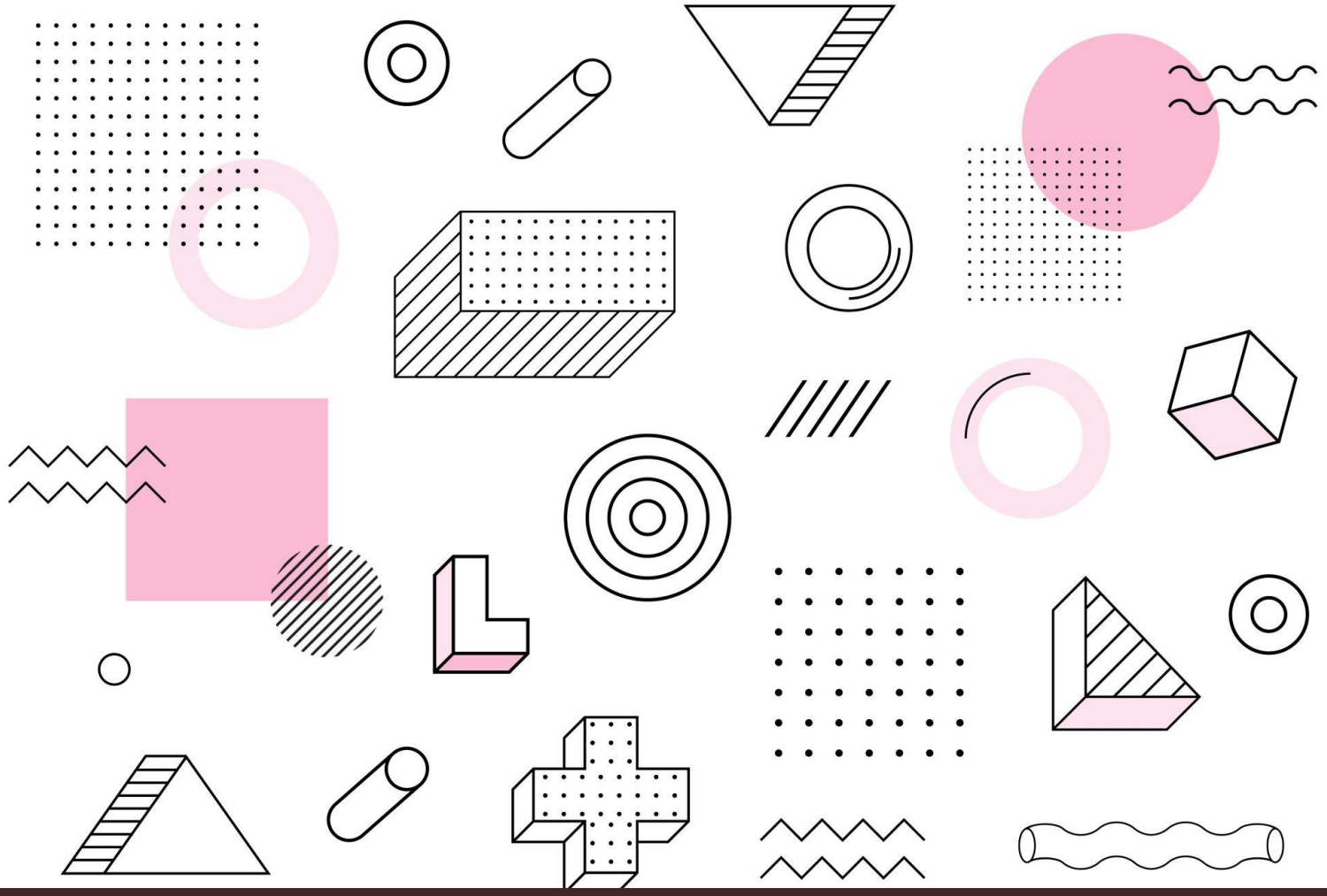


1학기 고급자바 실습

week 7-1(수업 8주차 실습)

김민진(18)

김지희(18)

문의 메일 : genie02166@duksung.ac.kr

Part2. 람다식 – 함수형 인터페이스

1) Funtion<T,R> 인터페이스

2) BiFuntion<T,U,R> 인터페이스

3) Predicate 그룹 인터페이스

4) Supplier 그룹 인터페이스

5) Consumer 그룹 인터페이스

6) Operator 그룹 인터페이스

1) Function<T,R> 인터페이스

FunctionTest.java

```java
package week7_1;

import java.util.function.Function;

public class FunctionTest {
    public static void main(String[] args) {
        Function<Integer,String> M_fun = (i) -> Integer.toString(i);
        System.out.println(M_fun.apply(100).length());
        System.out.println(M_fun.apply(1000).length());
    }
}
```

