CS2313 Computer Programming

LT1 – Introduction to Programing

香港城市大學
City University of Hong Kong
專業 創新 胸懷全球
Professional · Creative
For The World

Outline

- About the Course
- What is a Computer?
- What is a Computer Program?
- Programming Language
- Programmer and Artist
- Basic Concept of Programming
- Simple Program

CS2313 Computer Programming

Prof. Wang, Shiqi (王詩淇)

Research:

Image/Video Compression
Visual Quality Assessment
Artificial Intelligence
Image Processing, Retrieval and Analysis
Multimedia forensics

Office: ***AC1-Y6414***
Phone: 3442 7341
Email: shiqwang@cityu.edu.hk
Website: http://www.cs.cityu.edu.hk/~shiqwang/

You teach me!

Language: Mandarin (Putonghua), English, Cantonese?

Research

Photo Quality Assessment

Image Synthesis

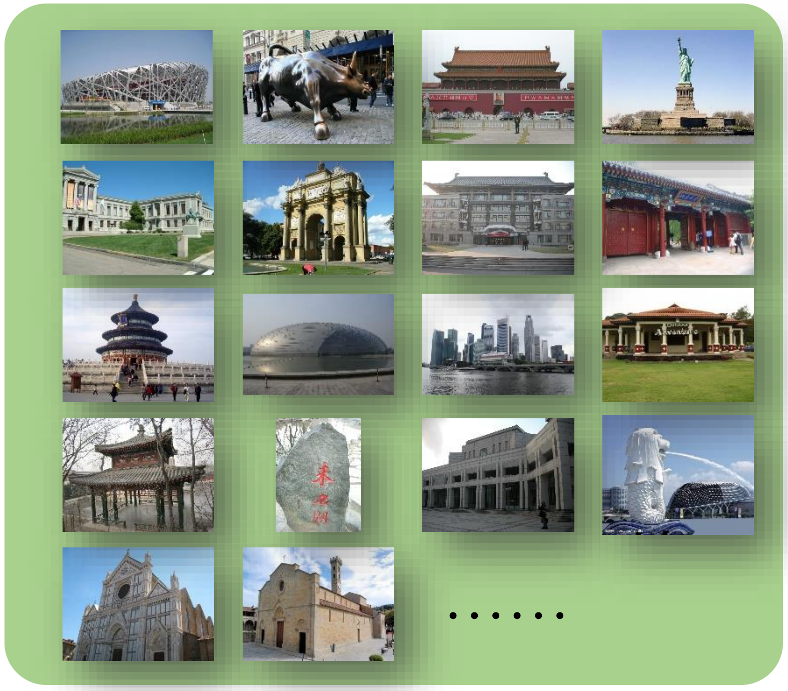


Image Retrieval

http://genekogan.com/works/style-transfer/

What You Will Learn

Course outcome

Program in **C++**

1. Explain the structure of an **object-oriented** computer program.
2. **Analyze**, **test** and **debug** computer programs.
3. **Solve** a task by applying effective programming techniques, which involve advanced skills like using recursion and dynamic data structures.
4. **Design** and **construct** well-structured programs with good programming practices.

Lecture and Lab Sessions

- Lecture and Lab sessions
  - Three hours **lecture** in Lecture Theater
  - One hours “**hands-on**” practice in lab.
  - Analyzing simple problems and implementing computer programs.

Course Assessment

- **Coursework: 40%**
  - One midterm Quiz: **20%**
  - Two assignments: **7% + 8%**
  - Date will be announced later
  - Lab exercise**: 5%**
    - Submit your program with ***correct**** output

- **Final Exam: 60%**
  - Two hour exam.

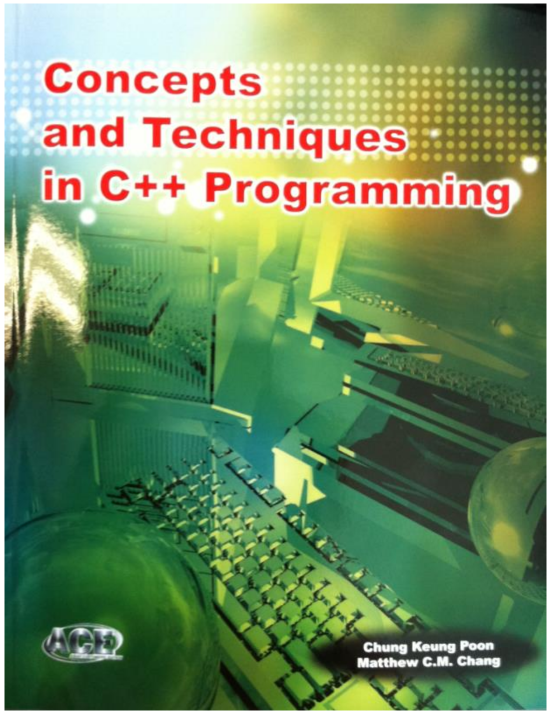
Course Schedule (tentative)

Wk	Lecture Topic	Lab Topic
1	Introduction, simple programs	Intro
2	The C++ programming language, operators, data Types	Simple programs & PASS
3		
4	Flow control (if, switch)	Simple programs & operators
5	Flow control (for, while)	Flow control (if, switch)
6	Arrays (1D and 2D)	Flow control (for, while)
7	Functions	Intro to VS Debugger
8	Class and Object I (TBL)	Arrays
9	Mid-term	Functions
10	Class and Object II (TBL)	Class and Object
11	Strings/Pointers (pass by ref)	Strings
12	Pointers (arrays) File I/O, Other topics (if time)	Pointers (pass by ref)
13	Revision	Pointers (arrays), File I/O

Learning Resources

- Course website on **Canvas**
  - Lecture slides.
  - Lab exercises.
  - Announcements.
  - Lecturer, TA.

- Textbook
  - **Main**: ***Concepts and Techniques in C++ Programming***, by Poon and Chang, McGraw Hill, 2007 and new edition.
  - **Reference**: ***Absolute C++***, by Savitch, Pearson/Addition-Wesley, 4th edition and newer.



Learning Resources

- **Visual Studio** software
  - Development platform (edit and compile programs, etc.)
  - Available in CSC Labs, 2/F (through CSC network)
  - Can also download for programming on PC at home
    - Through CS Lab network (Change password after first login)
    - http://msdnaa.cs.cityu.edu.hk

- **PASS** (Program Assignment aSsessment System)
  - Test programs and submit assignments
  - How to access (Change password after first login)
    - https://pass3.cs.cityu.edu.hk/index.jsp

