Object Oriented Programming by C++

C++ Basic (1/2)

Orientation, and Basic Structure of C++ Program

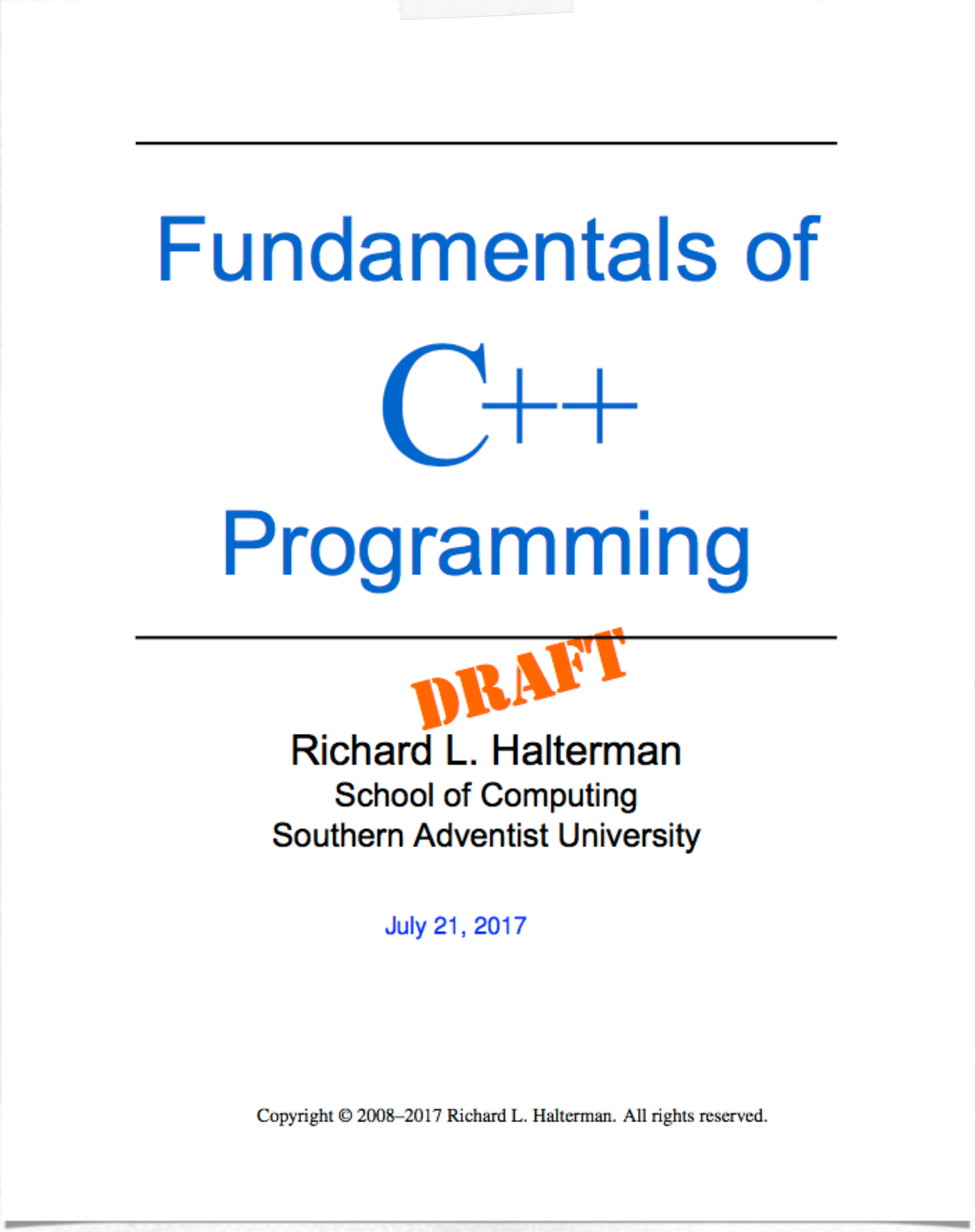
2017. 8.

