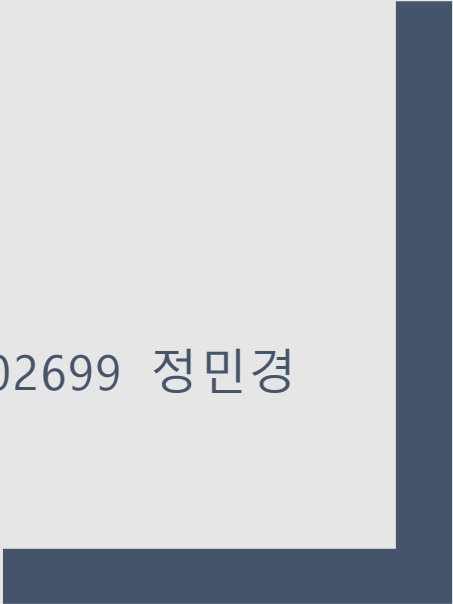




2023\_compiler

# JAVA BYTECODE 다뤄보기

202102699 정민경



# 1. JVM assembly 코드(Test.j)를 jasmin으로 .class 파일로 변환 후 실행

- Test.j -> Test.class 로 변환

```
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/Downloads/jasmin-2.4
ls
Readme.txt      build.sh      changes.txt   examples      jasmin.jar    license-ant.txt  makefile
build.bat      build.xml     docs          html2x        lib           license-jasmin.txt src
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/Downloads/jasmin-2.4
ls | grep "Test"
(base) x jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/Downloads/jasmin-2.4
java -jar jasmin.jar ../Test.j
Generated: Test.class
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/Downloads/jasmin-2.4
ls
Readme.txt      build.bat      build.xml     docs          html2x        lib           license-jasmin.txt src
Test.class      build.sh      changes.txt   examples      jasmin.jar    license-ant.txt  makefile
Test.class
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/Downloads/jasmin-2.4
ls | grep "Test"
Test.class
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/Downloads/jasmin-2.4
```

- Jasmin 을 download 받아서 압축 해제 후 jasmin.jar 파일을 사용해 Test.j 를 compile 해봄. (.class 파일로 변환)  
변환 전과 후에 ls 명령어를 통해 Test.class 파일이 잘 생성되었는지 확인함.

- Test.class 실행

```
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/Downloads/jasmin-2.4
ls
Readme.txt      build.bat      build.xml     docs          html2x        lib           license-jasmin.txt src
Test.class      build.sh      changes.txt   examples      jasmin.jar    license-ant.txt  makefile
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/Downloads/jasmin-2.4
java Test
34
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/Downloads/jasmin-2.4
```

- 앞선 과정에서 만든 Test.class 파일을 java 로 실행했을 때 "34" 라는 결과가 출력 되는 것을 확인함.

## 2. 수기로 작성한 Java 코드를 javac로 컴파일, javap로 disassemble해서 결과 확인

- Javac 로 compile 후 생성된 Test.class 실행결과

```
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/CNU/ITC_23/hw10/week10/src 1/2 main ±+ ls
Brewfile Main.java Test.java
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/CNU/ITC_23/hw10/week10/src 1/2 main ±+ javac Test.java
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/CNU/ITC_23/hw10/week10/src 1/2 main ±+ ls
Brewfile Main.java Test.class Test.java
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/CNU/ITC_23/hw10/week10/src 1/2 main ±+ java Test
34
```

- Test.j 코드를 보고 수기로 작성한 Test.java 파일을 javac 를 사용해 Test.class 로 컴파일 후 실행해 "34" 라는 결과가 출력되는 것을 확인.

- Javac 로 compile 한 Test.class 파일을 javap 로 disassemble 한 결과 (-c option)

```
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/CNU/ITC_23/hw10/week10/src 1/2 main ±+ ls
Main.java Test.java
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/CNU/ITC_23/hw10/week10/src 1/2 main ±+ javac Test.java
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/CNU/ITC_23/hw10/week10/src 1/2 main ±+ ls
Main.java Test.class Test.java
(base) jeongmingyeong@jeongmingyeong-ui-MacBookAir ~/CNU/ITC_23/hw10/week10/src 1/2 main ±+ javap -c Test.class
Compiled from "Test.java"
public class Test {
    public Test();
        Code:
            0: aload_0
            1: invokespecial #1                  // Method java/lang/Object."<init>":()V
            4: return

    public static int add(int, int);
        Code:
            0: iload_0
            1: iload_1
            2: iadd
            3: istore_2
            4: iload_2
            5: ireturn

    public static void main(java.lang.String[]);
        Code:
            0: bipush       33
            2: istore_1
            3: getstatic    #7                  // Field java/lang/System.out:Ljava/io/PrintStream;
            6: iload_1
            7: iconst_1
            8: invokestatic #13                 // Method add:(II)I
            11: invokevirtual #19                // Method java/io/PrintStream.println:(I)V
            14: return
}
```

- -c option 을 주어 disassemble 했을 때의 결과 확인. 이 결과는 제공된 Test.j 와 매우 흡사한 결과 도출.

