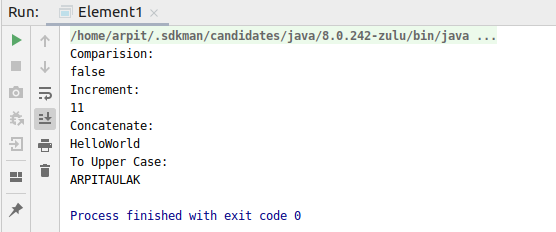
Java 8 Features

Exercise

1. 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)



Code:

**package** Java8;

**import** java.util.Objects;

**interface** First{

**boolean** compare(**int** a,**int** b);

}

**interface** Second{

**int** increment(**int** x);

}

**interface** Third{

String concat(String s1, String s2);

}

**interface** Fourth{

String upper(String s);

}

**public class** Element1 {

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

*//part1*

First first=(a,b)->{

**return** a>b;

};

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

System.***out***.println(first.compare(1,2));

*//part 2*

Second second=(e)->{

**return** e=e+1;

};

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

System.***out***.println(second.increment(10));

*//part 3*

Third third=(a,b)->{

**return** a+b;

};

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

System.***out***.println(third.concat(**"Hello"**,**"World"**));

*//part 4*

Fourth fourth=(e)->{

**return** e.toUpperCase();

};

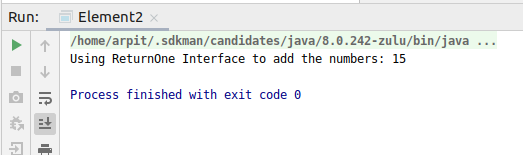
System.***out***.println(**"To Upper Case:"**);

System.***out***.println(fourth.upper(**"arpitaulak"**));

}

}

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



Code:

**package** Java8;

**interface** ReturnOne{

**int** function(**int** a, **int** b);

}

**public class** Element2 {

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

ReturnOne obj = (a,b)->{

**return** a+b;

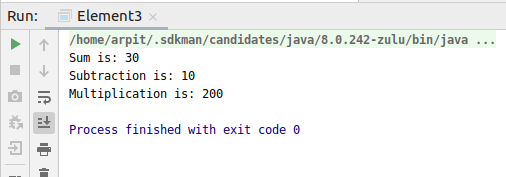
};

System.***out***.println(**"Using ReturnOne Interface to add the numbers: "**+obj.function(10,5));

}

}

1. 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.



Code:

**package** Java8;

**interface** Example{

**int** addSub(**int** a,**int** b);

}

**interface** Example2{

**int** multiply(**int** a,**int** b);

}

**public class** Element3 {

**static int** mult(**int** a,**int** b){

**return** a\*b;

}

**int** sub(**int** a,**int** b){

**return** a-b;

}

**int** add(**int** a,**int** b){

**return** a+b;

}

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

Example example = **new** Element3()::add;

System.***out***.println(**"Sum is: "**+example.addSub(10,20));

Example example1 = **new** Element3()::sub;

System.***out***.println(**"Subtraction is: "**+example1.addSub(20,10));

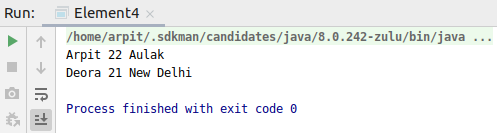
Example2 example2 = Element3::*mult*;

System.***out***.println(**"Multiplication is: "**+example2.multiply(10,20));

}

}

1. Create an Employee Class with instance variables (String) name, (Integer)age, (String)city and get the instance of the Class using constructor reference



Code:

**package** Java8;

**interface** Employable

{

Employee details(String s, **int** a, String c);

}

**class** Employee

{

String **name**;

**int age**;

String **city**;

Employee(String s, **int** a, String c)

{

**name** = s;

**age** = a;

**city** = c;

}

String getDetails()

{

**return this**.**name**+**" "**+**this**.**age**+**" "**+**this**.**city**;

}

}

**public class** Element4 {

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

Employable employable = Employee::**new**;

Employee obj1 = employable.details(**"Arpit"**, 22, **"Aulak"**);

Employee obj2 = employable.details(**"Deora"**, 21, **"New Delhi"**);

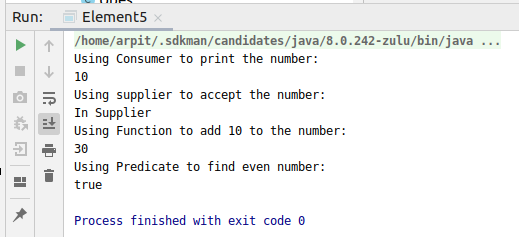
System.***out***.println(obj1.getDetails());

System.***out***.println(obj2.getDetails());

}

}

1. Implement following functional interfaces from java.util.function using lambdas:
   * (1) Consumer
   * (2) Supplier
   * (3) Predicate
   * (4) Function



Code:

**package** Java8;

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

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

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

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

**public class** Element5 {

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

*//part 1*

System.***out***.println(**"Using Consumer to print the number:"**);

Consumer<Integer> consumer = e-> System.***out***.println(e);

consumer.accept(10);

*//part 2*

System.***out***.println(**"Using supplier to accept the number:"**);

Supplier<String> supplier = ()-> {

**return "In Supplier"**;

};