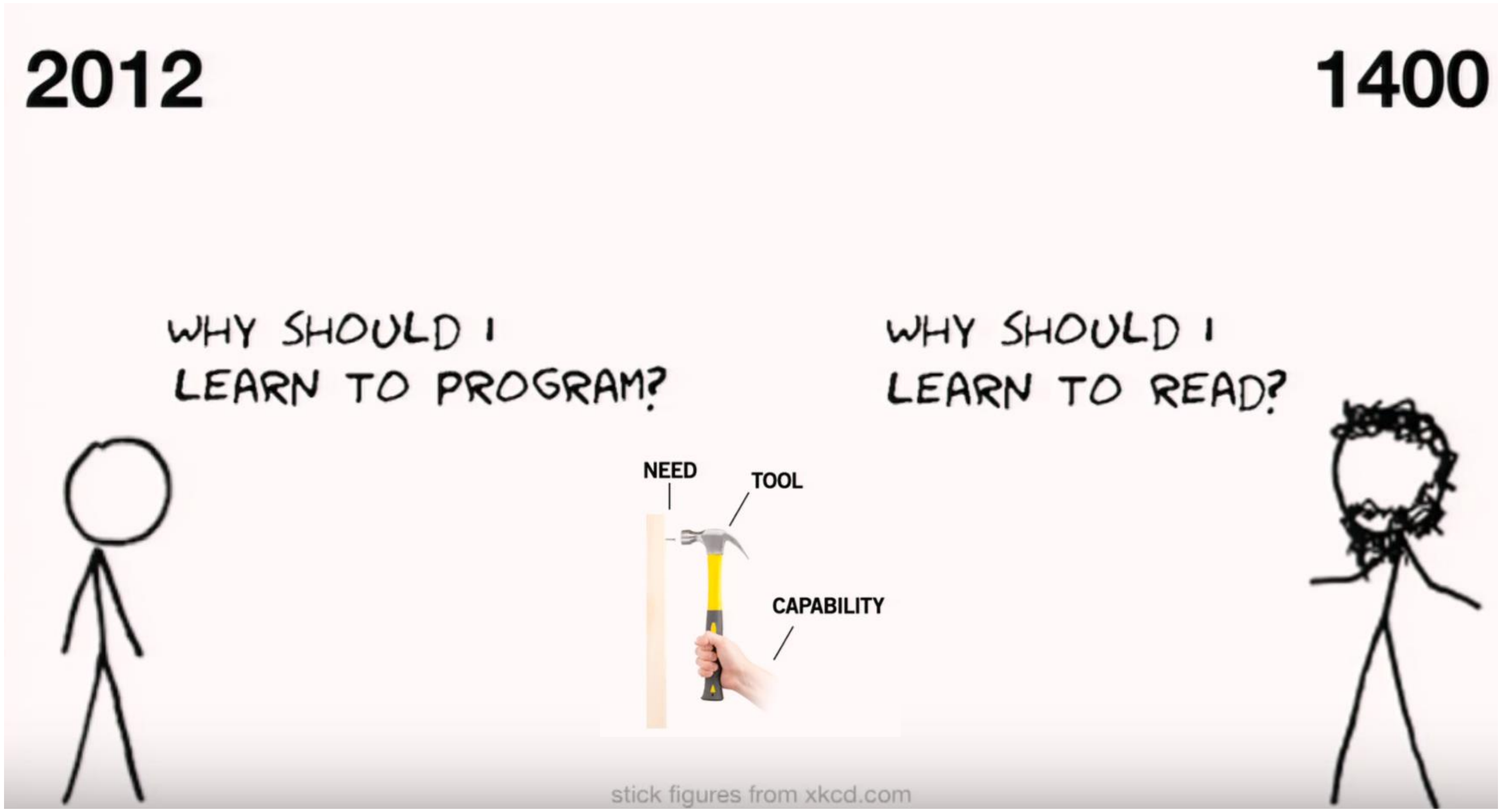
Academic Honesty

- CityU has revised *Rules of Academic Honesty* and has required all students to complete an online tutorial on subject and declare your understanding.
- Plagiarism…
  - It is serious fraud to plagiarize others' work.
  - Punishment ranges from warning to course failure.
- How to prevent plagiarism...
  - Finish the assignments by yourself! You can talk about ideas on how to do the assignment, but you have to write the program yourself.
  - Protect your code; don't give it away as a "reference" copy.
  - In plagiarism cases, we treat the giver and the copier as both guilty.
  - You hurt your own grades by not reporting cheating.
- As instructors...
  - We have responsibility to report academic dishonesty cases so as not to compromise the quality of education.
  - We take suspected plagiarism cases very seriously.

How to Get an A

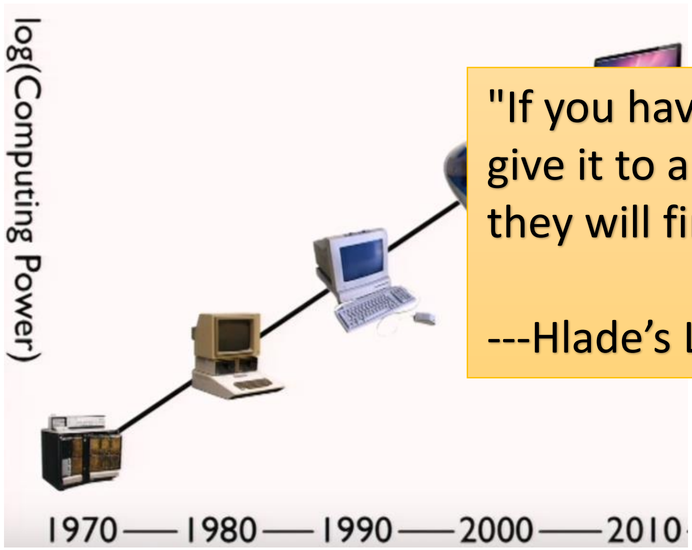
- **Studying…**
  - You are recommended to study the relevant notes or handouts before attending the lecture or lab session.
  - Review as soon as possible to maximize retention.
- **Practice…**
  - **Do the lab exercise yourself** and repeat the practice for better learning.
  - If you get help on the labs, don't just blindly accept it, but try to understand what each part of the code is doing.
  - Do the self-study exercises.
- **Assignments…**
  - Start work on the assignments **when they are released**, and come up with a good plan to finish it.
  - Many times fixing problems in your program will take **longer** than you expect, so make sure you have plenty of time to complete the assignment.

Why you should learn to program


2012
1400
WHY SHOULD I LEARN TO PROGRAM?
WHY SHOULD I LEARN TO READ?
NEED
TOOL
CAPABILITY
stick figures from xkcd.com

Why you should learn to program (Christian Genco)

- Programming makes you smarter
- Computers grow faster
- Lazy
  - If there is a job that takes ten seconds every day, a computer scientist may spend months to make a tool himself to save 5

