UCLA Samueli Computer Science

CS31 Week 1 Discussion

Fall 2021, Section 1C

Mingyu Derek Ma mdma@ucla.edu

Thanks Muhao Chen, Rosa Garza for their shared content

https://derek.ma/cs31 for slides and other discussion materials

About me and this section

- About me
  - 3rd year PhD student in CS
  - Working with Prof. Nanyun (Violet) Peng
  - Research interests: Natural Language Processing, Machine Learning
- Office Hours
  - Monday 4:30pm-7:30pm on Zoom
  - Friday 4:30pm-5:30pm at Boelter 3256S
- Email
  - mdma@ucla.edu (please include CS31 in the subject, I'll response within 24 hours!)
- Discussion section materials
  - CCLE Discussions folder
  - https://derek.ma/cs31

LA

- Jessica Lee

Coming deadlines

- Project 1: Monday, Oct 4 11pm PST

Abstraction levels of languages

- Machine Language
  - Binary
  - Difficult for human to understand
  - Easy for computers
- Assembly Language
- Higher-level Language
  - Machine map it to machine language and execute the instructions
- Human Language?

Write a program

```
#include <iostream>
using namespace std;

int main() {
	int a = 1, b = 1;
	cout<< a + b << endl;
}
```

- Include the <iostream> library to use standard input output variable like “cout”
- Use namespace std
  - Namespace is a collection of name definitions
  - Same function name can be given different definitions in two namespaces
- ”endl” : output a new line
- main() function: where the C++ program begins its logic
- Case-sensitive

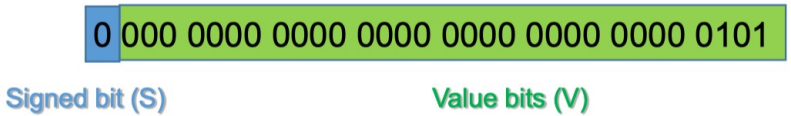
Variables

```
#include <iostream>
using namespace std;

int main() {
	int a = 1, b = 1;
	cout<< a + b << endl;
}
```

- Begin with letter or underscore _
- Rest of characters must be letters, digits or _
  - `x, x1, x_1, _abc, ABC123z7`
- Case sensitive
  - `timeUsed` is different from `timeused`
- Good practice is to use “Camel-Casing”
  - `exampleVariable`, `thisIsCamelCasing`
- Declaring variables
  - Need to specify data types
  - `string var`
  - `int balance`

Data Types



- Integer
  - `int year;`
  - Consumes 4 bytes / 32 bits in memory
  - Range -2147483648~2147483647 ($-2^{31} \sim 2^{31}-1$) for 32-bit machines
  - 5: 0000 0000 0000 0000 0000 0000 0000 0101
  - -5: 1111 1111 1111 1111 1111 1111 1111 1011
- Double
  - Floating point numbers
  - `double piValue;`
  - Consumes 8 bytes / 64 bits in memory

Assigning Variables

- Set the variable value to the right hand side one of the expression
- Examples
  - `count = count + 1;`
  - `numOfDays = numOfWeeks * 7;`
- Unassigned variables have garbage values
  - To be safe, assign them at declaration
  - `int minimumNumber = 3;`

cout

- `cout` and `cin` are part of the library iostream
- Print to the console
  - `cout << “Hello World!”;`
- Print variable and string together
  - `cout << "This year is " << thisYear << ", and this month is " << thisMonth << ". " << endl;`
- Print newline
  - `cout << thisYear << " is this year. " << endl;`
  - `cout << thisYear << " is this year. \n";`
- Double precision
  - `cout.setf(ios::fixed);`
  - `cout.setf(ios::showpoint);`
  - `cout.precision(2);`

`cin`

- Receives input from console, sets variable equal to the value typed in at the keyboard
- Example
  - `cin >> dateNumber;`
  - `cin >> dateNumber >> hourNumber;`
- Input needs to match the variable type declared ahead
  - Otherwise the program stops

Comment

- `// Write single-line comment`
- ```
  /* This is a multi-line comment
      with more explanations
      and details
   */
  ```
  ```

Compile a program

- Compile using Xcode / Visual C++
- g++ on a Linux server
  - Run these
    - `g++ -g source_code.cpp –o target`
    - `./target`
  - Compiler used on SEASnet server, cannot be installed on local machines
  - Encouraged to run your programs here before submit (so that you can get error messages from different perspectives)
  - Used to grade projects
- Only compiled language need compilation
  - Compiled language (C++, C, Java), the compiler will translate the program directly into machine code of the target machine
  - Interpreted language (Python, Bash), the source code is not directly run by the target machine

Remote Linux server

- Make sure you are in UCLA network
- Access the machine using `ssh`
  - `ssh USERNAME@cs31.seas.ucla.edu`
- Set up g31
  - `setupg31`
- Transfer files
  - `scp path/on/local/machine USERNAME@cs31.seas.ucla.edu:path/on/remote`
    - `/path/on/local example: /Users/ma/xcode_workspace/hello/hello/hello.cpp`
    - `/path/on/remot example: ~/Desktop`
  - `sftp USERNAME@cs31.seas.ucla.edu`
    - `put path/on/local/machine`
    - `get path/on/remote/machine`
  - Software like FileZilla

Remote Linux server

- Compile to build an executable file from the source files
  - `g31 –o hello hello.cpp`
- Execute the program
  - `./hello`
- More instruction: g++ with Linux (ucla.edu)

Errors

- Compile error
  - Fails to compile
  - Syntax errors, library errors, link errors, etc
  - Compiler may provide some error message to help us locate the problem
  - For example
    - Missing semicolons, missing code block brackets, missing namespace of #include definitions, misspelled variables or names
- Logical error
  - Compiles successfully
  - Program may run well
  - Program may crash
  - Gives incorrect results/undefined behaviors
  - You can print out intermediate variable values to debug
  - For example: division by 0, overflow

Bit Overflow

- A bit overflow occurs when an arithmetic operation attempts to create a numeric value that is too large to be represented within the available storage space
- For example, assign 99999999999999999999999999999999 to a variable
- A type of logical error

- For example
  - int a = 2147483647 // 0111 1111 … 1111, 2^31-1
  - a = a + 1

Project 1 Tips

- Name files follow the instruction (must by exact match)
- Test on both your local machine and g31 server
- In one project, there can only be one cpp file, if there are multipile main functions in the same project (For example, both original.cpp and logic_error.cpp in the project and both have main functions), then the project won't run
- You can create several projects separately, and copy .cpp files into a folder and zip them (the zip file should contain exactly 3 .cpp files and a report file)
    - Have more than one main function in a project in VS
    - Have more than one main function in a project in Xcode

Thank You