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

*//part 3*

System.***out***.println(**"Using Function to add 10 to the number: "**);

Function<Integer, Integer> function =(e)->{

**return** e+10;

};

System.***out***.println(function.apply(20));

*//part 4*

System.***out***.println(**"Using Predicate to find even number: "**);

Predicate<Integer> predicate = e->{

**return** e%2==0;

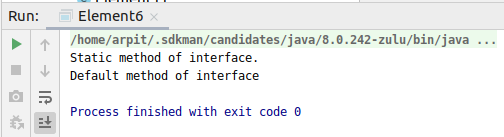
};

System.***out***.println(predicate.test(20));

}

}

1. Create and access default and static method of an interface.



Code:

**package** Java8;

**interface** Test{

**static void** hello(){

System.***out***.println(**"Static method of interface."**);

}

**default void** show(){

System.***out***.println(**"Default method of interface"**);

}

}

**public class** Element6 **implements** Test{

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

Element6 obj=**new** Element6();

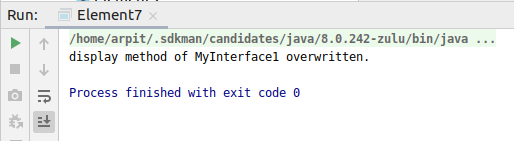
Test.*hello*();

obj.show();

}

}

1. Override the default method of the interface.



Code:

**package** Java8;

**interface** MyInterface1{

**public static int *num*** = 100;

**public default void** display() {

System.***out***.println(**"display method of MyInterface1"**);

}

}

**public class** Element7 **implements** MyInterface1 {

**public void** display() {

System.***out***.println(**"display method of MyInterface1 overwritten."**);

}

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

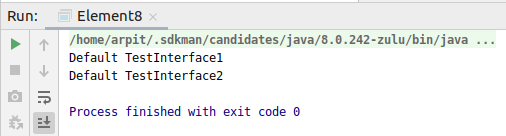
Element7 obj = **new** Element7();

obj.display();

}

}

1. Implement multiple inheritance with default method inside interface.



Code:

**package** Java8;

**interface** TestInterface1

{

**default void** show()

{

System.***out***.println(**"Default TestInterface1"**);

}

}

**interface** TestInterface2

{

**default void** show()

{

System.***out***.println(**"Default TestInterface2"**);

}

}

**class** Element8 **implements** TestInterface1, TestInterface2

{

*// Overriding default show method*

**public void** show()

{

TestInterface1.**super**.show();

TestInterface2.**super**.show();

}

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

{

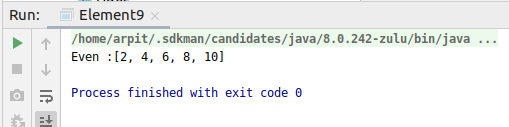
Element8 d = **new** Element8();

d.show();

}

}

1. Collect all the even numbers from an integer list.



Code:

**package** Java8;

**import** java.util.Arrays;

**import** java.util.List;

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

**public class** Element9 {

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

List<Integer> evenList = Arrays.*asList*(1,2,3,4,5,6,7,8,9,10);

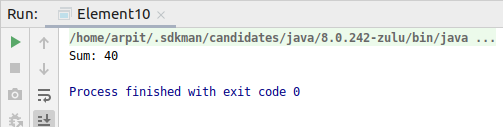
List result= evenList.stream().filter(e->e%2==0).collect(Collectors.*toList*());

System.***out***.println(**"Even :"**+result);

}

}

1. Sum all the numbers greater than 5 in the integer list.



Code:

**package** Java8;

**import** java.util.Arrays;

**import** java.util.List;

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

**public class** Element10 {

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

List<Integer> evenList = Arrays.*asList*(1,2,3,4,5,6,7,8,9,10);

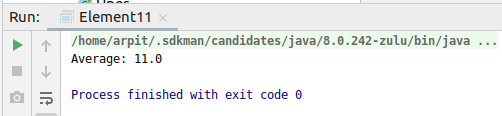
Integer result= evenList.stream().filter(e->e>5).collect(Collectors.*summingInt*(e->e));

System.***out***.println(**"Sum: "**+result);

}

}

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



Code:

**package** Java8;

**import** java.util.Arrays;

**import** java.util.List;

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

**public class** Element11 {

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

List<Integer> evenList = Arrays.*asList*(1,2,3,4,5,6,7,8,9,10);

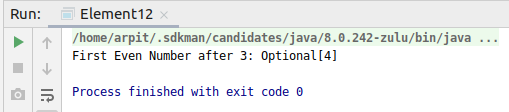
Double result= evenList.stream().map(e->e\*2).collect(Collectors.*averagingInt*(e->e));

System.***out***.println(**"Average: "**+result);

}

}

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



Code:

**package** Java8;

**import** java.util.Arrays;

**import** java.util.List;

**import** java.util.Optional;

**public class** Element12 {

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

List<Integer> evenList = Arrays.*asList*(1,2,3,4,5,6,7,8,9,10);

Optional<Integer> result= evenList.stream().filter(e->e%2==0).filter(e->e>3).findFirst();

**if**(result.isPresent())

{

System.***out***.println(**"First Even Number after 3: "**+result);

}

**else**

{

System.***out***.println(**"No result found"**);

}

}

}