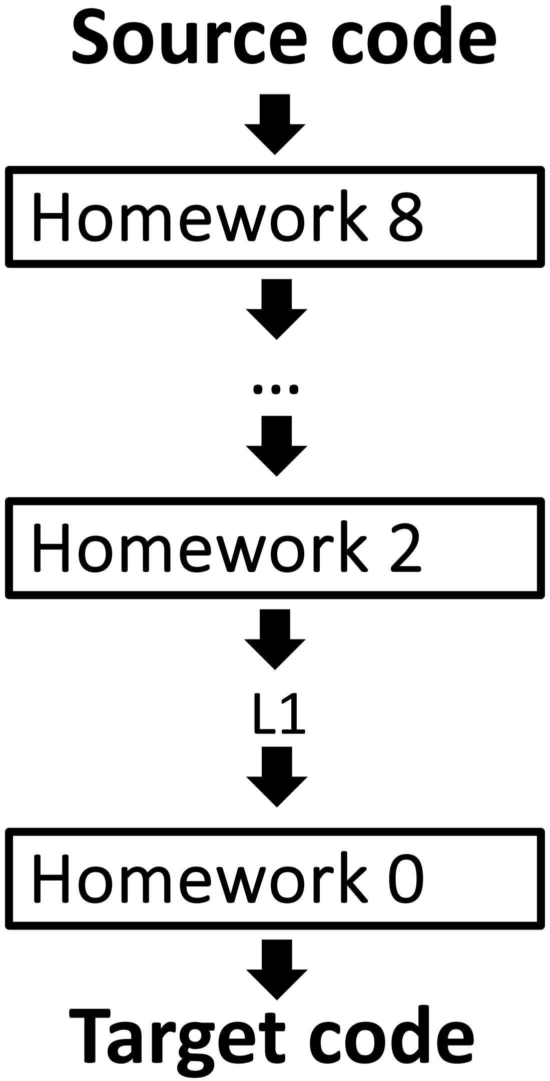
C⚙mpiler

C⚙nstruction

The CC framework

Simone Campanoni
simone.campanoni@northwestern.edu

In this class



Source code

Homework 8

…

Homework 2

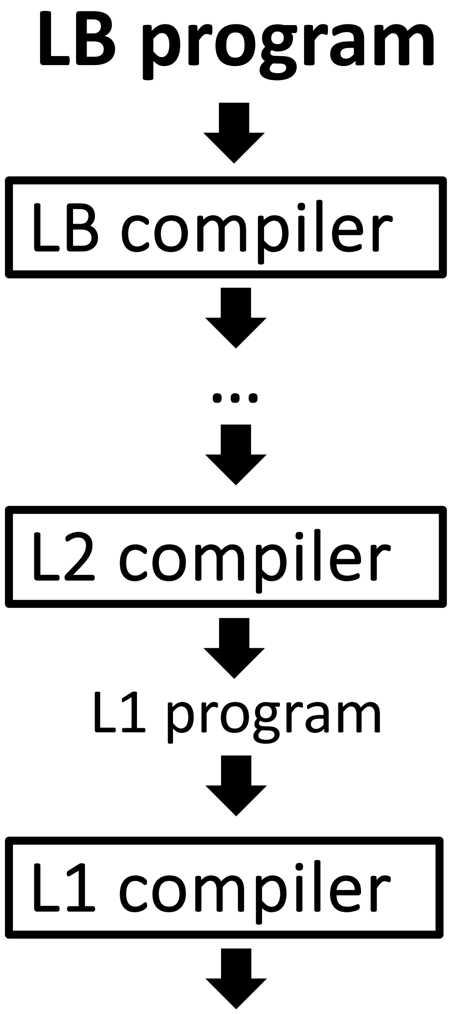
L1

Homework 0

Target code

- The final compiler is built as a sequence of internal compilers
- Each internal compiler translates from a source language to a target language
- Source and target languages are always different
- All languages are written/read into/from files
- Each homework is a standalone compiler

