The Introduction of C++

闫宏飞

Institute of Network
Peking University

Oct 9, 2007
1. 简介
2. 语言的演化
3. C and C++
4. 程序设计环境
5. 简单程序
6. 课程小结
A computer program is a collection of instructions that describe a task, or set of tasks, to be carried out by a computer.

The term computer program may refer to source code, written in a programming language, or to the executable form of this code. Computer programs are also known as software, applications programs, system software or simply programs.
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(assemble)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
Machine Languages

- only language computer directly understands
  - “Natural language” of computer
  - Defined by hardware design
- Generally consist of strings of numbers
  - Ultimately 0s and 1s
- Instruct computers to perform elementary operations
- Cumbersome for humans
- Example
  - +1300042774
  - +1400593419
  - +1200274027
Assembly Languages

- English-like abbreviations representing elementary computer operations
- **Clearer** to humans
- Incomprehensible to computers
- Convert to machine language by translator programs (**assemblers**)
- Example
  - load basepay
  - add overpay
  - store grosspay
High-Level Languages

- Similar to everyday English
  - Uses common mathematical notations
- Single statements accomplish substantial tasks
- Converted to machine language by translator programs (compilers)
- Interpreter programs
  - Directly execute high-level language programs
  - Execute more slowly than the compiled program
- Example
  - grossPay = basePay + overTimePay

C++ Programming Style

http://docs.google.com/Doc?id=dfmjmnmq_24fnwtdq&pli=1
程序设计

程序＝ 算法 + 数据结构

- 程序设计是一种目标明确的智力劳动
- 程序的质量决定了软件的质量
- 概念、工具、方法以及方法学
程序设计的基本步骤

1. 分析(analysis): 分析问题并建立模型
2. 设计/design): 数据结构(data structure)和算法(algorithm)设计
3. 编码(coding): 用一种程序语言实现设计方案
4. 调试(debugging): 排查程序中的错误
History of C

- Evolved from BCPL and B
  - Developed by Dennis Ritchie (Bell Laboratories)

- Development language of UNIX

- Hardware independent
  - Can write portable programs

- ANSI and ISO standard for C published in 1990
  - ANSI/ISO 9899: 1990
History of C++

- **Extension of C**
  - Developed by Bjarne Stroustrup (Bell Laboratories) in early 1980s

- Provides new features to “spruce up” C

- Provides capabilities for object-oriented programming
  - Objects: reusable software components
    - Model items in the real world
  - Object-oriented programs
    - Easier to understand, correct and modify
Programming languages - C++

`C++` is a general purpose programming language based on the `C` programming language as described in ISO/IEC 9899:1990 Programming languages - `C(1.2)`.

In addition to the facilities provided by `C`, `C++` provides additional data types, classes, templates, exceptions, namespaces, inline functions, operator overloading, function name overloading, references, free store management operators, and additional library facilities.
The fundamental storage unit in the C++ memory model is the byte. A byte is at least large enough to contain any member of the basic execution character set and is composed of a contiguous sequence of bits, the number of which is implementation-defined. The least significant bit is called the low-order bit; the most significant bit is called the high-order bit. The memory available to a C++ program consists of one or more sequences of contiguous bytes. Every byte has a unique address.
C++ Standard Library

- **C++ programs**
  - Built from pieces called classes and functions

- **C++ Standard Library**
  - Rich collections of existing classes and functions
  - Reusable in new applications
Reusable software components that model real world items

Meaningful software units
- Time objects, paycheck objects, record objects, etc.
- Any noun can be represented as an object

More understandable, better organized and easier to maintain than procedural programming

Libraries of reusable software
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

- **Load**
  - Transfer executable image to memory

- **Execute**
  - Execute the program one instruction at a time
Typical C++ Development Environment (cont.)

http://net.pku.edu.cn/~course/cs101/image/cpp_env.png
**Input/output**

- **cin**
  - Standard input stream
  - Normally inputs from keyboard

- **cout**
  - Standard output stream
  - Normally outputs to computer screen

- **cerr**
  - Standard error stream
  - Displays error messages
### 独立开发工具

编辑、编译、调试等是单独的工具，通常是命令行界面

- **编辑工具**: vi
- **编译工具**: g++
- **调试工具**: gdb
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

