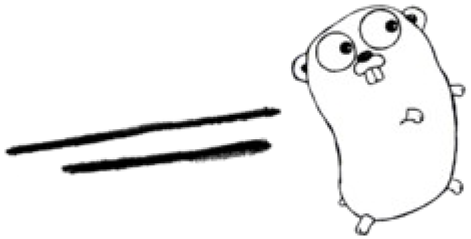
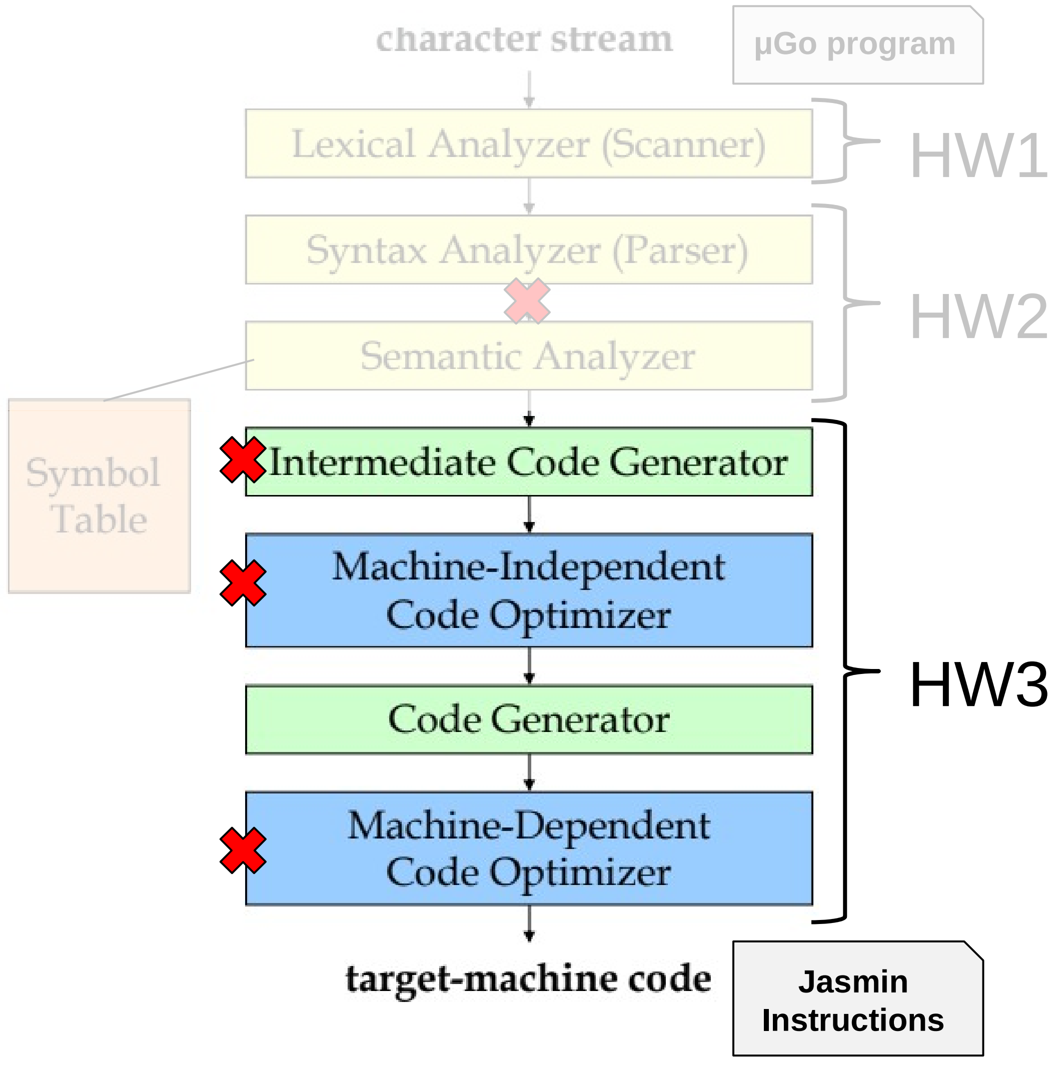
Compiler Construction