The CC framework



LB program

LB compiler

…

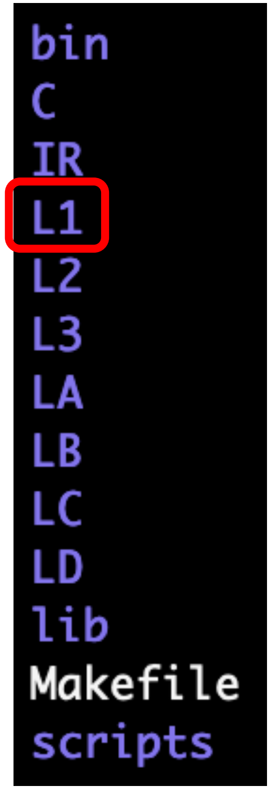
L2 compiler

L1 program

L1 compiler

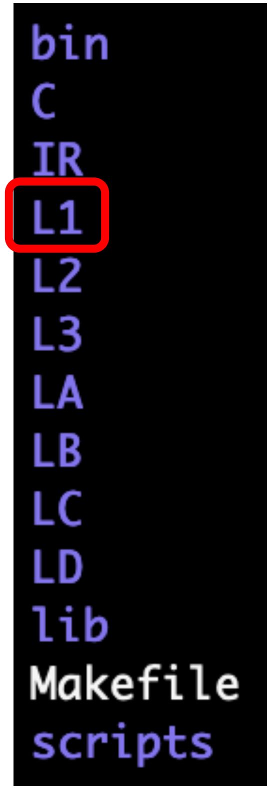
x86_64 standalone binary

- 322_framework.tar.bz2
- All compilers you will build will be included in the CC framework
- Each compiler is a standalone program
- The framework includes tests and scripts to help you succeeding in designing each compiler



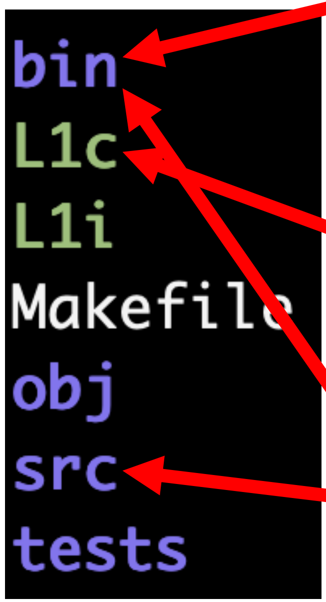
```
bin
C
IR
L1
L2
L3
LA
LB
LC
LD
lib
Makefile
scripts
```

The CC framework



```
bin
C
IR
L1
L2
L3
LA
LB
LC
LD
lib
Makefile
scripts
```

A compiler in the framework



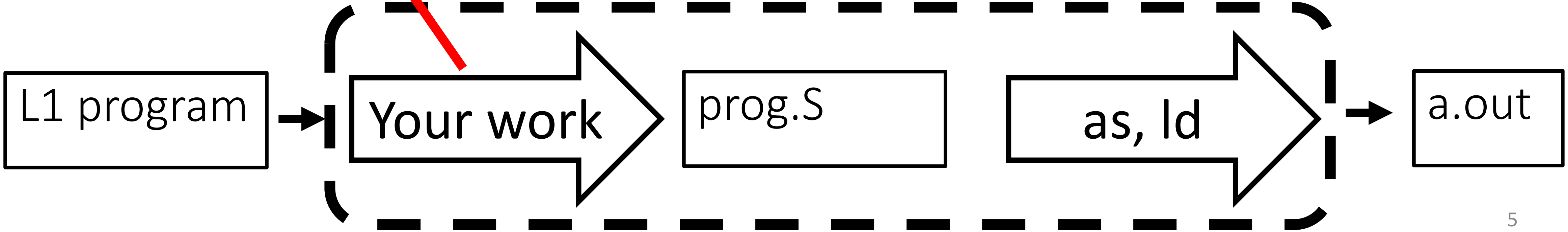
```
bin
L1c
L1i
Makefile
obj
src
tests
```

The binary that generates prog.S (x86_64 assembly file) from an L1 program

You have our binaries to test your L1 programs you will develop

What you invoke to compile an L1 program

C++ code you will develop



L1 program

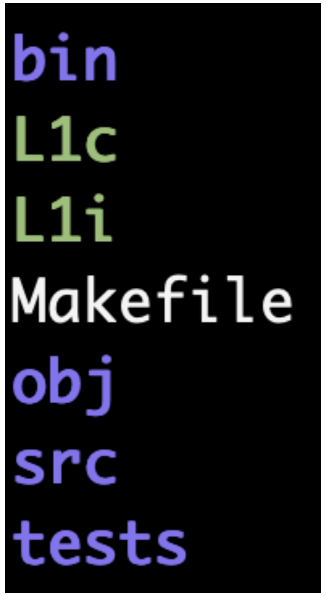
Your work

prog.S

as, ld

a.out

Invoking the L1 compiler (L1c)



```
bin
L1c
L1i
Makefile
obj
src
tests
```

- ./L1c OPTIONS_OF_THE_BINARY AN_L1_program.L1

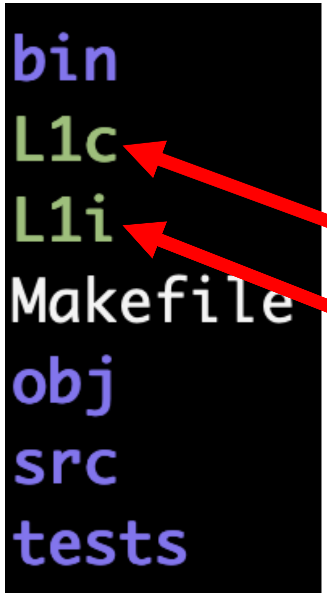
  ./L1c tests/tests3.L1

  ./L1c –v tests/tests3.L1

- You will find both
  prog.S (assembly file generated by bin/L1) and
  a.out (executable)

