Introduction to Computer and C++ Programming

Dr. Z. Aliyazicioglu
Cal Poly Pomona
Electrical & Computer Engineering

Computer System

Hardware
Software

Computer hardware
CPU - Primary Storage
Input Devices
Output Devices
Auxiliary Storage Devices

Computer Software

Software
System Software
Application Software

- System Software: manage the hardware resources of a computer and perform required information processing tasks.
- Operating system provides services such as a user interface, file and database access, and interfaces to communication
- System support software provides system utilities.
- System development software converts your program into machine language for execution

Application software:
- General purpose software is purchased from a software developer and can be used more than one application
- Application specific software can be used only for its intended purpose.

Time-Sharing Environment

- Many user are connected to one or more computers. These computers may be minicomputer or central mainframe. All of the computing must be done by the central computer
Client/Server Environment
Split the computing function between a central computer and users’ computers. Users’ computers are called the client.

History of Computer Languages
We must use a computer language to write a program for a computer. Hundreds of computer languages are in use today.
- 1940s Machine Languages
- 1950s Assembly Languages
- 1960s High Level Languages
- 1970s
- 1980s
- 1990s Natural Languages

High-Level Languages
- FORTRAN FORmula TRANslations
- BASIC Beginners All-purpose Symbolic Instructional Code
- PASCAL
- COBOL C0mmon Business Oriented Language
- C Combines the power of an assembly language with the ease of use and portability of a high-level language
- ADA Designed for real-time distributed systems.
- LISP Used in artificial intelligence applications

History of C and C++
- BCPL language was developed in 1967 by Martin Richards for writing operating system and compiler.
- Ken Thomas modeled many features in his language B after BCPL and used it to create early version of UNIX Operating system at Bell Lab.
- C language was evolved from B by Dennis Ritchie at Bell Lab in 1972. C used many important concepts of B and BCPL and adding other features.
- C++ language was evolved from C by Bjarne Stroustrup in early 1980s at Bell Lab. It has new features and important one is that it is object-oriented programming.
- Visual C++ is developed in 1990s. Developers use graphical tools to create applications.

Introduction to Microsoft.NET
- Microsoft introduced .NET in June 2000
- Strategy is its independence from a specific language or platform
- It is new software development model that allows applications created in disparate programming languages to communicate each other
- Creates Web-based applications

Structured Programming
- In 1960s, people realize that software development was complex activity.
- Writing structured program is clear to understand, easy to test and debug, and easy to modify.
- Pascal programming language was developed in 1971 and Blaise Pascal was designed for teaching structured programming in academic area.
- ADA programming language was developed during 1970s under DOD. One important capability of ADA is multitasking. This allows to programmer to specify many activities are to occur parallel.
Object Technology
- The goal is building software quickly, correctly, and economically.
- Software developers discovered that using a modular, object-oriented design and implementation approach can make software much more productive.
- Object-oriented programs are often easier to understand, correct and modify.
- Object-oriented programming became widely used in the 1980s.

C++ Standard Library
- C++ has rich collections of existing classes and functions in the C++ standard library.
- First is learning the C++ standard library.
- Second is learning how to use the classes and functions.

C++ Standard Library
- C++ has rich collections of existing classes and functions in the C++ standard library.
- First is learning the C++ standard library.
- Second is learning how to use the classes and functions.

Problem Solving
Art
- Ask the right questions in order to clarify the problem.
- Analyze the problem to extract its essential features.
- Determine whether any constraints or simplifying assumptions can be applied to facilitate the problem solution.

Science
- Knowledge of the problem environment
- Knowledge of the formulas or equations that characterize the environment

Good Problem Solver → Good Programmer

Software Development Method
1- Requirements Specifications
- State the problem and gain a clear understanding of what is required for its solution
2- Analysis
- Identify the problem's inputs
- Outputs
- Constraints on the solutions.
3- Design
- Develop a list of steps called an algorithm to solve the problem.
- Verify that it is correct.

Software Development Method (cont)
4- Implementations
- Implement the algorithm as a program
5- Verification and Testing
- Test the program and verify that it meets the original requirements specified for the program.
- Don't rely on just one test case. Run the program using several different sets of data.

