

Formal Languages and Compilers Laboratory

Introduction

Daniele Cattaneo

Material based on slides by Alessandro Barengi and Michele Scandale

Formal Language Theory in Practice

Regular expressions, formal grammars, LR and LL parsing...

“Compilers are commodities!

I will never need this stuff in my job, I’m losing my time!”

This is a **mistake**! Compiler technology has many applications:

- Want to do some bulk processing on text files?
 - Most text editors support **regular expressions**
- Need to read a configuration file in a custom format?
- Need to read/design a complex file format?
- Need to create a **domain-specific** language?
 - **Parser generators** are the perfect tool to make a parser quickly
- **Your scientific calculator has a parser in it**

Topics

In these 5 laboratory lessons we will see how theoretical concepts are applied **in practice**

- The standard regular expression syntax
- Standard unix tools for text editing
- Parser generation with **flex** and **bison**
- The internal organization and workflow of a real-world compiler
- How to modify and extend a simple compiler called **ACSE**

Modification of ACSE will be your **final proof** at the exam

Exam

The lab is 20% of the exam score:

- You need **to pass** the lab exam in order to pass the whole exam
- The minimum score to pass the lab test is $\frac{15}{30}$
- Bonus question for **laboratory** laude (not evaluated otherwise)
 - Theory laude does **not** exist
 - If lab grade = 30L and rounded theory grade = 30:
get 30L final grade
 - If lab grade = 30L and rounded theory grade < 30:
get +1 to the final grade

The lab exam is usually held before the theory exam.

The only material allowed during the exam is the ACSE reference header

Exam

Truth table

Lab grade	Theory grade (rounded)	Final Grade
< 15	< 15	FAIL
< 15	≥ 15	FAIL
≥ 15	< 15	FAIL
$\geq 15, \leq 30$	≥ 15	$\frac{4}{5}\text{theory} + \frac{1}{5}\text{lab}$
30L	$\geq 15, < 30$	$\frac{4}{5}\text{theory} + \frac{1}{5}\text{lab} + 1$
30L	30	30L

Requirements & Assumptions

You are expected to meet or exceed the following requirements:

- Have a good command of the **C language**
- **Be able to use a standard UNIX compilation toolchain**
 - Covered in Computer Science 101 / Fondamenti di Informatica
- Know how a C construct is translated into **assembly**
 - Covered in Computer Architectures and Operating Systems
- To employ all the above in a thoughtful way

Your brain must be turned on

Requirements & Assumptions

Laboratory means **hands-on practice**
Hands-on practice means **writing code**

Sadly there is no time to let you write code in class.

But you must do it at home!

To get started:

- 1 Download the **Unix Setup Guide** you can find on **WeBeep**
- 2 Follow the instructions
- 3 Download the zip files with the **code examples shown in class**
- 4 **Run them** in the environment you have prepared
- 5 **Do the assignments that I am going to give at each lecture**

The harsh truth

Doing the assignments is optional
How should you check if they are correct?

Compile and test them!

50% of the people doing the laboratory exam **fail**

The reason?

They don't **compile and test their solutions**
when exercising!

Rote memorization won't help you!
Just looking at the solutions is not enough!

Every exam term is different!

Where do I study for the Lab?

All the contents of the Lab lessons can be found here:

- The **bison** docs:
https://www.gnu.org/software/bison/manual/html_node/index.html
- The **flex** docs:
<https://westes.github.io/flex/manual/>
- The **ACSE** docs and source code:
On WeBeeP

Doubts?

The slides are incomplete/unclear?

Read these documents **first!**

Is everything clear?

Then let's get started!