Programming Assignment 3

Generate Java Assembly Code for µGo

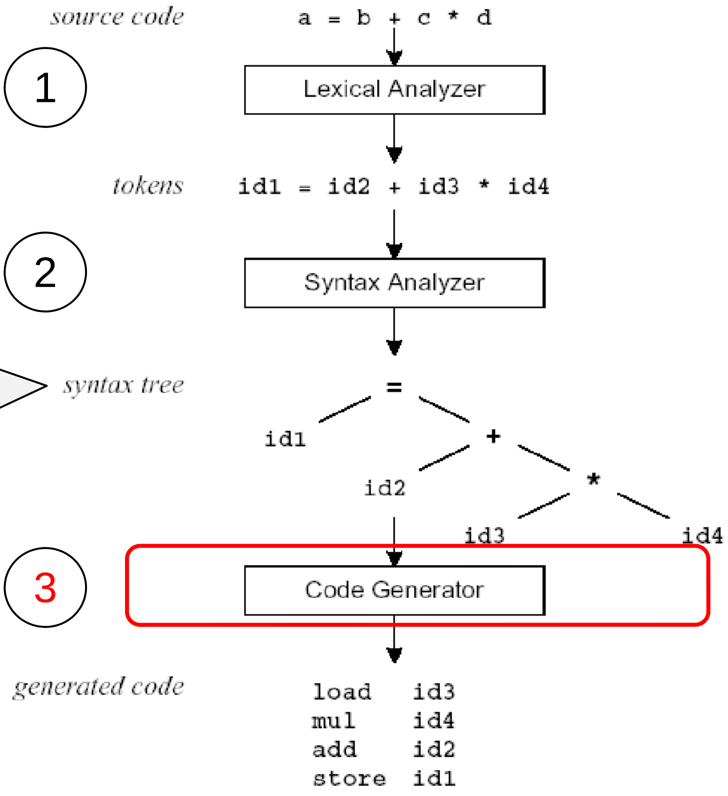


Project Outline



What to do in this Assignment?

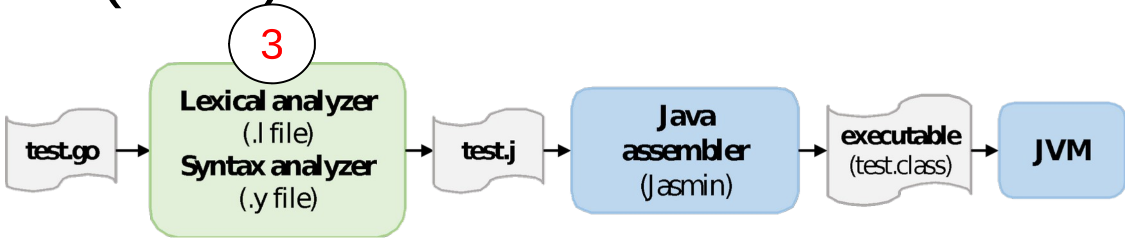
- To accomplish the last step of building your *μGO* compiler, which converts the *μGO* program into the Java assembly code.



- Code Generation:
  - ***Inject*** the Jasmin assembly instructions into your flex/bison code developed in the previous assignments.

