

11주차 실습 보고서



강의명	객체지향프로그래밍및실습
담당교수	류기열
학과 학년	소프트웨어학과 2학년
학번	202220209
작성자	이육준

2025.11.13.

□ 1

(가) local inner class

```
J AnonymousInnerClassTest.java 1 X
JAVA > 11 > J AnonymousInnerClassTest.java > TalkingClock
30  class Talkingclock
37      public void start(int interval, boolean beep)
46          // printing BEEP instead of making a sound
47          // if (beep) System.out.println("---BEEP---");
48
49          //
50          //
51          // var timer = new Timer(interval, listener);
52          // timer.start();
53
54          //
55          class Listener implements ActionListener{
56              public void actionPerformed(ActionEvent event){
57                  System.out.println("At the tone, the time is " + Instant.ofEpochMilli(event.getWhen()));
58                  if (beep) System.out.println(" ---BEEP---");
59              }
60          }
61          ActionListener listener = new Listener();
62          var timer = new Timer(interval, listener);
63          timer.start();
64
65      }
```

PROBLEMS 54 OUTPUT DEBUG CONSOLE TERMINAL PORTS

- lwj@lwj-code:~/workspace/JAVA/11\$ java AnonymousInnerClassTest
At the tone, the time is 2025-11-13T05:26:01.485Z
---BEEP---
At the tone, the time is 2025-11-13T05:26:02.484Z
---BEEP---
At the tone, the time is 2025-11-13T05:26:03.484Z
---BEEP---
At the tone, the time is 2025-11-13T05:26:04.485Z
---BEEP---
At the tone, the time is 2025-11-13T05:26:05.485Z
---BEEP---
At the tone, the time is 2025-11-13T05:26:06.485Z

(L) lambda exp

The screenshot shows a Java code editor with the following code:

```
J AnonymousInnerClassTest.java 1 X
JAVA > 11 > J AnonymousInnerClassTest.java > ...
30   class TalkingClock
37     public void start(int interval, boolean beep)
64
65     //L
66     ActionListener listener = (ActionEvent event) -> {
67       System.out.println("At the tone, the time is " + Instant.ofEpochMilli(event.getWhen()));
68       if (beep) System.out.println("BEEP---");
69     };
70     Timer t = new Timer(interval, listener);
71     t.start();
72   }
73 }
74 }
```

The code defines a `TalkingClock` class with a `start` method. The `start` method creates a `Timer` that triggers an `ActionListener`. The `ActionListener` prints the current time and a beep sound if specified.

Below the code editor, there is a terminal window showing the output of running the code:

```
PROBLEMS 54 OUTPUT DEBUG CONSOLE TERMINAL PORTS
● lwj@lwj-code:~/workspace/JAVA/11$ javac AnonymousInnerClassTest.java
● lwj@lwj-code:~/workspace/JAVA/11$ java AnonymousInnerClassTest
At the tone, the time is 2025-11-13T05:30:00.583Z
---BEEP---
At the tone, the time is 2025-11-13T05:30:01.582Z
---BEEP---
At the tone, the time is 2025-11-13T05:30:02.583Z
---BEEP---
At the tone, the time is 2025-11-13T05:30:03.583Z
---BEEP---
```

□ 2

The screenshot shows a Java code editor interface. At the top, there are two tabs: "AnonymousInnerClassTest.java 1" and "InnerClassTest.java 1 X". The "InnerClassTest.java 1 X" tab is active. Below the tabs, the file path is shown as "JAVA > 11 > J InnerClassTest.java > InnerClassTest > main(String[])". The code itself is as follows:

```
11 | /*
12  public class InnerClassTest
13  {
14      public static void main(String[] args) throws InterruptedException
15      {
16          var clock = new TalkingClock(interval: 1000, beep: false);
17          // clock.start();
18          TalkingClock.TimePrinter timePrinter = clock.new TimePrinter();
19          Timer timer = new Timer(delay: 1000, timePrinter);
20          timer.start();
21
22          Thread.sleep(millis: 10000); // main thread sleeps here for 10 seconds
23          // keep program running until the user selects "OK"
24          // JOptionPane.showMessageDialog(null, "Quit program?");
25          System.exit(status: 0);
26      }
27  }
```

Below the code editor, there is a navigation bar with tabs: PROBLEMS (53), OUTPUT, DEBUG CONSOLE, TERMINAL (underlined), and PORTS.