Sungwon Lee / Professor

Email: drsungwon@khu.ac.kr
Web: http://mobilelab.khu.ac.kr/

Textbook & Copyright

- Textbook: http://python.cs.southern.edu/cppbook/progcpp.pdf
- Sample Codes: https://github.com/halterman/CppBook-SourceCode

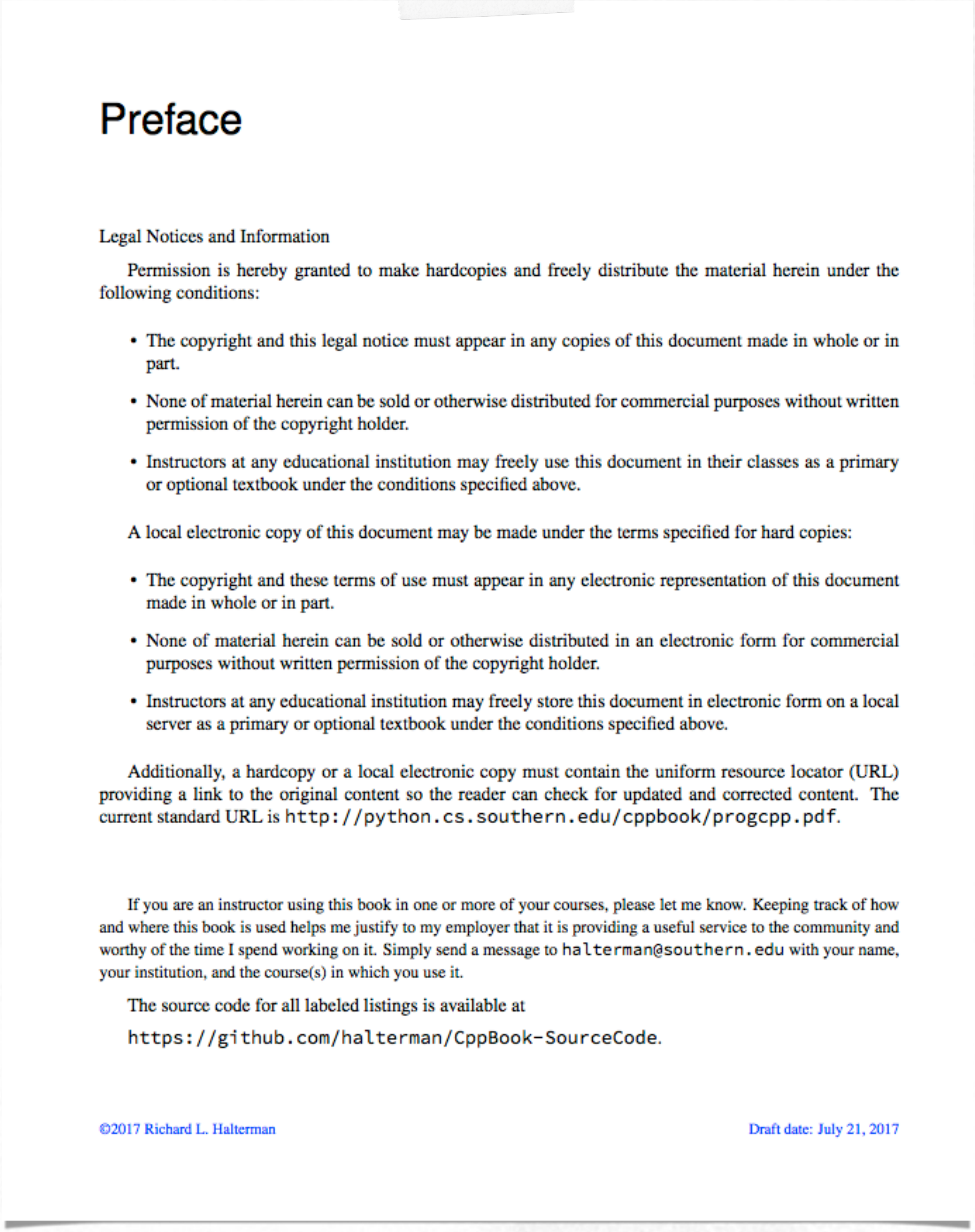


Fundamentals of C++ Programming

DRAFT

Richard L. Halterman
School of Computing
Southern Adventist University

July 21, 2017



Preface

Legal Notices and Information

Permission is hereby granted to make hardcopies and freely distribute the material herein under the following conditions:

- The copyright and this legal notice must appear in any copies of this document made in whole or in part.
- None of material herein can be sold or otherwise distributed for commercial purposes without written permission of the copyright holder.
- Instructors at any educational institution may freely use this document in their classes as a primary or optional textbook under the conditions specified above.