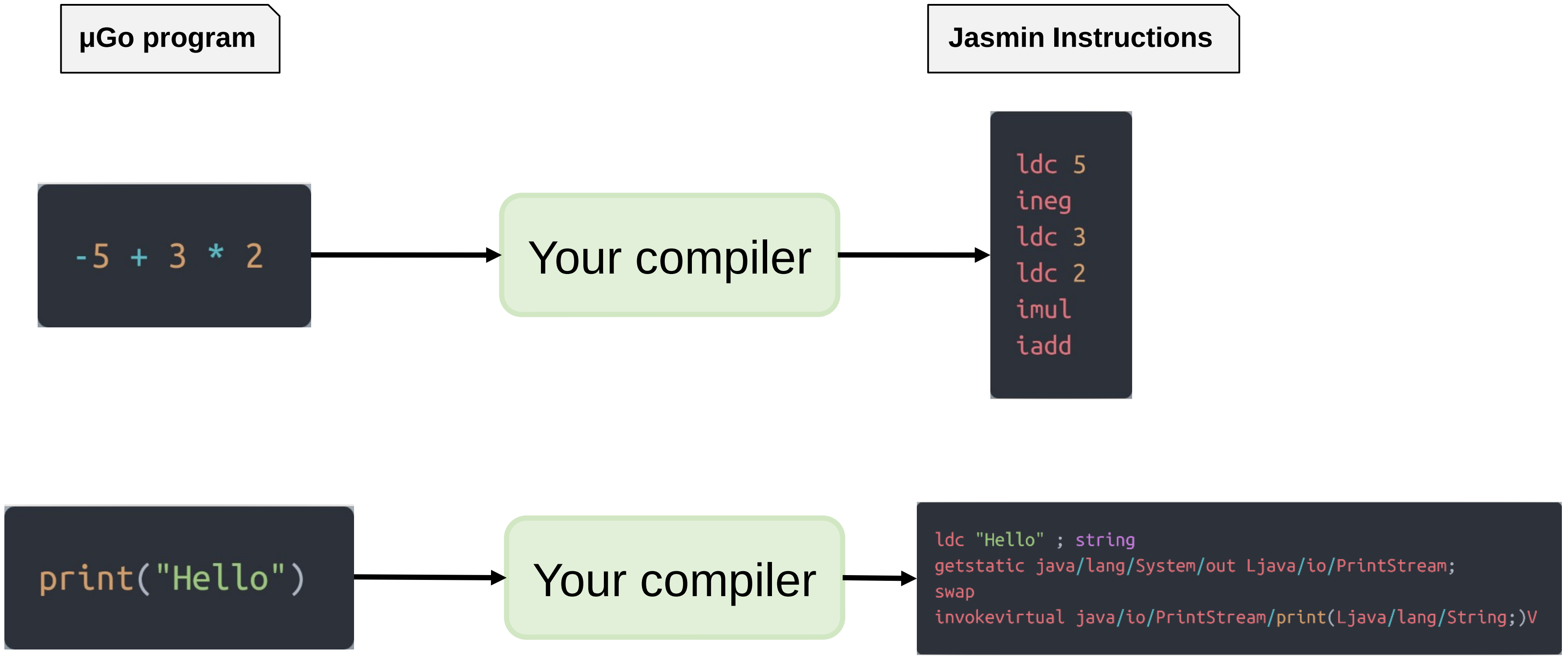
What to do in this Assignment? (cont.)

- Your compiler generates the Jasmin assembly code (test.j) for the given input program (test.go).
- The generated code will then be translated to the Java bytecode (test.class) by the Java assembler, Jasmin.
- The generated Java bytecode should be run by the Java Virtual Machine (JVM).



- In this assignment,
  - TAs give the score based on your .j file and the JVM **execution results**.
  - The flex/bison files need to print out the error messages as hw2 did.

Simple examples



```
-5 + 3 * 2
```

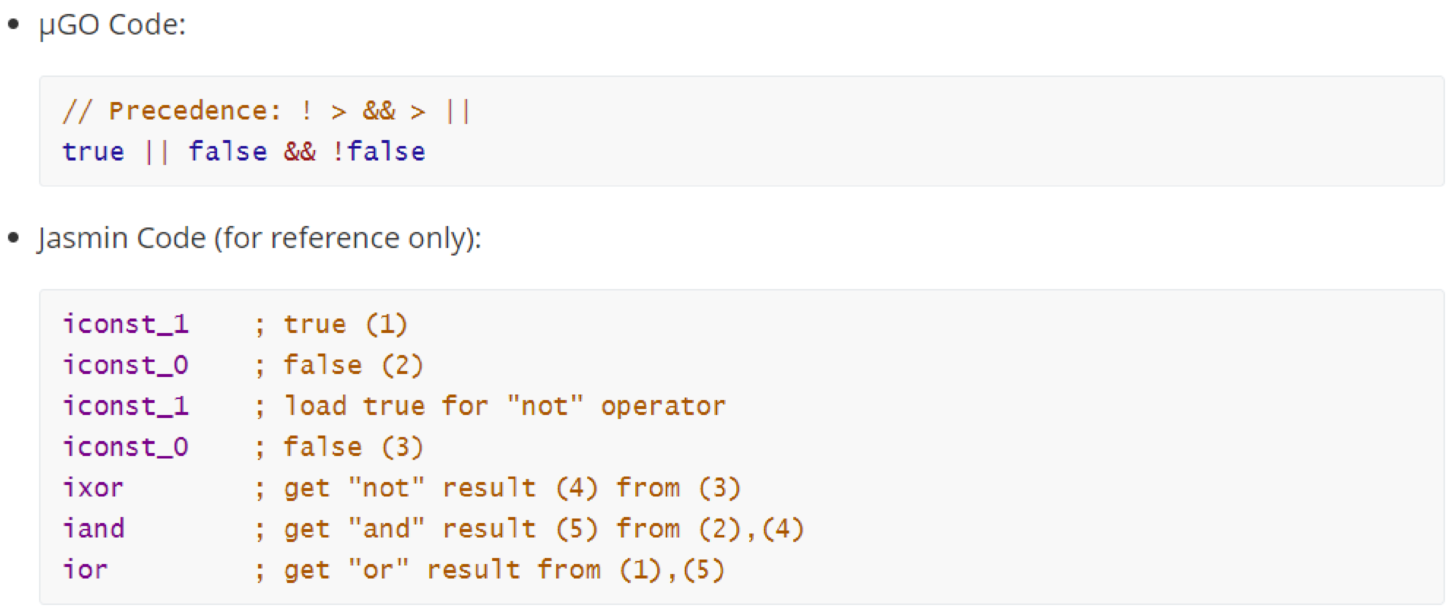
```
ldc 5
ineg
ldc 3
ldc 2
imul
iadd
```

```
print("Hello")
```

```
ldc "Hello" ; string
getstatic java/lang/System/out Ljava/io/PrintStream;
swap
invokevirtual java/io/PrintStream/print(Ljava/lang/String;)V
```

