Lamda Assignments

Q1

```java

@FunctionalInterface
interface Arithmetic {

    int operation(int n1, int n2);

}

public class Main {

    public static void main(String[] args) {

        //Lambda expression for addition
        Arithmetic addition = (int n1, int n2) -> n1 + n2;

        //Lambda expression for subtract
        Arithmetic subtraction = (int n1, int n2) -> n1 - n2;

        //Lambda expression for multiply
        Arithmetic multiplication = (int n1, int n2) -> n1 * n2;

        //Lambda expression for division
        Arithmetic division = (int n1, int n2) -> n1 / n2;

        //Lambda expression for division
        Arithmetic modulo = (int n1, int n2) -> n1 % n2;

        System.out.println("Addition : " + addition.operation(100, 600));

        System.out.println("Subtraction : " + subtraction.operation(300, 100));

        System.out.println("Multiplication: " + multiplication.operation(30, 5));

        System.out.println("Division : " + division.operation(10, 2));

        System.out.println("Modulo : " + modulo.operation(10, 6));

    }

}

```

```
Console
<terminated> Main (1) [Java Application] C:\Program Files\Java\jd
Addition : 700
Subtraction : 200
Multiplication: 150
Division : 5
Modulo : 4
```

Q 3

```java
import java.util.function.Predicate;
public class PredicateDemo {
    public static void main(String[] args)
    {
        Predicate<Integer> greaterThanTen = (i) -> i > 10;

        // Creating predicate
        Predicate<Integer> lowerThanTwenty = (i) -> i < 20;
        boolean result = greaterThanTen.and(lowerThanTwenty).test(15);
        System.out.println(result);

        // Calling Predicate method
        boolean result2 = greaterThanTen.and(lowerThanTwenty).negate().test(15);
        System.out.println(result2);
    }
}
```

```
<terminated> PredicateDemo [Java Application] C:\Pr
true
false
```

```java
import java.util.ArrayList;
import java.util.List;
import java.util.function.Supplier;

public class SupplierDemo {

    public static void main(String[] args) {



        List<String> names = new ArrayList<String>();
        names.add("Riddhi");
        names.add("Riya");
        names.add("Priya");
        names.add("Anuj");
```

```
        names.stream().forEach((item)-> {
            printNames(()-> item);
        });
    }

private static void printNames(Supplier<String> supplier) {
        System.out.println(supplier.get());
    }
}
```

```
Riddhi
Riya
Priya
Anuj
```

```
import java.util.ArrayList;

import java.util.List;

import java.util.function.Consumer;

import java.util.function.Predicate;


public class ConsumerTest {


   public static void main(String[] args) {


      System.out.println("E.g. #1 - Java8 Consumer Example\n");


      Consumer<String> consumer = ConsumerTest::printNames;

      consumer.accept("C++");

      consumer.accept("Java");
```

```
        consumer.accept("Python");
        consumer.accept("Ruby");
    }

    private static void printNames(String name) {
        System.out.println(name);
    }
}
```

```
C++
Java
Python
Ruby
```

Q2

```java
class Order {

  int id;
  String status;
  int price;

  public Order(int id, String status, int price){

  super();
  this.id = id;
  this.status = status;
  this.price = price;
  }

  public int getId(){
  return id;
  }
  public void setId(int id){
  this.id = id;
  }

  public String getStatus(){
  return status;
  }
  public void setStatus(String Status){
  this.status = status;
  }

  public int getPrice(){
      return price;
      }
      public void setPrice(String Price){
      this.price = price;
      }
}
```

eclipse-workspace - lamdaProject/src/lamdaProject/OrderExample.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

LamdaEx.java | Person.java | ConsumerTest.java | OrderExample.java

```java
package lamdaProject;

import java.util.ArrayList;
import java.util.List;
import java.util.stream.Stream;

public class OrderExample {

    public static void main(String[] args) {
        List<Order>list = new ArrayList<>();
        list.add(new Order(1, "Accepted",30000));
        list.add(new Order(2, "Completed", 40000));
        list.add(new Order(3, "Accepted", 15000));
        list.add(new Order(4, "Completed",5000));

        System.out.println("Orders having prize above 10000");
        Stream<Order> order1 = list.stream().filter(p -> p.price > 10000);
        order1.forEach(order -> System.out.println(order.price + ":" + order.status));

        System.out.println("Orders having status as Accepted");
        Stream<Order> order2 = list.stream().filter(p -> p.getStatus().matches("Accepted"));
        order2.forEach(order -> System.out.println(order.id + ":" + order.price+ " " + order.status));


        System.out.println("Orders having status as Completed");
        Stream<Order> order3 = list.stream().filter(p -> p.getStatus().matches("Completed"));
        order2.forEach(order -> System.out.println(order.id + ":" + order.price+ " " + order.status));

    }

}
```