실행 결과

Console

<terminated>

3

4

2) BiFunction<T,U,R> 인터페이스

FunctionTest2.java

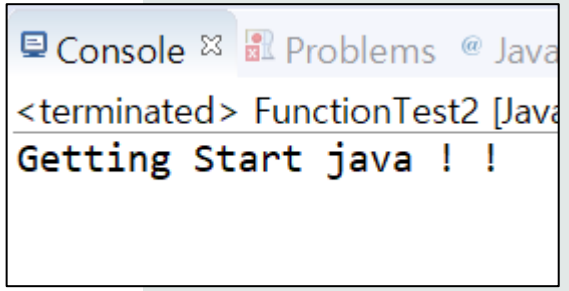
```java
package week7_1;

import java.util.function.BiFunction;
import java.util.function.Function;

public class FunctionTest2 {
    public static void main(String[] args) {
        BiFunction<String,String,String> bi = (x, y) -> {
            return x + y;
        };

        Function<String,String> f = x -> x + " !";
        System.out.println(bi.andThen(f).andThen(f).apply("Getting Start", " java"));
    }
}
```

실행 결과



3) Predicate 그룹 인터페이스

PredicateTest.java

```java
package week7_1;

import java.util.function.IntPredicate;
import java.util.function.Predicate;

public class PredicateTest {
    public static void main(String[] args) {
        Predicate<String> i = (s) -> s.length() > 10;
        System.out.println(i.test("getting start java"));

        IntPredicate p1 = n -> (n % 3) == 0;
        IntPredicate p2 = n -> (n % 5) == 0;

        IntPredicate p_res = p1.and(p2);

        System.out.println(p_res.test(3));
        System.out.println(p_res.test(4));

        IntPredicate p_res02 = p1.or(p2);
        System.out.println(p_res02.test(5));
        System.out.println(p_res02.test(15));

        Predicate<String> str = Predicate.isEqual("Dominica_kim");
        System.out.println(str.test("Dominica_kim"));
    }
}
```

실행 결과

Console

<terminated>

```
true
false
false
true
true
true
```

4) Supplier 그룹 인터페이스

supplier.java

```java
package week7_1;

import java.util.function.DoubleSupplier;
import java.util.function.IntSupplier;
import java.util.function.Supplier;

public class supplier {
    public static void main(String[] args) {
        Supplier<String> s = () -> "abc";
        System.out.println(s.get()); // abc 리턴

        IntSupplier y = () -> 100;
        System.out.println(y.getAsInt()); //100 리턴

        DoubleSupplier d = () -> 90.7;
        System.out.println(d.getAsDouble()); //90.7 리턴
    }
}
```

실행 결과

Problems @ Javadoc Decl

<terminated> supplier [Java Appl

```
abc
100
90.7
```

5) Consumer 그룹 인터페이스

consumer.java

```java
package week7_1;
import java.util.function.Consumer;

public class consumer {
    public static void main(String[] args) {
        Consumer<String> c = s -> System.out.println(s);
        c.accept("abc");

        Consumer<String> c1 = s -> System.out.println("c1 = " + s);
        Consumer<String> c2 = s -> System.out.println("c2 = " + s);
        Consumer<String> c_res = c1.andThen(c2); //c1, c2 순서대로 실행
        c_res.accept("abc");
    }
}
```

실행 결과

Problems @ Javadoc

<terminated> consumer

```
abc
c1 = abc
c2 = abc
```

6) Operator 그룹 인터페이스

*operator.java

```java
package week7_1;

import java.util.function.BinaryOperator;
import java.util.function.IntUnaryOperator;
import java.util.function.UnaryOperator;

public class operator {
    public static void main(String[] args) {
        IntUnaryOperator op1 = n -> n*10; //단항 연산 실행
        IntUnaryOperator op2 = n -> n+1; //단항 연산 실행
        IntUnaryOperator op = op1.compose(op2);
        System.out.println(op.applyAsInt(2));

        UnaryOperator<String> op02 = UnaryOperator.identity();
        System.out.println(op02.apply("aaa"));

        BinaryOperator<String> op03 = BinaryOperator.minBy((s1,s2)->s1.compareTo(s2));
        String s_res1 = op03.apply("aaa", "ddd");
        System.out.println("aaa, ddd 중 작은 값 리턴 : " + s_res1);

        BinaryOperator<String> op04 = BinaryOperator.maxBy((s1,s2)->s1.compareTo(s2));
        String s_res2 = op04.apply("aaa", "ddd");
        System.out.println("aaa, ddd 중 큰 값 리턴 : " + s_res2);
    }
}
```

실행 결과

Problems @ Javadoc Declaration

<terminated> operator [Java Applicatior

```
30
aaa
aaa, ddd 중 작은 값 리턴 : aaa
aaa, ddd 중 큰 값 리턴 : ddd
```

출석 과제 (5/10 월 오후 11:55 마감)

Q. 람다식에 대한 설명으로 틀린 것은 무엇입니까?

1) 람다식은 함수적 인터페이스의 익명 구현 객체를 생성한다.

2) 매개 변수가 없을 경우 ( ) -> {…} 형태로 작성한다.

3) (x,y) -> {return x+y;}는 (x,y) -> x+y로 바꿀 수 있다.

4) @FunctionalInterface가 기술된 인터페이스만 람다식으로 표현이 가능하다.

위의 기간까지 고급 자바 실습
DS-CLASS 에 제출하시기 바랍니다.