Week 1: Hello World!

Ling Ding

Slides modified from Muhao Chen, with permission

Ling Ding

- Teaching Associate
- Email address: lingding@cs.ucla.edu
- Office Hours:
  - Tuesdays 4:30 ~ 7:30 PM BH3256S
- Personal office
  - BH3551

My research

- 3rd-year PhD student in ScAi Lab
- Databases and Data Management
  - Recursive SQL, Datalog
  - Probabilistic Databases
  - Data Integration, Data cleaning, Provenance
  - Spatial-temporal Databases
  - Etc.

About CS31

- Skills for programming using C++ (without data structures)
- Basic principles of memory allocation
- Basic knowledge of object-oriented programming

Outline

- Review
- How to compile programs
- Debugging using Visual C++
- Project 1

Outline

- *Review*
- How to compile programs
- Debugging using Visual C++
- Project 1

Review

- What is a program?
  - A **sequence of rules and instructions** that describe the logic of specific tasks to be processed by the computer.
    - To calculate some formulas, to train some mathematical models.
    - Operating systems, databases, compilers, network system …
    - Websites, games …

Review

- What is a programming language?

	Human Language	Programming Language	Machine Language
	English Spanish Italian ...	C C++ Java ...	binary numbers
for humans	easy	medium	difficult
for computers	difficult	medium	easy

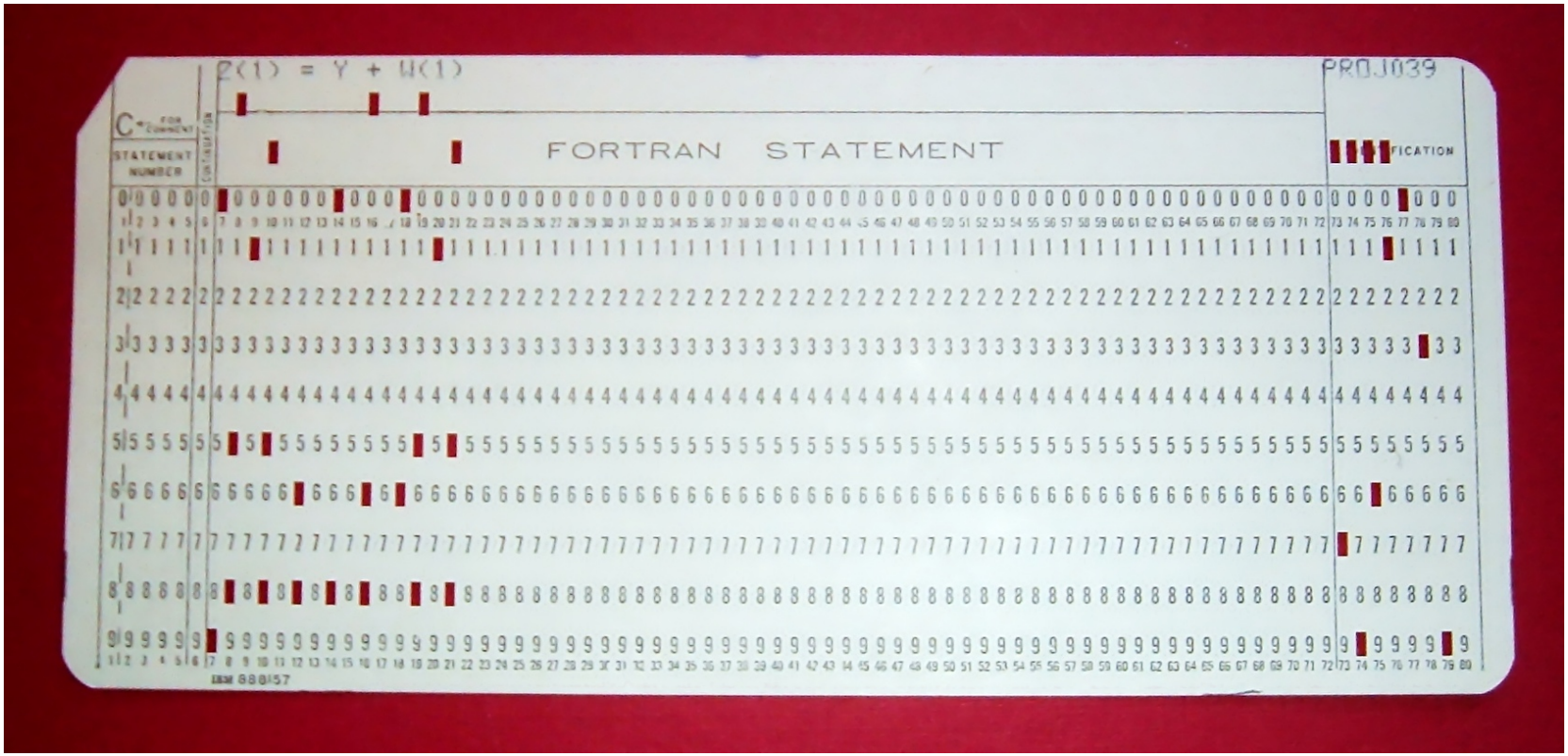
- A language of medium difficulty to both us and computer. We use it to represent the procedural logic of instructions. Machines map it to machine language and execute the instructions.