Debugging
- Making mistakes is a natural part of learning.
- Debugging is the finding and fixing of program mistakes (errors)

Porting
- Adapt the program so that it runs on a different computer system.
Stepwise Refinement

- The refinement of an outline into more detailed steps can be done with:
  - Pseudocode uses English-like Statements to describe the steps in an algorithm
  - Flowchart uses a diagram to describe the steps in algorithm

Case Study
Finding the Area and circumference of a circle

1- Take the radius of a circle and compute and print its area and circumference
2- Inputs: Circle radius
   Outputs: Area of the circle
            Circumference of the circle
   Constants: PI = 3.14159
   Formula: area = \pi r^2
             circumference = 2 \pi r
3- Get circle radius
   Calculate area
   Calculate circumference
   Display area and circumference

Psedocode & Flowchart

<table>
<thead>
<tr>
<th>Operation</th>
<th>Pseudocode</th>
<th>Flowchart</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>read radius</td>
<td>read radius</td>
</tr>
<tr>
<td>Compute</td>
<td>set area to ( \pi ) * radius^2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>and circumference to 2 * \pi * radius</td>
<td></td>
</tr>
<tr>
<td>Output</td>
<td>print radius, area and circumference</td>
<td>print radius, area</td>
</tr>
</tbody>
</table>

Source Code

```c
// Calculate and displays the area and circumference of a circle
#include <stdio.h>
#define PI 3.14159

int main(void)
{
    double radius, area, circum;
    printf(" Enter radius > ");
    scanf("%lf ", &radius);
    area = PI * radius * radius;
    circum = 2 * PI * radius;
    printf(" The area is %f\n", area);
    printf("The circumference is %f\n", circum);
    return(0);
}
```

Using .NET Environment

- Start up Microsoft Visual studio.NET. The following window should be displayed
  - Click New Project
  - Under the File menu, point to New, and then click Project
Click the Application Settings text on the left side.
The right side dialog box is changed to show the
current wizard settings.
Select Console Application and Empty Project, click
finish.

Adding a C++ Source File to the project
Either right-click the prog1 icon in Solution Explorer, point to
Add, and then click Add New Item, or
Click the Add New Item Button on the toolbars.

Select C++ File (.cpp) from the Templates pane on
the right.
Type prog1.cpp in the Name box, and click Open.

Type the source code for the program, as shown
here.

To build the executable, select Build Solution from
the Build menu.
If no error, the message is Build: 1 succeeded, 0 failed, 0 skipped.
Once you have successfully built the project,
Choose Start Without Debugging from the Debug menu to run the program

My first program

```
// Program: pr1.cpp
// A first program in C++
#include <iostream>
using std::cout;

int main()
{
    cout << "Welcome to C++!\n";
    return 0; // indicate that program ended successfully
}
```

Main is a program building block called a function. It prints a string on the screen.

Some Common Escape Sequence

- \n New line. Goes to beginning of the next line.
- \t Horizontal tab. Move the screen cursor to the next tab stop.
- \r Carriage return. Position the screen cursor to the beginning of the current line.
- \a Alert. Sound the system.
- \ Backslash. Print a backslash character.
- \" Double quote. Print double quote character.

Example

```
// pr2.cpp
// Printing a line with multiple statements
#include <iostream>
using std::cout;

int main()
{
    cout << "Welcome ";
    cout << "to C++!\n";
    return 0; // indicate that program ended successfully
}
```

Output

Example

```
// pr5.cpp
// Printing multiple lines with a single statement
#include <iostream>
using std::cout;

int main()
{
    cout << "Welcome\n\nto\n\nC++!\n";
    return 0; // indicate that program ended successfully
}
```

Using new-style header files

```
// Using new-style header files
#include <iostream>

using namespace std;

int main()
{
    cout << "Welcome to C++!\n";
    cout << "Welcome to C++!\n";
    return 0; // indicate that program ended successfully
}
```

Output
Example:

```cpp
// Program: pr6.cpp
// Printing multiple lines to draw a head
#include <iostream>
using std::cout;
using std::endl;

int main()
{
    cout << "    ||||||||||||||||  " <<endl;
    cout << "    |              |  " <<endl;
    cout << "    |    o    o    |  " <<endl;
    cout << "   _|              |_ " <<endl;
    cout << "  |_                _|" <<endl;
    cout << "    |   |_______|  |  " <<endl;
    cout << "    |              |  " <<endl;
    cout << "    \_____________/  " <<endl;
    return 0;  // indicate that program ended successfully
}
```

```
// prog4.cpp
// Printing multiple lines to draw glasses face
#include <iostream>
using std::cout;
using std::endl;

int main()
{
    cout << "    ||||||||||||||||  " <<endl;
    cout << "    |              |  " <<endl;
    cout << "    |    _    _    |  " <<endl;
    cout << "    |  -|o|--|o|- |  " <<endl;
    cout << "   _|    - - |_ " <<endl;
    cout << "  |_                _|" <<endl;
    cout << "    |   |______|   |  " <<endl;
    cout << "    |              |  " <<endl;
    cout << "    \_____________/  " <<endl;
    return 0;  // indicate that program ended successfully
}
```