Simple examples (cont.)

- We also give several examples in the appended document
- However, the corresponding Jasmin codes are just for reference, so you can write your own version while it can produce the same program outputs.



- µGO Code:

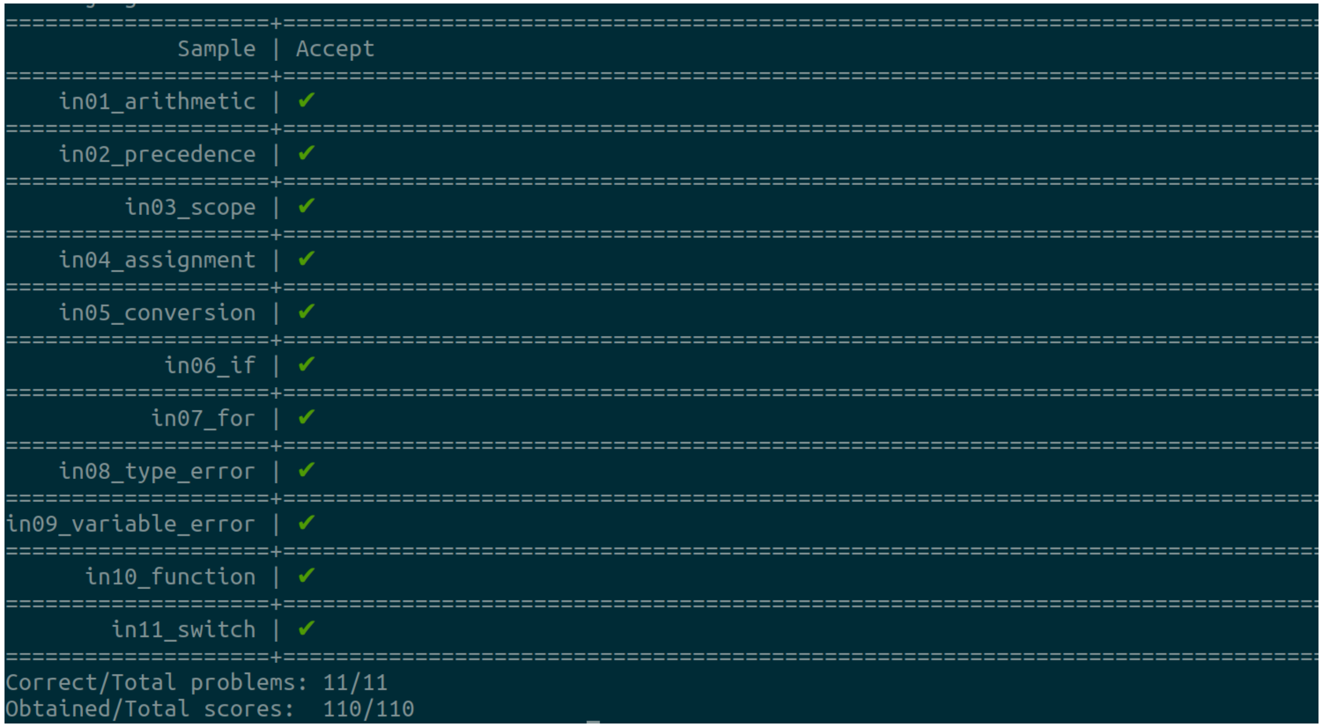
```
// Precedence: ! > && > ||
true || false && !false
```

- Jasmin Code (for reference only):

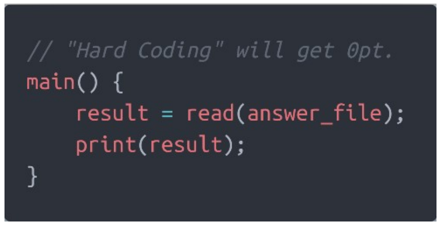
```
iconst_1    ; true (1)
iconst_0    ; false (2)
iconst_1    ; load true for "not" operator
iconst_0    ; false (3)
ixor        ; get "not" result (4) from (3)
iand        ; get "and" result (5) from (2),(4)
ior         ; get "or" result from (1),(5)
```

Assignment Requirements

- Each test case is 10pt and the total score is 110pt.