- You can run the program now
  ./a.out

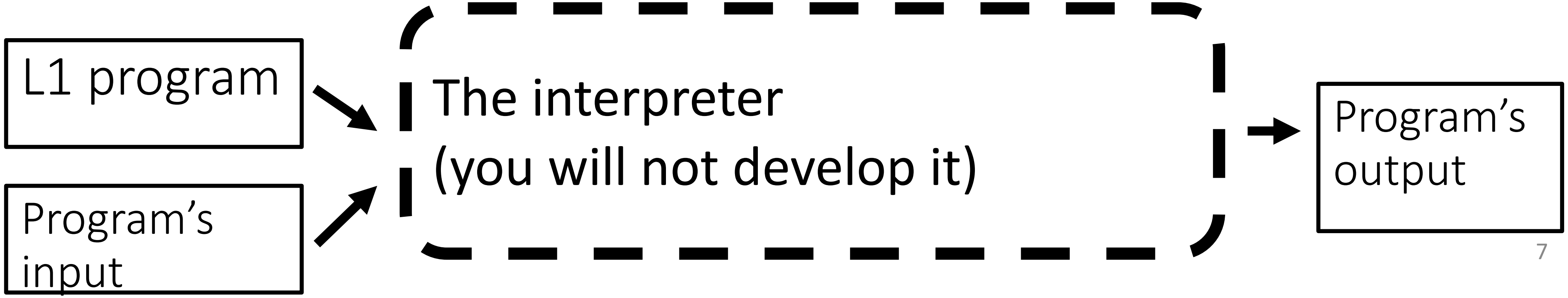
An interpreter in the framework



```
bin
L1c
L1i
Makefile
obj
src
tests
```

What you invoke to compile an L1 program

What you invoke to run the interpreter for an L1 program



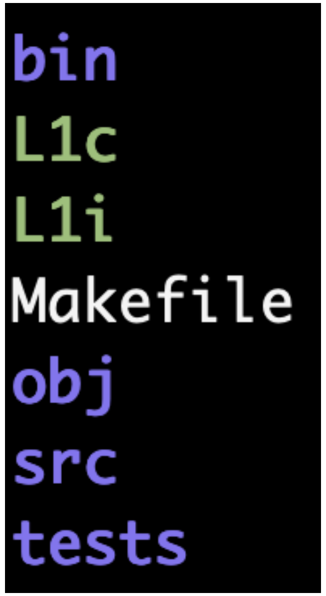
L1 program

Program's input

The interpreter
(you will not develop it)

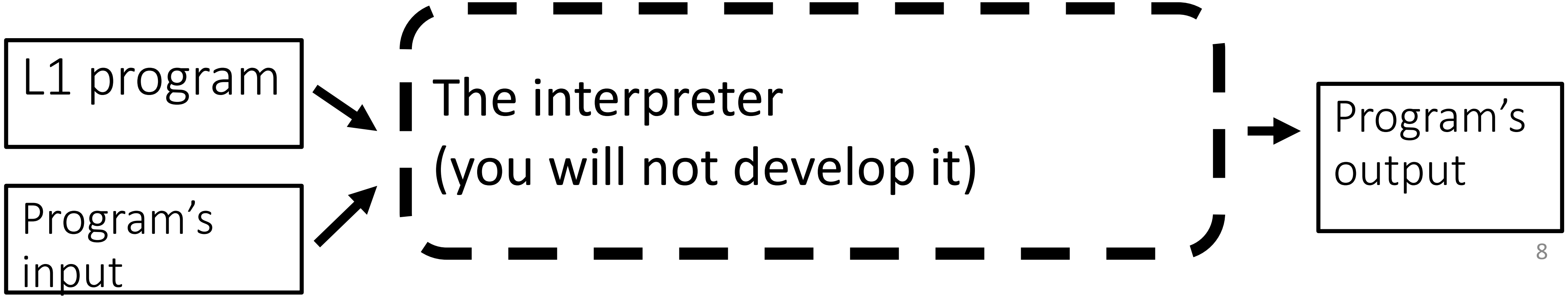
Program's output

An interpreter in the framework



```
bin
L1c
L1i
Makefile
obj
src
tests
```

- The L1 interpreter performs the following operation for every L1 instruction
  - It checks if the L1 instruction that is about to be executed is correct
  - If the instruction isn't correct, then it aborts printing the bug found in the L1 program
  - Otherwise, the interpreter executes that instruction
- ./L1i YOUR_PROGRAM.L1