A local electronic copy of this document may be made under the terms specified for hard copies:

- The copyright and these terms of use must appear in any electronic representation of this document made in whole or in part.
- None of material herein can be sold or otherwise distributed in an electronic form for commercial purposes without written permission of the copyright holder.
- Instructors at any educational institution may freely store this document in electronic form on a local server as a primary or optional textbook under the conditions specified above.

Additionally, a hardcopy or a local electronic copy must contain the uniform resource locator (URL) providing a link to the original content so the reader can check for updated and corrected content. The current standard URL is `http://python.cs.southern.edu/cppbook/progcpp.pdf`.

If you are an instructor using this book in one or more of your courses, please let me know. Keeping track of how and where this book is used helps me justify to my employer that it is providing a useful service to the community and worthy of the time I spend working on it. Simply send a message to `halterman@southern.edu` with your name, your institution, and the course(s) in which you use it.

The source code for all labeled listings is available at

`https://github.com/halterman/CppBook-SourceCode`.

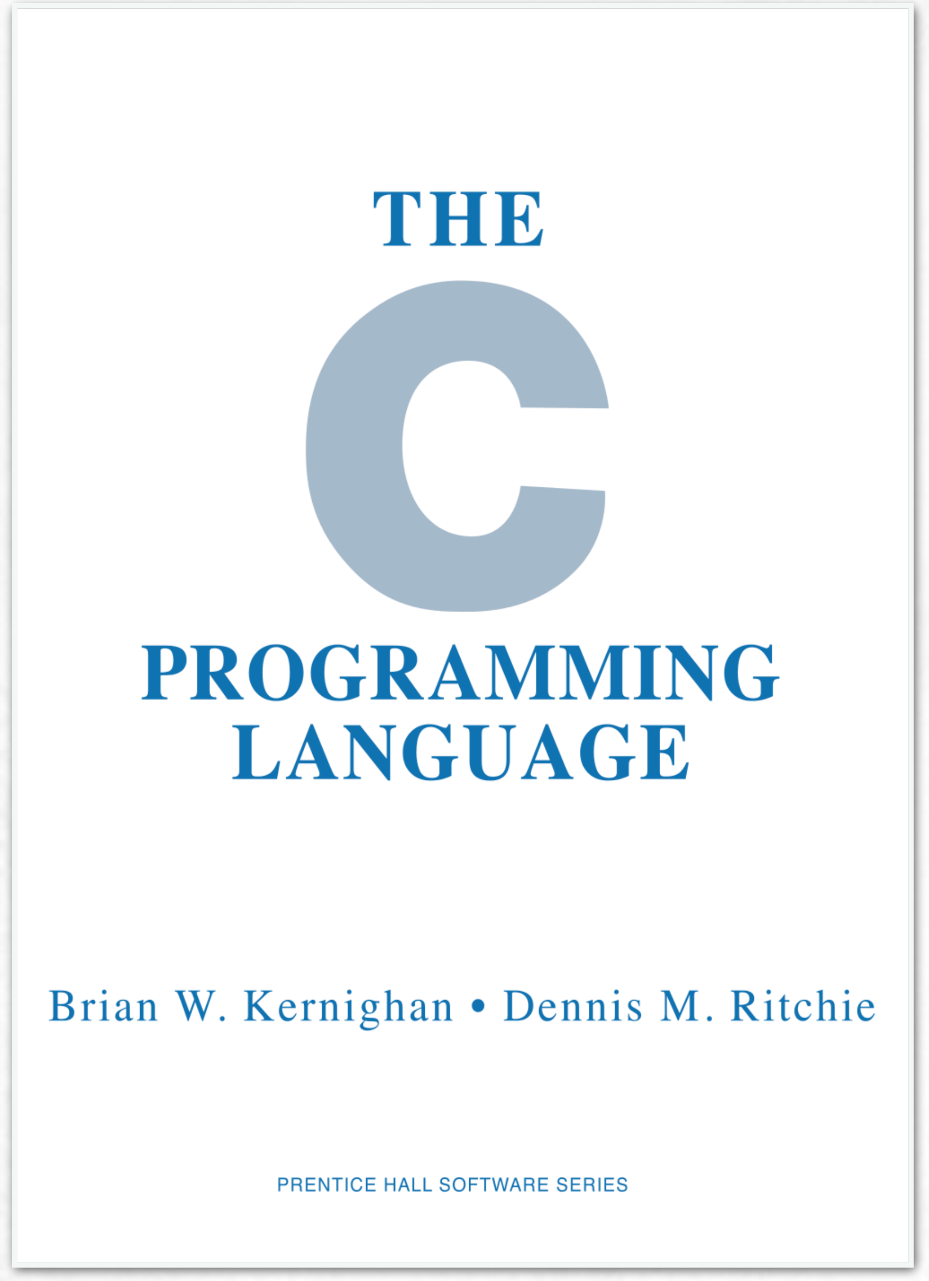
©2017 Richard L. Halterman Draft date: July 21, 2017