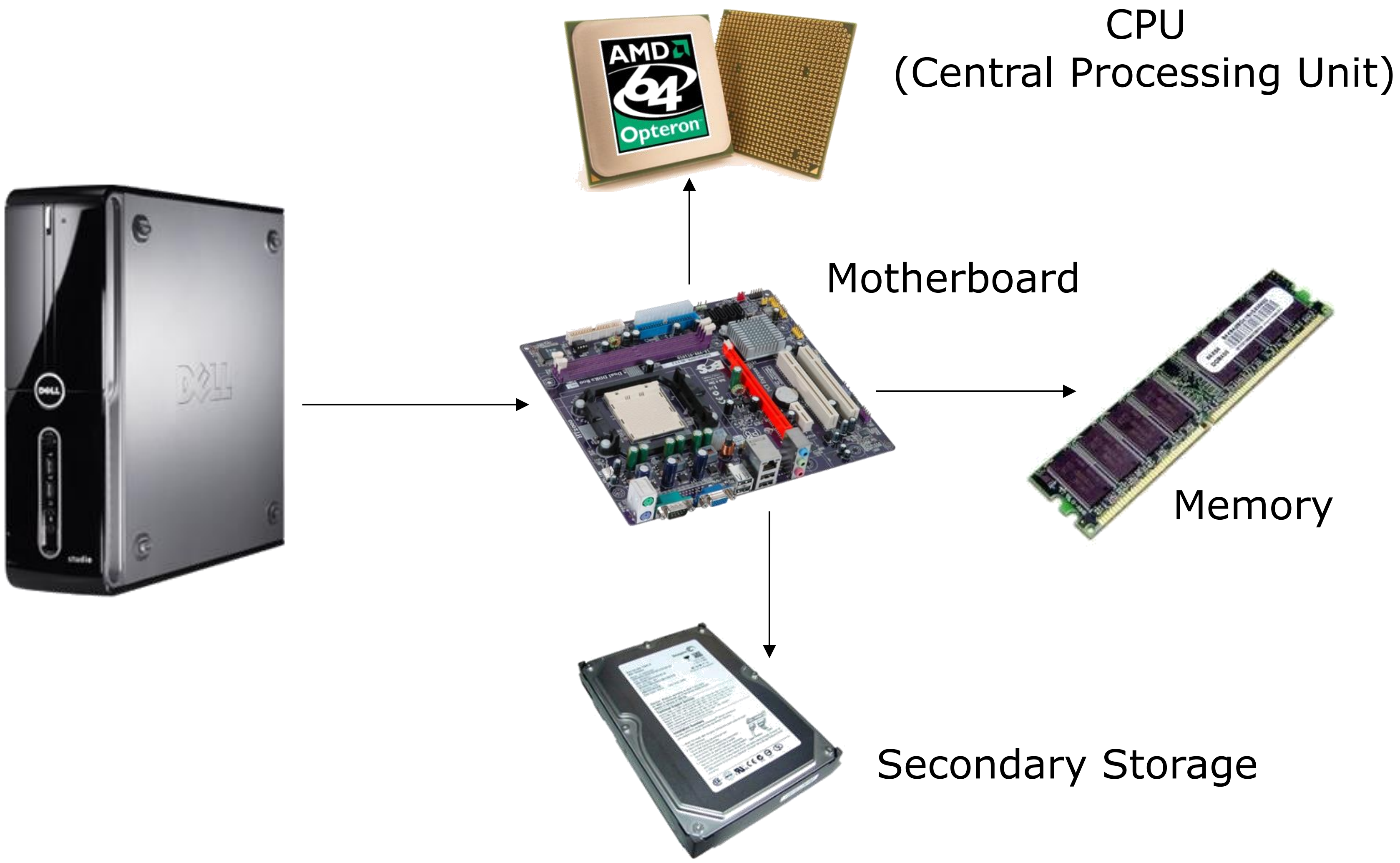


"If you have a difficult task,
give it to a lazy person;
they will find an easier way to do it.“

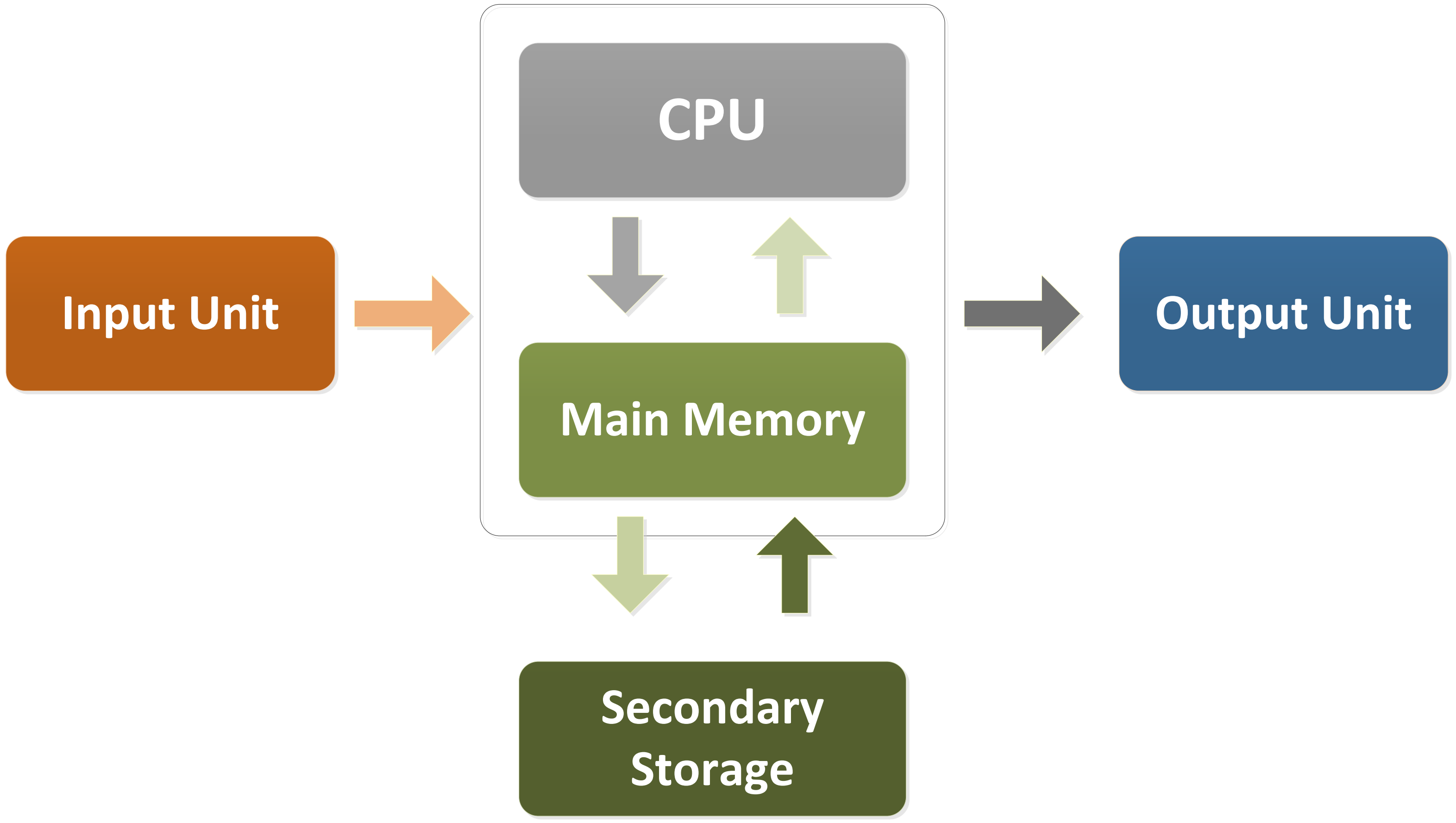
---Hlade’s Law

Any Questions?

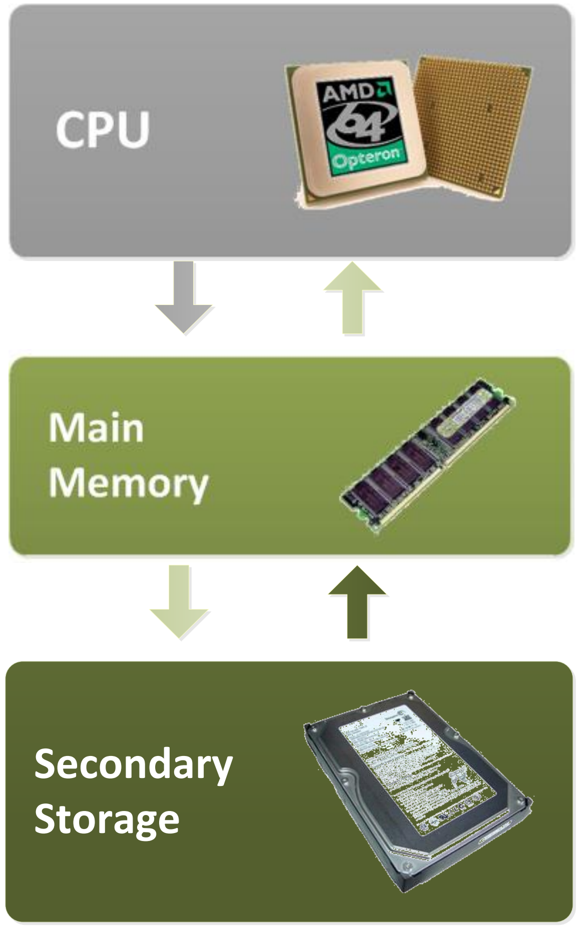
What is a Computer



Stored Program Computer (Von Neumann Machines)



Personal Computer



CPU (Central Processing Unit): Read instruction from main memory and execute the instruction. Update main memory value or send instruction to motherboard.