An Addition Program

```cpp
// prog5.cpp
// Addition program
#include <iostream>
using std::cout;
using std::cin;
using std::endl;

int main()
{
    int integer1, integer2, sum;  // declaration
    cout << "Enter first integer
";
    cin >> integer1;  // read an integer
    cout << "Enter second integer
";
    cin >> integer2;  // read an integer
    sum = integer1 + integer2;  // assignment of sum
    cout << "Sum is " << sum << endl;  // print sum
    return 0;  // indicate that program ended successfully
}
```

### Arithmetic Operation

<table>
<thead>
<tr>
<th>C++ Operation</th>
<th>Algebraic Operation</th>
<th>C++ Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addition</td>
<td>+</td>
<td>f + 7</td>
</tr>
<tr>
<td>Subtraction</td>
<td>-</td>
<td>p - c</td>
</tr>
<tr>
<td>Multiplication</td>
<td>*</td>
<td>b * m</td>
</tr>
<tr>
<td>Division</td>
<td>/</td>
<td>x / y</td>
</tr>
<tr>
<td>Modulus</td>
<td>%</td>
<td>r mod s</td>
</tr>
</tbody>
</table>

### Precedence of Arithmetic Operators

<table>
<thead>
<tr>
<th>Operators</th>
<th>Operations</th>
<th>Order of Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>()</td>
<td>Parentheses</td>
<td>Evaluate first: If there are several, evaluate left to right</td>
</tr>
<tr>
<td>*,/,%</td>
<td>Multiplication Division Modulus</td>
<td>Evaluate second: If there are several, evaluate left to right</td>
</tr>
<tr>
<td>+,-</td>
<td>Addition Subtraction</td>
<td>Evaluate last: If there are several, evaluate left to right</td>
</tr>
</tbody>
</table>

### Equality and relational Operators

<table>
<thead>
<tr>
<th>Standard algebraic equality or relational operator</th>
<th>C++ equality or relational operator</th>
<th>Example of C++ condition</th>
<th>Meaning of C++ condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>==, !=</td>
<td>==, !=</td>
<td>x == y, x != y</td>
<td>x is equal to y, x is not equal to y</td>
</tr>
<tr>
<td>&gt;, &gt;=</td>
<td>&gt;, &gt;=</td>
<td>x &gt; y, x &gt;= y</td>
<td>x is greater than y, x is greater than or equal to y</td>
</tr>
<tr>
<td>&lt;, &lt;=</td>
<td>&lt;, &lt;=</td>
<td>x &lt; y, x &lt;= y</td>
<td>x is less than y, x is less than or equal to y</td>
</tr>
</tbody>
</table>

**Meaning of C++ condition**

- `==`: equality operator
- `!=`: inequality operator
- `>`: greater than operator
- `>=`: greater than or equal to operator
- `<`: less than operator
- `<=`: less than or equal to operator
Using equality and relational operations

```cpp
#include <iostream>
using std::cin;
using std::cout;
using std::endl;

int main()
{
    int num1, num2;
    cout << "Enter two integers, and I will tell you
    the relationships they satisfy: ";
    cin >> num1 >> num2; // read two integers

    if ( num1 == num2 )
        cout << num1 << " is equal to " << num2 << endl;
    if ( num1 != num2 )
        cout << num1 << " is not equal to " << num2 << endl;
    if ( num1 < num2 )
        cout << num1 << " is less than " << num2 << endl;
    if ( num1 > num2 )
        cout << num1 << " is greater than " << num2 << endl;
    if ( num1 <= num2 )
        cout << num1 << " is less than or equal to " << num2 << endl;
    if ( num1 >= num2 )
        cout << num1 << " is greater than or equal to " << num2 << endl;

    return 0; // indicate that program ended successfully
}
```

Homework-1

Question 1

Write a program that print your initial in large format such as in 8 lines

Output:
```
Your initial: 8
```

Question 2

Write a program that inputs three integer from the keyboard and print the sum, average, product, smallest, and largest of these numbers. The screen dialogue should appear as follow

```
Input three different integers: 13 27 14
Sum is 54
Average is 18
Product is 4914
Smallest is 13
Largest is 27
```

Problem

Given the algebraic equation
\[ y = ax^3 + y \]

Write the correct C++ statements for this equation.

```
y = a * pow(x, 3) + y;
y = a * (x * x * x) + y;
```
Using only the techniques you learned in this chapter, write a program that calculates the squares and cubes of the number from 0 to 10 and uses tabs to print the following table of values.

<table>
<thead>
<tr>
<th>Number</th>
<th>Square</th>
<th>Cube</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>27</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>64</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
<td>125</td>
</tr>
</tbody>
</table>