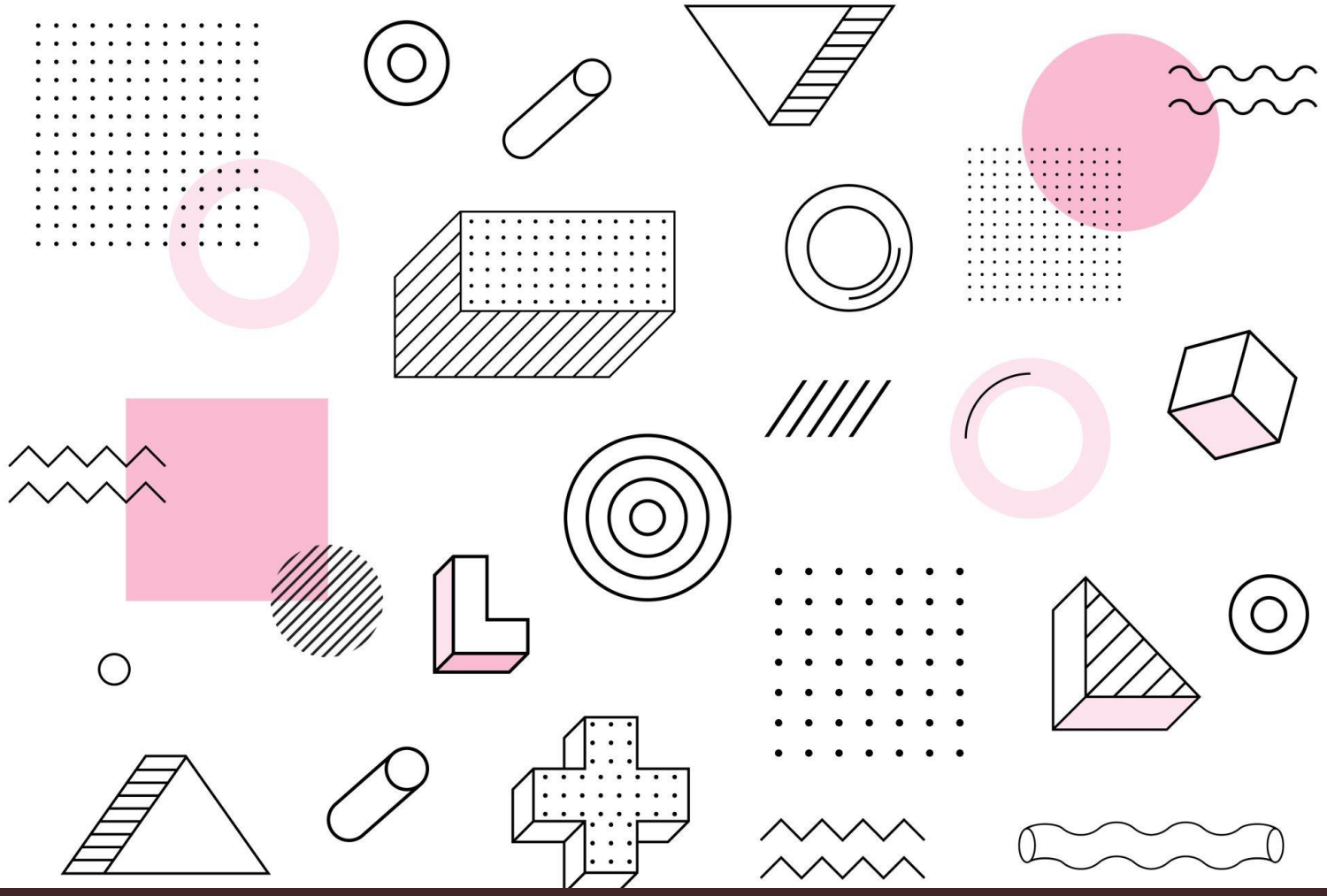

1학기 고급자바 실습

week 9-1 (수업 11주차 실습)

김민진(18)

김지희(18)

문의 메일 : genie02166@duksung.ac.kr

Part1. 멀티 스레드

1) 스레드 종료

- stop 플래그
- interrupt()

2) 스레드 상태 제어

- yield()
- join()

3) 문제 풀어보기

1) 스레드 종료
- stop 플래그

StopFlagExample.java

```java
package  week9_1;

public class StopFlagExample {
    public static void main(String[] args)  {
        PrintThread1 printThread = new PrintThread1();
        printThread.start();

        try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
        }

        printThread.setStop(true);
    }
}
```

PrintThread1.java

```java
package week9_1;

public class PrintThread1 extends Thread {
    private boolean stop;

    public void setStop(boolean stop) {
      this.stop = stop;
    }

    public void run() {
        while(!stop) {
            System.out.println("실행 중");
        }
        System.out.println("자원 정리");
        System.out.println("실행 종료");
    }
}
```

실행 결과

```
실행 중
실행 중
실행 중
자원 정리
실행 종료
```

1) 스레드 종료
- interrupt()

PrintThread2.java | InterruptExample.java

```java
package week9_1;

public class PrintThread2 extends Thread {
    public void run() {

        try {
            while(true) {
                System.out.println("실행 중");
                Thread.sleep(1);
            }
        } catch(InterruptedException e) {
        }

        //how2
        /*
        while(true) {
            System.out.println("실행 중");
            if(Thread.interrupted()) {
                break;
            }
        }
        */

        System.out.println("자원 정리");
        System.out.println("실행 종료");
    }
}
```