Main Memory: fast storage of program and data in action.

Secondary Storage: Storage of program and data files.

Personal Computer

Input Unit

Input Unit: Get input from user or external environment.

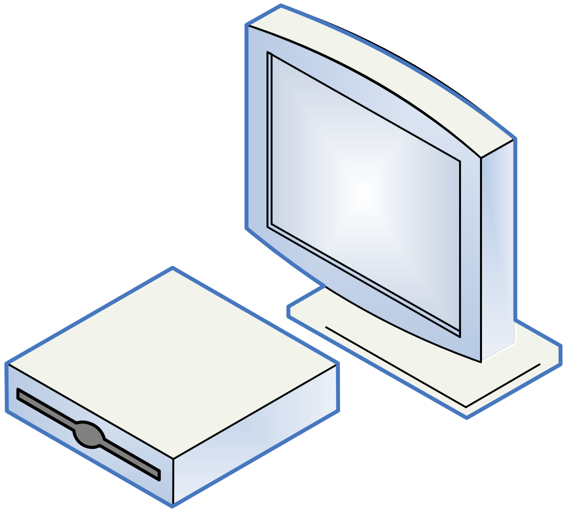
Output Unit

Output Unit: Show result to user or other programs.

What is a Computer Program

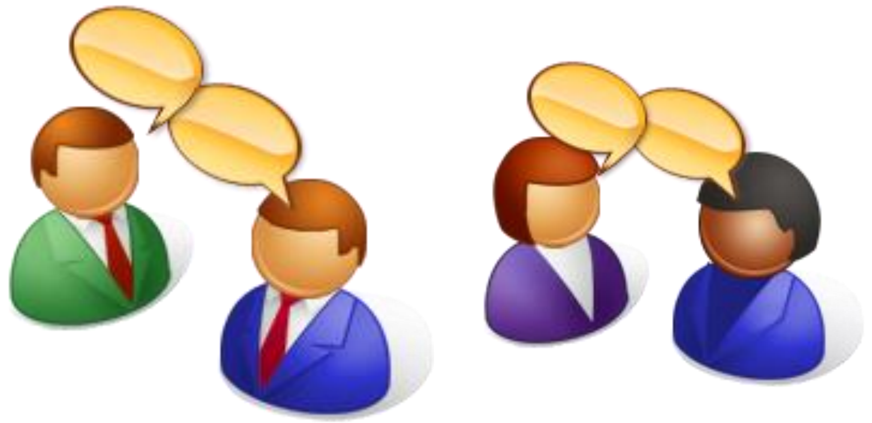
- A list of instructions that instructs a computer to do some tasks.

Timer Recording	Program X
1. Turn on	int x=10;
2. Set Channel to **ch01**	int y=11;
3. Set Date to **1/5/2009**	y+=x;
4. Set Time to **3:00am**	System.out.println(y);
5. Confirm setting	System.out.println(x);



A Computer Program

- A way to **communicate** with computers
- Written in computer Language
- Computer are picky, it needs you to tell them exactly what to do
  - Look out! It is raining hard
    - Be careful?
    - Look out of the window?



Programming Languages

- To write a program for a computer, we must use a **computer language**.

Machine Language

Language directly understood by the computer

binary code

Symbolic Language

English-like abbreviations representing elementary computer operations

assembly language

High-level Language

Close to human language.

Example: *a* = *a* + *b*

[add values of *a* and *b*, and store the result in *a*, replacing the previous value]

C, C++, Java, Basic

PROGRAM 1-1 The Multiplication Program in Machine Language

```
         00000000 00000100 0000000000000000
01011110 00001100 11000010 0000000000000010
         11101111 00010110 0000000000000101
         11101111 10011110 0000000000001011
11111000 10101101 11011111 0000000000010010
         01100010 11011111 0000000000010101
11101111 00000010 11111011 0000000000010111
11110100 10101101 11011111 0000000000011110
00000011 10100010 11011111 0000000000100001
11101111 00000010 11111011 0000000000100100
01111110 11110100 10101101
11111000 10101110 11000101 0000000000101011
00000110 10100010 11111011 0000000000110001
11101111 00000010 11111011 0000000000110100
         01010000 11010100 0000000000111011
                  00000100 0000000000111101
```

The only language understood by computer hardware is machine language.

PROGRAM 1-2 The Multiplication Program in Symbolic Language

```
        entry   main,^m<r2>
        subl2   #12,sp
        jsb     C$MAIN_ARGS
        movab   $CHAR_STRING_CON

        pushal  -8(fp)
        pushal  (r2)
        calls   #2,SCANF
        pushal  -12(fp)
        pushal  3(r2)
        calls   #2,SCANF
        mull3   -8(fp),-12(fp),-
        pusha   6(r2)
        calls   #2,PRINTF
        clrl    r0
        ret
```

Symbolic language uses symbols, to represent the various machine language instructions.

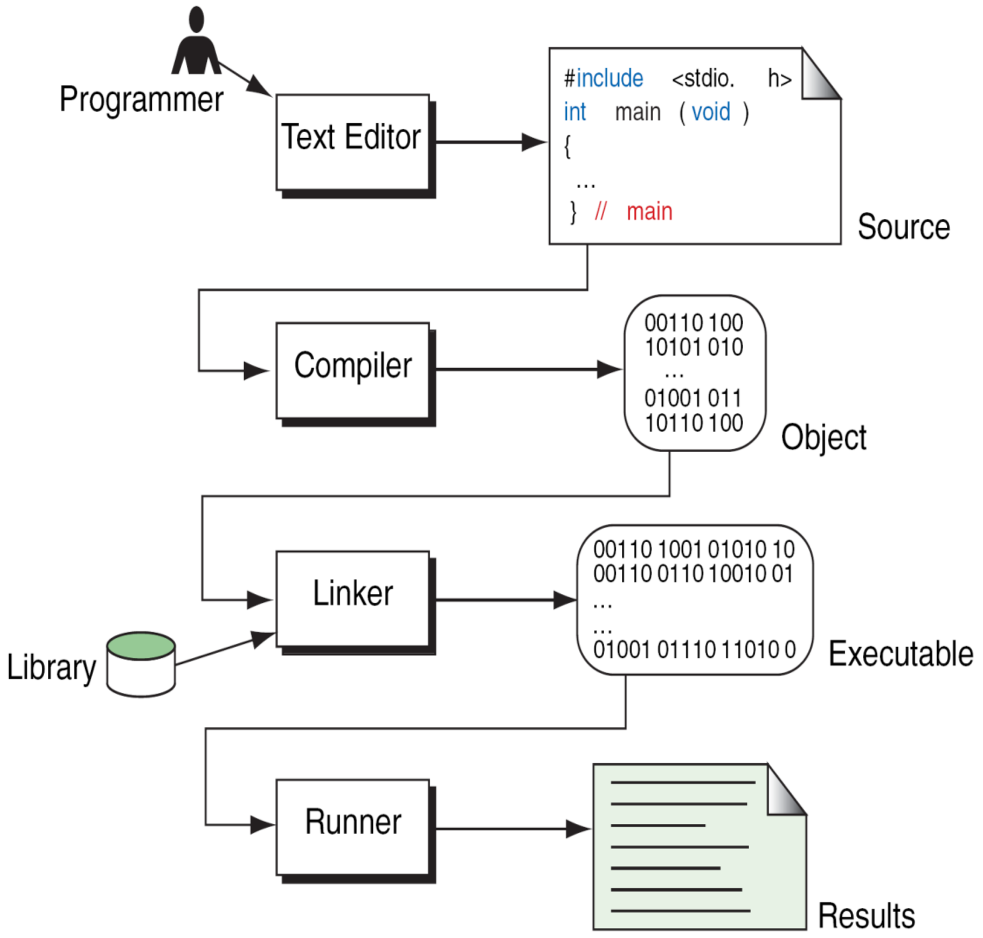
PROGRAM 1-3 The Multiplication Program in C

```c
/* This program reads two integers from the keyboard
   and prints their product.
      Written by:
      Date:
*/
#include <stdio.h>

int main (void)
{
// Local Definitions
   int number1;
   int number2;
   int result;

// Statements
   scanf  ("%d", &number1);
   scanf  ("%d", &number2);
   result = number1 * number2;
   printf ("%d", result);
   return 0;
}  // main
```

