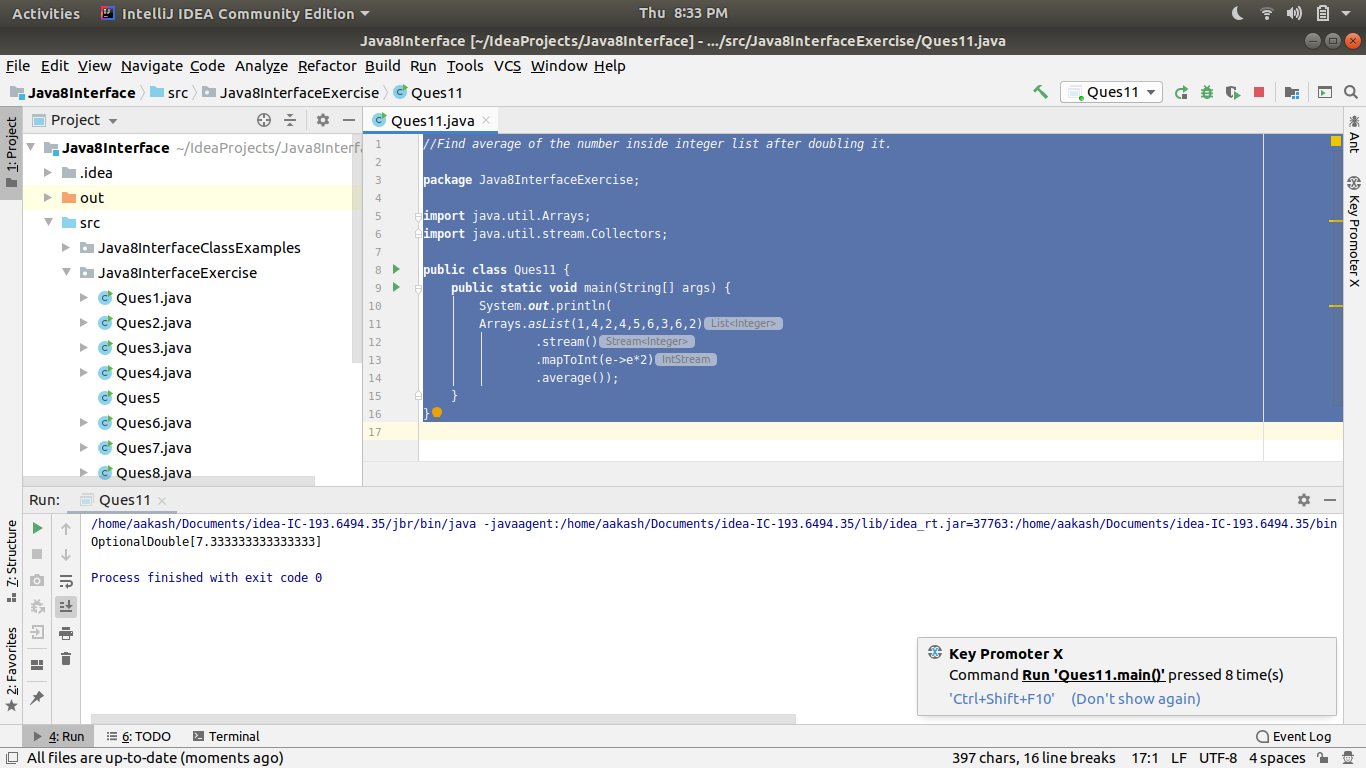
.stream()

.mapToInt(e->e\*2)

.average());

}

}



*//Find the first even number in the integer list which is greater than 3.*

**package** Java8InterfaceExercise;

**import** java.util.Arrays;

**import** java.util.List;

**import** java.util.function.Function;

*//import java.util.function.Predicate;*

**public class** Ques12 {

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

System.***out***.println(Arrays.*asList*(2,3,4,5,6)

.stream()

.filter(e->e%2==0)

.filter(e->e>3)

.findFirst().orElse(-1));

*// Function<List<Integer>, Integer> object = (list)->{*

*//// int max = list.get(0);*

*// for (Integer i :list) {*

*// if(i%2==0 && i>3){*

*// return i;*

*// }*

*// }*

*// System.out.println("Element not found.");*

*// return 0;*

*// };*

*//*

*// System.out.println(object.apply(list1));*

}

}