Contents

- C & C++ Language
- Language Popularity
- Software Build Process and Tools
- Open Source Resources for Software Programming
- Example Program - *Empty*
- General Structure of C++ Program
- Software Development Stage

C was

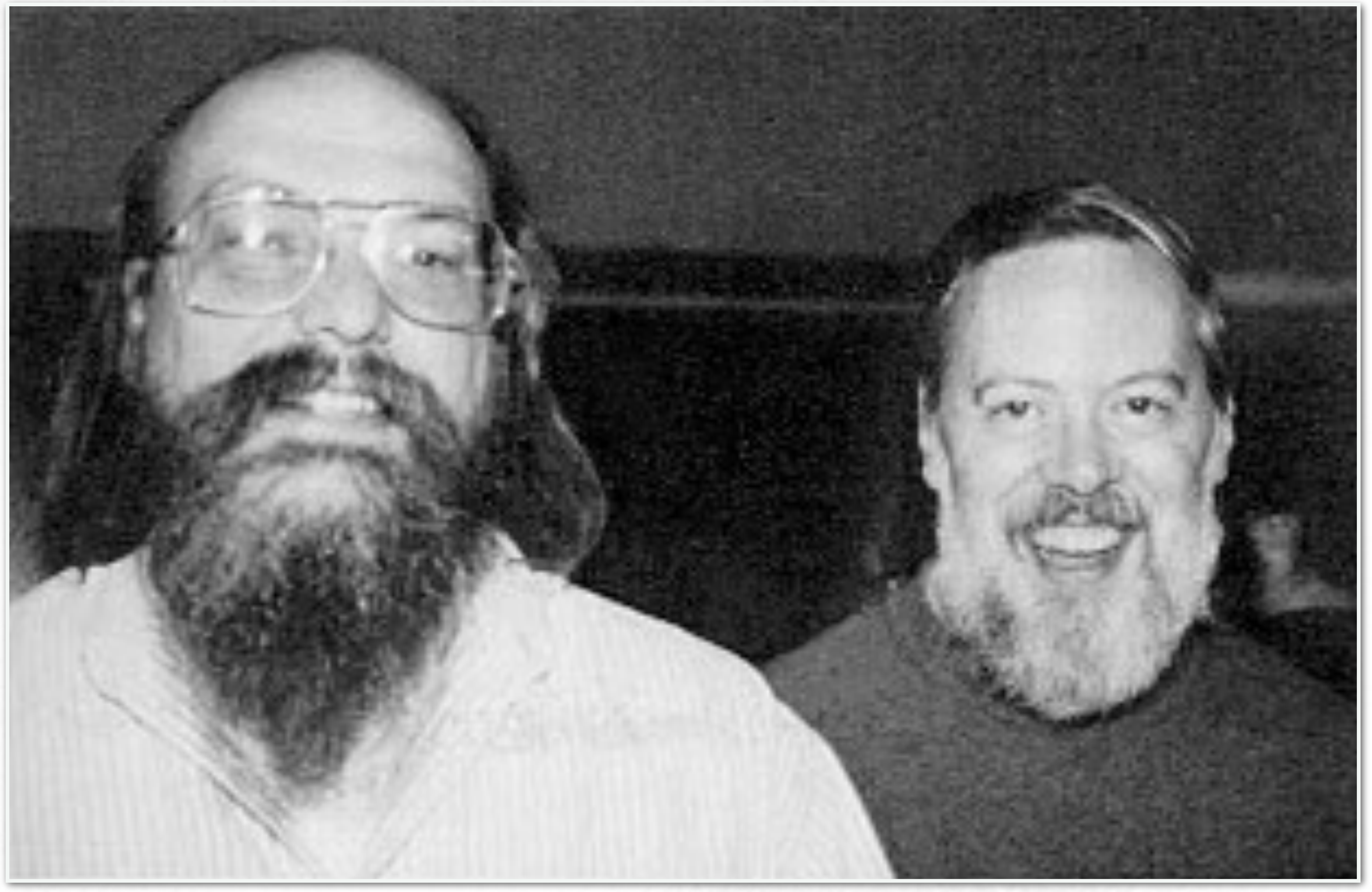
- Originally developed by Dennis Ritchie between 1969 and 1973 at Bell Labs, and used to re-implement the Unix operating system
- It has since become one of the most widely used programming languages of all time, with C compilers from various vendors available for the majority of existing computer architectures and operating systems
- It has been standardized by the American National Standards Institute (ANSI) since 1989 (see ANSI C) and subsequently by the International Organization for Standardization (ISO)