The terminal output window shows the following logs:

- lwj@ljwj-code:~/workspace/JAVA/11\$ javac InnerClassTest.java
- lwj@ljwj-code:~/workspace/JAVA/11\$ java InnerClassTest
- At the tone, the time is 2025-11-13T05:40:47.375Z
- At the tone, the time is 2025-11-13T05:40:48.373Z
- At the tone, the time is 2025-11-13T05:40:49.374Z

□ 3

(가)

```
● lwj@lwj-code:~/workspace/JAVA/11$ javac PollTest.java
● lwj@lwj-code:~/workspace/JAVA/11$ java PollTest
CEException in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 3 out of bounds for length 3
at PollTest.main(PollTest.java:8)
```

Array의 size를 넘는 index에 대한 접근을 처리하는 Exception. int[] data의 length는 4인데, int[] freq[3]으로 선언된 것에서 freq[3]에 대해 접근을 할 수 없는데, 코드의 반복문 과정에서는 freq[3]을 접근하려고 하니 Exception이 발생함.

(나)

```
1 public class PollTest
3     public static void main(String[] args)
5         int[] data = { 1, 3, 1, 2 };
6         int[] freq = new int[3]; // initialized by 0
7         for (int i = 0; i < data.length; i++) {
8             try{
9                 freq[data[i]]++;
10                System.out.print(s: "C");
11            }
12            catch(ArrayIndexOutOfBoundsException e){
13                System.out.println(x: "E");
14            }
15        }
16        System.out.println();
17        for (int i = 1; i < freq.length; i++) {
18            System.out.printf(format: "%d:%d%n", i, freq[i]);
19        }
```

PROBLEMS 53 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
● lwj@lwj-code:~/workspace/JAVA/11$ javac PollTest.java
● lwj@lwj-code:~/workspace/JAVA/11$ java PollTest
CE
CC
1:2
2:1
```

반복문은 총 4회 반복된다. 1번 시도에서는 freq[1]의 값을 1증가, 2번 시도에서는 freq[3]의 값을 1증가하는데 이 때 freq[3]은 접근할 수 없는 영역이므로 Exception이 발생하기에 try문에서 catch문으로 이동하여 sout("E"); 3번 시도에서는 freq[1]의 값을 1증가, 3번 시도에서는 freq[2]의 값을 1증가, 첫 번째 for문을 통해 freq[1] = 2, freq[2]=1이 되어 2번 째 반복문에서 1:2, 2:1이 출력됨.

□ 4

(가)

```
3  class FileInputStreamTest {
4      public static FileInputStream foo(String fileName) throws FileNotFoundException
5      {
6          System.out.println("foo: Started");
7          FileInputStream fis = new FileInputStream(fileName);
8          System.out.println("foo: Returned");
9          return fis;
10     }
11
12    public static void main(String args[])
13    {
14        FileInputStream fis = null;
15        String fileName = "foo.txt";
16
17        System.out.println("main: Started");
18        try{
19            fis = foo(fileName);
20        }
21        catch(FileNotFoundException e){
22            System.out.println("파일이 존재하지 않는다");
23        }
24
25        System.out.println("main: Ended");
26    }
27 }
```

PROBLEMS 52 OUTPUT DEBUG CONSOLE TERMINAL PORTS

- lwj@lwj-code:~/workspace/JAVA/11\$ javac FileInputStreamTest.java
- lwj@lwj-code:~/workspace/JAVA/11\$ java FileInputStreamTest
main: Started
foo: Started
파일이 존재하지 않는다
main: Ended

(다)

```
13  public static void main(String args[])
14  {
15      FileInputStream fis = null;
16      Scanner in = new Scanner(System.in);
17      while(true){
18          try{
19              String fileName = in.next();
20              fis = foo(fileName);
21              System.out.println("main: Started");
22              break;
23          }
24          catch(FileNotFoundException e){
25              System.out.println("파일이 존재하지 않음. 재입력 요망");
26          }
27      }
28  }
```

PROBLEMS 53 OUTPUT DEBUG CONSOLE TERMINAL PORTS

- lwj@ljw-code:~/workspace/JAVA/11\$ java FileInputStreamTest
goo.txt
foo: Started
파일이 존재하지 않음. 재입력 요망
hoo.txt
foo: Started
파일이 존재하지 않음. 재입력 요망
foo.txt
foo: Started
foo: Returned
main: Started
main: Ended

□ 코드 전문

AnonymousInnerClassTest.java

```
import java.awt.*;
import java.awt.event.*;
import java.time.*;
import javax.swing.*;

/**
 * This program demonstrates anonymous inner classes.
 * @version 1.12 2017-12-14
 * @author Cay Horstmann
 */
public class AnonymousInnerClassTest
{
    public static void main(String[] args) throws InterruptedException
    {
        var clock = new TalkingClock();
        clock.start(1000, true);

