# LAB 10

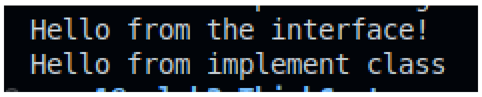
**Programming Exercise:**

**Program set 1:**

1. Create an interface Greeter with a default method greet() that prints "Hello from the interface!". Implement this interface in a class and invoke the method.

```
interface Greeter{
   default void greet(){
       System.out.println("Hello from the interface!");
   }
}

class Imp implements Greeter{
   public void print(){
       System.out.println("Hello from implement class");
   }
}

public class Main {
   public static void main(String[] args) {
       Imp obj = new Imp();
       obj.greet();
       obj.print();
   }
}
```

OUTPUT :

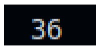


```
Hello from the interface!
Hello from implement class
```

2. Create an interface MathUtil with a static method square(int n) that returns the square of a number. Call this method from main().

```java
interface MathUtil{
   static int square(int n){
       return n*n;
   }
}

public class Main {
   public static void main(String[] args) {
       System.out.println(MathUtil.square(6));
   }
}
```

OUTPUT :

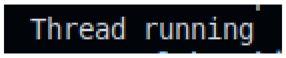


```
36
```

3. Write a lambda expression to create a Runnable that prints "Thread running" and executes it.

```java
public class Main {
   public static void main(String[] args) {
       Runnable r = () -> System.out.println("Thread running");
       Thread t = new Thread(r);
       t.start();
   }
}
```

OUTPUT :

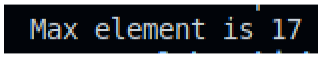


```
Thread running
```

4. Write a Java program to create a functional interface MaxFinder with a method max(int a, int b). Use a lambda expression to find and display the maximum of two numbers.

```java
interface MaxFinder{
   int max(int a,int b);
}
public class Main {
   public static void main(String[] args) {
       MaxFinder max_obj = (a,b) -> {
           if(a>b)
               return a;
           else
               return b;
       };
       System.out.println("Max element is " + max_obj.max(6,17));
   }
}
```

OUTPUT :

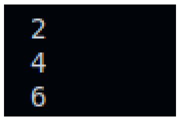


```
Max element is 17
```

5. Given a list of integers, use Stream API to filter and print only the odd numbers.

```java
import java.util.*;

public class Main {
   public static void main(String[] args) {
       List<Integer> list = new ArrayList<>(List.of(1,2,3,4,5,6));
       list.stream()
           .filter(n->n%2==0)
           .forEach(System.out::println);
   }
}
```

OUTPUT :

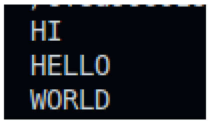


```
2
4
6
```

6. Given a list of strings, use Stream API to convert each string to uppercase and print the result.

```java
import java.util.*;

public class Main {
   public static void main(String[] args) {
 List<String> list = new ArrayList<>(List.of("hi","hello","world"));
       list.stream()
           .map(world->world.toUpperCase())
           .forEach(System.out::println);
   }
}
```

OUTPUT :

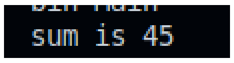


```
HI
HELLO
WORLD
```

7. Given a list of integers, use Stream API reduce() to find the sum of all elements.

```java
import java.util.*;

public class Main {
   public static void main(String[] args) {
                                 List<Integer>     list     =     new
ArrayList<>(List.of(1,2,3,4,5,6,7,8,9));
       int sum = list.stream()
                   .reduce(0,(a,b) -> a+b);
       System.out.println("sum is " + sum);
   }
}
```

OUTPUT :

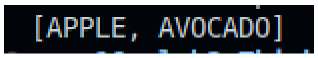


```
sum is 45
```

8. Given a list of names, use Stream API to filter names starting with letter 'A' and collect them into a new list.

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

public class Main {
   public static void main(String[] args) {
                                  List<String>     list     =     new
ArrayList<>(List.of("APPLE","ORANGE","BANANA","AVOCADO"));
       List<String> newlist = list.stream()
                               .filter(s -> s.charAt(0)=='A')
                               .collect(Collectors.toList());
       System.err.println(newlist);
   }
}
```

OUTPUT :



```
[APPLE, AVOCADO]
```

9. Create an Optional<String> with a value. If the value is present, print it using ifPresent().