Review

- Example
- In Human language:
  - Print out “Hello World!” on the screen.
- In a programming language:
  - cout << “Hello World!” << endl;

Compiler.
VC++, g++, etc

- In machine language:
  - 01011011111010101011010110 …



A punch card used in 1960s to program Fortran (the very early programming language which was extremely close to machine languages)

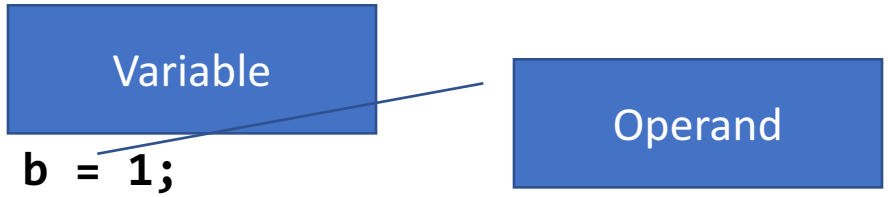
Outline

- Review
- *How to compile programs*
- Debugging using Visual C++
- Project 1

Compile a Program

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

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



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

Compilers

- Compiling with a Visual Studio (VC++)
  - Wysiwyg
  - ** (Choose win32/win64 console application when you create the project!)

- Compiling with g++
  - g++ -g source_code.cpp -o target
  - ./target

Errors

- What is compile error?
  - Fail in compiling.
  - Syntax errors, library errors, link errors, etc.
- What is logical error?
  - Compiled successfully.
  - Program may run well / Or may crash (e.g. infinite loop, over-allocated memory, etc).
  - Give results that are not as expected.

Outline

- Review
- How to compile programs
- *Debugging using Visual C++*
- Project 1

Debugging using Visual C++

- Why do I care?
- Development environment
  - Editor
  - Compiler
  - Debugger
  - IDE
- Debugging
  - Locate a bug
  - Find the cause
  - Fix the bug
- Debugging in action

Why do I care?

- Why don’t you?
  - Write code →Compile → Test → Something wrong → Debug
- Understanding the concepts is critical. Also the “how to “ helps you save hours or days (Instead, you can use that time to sleep, watch movies, …)
- This is a general debugging tutorial, using Visual C++ as a example

Development Environment

- Text Editor
  - Simple application that lets you create raw (unformatted) documents, NO fancy features (NO bullet points, NO underlining, etc.)
- Compiler
  - Complex application that converts your source code to machine language
  - E.g. Microsoft C/C++ compiler | gcc, g++

Development Environment (cont.)