THE C PROGRAMMING LANGUAGE

Brian W. Kernighan • Dennis M. Ritchie

PRENTICE HALL SOFTWARE SERIES



Reference: https://en.wikipedia.org/wiki/C_(programming_language)

C is

- An imperative procedural language
  - It was designed to be compiled using a relatively straightforward compiler, to provide low-level access to memory, to provide language constructs that map efficiently to machine instructions, and to require minimal run-time support

- Despite its low-level capabilities, the language was designed to encourage cross-platform programming
  - A standards-compliant and portably written C program can be compiled for a very wide variety of computer platforms and operating systems with few changes to its source code
  - The language has become available on a very wide range of platforms, from embedded microcontrollers to supercomputers

Reference: https://en.wikipedia.org/wiki/C_(programming_language)

C++ was

- In 1979, Bjarne Stroustrup, a Danish computer scientist, began work on "C with Classes", the predecessor to C++
- Initially, Stroustrup's "C with Classes" added features to the C compiler, Cpre, including classes, derived classes, strong typing, inlining and default arguments
- In 1983, "C with Classes" was renamed to “C++"
- In 1985, the first edition of The C++ Programming Language was released, which became the definitive reference for the language, as there was not yet an official standard
- In 1989, C++ 2.0 was released, followed by the updated second edition of The C++ Programming Language in 1991
- As of 2017, C++ remains the third most popular programming language, behind Java and C