```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Optional<String> op = Optional.ofNullable("java");
        op.ifPresent(s -> System.out.println(s));
    }
}
```

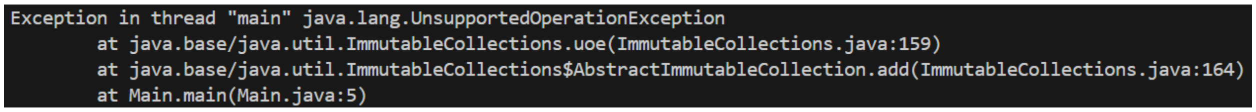
OUTPUT :

```
java
```

10. Create an immutable list using List.of() and attempt to add a new element. What is the outcome?

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        List<Integer> list = List.of(1,2,3,4,5,6);
        list.add(9);
    }
}
```

OUTCOME :

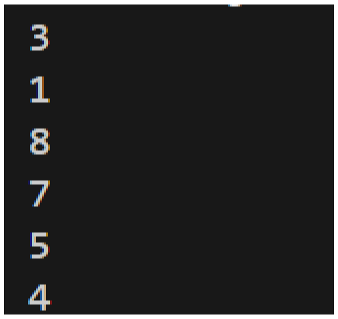


```
Exception in thread "main" java.lang.UnsupportedOperationException
        at java.base/java.util.ImmutableCollections.uoe(ImmutableCollections.java:159)
        at java.base/java.util.ImmutableCollections$AbstractImmutableCollection.add(ImmutableCollections.java:164)
        at Main.main(Main.java:5)
```

11. Create an immutable set using Set.of() and print all elements using forEach() with a method reference.

```java
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Set<Integer> set = Set.of(1,5,3,4,7,8);
        set.forEach(System.out::println);
    }
}
```

OUTPUT :



```
3
1
8
7
5
4
```

**Program 2:**

Design a Java program to process student results in a university portal system.

Each student record contains: name, marks, passed status

The system should:

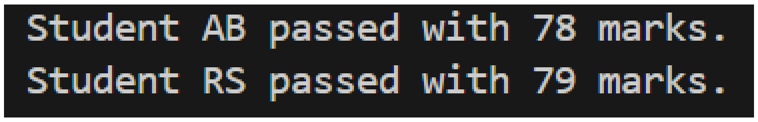
1. Filter students who have passed and scored more than 60 marks.
2. Convert each filtered student into a formatted message.
3. Display the messages on the console.

Tasks to Perform:

- Create a Student class with required fields.
- Store at least 5 student objects in a list.
- Use:
    - Predicate<Student> to filter students
    - Function<Student, String> to format output
    - Consumer<String> to display results
- Use lambda expressions (no traditional filtering loops).
- Use method reference for printing (if applicable).

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

class Student{
    private String name;
    private int marks;
    private boolean passed_status;

    public Student(String name,int marks,boolean passed_status) {
        this.name = name;
        this.marks = marks;
        this.passed_status = passed_status;
    }

    public String getName() { return name; }
    public int getMarks() { return marks; }
    public boolean isPassed() { return passed_status; }
}

public class Main {
    public static void main(String[] args) {
        List<Student> list = new ArrayList<>();
        list.add(new Student("AB",78,true));
        list.add(new Student("CD",58,true));
        list.add(new Student("PQ",33,false));
        list.add(new Student("RS",79,true));
        list.add(new Student("XY",22,false));

        Predicate<Student> filter_condition =
                       s -> s.isPassed() && s.getMarks() > 60;
        Function<Student, String> formatter =
                       s -> "Student " + s.getName() +
                       " passed with " + s.getMarks() + " marks.";
        Consumer<String> display = System.out::println;
        list.stream()
                .filter(filter_condition)
                .map(formatter)
                .forEach(display);
    }
}
```

OUTPUT :



```
Student AB passed with 78 marks.
Student RS passed with 79 marks.
```