high-level languages are easier for us to understand.

Building a C++ Program

- **Writing** source code in C++.
  - e.g. hello.cpp
- **Preprocessing**
  - Processes the source code for compilation.
- **Compilation**
  - Checks the grammatical rules (syntax).
  - Source code is converted to object code in machine. language (e.g. hello.obj).
- **Linking**
  - Combines object code and libraries to create an executable (e.g. hello.exe).
  - Library: common functions (input, output, math, etc).



There are Many Programming Languages in the World!!

ActionScript Ada ASP.NET Assembler Basic C C++ C# Cobol Cobra CODE ColdFusion Delphi Eiffel Fortran FoxPro GPSS HTML J# J++ Java JavaScript JSP LISP Logo LUA MEL Modula-2 Miranda Objective-C Perl PHP Prolog Python SQL Visual Basic Visual Basic.NET VBA Visual-FoxPro

The C Language

- Developed by Dennis Ritchie and Ken Thompson during 1972-1973 at Bell labs.

- The first language they developed at bell labs was known as B language, which was later followed by the C language.
- During 1970s at Bell lab they developed the Unix operating system.
- They found that due to the limitations of B language, it was incapable of building Unix. So they invented C language.

Programming Languages

- Programming languages usually differ in two aspects:
  - Language Syntax.
  - Standard libraries/software development kit (SDKs)/functions.
- Java

```
if (a>b){
     System.out.println("a is larger than b");
}else{
    System.out.println("a is smaller than or equal to b");
}
```

- Pascal

```
if a>b then
    writeln('a is larger than b');
else
    writeln('a is smaller than or equal to b');
```

Programming Languages

- Syntax is well-defined, no exception:
  - `if (…){…}else{…}`
  - `for (;;;){…}`
  - `while (){…}`

- Basic Components:
  - Variable / structure /function declaration/ function access.
  - Conditional statement.
  - Iteration statement.
  - SDK (software development kit)/built-in functions.

Artist and Programmer

Some Difficulties

- Computer only follows instructions. It won’t solve problems by itself.

- Programmer needs to:

1. Develop an appropriate solution (logic).

2. Express the solution in programming language (implementation).

3. Validate the logic and implementation (testing).

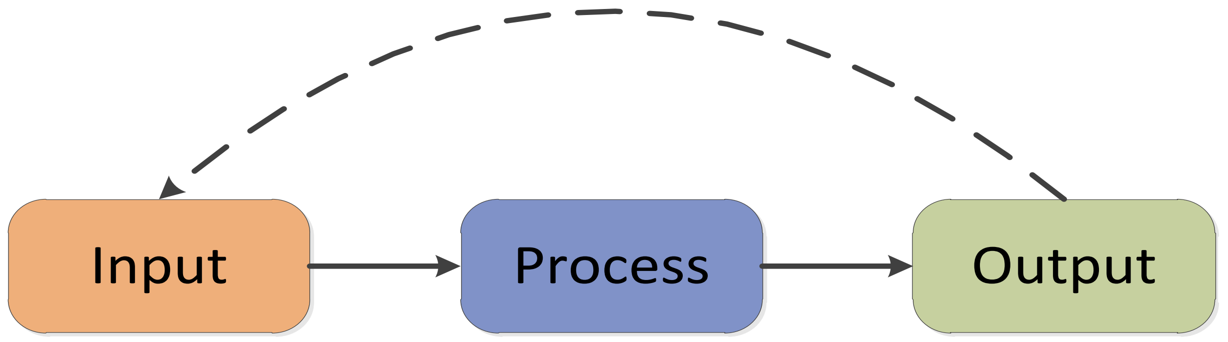
Requirements-Computers are picky

- Correct syntax
- Correct logic
- Efficient
- Running properly under various constraints
- Scalability, Maintainability
- Platform independent

Basic Concept of Programming

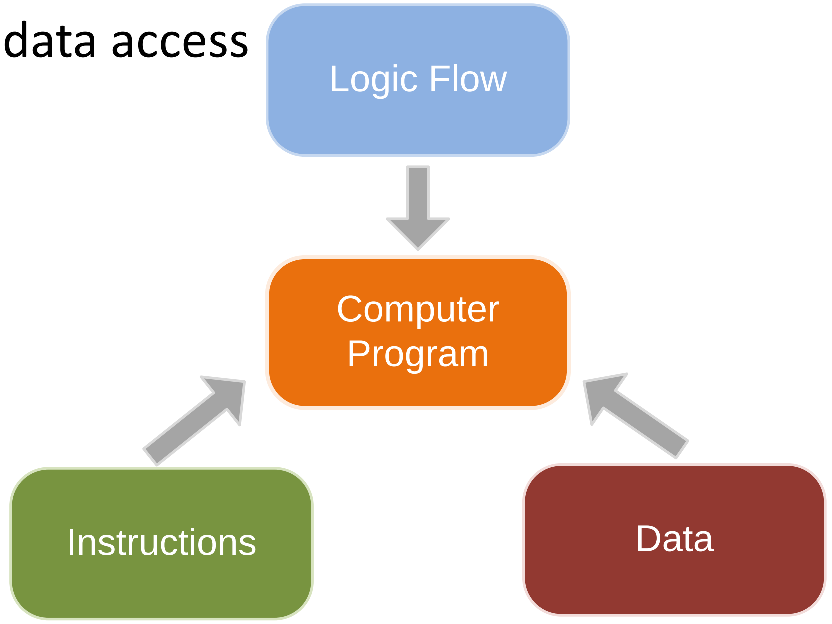
Computer Program (External View)

- Basic elements of a **computer program**
  - Input
  - Process
  - Output



Computer Program (Internal View)