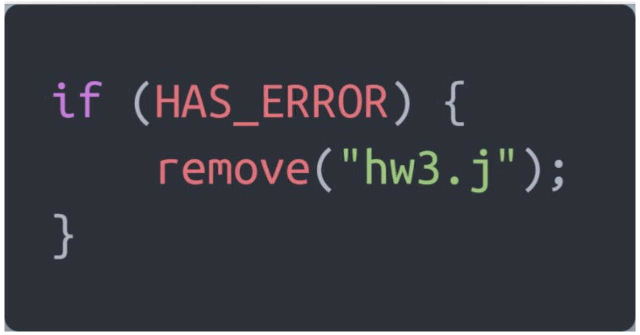
```
==================+====================================================================
           Sample | Accept
==================+====================================================================
   in01_arithmetic | ✔
==================+====================================================================
   in02_precedence | ✔
==================+====================================================================
        in03_scope | ✔
==================+====================================================================
   in04_assignment | ✔
==================+====================================================================
   in05_conversion | ✔
==================+====================================================================
           in06_if | ✔
==================+====================================================================
          in07_for | ✔
==================+====================================================================
   in08_type_error | ✔
==================+====================================================================
in09_variable_error | ✔
==================+====================================================================
     in10_function | ✔
==================+====================================================================
       in11_switch | ✔
==================+====================================================================
Correct/Total problems: 11/11
Obtained/Total scores:  110/110
```



```
// "Hard Coding" will get 0pt.
main() {
    result = read(answer_file);
    print(result);
}
```

Assignment Requirements (cont.)

- When ERRORs occur during the parsing phase,
  - Print out **<u>ALL</u>** error messages, as Assignment 2 did, and
  - **<u>DO NOT</u>** generate the Java assembly code (.j file).

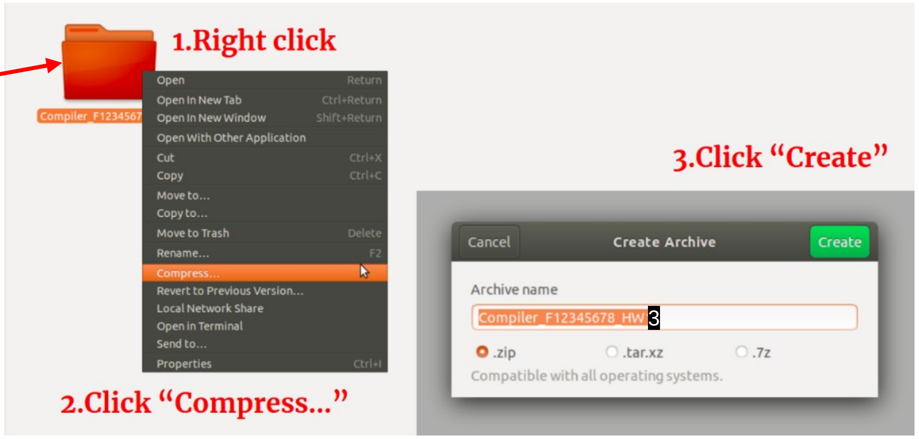


```
if (HAS_ERROR) {
    remove("hw3.j");
}
```

Submission

- Upload your homework to Moodle.
- The expected arrangement of your codes:
  - Only .zip and .rar types of compression are allowed.
  - The directory should be organized as:

```
Compiler_StudentID_HW3.zip/
└── Compiler_StudentID_HW3/
        ├── compiler_hw3.l
        ├── compiler_hw3.y
        ├── common.h
        ├── jasmin.jar
        └── Makefile
```



