Assignment - Beyond Java 8 Features - 1

Q1)Use iterator stream method to generate a stream

```java
package Beyond_Java_8_1.Q1;


import java.util.stream.Collectors;
import java.util.stream.IntStream;

public class Q1 {
    public static void main(String[] args) {


        System.out.println("\niterate() ");

        IntStream.iterate( seed: 0,  int i->i<10,  int i->i + 2).forEach(System.out::println);
    }
}
```

```
/usr/lib/jvm/java-1.21.0-openjdk-amd64/bin/java -javaagent:/home/akash/

iterate()
0
2
4
6
8

Process finished with exit code 0
```

Q2)Convert an Optional type into Stream

```
/usr/lib/jvm/java-1.21.0-openjdk-amd64/bin/java -javaagent:/home/akash/
HELLO

Process finished with exit code 0
```

```java
package Beyond_Java_8_1.Q2;

import java.util.Optional;

public class Q2 {
    public static void main(String[] args) {
        Optional<String> optional = Optional.of( value: "hello");

        optional.stream().map(String::toUpperCase).forEach(System.out::println);
    }
}
```

Q3)Use Of method to create List, Set and Map

```java
package Beyond_Java_8_1.Q3;

import java.util.ArrayList;
import java.util.List;
import java.util.Map;
import java.util.Set;

public class Q3 {
    public static void main(String[] args) {

        List<Integer> list = List.of(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
        Set<Integer> set = Set.of(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
        Map<Integer, String> map = Map.of( k1: 1,  v1: "Hello",  k2: 2,  v2: "World");

        System.out.println("List is : "+list);
        System.out.println("Set is : "+set);
        System.out.println("Map is : "+map);
    }
}

```

```
/usr/lib/jvm/java-1.21.0-openjdk-amd64/bin/java -javaagent:/home/akash/Downloads/idea-IU-251.26094.121/lib/idea_rt.jar=352
List is : [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Set is : [5, 6, 7, 8, 9, 10, 1, 2, 3, 4]
Map is : {1=Hello, 2=World}

Process finished with exit code 0
```

Q4)Create Unmodifiable List from a Steam

```java
package Beyond_Java_8_1.Q4;

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


public class Q4 {
    public static void main(String[] args) {
        List<Integer> list = Stream.of( ...values: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10).collect(Collectors.toUnmodifiableList());
        System.out.println("Immutable Copy of list : "+list);



    }
}
```

```
/usr/lib/jvm/java-1.21.0-openjdk-amd64/bin/java -javaagent:/home/akash/Downloads/idea-IU-251.26094.121/lib/idea_rt.j
Immutable Copy of list : [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

Process finished with exit code 0
```