Output

Console

```
<terminated> OrderExample [Java Application] C:\Program Files\Java\jdk-16.0.2\bin\javaw.exe  (11-Aug-202
Orders having prize above 10000
30000:Accepted
40000:Completed
15000:Accepted
Orders having status as Accepted
1:30000 Accepted
3:15000 Accepted
```

Q4

```java
package lamdaProject;


import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.Collections;


public class Oddlen {

    public static void main(String[] args) {
        List<String> list = new ArrayList<>(Arrays.asList("Riddhi", "priya", "Siddhi", "Raj", "Bhoomi"));
        list.removeIf(i -> i.length() % 2 != 0);
        System.out.println("Odd length names are as follows:" +list);
    }
}
```

Output:

```
<terminated> Oddlen [Java Application] C:\Program Files\Java\jdk-16.0.2\bin\javaw.exe  (11-Aug-2021, 7:59:30 pm – 7:59:33 pm)
Odd length names are as follows:[Riddhi, Siddhi, Bhoomi]
```

Q5

```java
public class firstLetter{
    public static void main(String[] args) {
        List<String> list = Arrays.asList("Riddhi", "Rathod");
        StringBuilder str = new StringBuilder();
        forEach(list, a-> str.append(a.charAt(0)));
        System.out.println(str);
    }

private static <String> void forEach(List<String> list, Consumer <String>consumer) {
    for(String t : list) {
        consumer.accept(t);


}



    }
}
```

Output

```
<terminated> firstLetter [Java Application] C:\
RR
```

Q6

```java
public class UpperCase {

    public static void main(String[] args) {
        List<String> list = new ArrayList<>(Arrays.asList("Riddhi", "priya", "Siddhi", "Raj", "Bhoomi"));
        System.out.println(list);
        list.replaceAll(e -> e.toUpperCase());
        System.out.println(list);
    }}
```

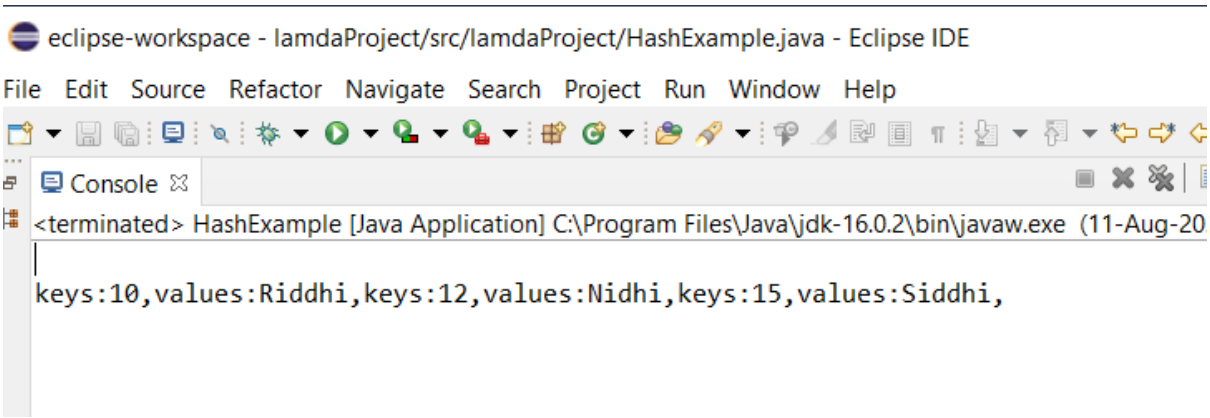
Output:

```
Console
<terminated> UpperCase [Java Application] C:\Program Files\Java\jdk-16.0.2\
[Riddhi, priya, Siddhi, Raj, Bhoomi]
[RIDDHI, PRIYA, SIDDHI, RAJ, BHOOMI]
```

Q7

```java

public class HashExample{
    public static void main(String[] args) {
        Map<Integer, String> map = new HashMap<Integer, String>();
        map.put(10, "Riddhi");
        map.put(15, "Siddhi");
        map.put(12, "Nidhi");

        StringBuilder s = new StringBuilder("");
        for(Map.Entry m:map.entrySet()) {

            s.append("keys:" + m.getKey() + ",");
            s.append("values:" + m.getValue() + ",");
        }

        System.out.println("\n" +s +"\n");
    }

private static Entry[] map(Object entrySet, Object object) {
        // TODO Auto-generated method stub
        return null;
    }

private static Object entrySet() {
    // TODO Auto-generated method stub
    return null;
}
        }
```

Output:



Q8

```java

public class Hello extends Thread{
    public void run() {

            System.out.println("Running....");
        }

        public static void main(String[] args) {
            Hello h = new Hello();
            h.start();

            List<Integer> numbers = new ArrayList<>();
            numbers.add(20);
            numbers.add(40);
            numbers.add(60);

            Consumer<List<Integer>>print=list -> list.stream().forEach(a -> System.out.println(a + "")
                    );
            print.accept(numbers);


        }
    }

```

Output: