# Day 4-Assignment

## Bollepalli Sai Sreekanth

1. Write a Program to implement Anonymous Class.

A.

Code:

```
package sre;
interface AnnonymousClass{
    public void show();
}
public class Q41 {

    public static void main(String[] args) {
            AnnonymousClass anom=new AnnonymousClass(){

                @Override
                public void show() {
                        System.out.println("This is annonumus class");

                }

            };
            anom.show();

    }

}
```

Output:

This is annonumus class

2. Write a Program to implement Two Abstract methods of an Interface using Anonymous Class.

A.
```
package sre;
interface AnnonymousClass{
	public void show();
	public void show1();
}
public class Q41 {

	public static void main(String[] args) {
		AnnonymousClass anom=new AnnonymousClass(){

			@Override
			public void show() {
				System.out.println("This is annonumus class");

			}

			@Override
			public void show1() {
				System.out.println("Hello, World");

			}

		};
		anom.show();
		anom.show1();

	}

}
```

Output:
This is annonumus class
Hello, World

3. Write a Program to give an example of interface reference (by which we can only call the abstract methods of the interface).

A.
```
package sre;
interface AnnonymousClass{
	public void show();
	public void show1();
}
class vari implements AnnonymousClass{

	@Override
	public void show() {
		System.out.println("Hi");

	}

	@Override
	public void show1() {
		System.out.println("Hello world");

	}
	public void show2() {
		System.out.println("How are You");
	}
}

public class Q43 {

	public static void main(String[] args) {
		AnnonymousClass anom=new vari();
		anom.show();
		anom.show1();
		//anom.show2(); cannot work


	}

}
```

Output:

```
Hi
Hello world
```

4. Write a Program to implement non parameterized Lambda Function

A.

Code:

```
package sre;

@FunctionalInterface
interface k{
	public void show();
}

public class Q44 {

	public static void main(String[] args) {
		k ref=()->{
			System.out.println("Hello world");
		};
		ref.show();


	}

}
```

Output:

Hello world

5.Write a Program to implement Lambda Function with Single Parameter.

A.

Code:

```
package sre;

@FunctionalInterface
interface k{
	public int showvalue(int a);
}

public class Q45 {
```

```
	public static void main(String[] args) {
		k ref=(a)->(a);
		System.out.println(ref.showvalue(10));


	}

}
```

Output:

10

6. Write a Program to implement Lambda Function with Multiple Parameter.

A.

Code:

```
package sre;

@FunctionalInterface
interface k{
	public int multiply(int a,int b);
}

public class Q44 {

	public static void main(String[] args) {
		k ref=(a,b)->(a*b);
		System.out.println(ref.multiply(10,5));


	}

}
```

Output:

50

7. Write a Program to implement Method Reference with Static and Non Static Method.

A.

Code:

```
package sre;
@FunctionalInterface
interface z{
	void k1();
}

public class Q47 {

	public static void show() {
		System.out.println("Method reference with static");

	}

	public void print() {
		System.out.println("Method reference without static");

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


		z ref=new Q47()::print;
		ref.k1();

		z ref2= Q47::show;
		ref2.k1();

	}

}
```

Output:

```
Method reference without static
Method reference with static
```

8. Write a Program to implement Bi Function with Method Reference and Lambda Function.

A.

Code:

```
package sre;

import java.util.function.BiFunction;

import java.util.function.Function;


class test5{

	public static int add(int a,int b) {

		return a+b;


	}

}


public class biFunction {


	public static void main(String[] args) {

		BiFunction<Integer,Integer,Integer> adder=test5::add;

		int result=adder.apply(10, 20);

		System.out.println("result using method reference");

		System.out.println(result);

		BiFunction<Integer,Integer,Integer> lambda=(a,b)->(a+b);

		int result1=lambda.apply(10, 20);

		System.out.println("result using lamba function");

		System.out.println(result1);


	}
```

```
}
```

Output:

```
result using method reference
30
result using lamba function
30
```

9. Write a Program to implement For Each Loop.

A.Code:

```
package sre;

import java.util.ArrayList;

public class Q49 {

	public static void main(String[] args) {
		ArrayList<String> countries = new ArrayList<String>();
		countries.add("India");
		countries.add("USA");
		countries.add("UK");
		countries.forEach((i)-> System.out.println(i));

	}

}
```

Output:

```
India
USA
UK
```

10. Write a Program to implement Stream

```
package sre;

import java.util.ArrayList;
import java.util.stream.Collectors;

public class Q49 {
```

```
	public static void main(String[] args) {
		ArrayList<String> countries = new ArrayList<String>();
		countries.add("India");
		countries.add("Russia");
		countries.add("Afganisthan");
		countries.add("USA");
		countries.add("Australia");
		countries.add("UK");
		countries.forEach((i)-> System.out.println(i));

		countries.stream().sorted((a,b)-
>b.compareTo(a)).forEach(System.out::println);
	}

}
```

Output:
Australia
UK
USA
UK
Russia
India
Australia
Afganisthan

# JavaScript questions:

1A.

```
Function odd(){
for (let i = 1; i < 100; i = i + 2) {
console.log(1);}
}
odd();
```

2A.

```
function factorial (integer) {
let  product = 1;
for (let i= 2; i <= integer; i++) {
product= product *i;
}
console.log(product);
} factorial(6);
```

3A.

```
function triarea(base, height) {
return 0.5 * base * height;}
console.log(triarea(2, 3));
```

4A.

```
function isPrime(n)
{

    if (n <= 1)
        return false;
    for (let i = 2; i <= (n); i++)
        if (n % i == 0)
            return false;

    return true;
}
onsole.log(isPrime(17));
```

5A.

```
Let  a=10;
Let b=20;
```

```
a=a+b;
b=a-b;
a=a-b;
console.log(a,b)
```