Q5)Demonstrate the use of :-

- repeat()
- strip()
- trim()
- isBlank()
- indent()
- transform()
- stripIndent()
- translateEscapes()
- tripleQuotes
- formatted()

```java
package Beyond_Java_8_1.Q5;

class A{  2 usages
    void Repeat(String str){  1 usage
        System.out.println("repeat() : " + str.repeat( count: 3));
    }
    void Strip(String s){  1 usage
        System.out.println("Using strip() : |" + s.strip() + "|");
    }
    void trim(String s){  1 usage
        System.out.println("Using trim()  : |" + s.trim() + "|");
    }
    void IsBlank(String s){  1 usage
        System.out.println("Using isBlank() : " + s.isBlank() );
    }
    void Indent(String s,int n){  1 usage
        System.out.println("Using indent() : "+s.indent(n));
    }
    void TransForm(String s){  1 usage
        s = s.transform( String string -> new StringBuilder(string)
                .reverse().toString());
        System.out.println("Using transForm() : "+s);
    }
    void StripIndent(String s){  1 usage
        System.out.println("Using stripIndent() : " + s.stripIndent());
    }
    void TranslateEscape(String s){  1 usage
        System.out.println("Using translateEscape() : "+s.translateEscapes());
    }
    void TripleQuotes(){  1 usage
      String json = """
              {
              name:Akash Kumar,
              Age: 23
              }
              """;

      System.out.println(json);
```

```java
class A{  2 usages
    void TripleQuotes(){  1 usage
    }

@   void Formatter(String s){  1 usage

        System.out.println("Using Formatter() : "+s.formatted( ...args: "12"));
    }

}
public class Q5 {
    public static void main(String[] args) {
        A obj = new A();
        obj.Repeat( str: "hello  ");
        System.out.println("\n");
        obj.Strip( s: "\u2002Hello\u2002");
        System.out.println("\n");
        obj.trim( s: "\u2002Hello\u2002");
        System.out.println("\n");
        obj.IsBlank( s: "");
        System.out.println("\n");
        obj.Indent( s: "Hello World",  n: 15);
        System.out.println("\n");
        obj.TransForm( s: "Hello World");
        System.out.println("\n");
        obj.StripIndent( s: """
                Line 1
                Line 2
                Line 3
                ;
            """);
        System.out.println("\n");
        obj.TranslateEscape( s: "\"Hello\\nWorld\"");
        System.out.println("\n");
        obj.TripleQuotes();
        System.out.println("\n");
        obj.Formatter( s: "Java %s");
```

```
/usr/lib/jvm/java-1.21.0-openjdk-amd64/bin/java -javaagent:/home/akash/Downloads/idea-IU-251.26094.121/lib/idea_rt.ja
repeat() : hello  hello  hello


Using strip() : |Hello|


Using trim()  : | Hello |


Using isBlank() : true


Using indent() :                Hello World



Using transForm() : dlroW olleH


Using stripIndent() :         Line 1
        Line 2
        Line 3
        ;



Using translateEscape() : "Hello
World"
```

```
Using translateEscape() : "Hello
World"


{
name:Akash Kumar,
Age: 23
}



Using Formatter() : Java 12
```

Q6)You are tasked with writing a processOrderStatus method that takes an OrderStatus enum as input and returns a descriptive string based on the order status.

Here's the OrderStatus enum:
public enum OrderStatus {
PENDING,
PROCESSING,
SHIPPED,
DELIVERED,
CANCELLED,
REFUNDED }
Your processOrderStatus method should adhere to the following rules:
For PENDING orders, return: "Order is awaiting confirmation."
For PROCESSING orders, return: "Order is being prepared."
For SHIPPED orders, return: "Order has been dispatched."
For DELIVERED orders, return: "Order has been successfully delivered."
For CANCELLED orders, return: "Order has been canceled."
For REFUNDED orders, return: "Refund has been issued for the order."
Use a single switch expression to achieve this.
Enhancements: yield Keyword: If your logic requires more complex processing within a case, demonstrate the use of the yield keyword to return a value from the switch expression.

```java
package Beyond_Java_8_1.Q6;

public class Q6 {
    enum OrderStatus {  3 usages
        PENDING,  2 usages
        PROCESSING,  1 usage
        SHIPPED,  1 usage
        DELIVERED,  1 usage
        CANCELLED,  1 usage
        REFUNDED  1 usage
    }

    public static String processOrderStatus(OrderStatus s) {  1 usage
        return switch (s) {
            case PENDING->"Order is awaiting confirmation.";
            case PROCESSING->"Order is being prepared.";
            case SHIPPED->"Order has been dispatched." ;
            case DELIVERED-> "Order has been successfully delivered.";
            case CANCELLED->"Order has been canceled.";
            case REFUNDED-> {
                String m = "Refund has been issued for the order.";
                yield m;
            }
            default->"Something Went Wrong";
        };
    }

    public static void main(String[] args) {

      OrderStatus s = OrderStatus.PENDING;
      String ans = processOrderStatus(s);

      System.out.println(ans);
    }
}
```

```
/usr/lib/jvm/java-1.21.0-openjdk-amd64/bin/java -javaagent:/home/akash/Downloads/idea-IU-251.26094.121/lib/idea_rt.jar=34325 -Dfile.encodi
Order is awaiting confirmation.

Process finished with exit code 0
```