Reference: https://en.wikipedia.org/wiki/C%2B%2B

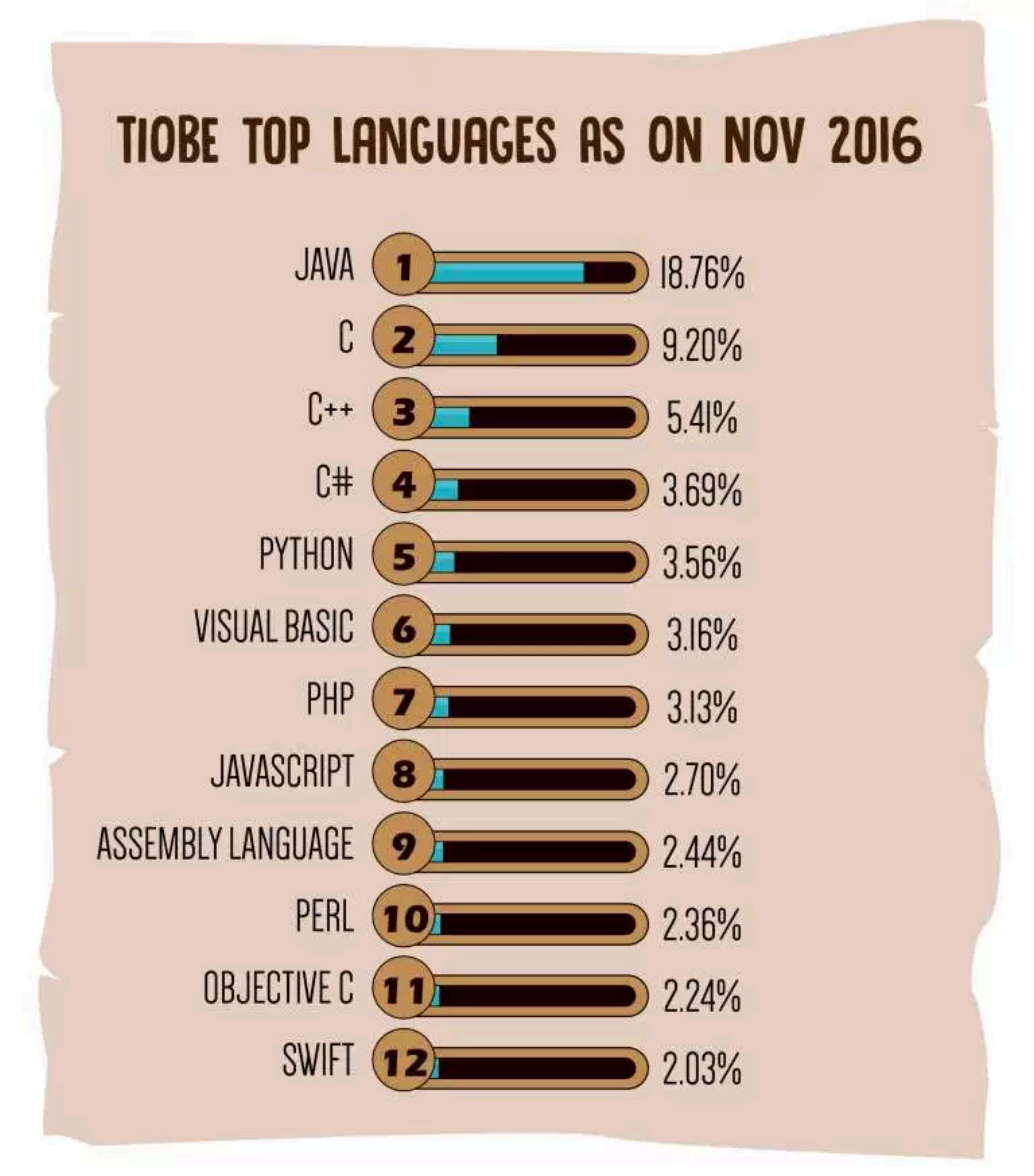
C++ is

- General-purpose programming language. It has imperative, object-oriented and generic programming features, while also providing facilities for low-level memory manipulation
- Bias toward system programming and embedded, resource-constrained and large systems, with performance, efficiency and flexibility of use as its design highlights
- C++ has also been found useful in many other contexts, with key strengths being software infrastructure and resource-constrained applications, including desktop applications, servers (e.g. e-commerce, web search or SQL servers), and performance-critical applications (e.g. telephone switches or space probes)
- C++ is a compiled language, with implementations of it available on many platforms
- Many vendors provide C++ compilers, including the Free Software Foundation, Microsoft, Intel, and IBM
- Many other programming languages have been influenced by C++, including C#, D, Java, and newer versions of C