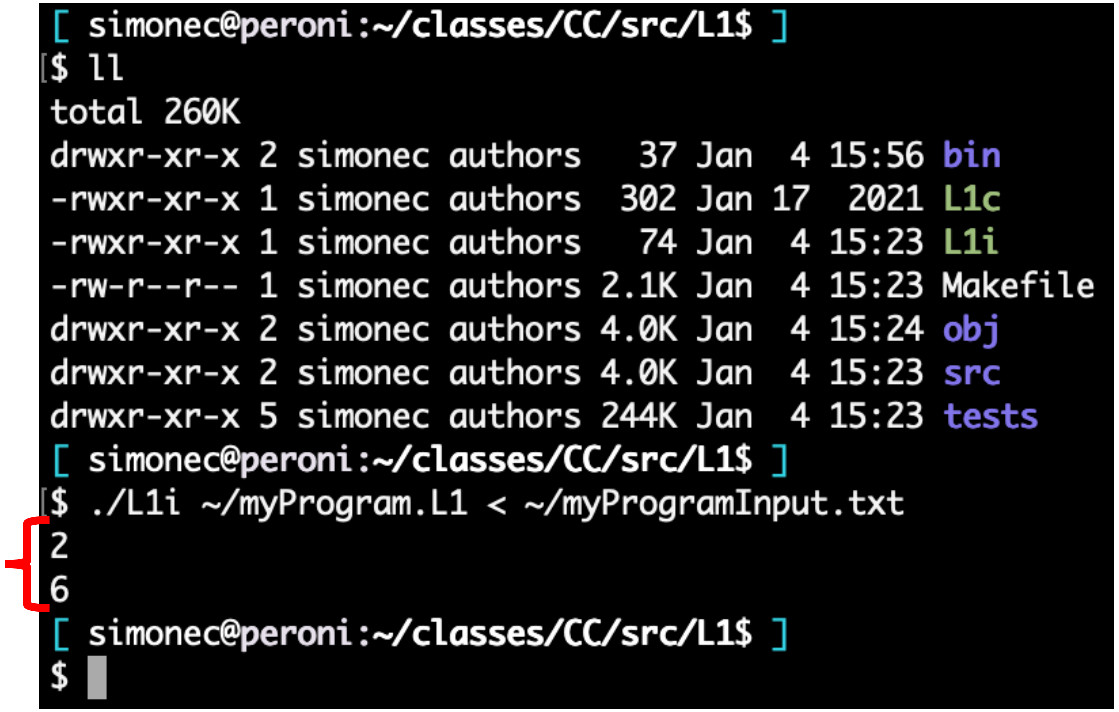
L1 program

Program's input

The interpreter
(you will not develop it)

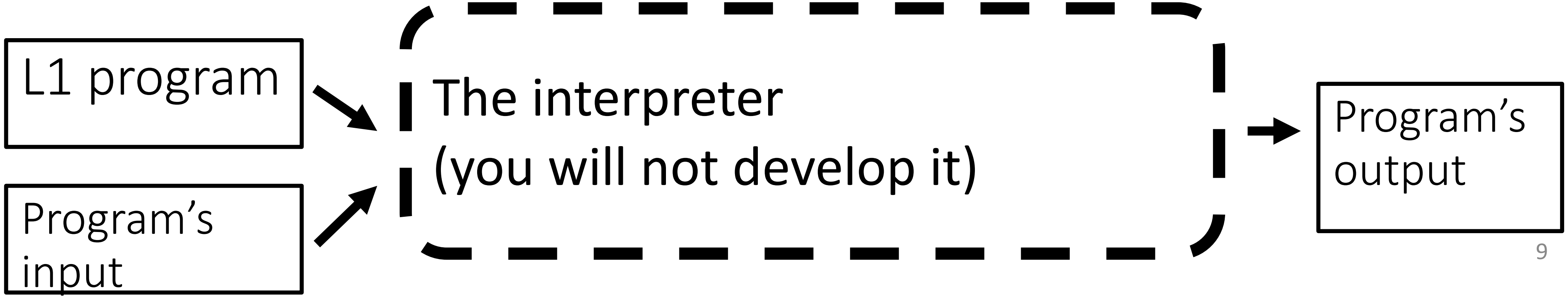
Program's output

An interpreter in the framework



```
[ simonec@peroni:~/classes/CC/src/L1$ ]
[$ ll
total 260K
drwxr-xr-x 2 simonec authors   37 Jan  4 15:56 bin
-rwxr-xr-x 1 simonec authors  302 Jan 17  2021 L1c
-rwxr-xr-x 1 simonec authors   74 Jan  4 15:23 L1i
-rw-r--r-- 1 simonec authors 2.1K Jan  4 15:23 Makefile
drwxr-xr-x 2 simonec authors 4.0K Jan  4 15:24 obj
drwxr-xr-x 2 simonec authors 4.0K Jan  4 15:23 src
drwxr-xr-x 5 simonec authors 244K Jan  4 15:23 tests
[ simonec@peroni:~/classes/CC/src/L1$ ]
[$ ./L1i ~/myProgram.L1 < ~/myProgramInput.txt
2
6
[ simonec@peroni:~/classes/CC/src/L1$ ]
$ 
```

