南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

CS323 Lab 1

Yepang Liu

liuyp1@sustech.edu.cn

Agenda

- Introduction to labs & course project
- Set up the environment for labs and project
- Find teammates for the project

What will we do in labs?

- Exercises to understand the core concepts learnt in lectures
- Tool tutorials to help you build a working compiler
- Cover some detailed content that cannot be covered in lectures, e.g., complex algorithms, etc.
- Q&A (we won't have much time for questions during lectures)
- Presentations / project inspections
- …

Lab Attendance & Exercises

- We will check attendance and exercises after course selection/drop period.

- It is fine if you cannot finish all lab exercises during class. You can continue to do the exercises after each class and submit your work before the end of the week.

Course Project

- **Team size:** 3-4 students (there is no additional bonus for small-sized teams)

- **Task 1: Research on languages and compilation techniques**
  - Presentation 1 (Sept. 30 & Oct. 14, tentative): Investigation of language features + the design of the language to be implemented
  - Presentation 2 (Dec. 9 & Dec. 16, tentative): Compilation techniques

Course Project

- **Team size:** 3-4 students (there is no additional bonus for small-sized teams)

- **Task 2: Language design and implementation**
  - Phase 1: Design your own language (within five weeks)
  - Phase 2: Lexical and syntax analysis (~3 weeks)
  - Phase 3: Semantic analysis (~2 weeks)
  - Phase 4: Intermediate code generation (~2 month)
  - Phase 5: Target code generation (~2 weeks)
  - Phase 6 (optional): Code optimization (~2 weeks)

Milestone check
Nov. 11 (tentative)

Final inspection
Dec. 23 (tentative)

Lab Tutorials

- We will provide several detailed tutorials to help you build a working compiler.
- Source language: **SPL** (SUSTech Programming Language), a C-like language without advanced features such as macros and pointers.
- Target language: **MIPS32** assembly language
- Useful tools[1]:
  - **Flex**: The Fast Lexical Analyzer (https://github.com/westes/flex)
  - **Bison**: A general-purpose parser generator (https://www.gnu.org/software/bison/)
  - **SPIM**: A simulator for MIPS32 programs (https://spimsimulator.sourceforge.net/)

[1]You are expected to implement your own language using the above tools or other tools, which we will provide recommendations later.

Install Flex

- Flex is a fast lexical analyser generator. It is a tool for generating programs that perform pattern-matching on text. Flex is a non-GNU free implementation of the well known Lex program.
  - GitHub repo: https://github.com/westes/flex
  - Manual: https://www.epaperpress.com/lexandyacc/download/flex.pdf

- Please follow the step below to install Flex on Ubuntu 18.04:
  - `sudo apt install flex` (you may need to update your package list with "`sudo apt update`" before installing flex)
  - If the installation is successful, type the command "`flex --version`". If you see "`flex 2.6.4`" (latest version in Sept. 2024), you are done.

* We have not tested other versions. There might be problems.

The latest avaiable version of a Ubuntu package can be found at https://packages.ubuntu.com/.

Install Bison

- Bison is a general-purpose parser generator that converts an annotated context-free grammar into a parser. You can use it to develop a wide range of language parsers, from simple desk calculators to complex programming languages.
  - Official website: https://www.gnu.org/software/bison/
  - Manual: https://www.gnu.org/software/bison/manual/

- Please follow the steps below to install Bison:
  - `sudo apt install bison`
  - If the installation is successful, type the command "`bison --version`". If you see "`bison (GNU Bison) 3.0.4…`" (latest version in Sept 2024), you are done.

Test Your C Programming Environment

- We will mostly use C for programming this semester.
- If you are not familiar with C, please check this quick guide:
  - https://www.tutorialspoint.com/cprogramming/c_quick_guide.htm
- Or learn the language via this interactive tutorial:
  - https://www.learn-c.org/
- To test your environment, please compile and run a hello world program:
  1. Clone our GitHub repo to local via https://github.com/sqlab-sustech/CS323-2024F.git (if you don't have git, install it by "`sudo apt install git`")
  2. Go to the directory "lab1" and run command "`make hello`". If you see a file "`hello.out`" generated, you are done. Continue to execute "`./hello.out`" and you will see a "`hello world!`" message in the terminal.
  3. If you fail to build the target, you may need to install gcc (via "`sudo apt install gcc`") and make (via "`sudo apt install make`") and repeat the second step.

C Programming Exercise (Optional)

- To warm up, let's do a linked list exercise

- In our definition, each node contains two fields (see `link_list.h` under the `lab1` directory):
  - For header node, the first field contains the number of nodes in the list, excluding itself; For other nodes, the first field contains an int value.
  - The second field is a pointer to the next node in the list and `NULL` if the current node is the last one in the list.

C Programming Exercise (Optional)

- We have provided the code for a few functions (see `link_list.c`). You are required to implement the following functions:

```c
/* insert val at position index */
void linked_list_insert(node *head, int val, int index);

/* delete node at position index */
void linked_list_delete(node *head, int index);

/* remove the first occurence node of val */
void linked_list_remove(node *head, int val);

/* remove all occurences of val */
void linked_list_remove_all(node *head, int val);
```

```c
/* get value at position index */
int linked_list_get(node *head, int index);

/* search the first index of val */
int linked_list_search(node *head, int val);

/* search all indexes of val */
node *linked_list_search_all(node *head, int val);
```

- When you finish, please compile your code and make a shared object file named "`libll.so`" via the command "`make libll`".
- To test your code, please run our provided python script (if you do not have python3, install it via "`sudo apt install python3`"):
  - `python3 ll_test.py` (see how many test cases you can pass)

Find Your Teammates ☺

- 【腾讯文档】CS323-2024F Project Teams https://docs.qq.com/sheet/DSlhPQUdYUkFKQ2JY?tab=BB08J2

- Please form your team by the end of the first week. Otherwise, you may miss some deadlines.