- Debugger
  - Complex application that lets you walk through the execution of your program
  - E.g. Microsoft Visual Studio debugger | gdb
- Integrated Development Environment (IDE)
  - An IDE = Editor + Compiler + Debugger + other fancy features
  - Visual C++ is an IDE

Overview of Debugging

- Bug = error
- Debug = try to fix the error
- Locate the bug
  - Narrow down which lines of code introduce the bug
- Find the bug
  - Understand why those lines of code do the wrong thing
- Fix the bug
  - Now what, how are you going to fix it?
- In this tutorial, we focus on how to use the Debugger to “Locate the bug”

Visual C++ debugger

- Starts vs. Start without Debugging
- Debugging techniques
  - Breakpoints
  - Trace execution (walk through): single stepping
  - Monitor variables and fuction calls
    - Watch variables
    - Function call stack

Start with vs. without Debugging

- Start without Debugging: Ctrl + F5
  - After execution, the console window is still there and you can see the results
  - However, it's not what we want for this tutorial since it won't stop at our breakpoints for us to examine the program
- Start (with Debugging): F5
  - The program will stop at the breakpoints so we can take a deeper look and debug it

Breakpoints

- Why we need them
  - To show the machine world that we have control over them
  - To stop the program at any line of code we want

- How to set/unset them
  - Click on the gray margin on the left side of the line of code
  - Unset: do the same thing

Trace (walk through) your program's execution

- Single stepping
  - Run at a time one piece of your code (one line of code or a function)
- Step in: F11
  - Step into a function
  - Pause at the beginning of the function
- Step over: F10
  - Execute a line of code no matter what it is (assignment, comparison, count, cin, functions…)
  - Pause at the nexe line of code
- Step out: Shift + F11
  - Get out of the function that I'm in right now (execute the rest of the function but don't show me what's going on)
  - Pause at the next line of code right after the funciton call.

Monitor variables and function calls

- Debugger windows:
  - Only available when you're in debugging mode.
  - Useful windows to keep track of your program
    - Breakpoints, variables, function calls
- Watch window:
  - Keep track of your variables' values
  - Short-hand calculator (evaluate variables, expressions)
- Call Stack window:
  - Keep track of function calls (who called me).
  - Useful with recursive calls

Debugging Recap

- IDE = editor + compiler + debugger + other fancy features
- Write code → Compile → Test → Something wrong → Debug
- Debugger: lets you trace/walk through the execution of your program
  - Locate the bug
  - Find the cause
  - Fix the bug
- Debugger techniques:
  - Set break points: stop the program at a line of code
  - Trace execution: single step through your program
  - Monitor variables: look at values of the variables

Debugging techniques and Tips

- Don’t duplicate code. Put repeated/reusable code in a function and call the function.
- To locate your bug, use binary search: repeated divide the search interval in half (put breakpoint at the middle, then at the middle of first half or second half, etc).
- Display line number next to line in the text editor in Visual C++
  - Tools / Options / Text editor / C++
- C++ help:
  - Google: C++ FAQ Lite

Outline

- Review
- How to compile programs
- Debugging using Visual C++
- *Project 1*

Project 1

- http://web.cs.ucla.edu/classes/fall18/cs31/

Project 1

- One thing we should pay attention to
  - In step 5, find input integer values that cause it to produce **incorrect**, **unusual**, or **nonsensical** results.
  - Is this to cause a compile error or a logical error?
  - Note: The variables numberSurveyed, forNewsom, and, forCox are **integer types**, so it is not the case to input floating type values like 12.3456.
  - What values should we input to trigger incorrect results?
  - Incorrect results: numberSurveyed != forNewsom + forCox (e.g. 3000, 2000, 2000)

Unusual or nonsensical results?

Data types

Type	SIze
int (Integer)	4Bytes (=32bits)
double (double precision float)	8Bytes (=64bits)

- Other datatypes: float, long int, unsigned int, char, boolean …

Int

- int: 4Bytes. Range: -2147483648~2147483647 (-2^{31}~2^{31}-1)

5 = 

-5 = 1 111 1111 1111 1111 1111 1111 1111 1011

If S = 0: Value = V
If S = 1. Value = V - 2^{31}

int a=2147483647; // 0 1111 … 1111, i.e. 2^{31} - 1
a += 1;

/// How much is a now?

Bit overflow

- In computer programming, 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

Project 1

- The zip file you submit must follow the instructions **exactly**. (Pay attention to how to name each cpp file and your zip file!)
- Be careful about compile error and logical error.
- Projects submitted after the due time will receive **reduced or no credit**.

Next week

- Data types and variables
- Operators
- Conditions
- Loops
- I/O

Thank you!