Program's output



L1 program

Program's input

The interpreter

(you will not develop it)

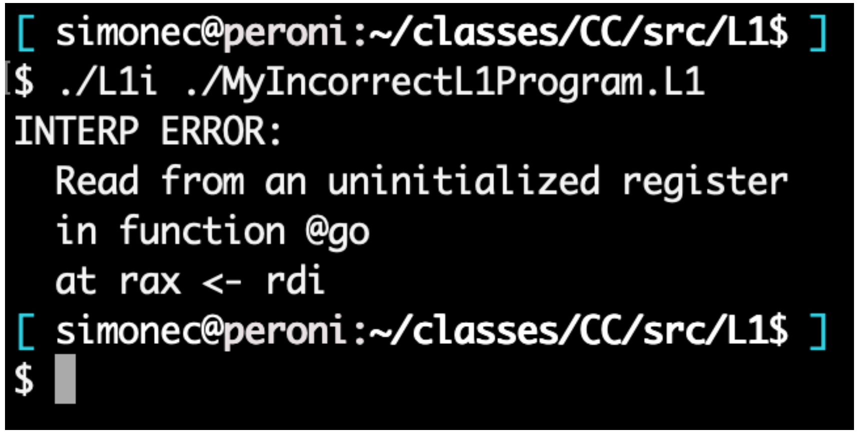
Program's output

Summary for L1 tools available in the framework

- **L1i: the interpreter**
  - It checks the correctness of runtime states for every L1 instruction of the running program
  - It is slow so be patient when you try to run programs that execute many instructions
  - It has been recently developed; so please be kind and let us know if you see problems
  - Possible outputs:
    - If the L1 program with its input is correct, then the interpreter prints the L1's program outputs
    - If the L1 program has a runtime bug (e.g., reading a register before writing to it), then the interpreter prints information about the runtime bug and where in the code it occurred

Summary for L1 tools available in the framework

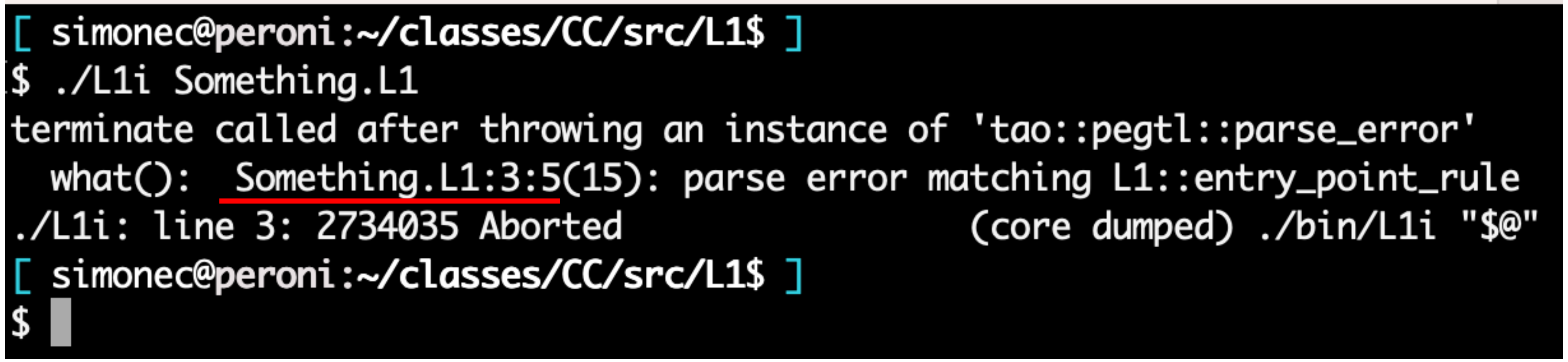
- **L1i: the interpreter**
  - Possible outputs:
    - If the L1 program with its input is correct,
      then the interpreter prints the L1's program outputs
    - If the L1 program has a runtime bug (e.g., reading a register before writing to it),
      then the interpreter prints information about the runtime bug
      and where in the code it occurred



```
[ simonec@peroni:~/classes/CC/src/L1$ ]
$ ./L1i ./MyIncorrectL1Program.L1
INTERP ERROR:
  Read from an uninitialized register
  in function @go
  at rax <- rdi
[ simonec@peroni:~/classes/CC/src/L1$ ]
$ 
```