- A list of instructions ordered logically
- Usually involve data access



Computer Program

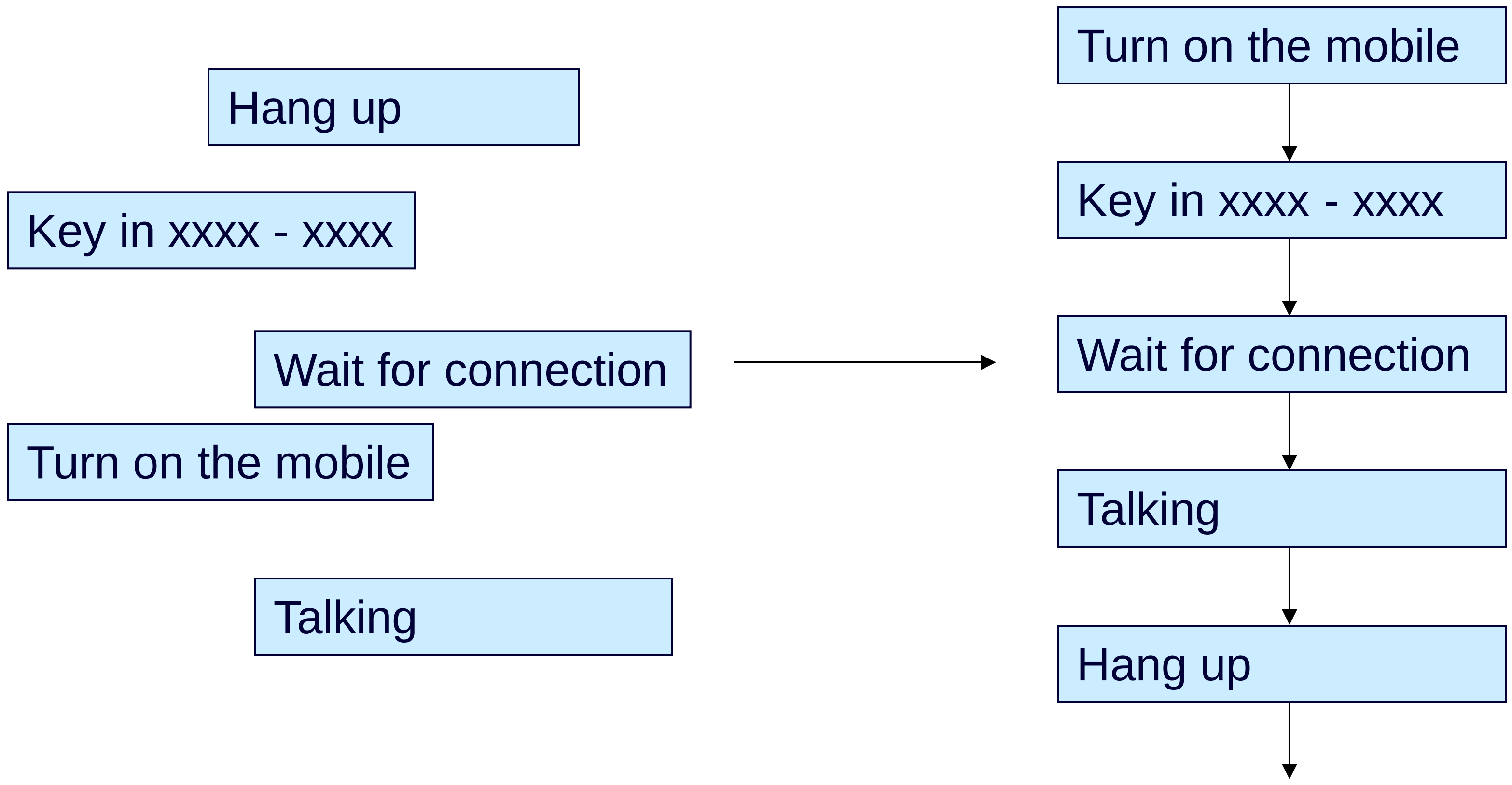
- Instructions
  - A set of predefined action that a computer can perform.
  - E.g. addition, subtraction, read , write.

- Logic Flow
  - Arrangement of Instructions
  - E.g. Calculate BMI **(Body Mass Index)**
    1. Read weight from keyboard.
    2. Read height from keyboard.
    3. Weight x weight/height.
    4. Write BMI to screen.

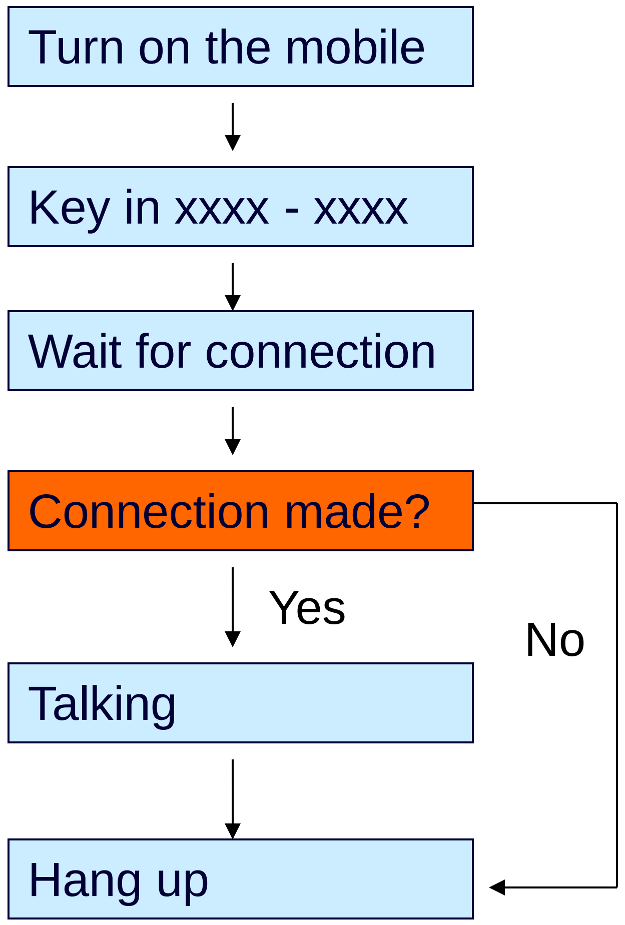
- Variable (data)
  - A space for temporarily store value for future process.

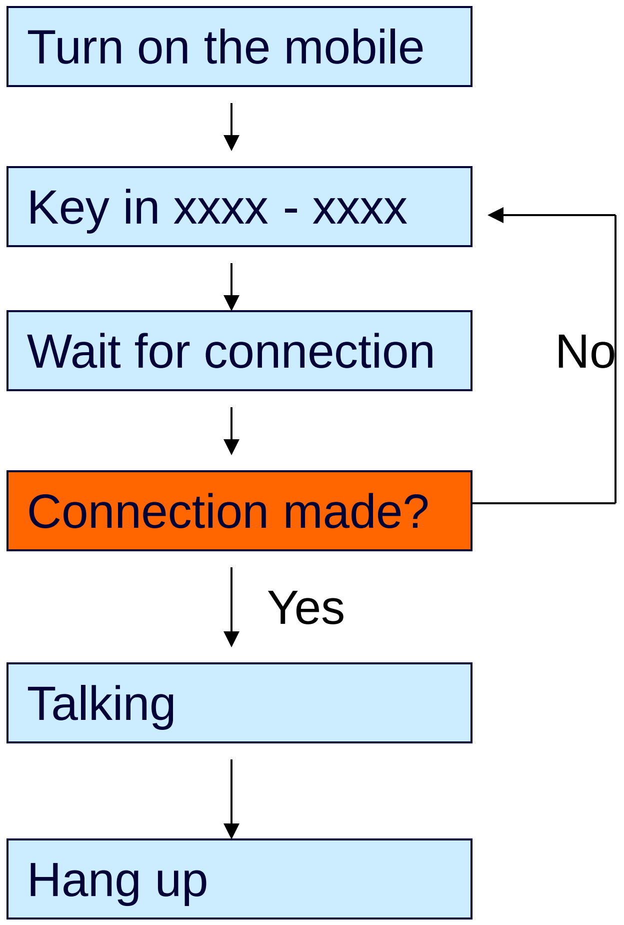
- Constant (data)
  - A value that will not be changed for the whole processing.