Archive name

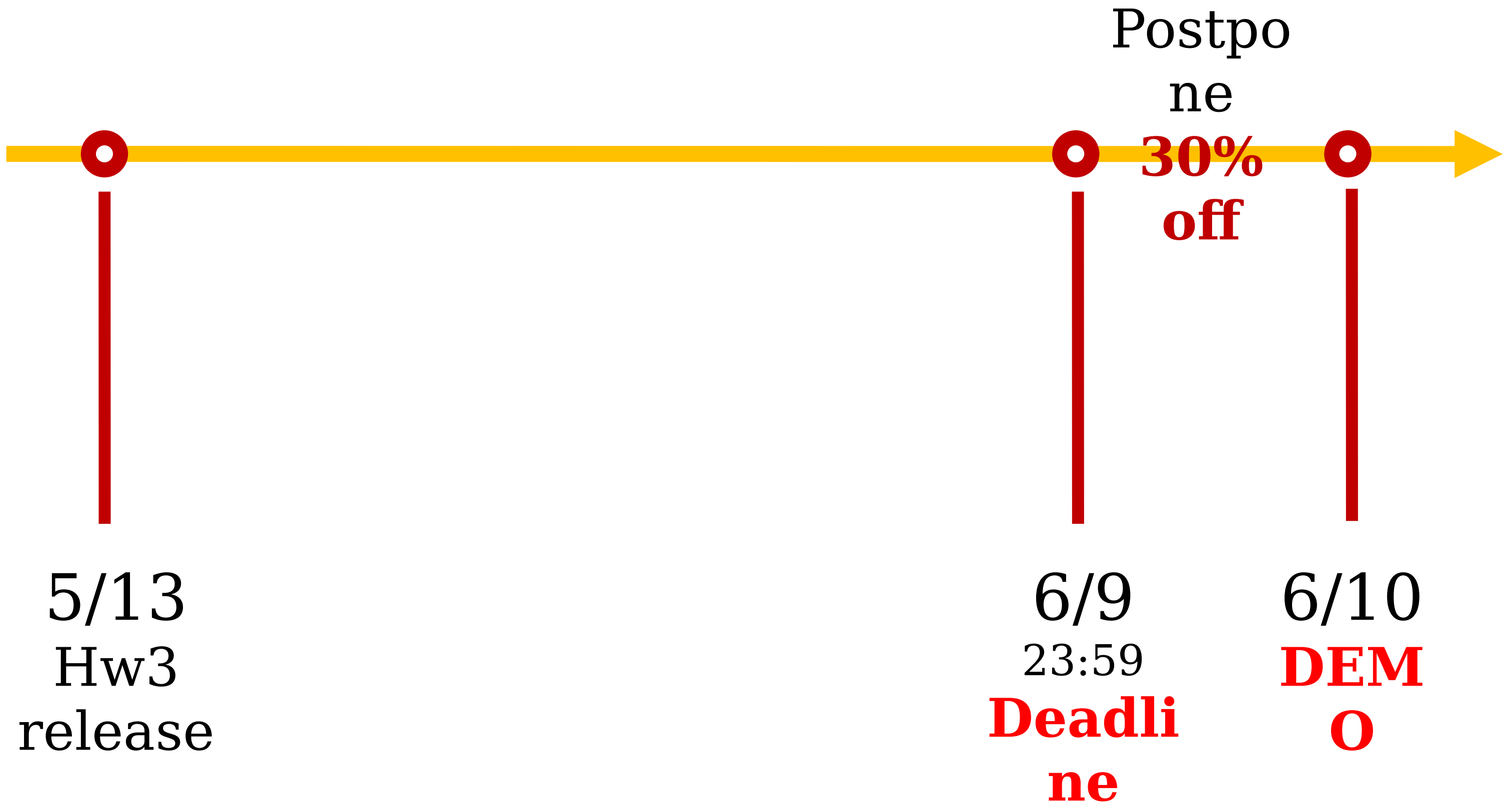
Compiler_F12345678_HW3

[✓ .zip] .tar.xz .7z

Compatible with all operating systems.

  - You will lose 10pt if your programs were uploaded in incorrect format!!!

Deadline



About DEMO

- Demo time is 12:00-18:30, 6/10
  - The demo is partitioned into several time periods.
  - We will open a Google Form for you to register your demo time slot.
  - Each time period allows less than 26 people to demo.
- Demo will be held in virtual
- You are responsible for your code.
  - If you cannot explain your code clearly, you score will be discounted.

- Please come to demo **ON TIME**.

How to Mail TAs

- Send mail to asrlab@csie.ncku.edu.tw, not any TA's mail!!
- Email subject starts with "[Compiler2022]"

QUESTIONS ?