Summary for L1 tools available in the framework

- **L1i: the interpreter**
  - Possible outputs:
    - If the L1 program with its input is correct,
      then the interpreter prints the L1’s program outputs
    - If the L1 program has a runtime bug (e.g., reading a register before writing to it),
      then the interpreter prints information about the runtime bug
      and where in the code it occurred
    - If the program is not an L1 program,
      then the interpreter prints the reason why the program do not adhere to the L1 language



```
[ simonec@peroni:~/classes/CC/src/L1$ ]
$ ./L1i Something.L1
terminate called after throwing an instance of 'tao::pegtl::parse_error'
  what():  Something.L1:3:5(15): parse error matching L1::entry_point_rule
./L1i: line 3: 2734035 Aborted                 (core dumped) ./bin/L1i "$@"
[ simonec@peroni:~/classes/CC/src/L1$ ]
$ 
```

Summary for L1 tools available in the framework

- **L1i: the interpreter**
  - It checks the correctness of runtime states for every L1 instruction of the running program
  - It is slow so be patient when you try to run programs that execute many instructions
  - It has been recently developed; so please be kind and let us know if you see problems
  - Possible outputs:
    - If the L1 program with its input is correct, then the interpreter prints the L1's program outputs
    - If the L1 program has a runtime bug (e.g., reading a register before writing to it), then the interpreter prints information about the runtime bug and where in the code it occurred
    - If the program is not an L1 program, then the interpreter prints the reason why the program do not adhere to the L1 language

Summary for L1 tools available in the framework

- **L1i: the interpreter**
  - It checks the correctness of runtime states for every L1 instruction of the running program
  - It is slow so be patient when you try to run programs that execute many instructions
  - It has been recently developed; so please be kind and let us know if you see problems
- **L1c: the compiler**
  - It assumes the correctness of the L1 program being compiled (if there is a bug in an L1 program, good luck finding it)
  - The generated binary is fast
  - It has been developed throughout the last 7 years.
    So do not be kind if you see problems, let us know about problems, and feel free to be angry at Simone

Suggestion about how to use L1 tools available in the framework

- **L1i: the interpreter**
  - Use it to check the correctness of your L1 programs

- **L1c: the compiler**
  - Use it to compare the binaries generated by our compiler with those generated by your L1 compiler

Testing your work

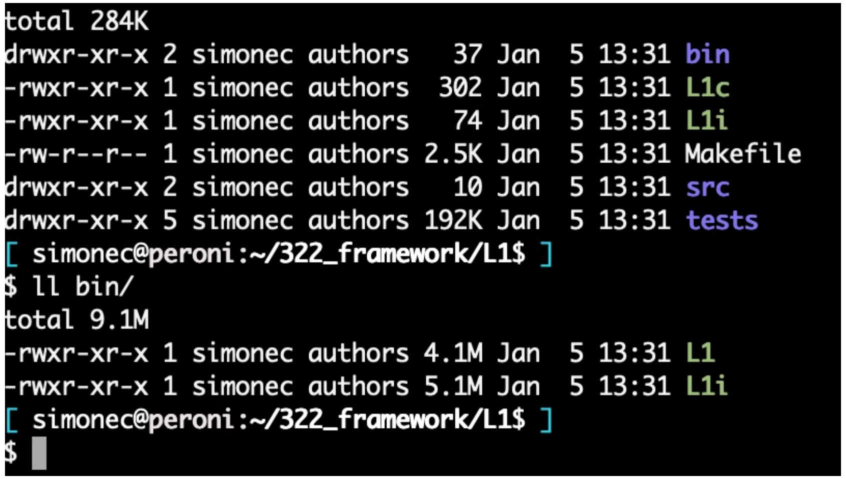
Host to use the CC framework for your assignments:

- Wilkinson lab
  gotham.ece.northwestern.edu, batman.ece.northwestern.edu, robin.ece.northwestern.edu, alfred.ece.northwestern.edu ,gordon.ece.northwestern.edu ,madhatter.ece.northwestern.edu ,joker.ece.northwestern.edu ,cobblepott.ece.northwestern.edu ,bane.ece.northwestern.edu ,nightwing.ece.northwestern.edu ,selina.ece.northwestern.edu ,ras.ece.northwestern.edu ,poisonivy.ece.northwestern.edu ,freeze.ece.northwestern.edu ,scarecrow.ece.northwestern.edu ,clayface.ece.northwestern.edu ,harley.ece.northwestern.edu ,killercroc.ece.northwestern.edu ,huntress.ece.northwestern.edu ,batgirl.ece.northwestern.edu ,riddler.ece.northwestern.edu ,hush.ece.northwestern.edu
- WOT systems
  murphy.wot.ece.northwestern.edu, finagle.wot.ece.northwestern.edu, hanlon.wot.ece.northwestern.edu, moore.wot.ece.northwestern.edu