PrintThread2.java | InterruptExample.java

```java
package week9_1;

public class InterruptExample {
    public static void main(String[] args)  {
        Thread thread = new PrintThread2();
        thread.start();

        try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
        }

        thread.interrupt();
    }
}
```

실행 결과

```
실행 중
실행 중
실행 중
자원 정리
실행 종료
```

2) 스레드 상태 제어
- yield()

Thread01.java | Thread02.java | yieldTest.java

```java
package week9_2;

public class Thread01 extends Thread{
    public void run() {
        for(int i=0; i<5; i++) {
            System.out.println("시작합니다 "+i);
            if(i ==1)
                Thread.yield();
        }
    }
}
```

Thread01.java | Thread02.java | yieldTest.java

```java
package week9_2;

public class Thread02 extends Thread{
    public void run() {
        for(int i=0; i<5; i++) {
            System.out.println("java study!");
        }
    }
}
```

Thread01.java | Thread02.java | yieldTest.java

```java
package week9_2;

public class yieldTest {
    public static void main(String[] args) {
        Thread01 th1 = new Thread01();
        Thread02 th2 = new Thread02();
        th1.start();
        th2.start();
    }
}
```

실행 결과

```
시작합니다 0
시작합니다 1
java study !
java study !
java study !
java study !
java study !
시작합니다 2
시작합니다 3
시작합니다 4
```

2) 스레드 상태 제어
- join()

joinTest.java

```java
package week9_2;

public class joinTest extends Thread{
    private int first, last;
    public int sum;

    public joinTest(int first, int last) {
        this.first = first;
        this.last = last;
    }

    public void run() {
        for(int i = first; i<= last; i++) {
            sum = sum+i;
        }
    }

    public static void main(String[] args) {
        joinTest th1 = new joinTest(1,5);
        joinTest th2 = new joinTest(6,10);

        th1.start();
        th2.start();

        try {
            th1.join();
            th2.join();
        } catch(InterruptedException e) {
            e.printStackTrace();
        }

        System.out.println("th1 sum : "+ th1.sum);
        System.out.println("th1 sum : "+ th2.sum);
        System.out.println("total sum : "+ (th1.sum+th2.sum));
    }
}
```

실행 결과

Problems @ Javadoc

<terminated> joinTest [Java

```
th1 sum : 15
th1 sum : 40
total sum : 55
```

3) 문제 풀어보기

Q1. 동영상과 음악을 재생하기 위해 두 가지 스레드를 실행하려고 합니다.

비어 있는 부분에 적당한 코드를 넣어 보세요.

ThreadExample.java | MovieThread.java | MusicRunnable.java

```
1  package week9_1;
2
3  public class MovieThread [          ] {
4      @Override
5      public void run() {
6          for(int i=0; i<3; i++) {
7              System.out.println("동영상을 재생합니다.");
8              try {
9                  Thread.sleep(1000);
10             } catch (InterruptedException e) {
11             }
12         }
13     }
14 }
```

ThreadExample.java | MovieThread.java | MusicRunnable.java

```
1  package week9_1;
2
3  public class MusicRunnable [          ] {
4      @Override
5      public void run() {
6          for(int i=0; i<3; i++) {
7              System.out.println("음악을 재생합니다.");
8              try {
9                  Thread.sleep(1000);
10             } catch (InterruptedException e) {
11             }
12         }
13     }
14 }
```

ThreadExample.java | MovieThread.java | MusicRunnable.java

```
1  package week9_1;
2
3  public class ThreadExample {
4      public static void main(String[] args) {
5
6          Thread thread1 = new Thread();
7          thread1.start();
8
9          Thread thread2 = new Thread([          ]);
10         thread2.start();
11     }
12 }
```

3) 문제 풀어보기

Q2. 메인 스레드에서 1초 후 MovieThread의 interrupt() 메소드를 호출해서 MovieThread를 안전하게 종료하고 싶습니다. 비어 있는 부분에 적당한 코드를 작성해보세요.

ThreadExample.java | MovieThread.java

```java
package week9_1_;

public class ThreadExample {
    public static void main(String[] args) {
        Thread thread = new MovieThread();
        thread.start();

        try { Thread.sleep(1000); } catch (InterruptedException e) {}

        thread.interrupt();
    }
}
```

ThreadExample.java | MovieThread.java

```java
package week9_1_;

public class MovieThread extends Thread {
    @Override
    public void run() {
        while(true) {
            System.out.println("동영상을 재생합니다.");
            
            
            
            
        }
    }
}
```

출석 과제 (5/24 월 오후 11:55 마감)

Q. 스레드 상태 제어를 하는 메소드에 대한 설명 중 틀린 것은 무엇입니까?

1) yield() 메소드를 호출한 스레드는 동일한 우선순위나 높은 우선순위의 스레드에게 실행 기회를 양보하고 자신은 실행 대기 상태가 된다.

2) sleep() 메소드를 호출한 스레드는 주어진 시간 동안 일시 정지 상태가 된다.

3) stop() 메소드는 스레드를 즉시 종료시키기 때문에 스레드 안전성에 좋지 못하다.

4) join() 메소드를 호출한 스레드가 종료할 때까지 join() 메소드를 멤버로 가지는 스레드는 일시 정지 상태가 된다.

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