Logic Flow

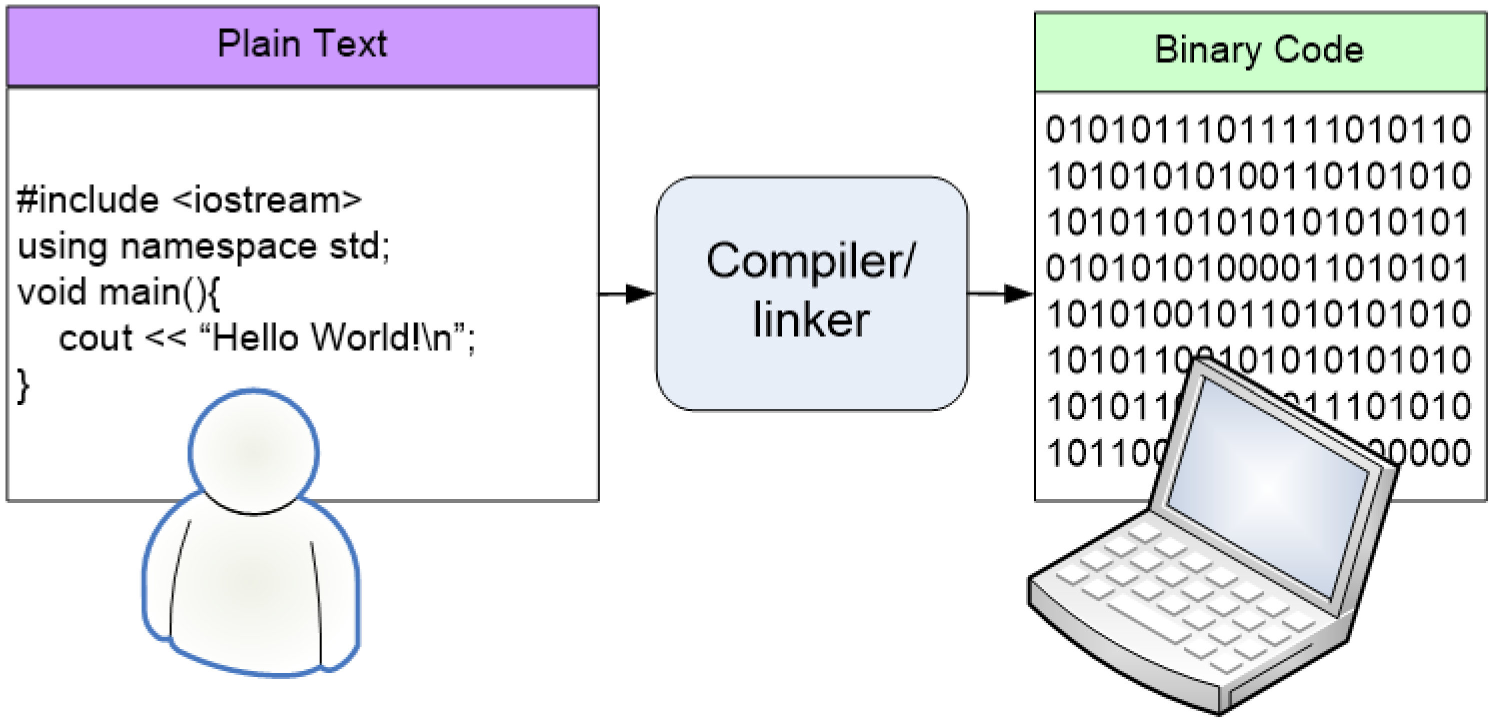

Hang up
Key in xxxx - xxxx
Wait for connection
Turn on the mobile
Talking
Turn on the mobile
Key in xxxx - xxxx
Wait for connection
Talking
Hang up

Logic Flow


Turn on the mobile
Key in xxxx - xxxx
Wait for connection
Connection made?
Yes
No
Talking
Hang up


Turn on the mobile
Key in xxxx - xxxx
Wait for connection
No
Connection made?
Yes
Talking
Hang up

Sample Program (Framework)



```
#include <iostream>
using namespace std;
void main(){
    cout << “Hello World!\n”;
}
```

Comments

```
/* The traditional first program in honor of
   Dennis Ritchie who invented C at Bell Labs
   in 1972 */
```

- Enclosed by “/*” and “*/” or begin with “//”

```
single line comments
// this is a single line comment
// each line must begin with “//” sign
```

Preprocessor Directive

- Give information / instruction to compiler for program creation

```
#include <iostream>
```

- Tells computer to load contents of a certain file / library.
- In this program, we include the library iostream into the program as it contains the definition of cout which is used to print something to the screen.
- No semi-colon at the end of the include directive.

```
using namespace std;
```

- directive
- Specifying that the standard (std) namespace is used such that we can use a shorthand name for the object cout.
  - std::cout <-> cout

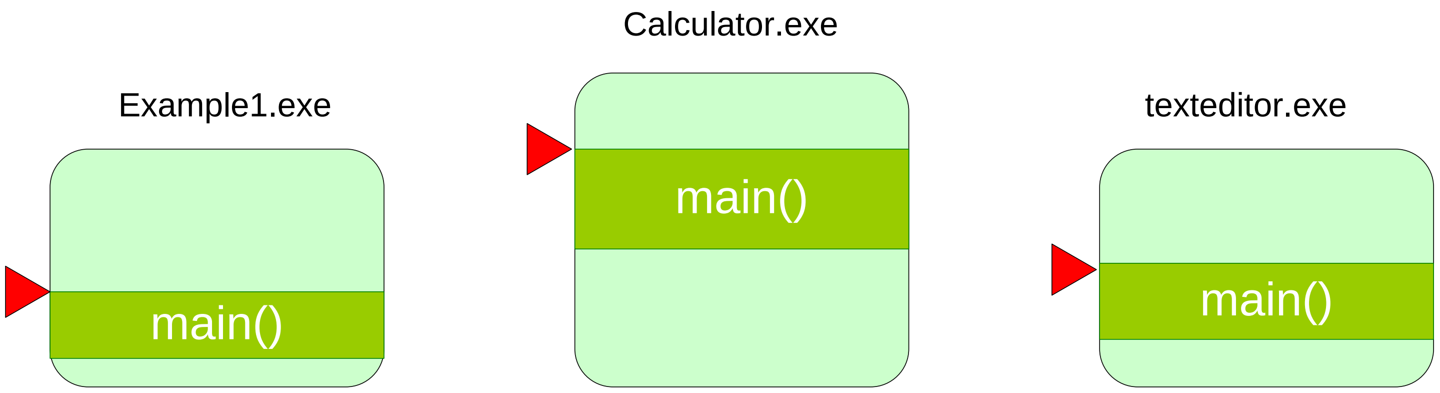
Functions

- When writing program, programmer usually **group related code** (instructions) into **functions** for easy design and maintenance.
- We will talk about function and how to write your own function in **later** lectures.

Function - main

```
void main()
{

}
```

- The starting point of program (the first function called by the computer).