Steps

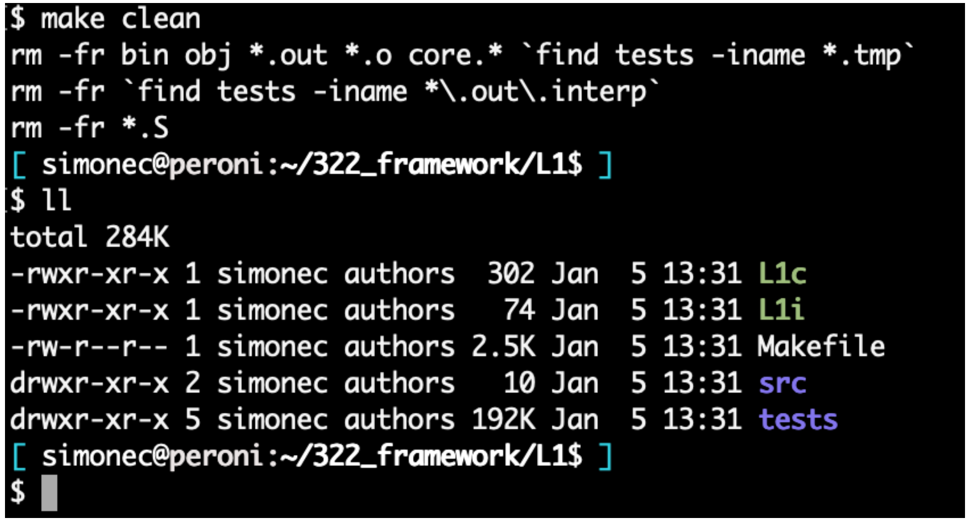
- **Login to it:** ssh YOUR_NET_ID@hanlon.wot.eecs.northwestern.edu
- **Run** “bash”
- **Run** “source /opt/rh/gcc-toolset-11/enable”

Recovering Simone’s binaries



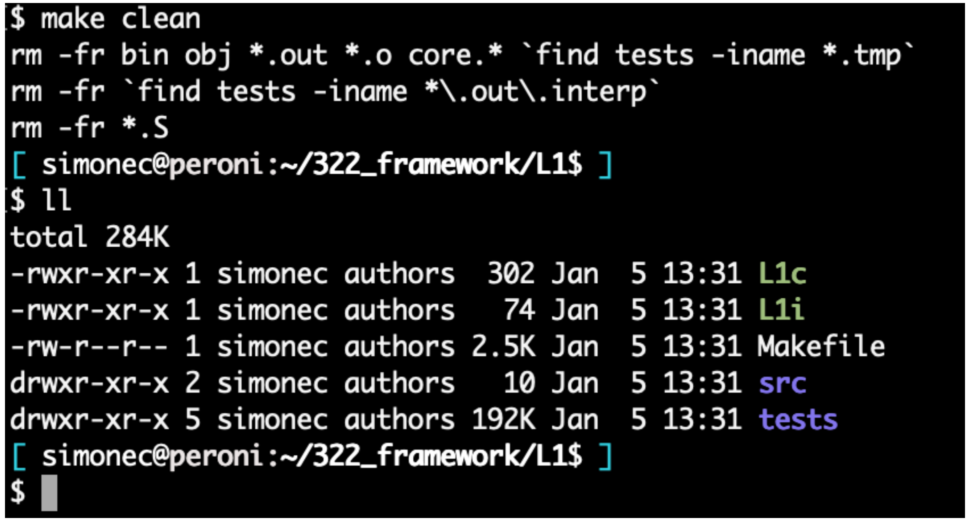
```
total 284K
drwxr-xr-x 2 simonec authors   37 Jan  5 13:31 bin
-rwxr-xr-x 1 simonec authors  302 Jan  5 13:31 L1c
-rwxr-xr-x 1 simonec authors   74 Jan  5 13:31 L1i
-rw-r--r-- 1 simonec authors 2.5K Jan  5 13:31 Makefile
drwxr-xr-x 2 simonec authors   10 Jan  5 13:31 src
drwxr-xr-x 5 simonec authors 192K Jan  5 13:31 tests
[ simonec@peroni:~/322_framework/L1$ ]
$ ll bin/
total 9.1M
-rwxr-xr-x 1 simonec authors 4.1M Jan  5 13:31 L1
-rwxr-xr-x 1 simonec authors 5.1M Jan  5 13:31 L1i
[ simonec@peroni:~/322_framework/L1$ ]
$
```