        Thread.sleep(10000); // main thread sleeps here for 10 seconds

        // keep program running until the user selects "OK"
        // JOptionPane.showMessageDialog(null, "Quit program?");
        System.exit(0);
    }
}

/**
 * A clock that prints the time in regular intervals.
 */
class TalkingClock
{
    /**
     * Starts the clock.
     * @param interval the interval between messages (in milliseconds)
     * @param beep true if the clock should beep
     */
    public void start(int interval, boolean beep)
    {
        // var listener = new ActionListener()
        // {
        //     public void actionPerformed(ActionEvent event)
        //     {
        //         System.out.println("At the tone, the time is "
        //             + Instant.ofEpochMilli(event.getWhen()));
    }
}
```

```

//           // if (beep) Toolkit.getDefaultToolkit().beep();
//           // printing BEEP instead of making a sound
//           if (beep) System.out.println("---BEEP---");

//       }
//   };
// var timer = new Timer(interval, listener);
// timer.start();

//JF
// class Listener implements ActionListener{
//     public void actionPerformed(ActionEvent event){
//         System.out.println("At the tone, the time is " +
Instant.ofEpochMilli(event.getWhen()));
//         if (beep) System.out.println("---BEEP---");
//     }
// }
// ActionListener listener = new Listener();
// var timer = new Timer(interval, listener);
// timer.start();

//LF
ActionListener listener = (ActionEvent event) -> {
    System.out.println("At the tone, the time is " +
Instant.ofEpochMilli(event.getWhen()));
    if (beep) System.out.println("---BEEP---");
};
Timer t = new Timer(interval, listener);
t.start();
}
}

```

InnerClassTest.java

```

import java.awt.*;
import java.awt.event.*;
import java.time.*;
import javax.swing.*;

/**
 * This program demonstrates the use of inner classes.
 * @version 1.11 2017-12-14
 * @author Cay Horstmann
 */
public class InnerClassTest
{

```

```
public static void main(String[] args) throws InterruptedException
{
    var clock = new TalkingClock(1000, false);
    // clock.start();
    TalkingClock.TimePrinter timePrinter = clock.new TimePrinter();
    Timer timer = new Timer(1000, timePrinter);
    timer.start();

    Thread.sleep(10000); // main thread sleeps here for 10 seconds
    // keep program running until the user selects "OK"
    // JOptionPane.showMessageDialog(null, "Quit program?");
    System.exit(0);
}

/**
 * A clock that prints the time in regular intervals.
 */
class TalkingClock
{
    private int interval;
    private boolean beep;

    /**
     * Constructs a talking clock
     * @param interval the interval between messages (in milliseconds)
     * @param beep true if the clock should beep
     */
    public TalkingClock(int interval, boolean beep)
    {
        this.interval = interval;
        this.beep = beep;
    }

    /**
     * Starts the clock.
     */
    public void start()
    {
        var listener = new TimePrinter();
        var timer = new Timer(interval, listener);
        timer.start();
    }

    public class TimePrinter implements ActionListener
    {
        public void actionPerformed(ActionEvent event)
        {

```

```

        System.out.println("At the tone, the time is "
            + Instant.ofEpochMilli(event.getWhen()));
        // if (beep) Toolkit.getDefaultToolkit().beep();
        // printing BEEP instead of making a sound
        if (beep) System.out.println("---BEEP---");
    }
}
}

```

PollTest.java

```

public class PollTest
{
    public static void main(String[] args)
    {
        int[] data = { 1, 3, 1, 2 };
        int[] freq = new int[3]; // initialized by 0
        for (int i = 0; i < data.length; i++) {
            try{
                ++freq[data[i]];
                System.out.print("C");
            }
            catch(ArrayIndexOutOfBoundsException e){
                System.out.println("E");
            }
        }
        System.out.println();
        for (int i = 1; i < freq.length; i++) {
            System.out.printf("%d:%d%n", i, freq[i]);
        }
    }
}

```

FileInputTest.java

```

import java.io.*;
import java.util.Scanner;

class FileInputTest {
    public static FileInputStream foo(String fileName) throws
FileNotFoundException
    {
        System.out.println("foo: Started");
        FileInputStream fis = new FileInputStream(fileName);
    }
}

```

```
System.out.println("foo: Returned");
return fis;
}

public static void main(String args[])
{
    FileInputStream fis = null;
    Scanner in = new Scanner(System.in);
    while(true){
        try{
            String fileName = in.next();
            fis = foo(fileName);
            System.out.println("main: Started");
            break;
        }
        catch(FileNotFoundException e){
            System.out.println("파일이 존재하지 않음. 재입력 요망");
        }
    }

    System.out.println("main: Ended");
}
}
```