- Javac 로 compile 한 Test.class 파일을 javap 로 disassemble 결과 (-verbose option)

```
Classfile /Users/jeongmingyeong/CNU/ITC_23/hw10/week10/src/Test.class
  Last modified 2023. 11. 3.; size 465 bytes
  SHA-256 checksum db0a7285519b4a5b656efa7ad102b4ad9234eba4ad6765b941f626af892d881b
  Compiled from "Test.java"
public class Test
  minor version: 0
  major version: 61
  flags: (0x0021) ACC_PUBLIC, ACC_SUPER
  this_class: #14                      // Test
  super_class: #2                      // java/lang/Object
  interfaces: 0, fields: 0, methods: 3, attributes: 1
Constant pool:
 #1 = Methodref      #2.#3            // java/lang/Object."<init>":()V
 #2 = Class          #4                // java/lang/Object
 #3 = NameAndType    #5:#6            // "<init>":()V
 #4 = Utf8           java/lang/Object
 #5 = Utf8           <init>
 #6 = Utf8           ()V
 #7 = Fieldref       #8.#9            // java/lang/System.out:Ljava/io/PrintStream;
 #8 = Class          #10               // java/lang/System
 #9 = NameAndType    #11:#12          // out:Ljava/io/PrintStream;
#10 = Utf8           java/lang/System
#11 = Utf8           out
#12 = Utf8           Ljava/io/PrintStream;
#13 = Methodref     #14.#15          // Test.add:(II)I
#14 = Class         #16               // Test
#15 = NameAndType   #17:#18          // add:(II)I
#16 = Utf8          Test
#17 = Utf8          add
#18 = Utf8          (II)I
#19 = Methodref     #20.#21          // java/io/PrintStream.println:(I)V
#20 = Class         #22               // java/io/PrintStream
#21 = NameAndType   #23:#24          // println:(I)V
#22 = Utf8          java/io/PrintStream
#23 = Utf8          println
#24 = Utf8          (I)V
#25 = Utf8          Code
#26 = Utf8          LineNumberTable
#27 = Utf8          main
#28 = Utf8          ([Ljava/lang/String;)V
```

```

#28 = Utf8      ([Ljava/lang/String;)V
#29 = Utf8      SourceFile
#30 = Utf8      Test.java
{
  public Test();
    descriptor: ()V
    flags: (0x0001) ACC_PUBLIC
    Code:
      stack=1, locals=1, args_size=1
        0: aload_0
        1: invokespecial #1          // Method java/lang/Object."<init>":()V
        4: return
    LineNumberTable:
      line 3: 0
      line 4: 4

  public static int add(int, int);
    descriptor: (II)I
    flags: (0x0009) ACC_PUBLIC, ACC_STATIC
    Code:
      stack=2, locals=3, args_size=2
        0: iload_0
        1: iload_1
        2: iadd
        3: istore_2
        4: iload_2
        5: ireturn
    LineNumberTable:
      line 7: 0
      line 8: 4

  public static void main(java.lang.String[]);
    descriptor: ([Ljava/lang/String;)V
    flags: (0x0009) ACC_PUBLIC, ACC_STATIC
    Code:
      stack=3, locals=2, args_size=1
        0: bipush      33
        2: istore_1
        3: getstatic   #7          // Field java/lang/System.out:Ljava/io/PrintStream;
        6: iload_1
        7: iconst_1
        8: invokestatic #13         // Method add:(II)I
       11: invokevirtual #19        // Method java/io/PrintStream.println:(I)V
       14: return
    LineNumberTable:
      line 12: 0
      line 13: 3
      line 14: 14
}
SourceFile: "Test.java"

```

- -v (verbose) option 을 주어 disassemble 을 한 결과로 Test.j 와 비슷한 코드 뿐만 아니라 descriptor 나 flag 와 같은 조금 더 자세한 정보까지 함께 출력되는 것을 확인할 수 있다.