运行并调试

```
$ gdb program
```
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>break [file]function</td>
<td>设置断点，或者break [file]linenumber</td>
</tr>
<tr>
<td>run [arglist]</td>
<td>启动待调试程序</td>
</tr>
<tr>
<td>bt</td>
<td>backtrace, 显示程序栈</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, 跳过函数入口</td>
</tr>
<tr>
<td>step</td>
<td>执行下一行, 跳进函数入口</td>
</tr>
<tr>
<td>list [file]function</td>
<td>显示程序停止位置的源程序</td>
</tr>
<tr>
<td>help [cmd]</td>
<td>显示cmd命令的使用帮助</td>
</tr>
<tr>
<td>quit</td>
<td>退出</td>
</tr>
</tbody>
</table>
"Hello, world!" is the first program one usually writes when learning a new programming language. The first Hello World program appeared in chapter 1.1 of the first edition of Kernighan & Ritchie’s original book about C, "The C Programming Language", in 1978.
"Hello, world!" in Assembly

```
$vi hello_world.asm
$asm -f elf hello_world.asm
$ld -s -o hello_world hello_world.o
$./hello_world

Hello, world!
```
section .text
    global _start ;must be declared for linker (ld)

msg    db  'Hello, world!',0xa ;our dear string
len    equ  $ - msg ;length of our dear string

_start: ;tell linker entry point
    mov    edx,len ;message length
    mov    ecx,msg ;message to write
    mov    ebx,1 ;file descriptor (stdout)
    mov    eax,4 ;system call number (sys_write)
    int    0x80 ;call kernel

    mov    eax,1 ;system call number (sys_exit)
    int    0x80 ;call kernel
"Hello, world!” in C++

HelloWorld.cpp

```cpp
#include <iostream>

// using namespace std;

int main(int argc, char* argv[]) {
    std::cout << "Hello, world!\n";
    // cin.get();
}

$g++ -g -Wall HelloWorld.cpp -o HelloWorld
```
Object orientation

- A natural way of thinking about the world and computer programs

Unified Modeling Language (UML)

- Graphical language that uses common notation
- Allows developers to represent object-oriented designs
Objects

- Reusable software components that model real-world items
- Examples are all around you
  - People, animals, cars, telephones, microwave ovens, etc.
- Have attributes
  - Size, shape, color, weight, etc.
- Exhibit behaviors
  - Babies cry, crawl, sleep, etc.; cars accelerate, brake, turn, etc.
Object-oriented design & Object-oriented language

- **Object-oriented design (OOD)**
  - Models real-world objects in software
  - Models communication among objects
  - Encapsulates attributes and operations (behaviors)
    - Information hiding
    - Communication through well-defined interfaces

- **Object-oriented language**
  - Programming in object oriented languages is called object-oriented programming (OOP)
  - C++ is an object-oriented language
    - Programmers can create user-defined types called classes
    - Contain data members (attributes) and member functions (behaviors)
Object-oriented design & Object-oriented language (Cont.)

- **Object-Oriented Analysis and Design (OOAD)**
  - Analyze program requirements, then develop solution
  - Essential for large programs
  - Plan in pseudocode or UML
The Unified Modeling Language

**History of the UML**
- Used to approach OOAD
- Object Management Group (OMG) supervised
  - Brainchild of Booch, Rumbaugh and Jacobson
- Version 2.1 is current version

**UML**
- Graphical representation scheme
- Enables developers to model object-oriented systems
- Flexible and extendible
教材 & 参考书

  - 中文版译名“C++大学教程-(第五版)”

Wrap-up

- discussed the different types of programming languages
- presented basic object technology concepts
  - including classes, objects, attributes, behaviors, encapsulation and inheritance
- learned about the history and purpose of the UML
  - UML the industry-standard graphical language for modeling software systems