Let’s assume that by mistake you run:



```
$ make clean
rm -fr bin obj *.out *.o core.* `find tests -iname *.tmp`
rm -fr `find tests -iname *\.out\.interp`
rm -fr *.S
[ simonec@peroni:~/322_framework/L1$ ]
$ ll
total 284K
-rwxr-xr-x 1 simonec authors  302 Jan  5 13:31 L1c
-rwxr-xr-x 1 simonec authors   74 Jan  5 13:31 L1i
-rw-r--r-- 1 simonec authors 2.5K Jan  5 13:31 Makefile
drwxr-xr-x 2 simonec authors   10 Jan  5 13:31 src
drwxr-xr-x 5 simonec authors 192K Jan  5 13:31 tests
[ simonec@peroni:~/322_framework/L1$ ]
$
```

Recovering Simone’s binaries

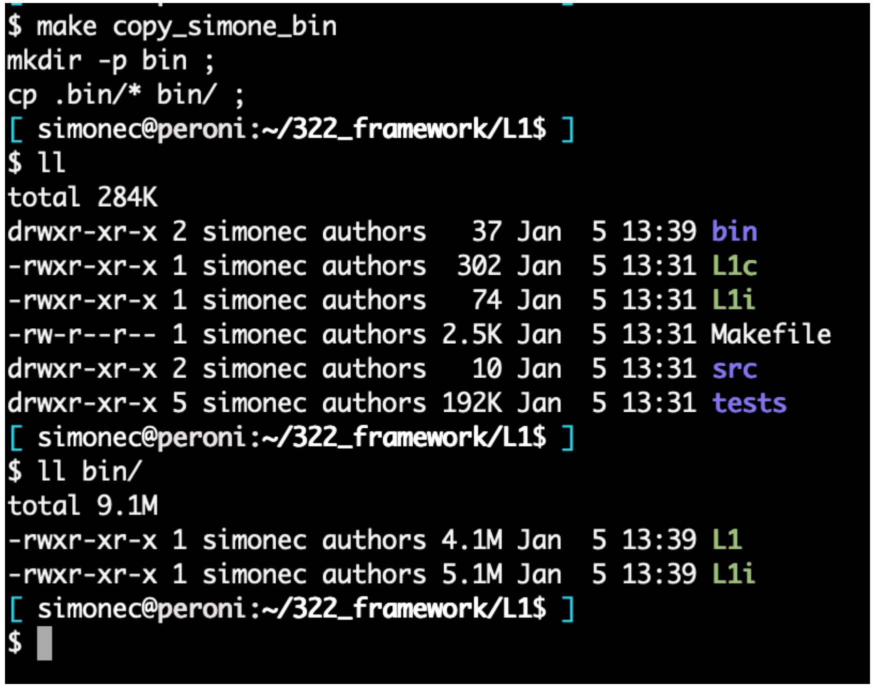


```
$ make clean
rm -fr bin obj *.out *.o core.* `find tests -iname *.tmp`
rm -fr `find tests -iname *\.out\.interp`
rm -fr *.S
[ simonec@peroni:~/322_framework/L1$ ]
$ ll
total 284K
-rwxr-xr-x 1 simonec authors  302 Jan  5 13:31 L1c
-rwxr-xr-x 1 simonec authors   74 Jan  5 13:31 L1i
-rw-r--r-- 1 simonec authors 2.5K Jan  5 13:31 Makefile
drwxr-xr-x 2 simonec authors   10 Jan  5 13:31 src
drwxr-xr-x 5 simonec authors 192K Jan  5 13:31 tests
[ simonec@peroni:~/322_framework/L1$ ]
$ 
```

Now we lost Simone’s binaries.

So you cannot run both
Simone’s compiler and interpreter

To recover them, run:



```
$ make copy_simone_bin
mkdir -p bin ;
cp .bin/* bin/ ;
[ simonec@peroni:~/322_framework/L1$ ]
$ ll
total 284K
drwxr-xr-x 2 simonec authors   37 Jan  5 13:39 bin
-rwxr-xr-x 1 simonec authors  302 Jan  5 13:31 L1c
-rwxr-xr-x 1 simonec authors   74 Jan  5 13:31 L1i
-rw-r--r-- 1 simonec authors 2.5K Jan  5 13:31 Makefile
drwxr-xr-x 2 simonec authors   10 Jan  5 13:31 src
drwxr-xr-x 5 simonec authors 192K Jan  5 13:31 tests
[ simonec@peroni:~/322_framework/L1$ ]
$ ll bin/
total 9.1M
-rwxr-xr-x 1 simonec authors 4.1M Jan  5 13:39 L1
-rwxr-xr-x 1 simonec authors 5.1M Jan  5 13:39 L1i
[ simonec@peroni:~/322_framework/L1$ ]
$ 
```

Always have faith in your ability

Success will come your way eventually

Best of luck!