Programming Languages and Development Environment

http://net.pku.edu.cn/~course/cs101/2008

Hongfei Yan
School of EECS, Peking University
10/8/2008
Outline

- Programming languages
- Development environment
Evolution

Evolution of computer languages

<table>
<thead>
<tr>
<th>Machine Languages</th>
<th>Symbolic Languages</th>
<th>High-Level Languages</th>
<th>Natural Languages (Evolving)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1940s</td>
<td>1950s</td>
<td>1960s</td>
<td>1970s</td>
</tr>
<tr>
<td>1980s</td>
<td>1990s</td>
<td>2000s</td>
<td></td>
</tr>
</tbody>
</table>
计算机语言的演化

• 机器语言
– 指令由 0 和 1 串组成，程序可直接执行

• 符号语言
– 用符号表示不同的机器语言指令，必须被汇编（assembler）为机器语言

• 高级语言
– 从关注计算机转移到关注问题本身。用语句（statement）构造应用逻辑。
– 通常不能直接执行，要编译（compile）或解释（interpret）执行。
Computer language

• The term **computer language** includes a large variety of artificial languages used to communicate with computers.
  – It is broader than the more commonly-used term **programming language**.
  – **Programming languages** are a subset of computer languages.
    • For example, HTML is a **markup language** and a computer language, but it is not traditionally considered a programming language.
  – **Machine code** is a computer language.
    • It can technically be used for programming, and has been (e.g. the original bootstrapper for [Altair BASIC](#)),
    • though most would not consider it a programming language.
Types of Computer Languages

• **Programming languages**
  – are the primary means by which developers of computing systems instruct a machine to organize or manipulate information or control physical devices.
  – Most software is written using one or more programming languages.
  – Common examples include C, C++, Java, BASIC, assembly languages, and many others.

• **Scripting languages**
  – are designed to control other software or to coordinate the actions of multiple software applications.

• **Machine code**
  – a non human-readable binary computer language which corresponds to the available instructions for a microprocessor.

• ……. 
What problem gave birth to programming languages?

• Before high level programming languages existed,
  – computers were programmed one instruction at a time using binary or hex.
    • a tedious job and a lot of errors.
  – Obviously, this job did not attract many people, so there was a shortage of programmers.
    • Expensive computers sat idle for long periods of time while software was being developed.
  – Software often cost two to four times as much as the computer.

• This led to the development of assemblers and assembly languages.
  – Programming became somewhat easier,
  – but many users still wanted floating point numbers and array indexing. Since these capabilities were not supported in hardware,

• high level languages had to be developed to support them.
What Makes a Language Hot?

- Ease of use
- Language Features
- Performance
- Corporate Support
- Experienced Programmers
- Legacy Code
Evolution of Programming Languages
PASCAL

• Pascal

N. Wirth 1984
1968

•

Blaise Pascal

• Pascal

Pascal
Niklaus Wirth

• In 1984 he won the Turing Award for developing a sequence of innovative computer languages.
  – Euler, Algol W, Pascal, Modula, Modula-2 and Oberon.

• His article *Program Development by Stepwise Refinement*, about the teaching of programming, is considered to be a classic text in software engineering.

• In 1975 he wrote the book *Algorithms + Data Structures = Programs*, which gained wide recognition and is still useful today.
生活在十九世纪中叶，是著名诗人拜伦的女儿——设计了巴贝奇分析机上解伯努利方程的一个程序，并证明当时的计算机狂人巴贝奇的分析器可以用于许多问题的求解。她甚至还建立了循环和子程序的概念。由于她在程序设计上的开创性工作，被称为世界上第一位程序员。

ADA

- Ada
- 1983-1995

- Ada
  1995

- Ada
  1983
• C programming language
  – was devised by Bell Labs of AT & T
  – as a system implementation language for the nascent Unix operating system.