Function - main

- `void main()`
  - `void` means there is no return value.
  - no semi-colon after `main()`.
  - C++ is case sensitive.
  - Incorrect: `void Main(), VOID main(), ...`

- `{ }`
  - Braces: left brace begins the body of a function. The corresponding right brace must end the function.

- Opening and closing
  - (……)
  - {……}
  - [……]
  - …

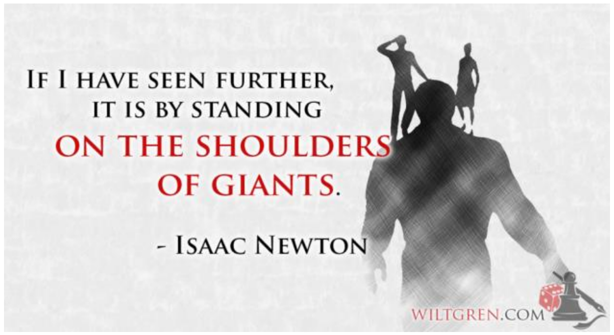
Library / SDK /Package

- Normally, we won’t write a program all by ourselves. Instead, we will reuse the code written by ourselves / other developers. Especially for the repeating tasks or low-level operation like disk I/O.

- The reusing code is well designed and pack a library / SDK / Package.

- Standard C++ program comes with a set of package to make programmer task easier.

- `cout` is one of the example.



Object - cout

```
cout << "Hello, world!\n";
```

- Object is a programming unit that store values (attributes) and provide functions (methods) to manipulate the values (we will elaborate this concept in future classes)

- `cout`: object provided by `iostream` library (package) for screen (console) output

- `<<`: output (also called insertion) operator that output values to an output device. In this case, the output device is `cout` (the screen)

- The value on the right hand side of the operator is the string you want to output

Object - cout

- **`\n`**
  - escape sequence: the character following backslash \ is not interpreted in the normal way.
  - represents a newline character: the effect is to advance the cursor on the screen to the beginning of the next line.
  - newline: position the character to the beginning of next line.

- **`\\`**
  - backslash: Insert the backslash character \\ in a string.

- **`\"`**
  - double quote: Insert the double quote character " in a string.

Simple Program

```
/* The traditional first program in honor of
   Dennis Ritchie who invented C at Bell Labs
   in 1972 */

#include <iostream>
using namespace std;
void main()
{
   cout << "Hello, world!\n”;
}
```

Includes a file

Specifies a library

main function, where to start

Starts a new line

Double quotation marks

Output stream object
Stream insertion operator

Summary

- Basic components of a computer program are:
  - Instructions.
  - Logic Flow.
  - Variable and Constant.
- A correct logic is important in programming.
- Programmer usually reuse code written by the others and the code is commonly in form of library / SDK / packages.
- `cout` is an object provided by iostream package for screen output.

Summary

- A simple C++ program will have

```
#include <iostream>             //A preprocessor
using namespace std;            //namespace declaration
void main(){
           /* the starting point of program
         execution  */
}
```

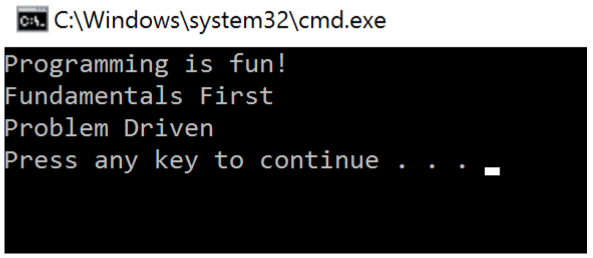
Summary

- Development cycle
  - Write a program in plan text via:
    - Text editor
      - Notepad, UltaEdit.
    - Integrated Development Environment (IDE)
      - E.g. Visual Studio 2015/2017, NetBean.
  - Compile the program
    - IDE / ANSI C++.
  - Execute the program
    - IDE / Console shell.
  - Debug the program

Computer Program

```
#include <iostream>
using namespace std;
int main()
{
  cout << "Programming is fun!" << endl;
  cout << "Fundamentals First" << endl;
  cout << "Problem Driven" << endl;
  return 0;
}
```

stream manipulator



```
Programming is fun!
Fundamentals First
Problem Driven
Press any key to continue . . .
```

Syntax of C++

- Like any language, C++ has an **alphabet** and **rules** for putting together words and punctuation to make legal program; that is called ***syntax*** of the language.

- C++ compiler detects any **violation** of the syntactic rules within the program.

- C++ compiler collects the characters of the program into ***tokens***, which form the basic vocabulary of the language.

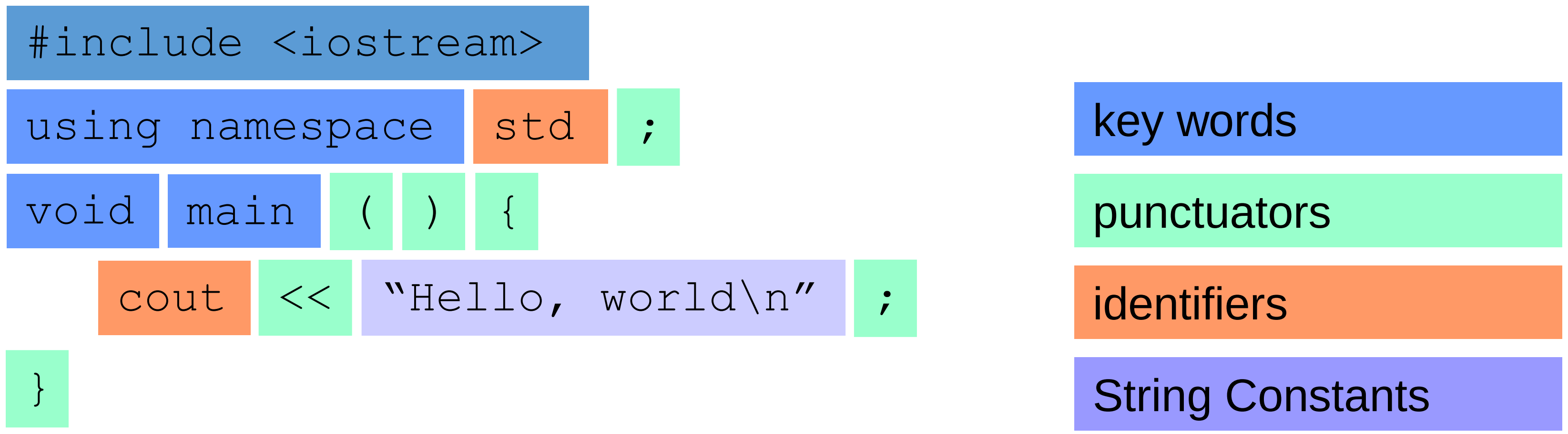
- ***Tokens*** are separated by space.

Syntax - Tokens

- Tokens in C++ can be categorized into:
  - *keywords*, e.g., `main`, `return`, `int`.
  - *identifiers*, e.g., user-defined variables and identifiers used in preprocessing statements.
  - *string constants*, e.g., `“Hello”`.
  - *numeric constants*, e.g., `7`, `11`, `3.14`.
  - *operators*, e.g., `++`.
  - *punctuators*, e.g., `;` and `,`.

Syntax – A Simple Program

```
#include <iostream>
using namespace std;
void main()
{
   cout << "Hello, world!\n";
}
```


#include <iostream>
using namespace
std
;
void
main
(
)
{
cout
<<
“Hello, world\n”
;
}
key words
punctuators
identifiers
String Constants