LAMBDA ASSIGNMENTS

1.PROGRAM

```
@FunctionalInterface
interface Arithmetic{
	int operations(int a,int b);

}
public class LambdaAssignment1 {

	public static void main(String [] args)
	{
		//performing the addition operation

		Arithmetic addition = (int a, int b)->(a+b);
		System.out.println("Addition is: "+addition.operations(10, 20));

		//performing the subtraction operation
		Arithmetic subtraction = (int a, int b)->(a-b);
		System.out.println("Subtraction is: "+subtraction.operations(100, 5));

		//performing the multiplication operation
		Arithmetic multiplication = (int a, int b)->(a*b);
		System.out.println("Multiplication is: "+multiplication.operations(110, 20));

		//performing the division//performing the addition operation operation
		Arithmetic division = (int a, int b)->(a/b);
		System.out.println("Division is: "+division.operations(500, 67));

	}
}
```

OUTPUT

```
C:\Users\GLMACHAD\Documents>javac LambdaAssignment1.java

C:\Users\GLMACHAD\Documents>java LambdaAssignment1
Addition is: 30
Subtraction is: 95
Multiplication is: 2200
Division is: 7

C:\Users\GLMACHAD\Documents>
```

2.PROGRAM

```
import java.util.ArrayList;

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

class Orders{

   String status;
   float price;
   public Orders( String status, float price) {
      super();

      this.status = status;
     this.price = price;
   }
}
```

```
public class LambdaAssignment2 {



        public static void main(String[] args) {
    List<Orders> list=new ArrayList<Orders>();
    list.add(new Orders("Order Status:Accepted",170000f));
    list.add(new Orders("Order Status:Completed",60000f));
    list.add(new Orders("Order Status:Accepted",370000f));
    list.add(new Orders("Order Status:Processing",2500f));
    list.add(new Orders("Order Status:Out For Delivery",150000f));
    list.add(new Orders("Order Status:Processing",5500f));
    list.add(new Orders("Order Status:Processing",6500f));

    // using lambda to filter data
    Stream<Orders> filtered_data = list.stream().filter(p -> p.price > 10000 &&
p.status.startsWith("Order Status:Accepted") || p.status.startsWith("Order Status:Completed"));

    // we will use lambda to iterate through collection
    filtered_data.forEach(Orders -> System.out.println("Order Price is "+Orders.price+ " &
"+Orders.status));
  }
}
```

OUTPUT

```
C:\Users\GLMACHAD\Documents>javac LambdaAssignment2.java
C:\Users\GLMACHAD\Documents>java LambdaAssignment2
Order Price is 170000.0 & Order Status:Accepted
Order Price is 60000.0 & Order Status:Completed
Order Price is 370000.0 & Order Status:Accepted
```

3.PROGRAM

```
import java.util.Arrays;

import java.util.function.Consumer;

import java.util.function.Function;

import java.util.function.Predicate;

import java.util.function.Supplier;


public class LambdaAssignment3 {


        public static void main(String[] args) {


                String[] str = {"Glenn", "Sam","kim"};




                Supplier<String> supplier = ()-> Arrays.toString(str) ;

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


                Consumer<String[]> consumer = (string) ->
System.out.println(Arrays.toString(string));

                consumer.accept(str);


                Predicate<String[]> predicate = (string) -> Arrays.toString(string).contains("Singh");

                System.out.println(predicate.test(str));


                Function<String[], String> function = (string) -> Arrays.toString(string);

                System.out.println(function.apply(str));


        }




}
```

OUTPUT

```
C:\Users\GLMACHAD\Documents>java LambdaAssignment3

[Glenn, Sam, kim]

[Glenn, Sam, kim]

false

[Glenn, Sam, kim]


C:\Users\GLMACHAD\Documents>
```