• Contribution:
  – Efficiency
  – Casting
  – Flexible array Library
  – Good portability
C

- C是由美国贝尔实验室的Dennis Ritchie在1972年设计开发的，开发目的是想成为一种编制"系统程序"的工具语言。

- Retchie等人首先用自己发明的C语言编写了Unix操作系统。以后C语言逐步发展成为开发系统软件的主要语言。

- C语言已成为最重要的软件系统开发语言，由此可见C语言在计算机领域地位之重要。
SIMULA

• Although never widely used, SIMULA is historically important.
  – Developed from 1964-67, SIMULA was designed as a language to aid in creating simulations.
  – SIMULA I was introduced in 1964, its sole purpose was system simulation.
  – Immediately after SIMULA I was finished, the designers decided to work on a new language which would contain many more features so that it could be used for more general purpose areas.
  – The result was SIMLUA 67.

• To help make simulation easier, SIMULA 67
  – introduced the concept of classes, including instancing and coroutine.
  – This was the beginning of data abstraction.
C++

- C++ was designed to provide
  - Simula's facilities for program organization
  - together with C's efficiency and flexibility for systems programming.
  - It was intended to deliver that to real projects.

- Contribution:
  - One line comment
  - Inline function
  - Multiple inheritance
  - Overloading operators and functions
C++

• C++ 是 C 语言基础上发展出的一种 “面向对象” 语言。它是由 Bjarne Stroustrup 在美国贝尔实验室开发的 (1979)。

• C++ 是 C 语言的一个扩充，– 修正了 C 语言的一些弱点和不足，使用起来更方便可靠；
– 以支持 “面向对象” (Object-Oriented, 简称为 OO) 的程序设计方法为基本目标，提供了一套支持面向对象程序设计的机制，如 “类” (class)、“对象” (object) 等等。

• 面向对象的方法被认为是开发复杂软件系统的一种有效途径，OO 程序设计语言也已经被广泛接受。

• C++ 是目前使用最广泛的一种面向对象的程序设计语言。
我们这门课程主要介绍 C++ 语言
Notations

• Programming languages
  – A notation for describing algorithms and data structure

• Program
  – A sentence of a programming language
Outline

• Programming languages
• Development environment
  –  
  –  
  –  
The compilation system

- The programs that perform the four phases (preprocessor, compiler, assembler, and linker) are known collectively as the compilation system.
- `gcc -o hello hello.c`
Building a program

```cpp
#include <iostream.h>

int main (void)
{
    int a;
    int b;
    ...
    cin >> ... ;
    ...
    return 0;
}
```
Typical C++ Development Environment

- **Edit**
  - Programmer writes program (and stores source code on disk)

- **Preprocess**
  - Perform certain manipulations before compilation

- **Compile**
  - Compiler translates C++ programs into machine languages

- **Link**
  - Link object code with missing functions and data

- **Execute**
  - Transfer executable image to memory
  - Execute the program one instruction at a time
GNU

•

– 编辑工具：vi
– 编译工具：g++
– 调试工具：gdb
程序调试

• 通常情况下，可执行文件中不包含对源程序的引用信息，如变量名，函数名，行号等。

• g++提供了`-g`开关，将源程序的信息存放在目标文件和可执行文件的符号表中，允许–调试器(debugger) gdb跟踪程序的执行–当程序崩溃的时候，检查程序崩溃前的状态。

– (debugger) gdb
– 
– $ gdb program
<table>
<thead>
<tr>
<th>常用命令</th>
<th>功能</th>
</tr>
</thead>
<tbody>
<tr>
<td>break [file] function</td>
<td>设置断点，或者break [file] linenumbers</td>
</tr>
<tr>
<td>run [arglist]</td>
<td>启动并调试程序</td>
</tr>
<tr>
<td>bt</td>
<td>显示程序堆栈</td>
</tr>
<tr>
<td>where</td>
<td>显示当前位置</td>
</tr>
<tr>
<td>print expr</td>
<td>打印表达式的值</td>
</tr>
<tr>
<td>c</td>
<td>continue，继续运行</td>
</tr>
<tr>
<td>next</td>
<td>执行下一行，跳过函数</td>
</tr>
<tr>
<td>step</td>
<td>执行下一行，跳进函数</td>
</tr>
<tr>
<td>list [file] function</td>
<td>显示程序停止位置的函数</td>
</tr>
<tr>
<td>help [cmd]</td>
<td>显示cmd命令的使用</td>
</tr>
<tr>
<td>quit</td>
<td>退出</td>
</tr>
</tbody>
</table>
"Hello, world!" in C++

```cpp
1 // HelloWorld.cpp
2 #include <iostream>
3 using namespace std;
4
5 int main()
6 {
7     cout << "Hello, world!\n";
8     // cin.get();
9
10    return 0;
11 }
```

```
$g++ -g -Wall HelloWorld.cpp -o HelloWorld
```
References

• cs101 course Q&A on Sept. 30, 2008

• MSYS tutorial
  http://net.pku.edu.cn/~course/cs101/2008/resource/tutorial1.doc

• Linux command tutorial
• 项目（Project）：编写程序的工作是以Project为单位。在开始一个新程序时，要先建立一个Project，之后在程序编写过程中所有与这个程序有关的文件都会包含在这个Project中。编制的程序可以有各种不同类型，编程环境为每种类型的程序准备了一个模板，用来生成程序的最初框架。在本课程里只介绍编写Console Application类型的程序。这类程序的特点是：程序运行中会打开一个类似于DOS操作系统的界面，所有键盘输入都是通过DOS界面进行的，而所有输出都是输出到DOS窗口中。
Source File

-  
-  
-  
-  
- C++
Compile

•

Link

•

Build

•
Debug

• 当程序出错时，可用调试工具发现错处的代码，进行改正。
• 所谓调试是指逐条执行或部分执行程序代码，并在执行过程中查看变量的值。
  – 当发现变量的值并非如我们预期或程序的执行逻辑并非我们预期时，就发现了错误，可以有针对性的改正。
Run

•

•

•
CodeBlocks

• 创建新工程
• 编辑源程序
• 编译/连接
• 运行
• 调试
CodeBlocks

[Image: Screenshot of CodeBlocks IDE with the New from template dialog open, showing various project templates including AVR Project, Code::Blocks plugin, Console application, Library, Emotion, GTK+ project, Inhibit project, Kernel Mode filter, Mach, GLUT project, Empty project, R-TK project, OpenGL project, and Core project.]
CodeBlocks
CodeBlocks
CodeBlocks
CodeBlocks
CodeBlocks
CodeBlocks
CodeBlocks
CodeBlocks
CodeBlocks
CodeBlocks
//     a b
#include <iostream>
using namespace std;

int main()
{
    int a,b;

    cin >> a >> b;
    cout << a + b << endl;
    // cin.get();

    return 0;
}
```cpp
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    float n[10], sum=0;
    for (int i=0; i<10; i++) {
        cin >> n[i];
        sum += n[i];
    }
    cout << "The average is " << fixed << setprecision(2) << sum/10 << endl;
    return 0;
}
```
Fred wants to buy land in Louisiana to build a house. In his investigation, he learned that due to the erosion of the Mississippi River, Louisiana is shrinking at a rate of 50 square miles per year. Since Fred wants to live in his new house until he dies, he wants to know if his house will be submerged.

After further research, Fred found that the land under erosion is semi-circular, with its center at (0,0). The straight edge of the semi-circle is the X-axis. The problem is to determine the year in which a given point (x, y) with y ≥ 0 will be submerged, given its coordinates.

Example: If the point (1.0, 1.0) is given, the output is: 

End of the first year.
```cpp
#include <iostream>
#include <math.h>
using namespace std;

int main()
{
    float x, y;
    int year;
    cin >> x >> y;
    year = (int) ceil (3.1416*(x*x + y*y)/2/50);
    cout << "Be corroded after " << year << " years.\n";
    return 0;
}
```