4.PROGRAM

```java
import java.util.ArrayList;

public class LambdaAssignment4 {



        public static void main(String[] args)
        {
          ArrayList<String> students = new ArrayList<String>();

          students.add("Glenn");
          students.add("kim");
          students.add("sam");
          students.add("nehal");
          students.add("kris");
          students.removeIf(m -> (m.length() % 2 != 0));

          //System.out.println("Students name Does not start with S");
          for (String str : students) {
            System.out.println(str);
          }
```

```
                /*System.out.println("-----------------------------------------------------");


                ArrayList<Integer> students1 = new ArrayList<Integer>();
                students1.add(32);
                students1.add(56);
                students1.add(67);
                students1.add(43);
                students1.add(87);
                students1.removeIf(n -> (n %2!=0));


                System.out.println("Students name with odd lengths is removed");
                for (int i: students1) {
                   System.out.println(i);
                }*/
              }
              }
```

5.PROGRAM

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.function.Function;

public class LambdaAssignment5 {

          public static void main(String[] args) {



                    List<String> str = Arrays.asList("Glenn", "Sam","kim");


                    Function<List<String>,List<String>> function = (string) -> {
```

```
                List<String> stringList = new ArrayList<String>();
                for (String s : string) {
                stringList.add(""+s.charAt(0));
        }  return stringList;};

        System.out.println(function.apply(str));

    }

}
```

OUTPUT

```
C:\Users\GLMACHAD\Documents>java LambdaAssignment5
[G, S, k]

C:\Users\GLMACHAD\Documents>
```

6.PROGRAM

```
import java.util.ArrayList;
import java.util.function.UnaryOperator;

class Op implements UnaryOperator<String> {
          public String apply(String str) {
            return str.toUpperCase();
          }
        }
public class LambdaAssignment6 {
        public static void main(String[] args) {
        ArrayList<String> list = new ArrayList<>();
           list.add("Hii");
           list.add("i am");
           list.add("Glenn Machado");
```

```
                    list.add("I am doing well");
                    list.add("Great.");
                     System.out.println("Contents of the list before conversion: "+list);
                    list.replaceAll(new Op());
                     System.out.println("\nContents of the list after replace operation: "+list);
                   }
                 }
```

OUTPUT

```
C:\Users\GLMACHAD\Documents>javac LambdaAssignment6.java

C:\Users\GLMACHAD\Documents>java LambdaAssignment6
Contents of the list before conversion: [Hii, i am, Glenn Machado, I am doing well, Great.]


Contents of the list after replace operation: [HII, I AM, GLENN MACHADO, I AM DOING WELL,
GREAT.]
```

7.PROGRAM

```
import java.util.HashMap;
import java.util.Map;
import java.util.Map.Entry;
import java.util.function.Function;

public class LambdaAssignment7 {

	public static void main(String[] args) {

		Map<Integer, String> map = new HashMap<>();
		map.put(1, "Glenn");
		map.put(2, "Machado");

		Function<Map<Integer, String>, StringBuilder> function = mapValues -> {
			StringBuilder sb = new StringBuilder();
			for (Entry<Integer, String> string : mapValues.entrySet()) {
```

```
                                sb.append(string.getKey());

                                sb.append(string.getValue());

                        }

                        return sb;

                };


                System.out.println(function.apply(map));


        }


}
```

OUTPUT

```
C:\Users\GLMACHAD\Documents>javac LambdaAssignment7.java


C:\Users\GLMACHAD\Documents>java LambdaAssignment7
1Glenn2Machado
```

8.PROGRAM

```
import java.util.Arrays;

import java.util.List;

import java.util.function.Consumer;


public class LambdaAssignment8 {





                public static void main(String[] args) {


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

```
                    Consumer<List<Integer>>dispList = (list1) -> {
                            for(Integer integer : list1) {
                                    System.out.print(integer + " ");
                                    }
                            };

                    Thread newthread = new Thread( ()-> dispList.accept(list) );

                    newthread.start();

            }
}
```

OUTPUT

```
C:\Users\GLMACHAD\Documents>javac LambdaAssignment8.java

C:\Users\GLMACHAD\Documents>java LambdaAssignment8
1 2 3 4 5 6 7 8 9
C:\Users\GLMACHAD\Documents>
```