Reference: https://en.wikipedia.org/wiki/C%2B%2B

Language Popularity

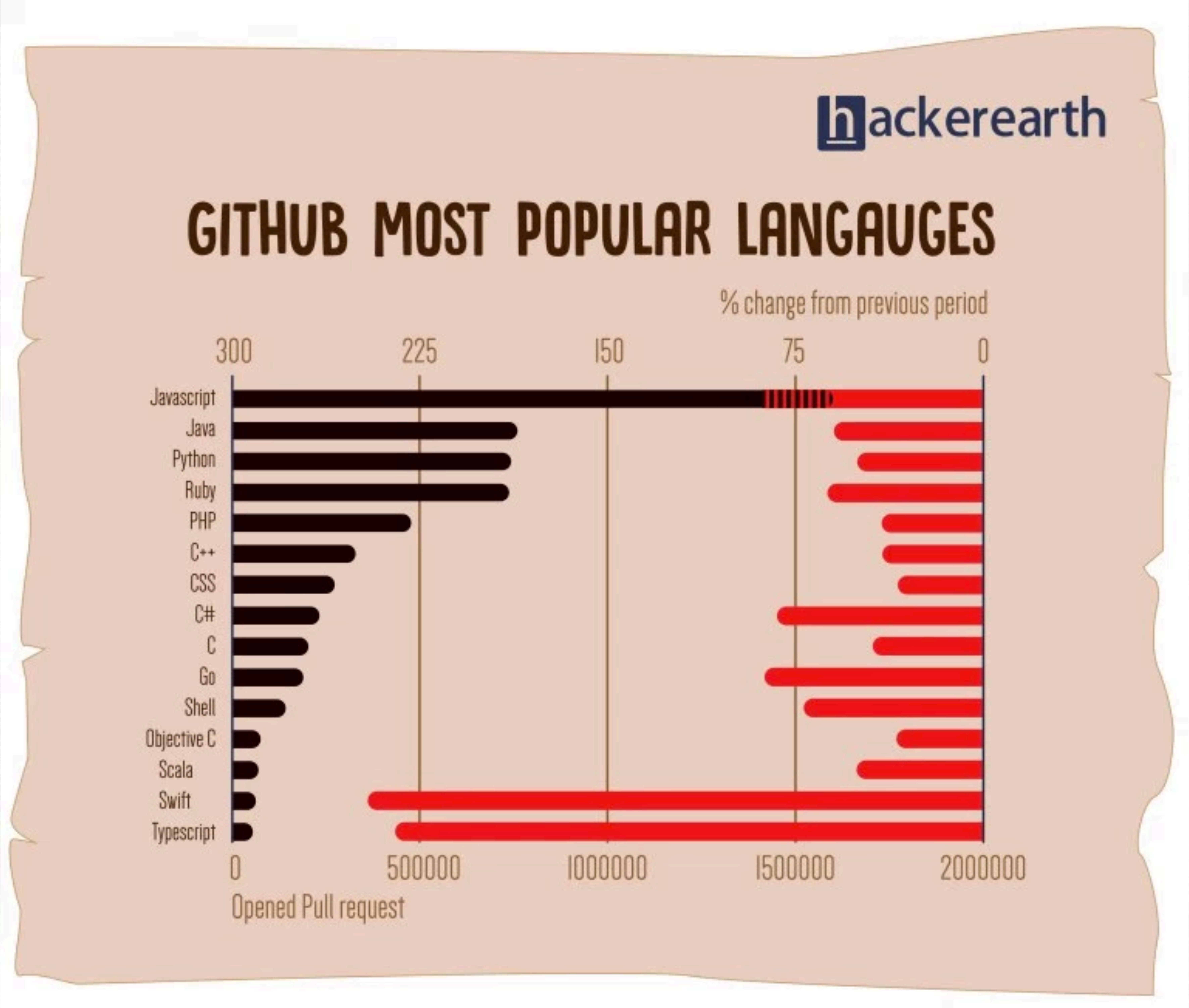
TIOBE Index



Reference: http://blog.hackerearth.com/2016/11/top-programming-language-2017.html

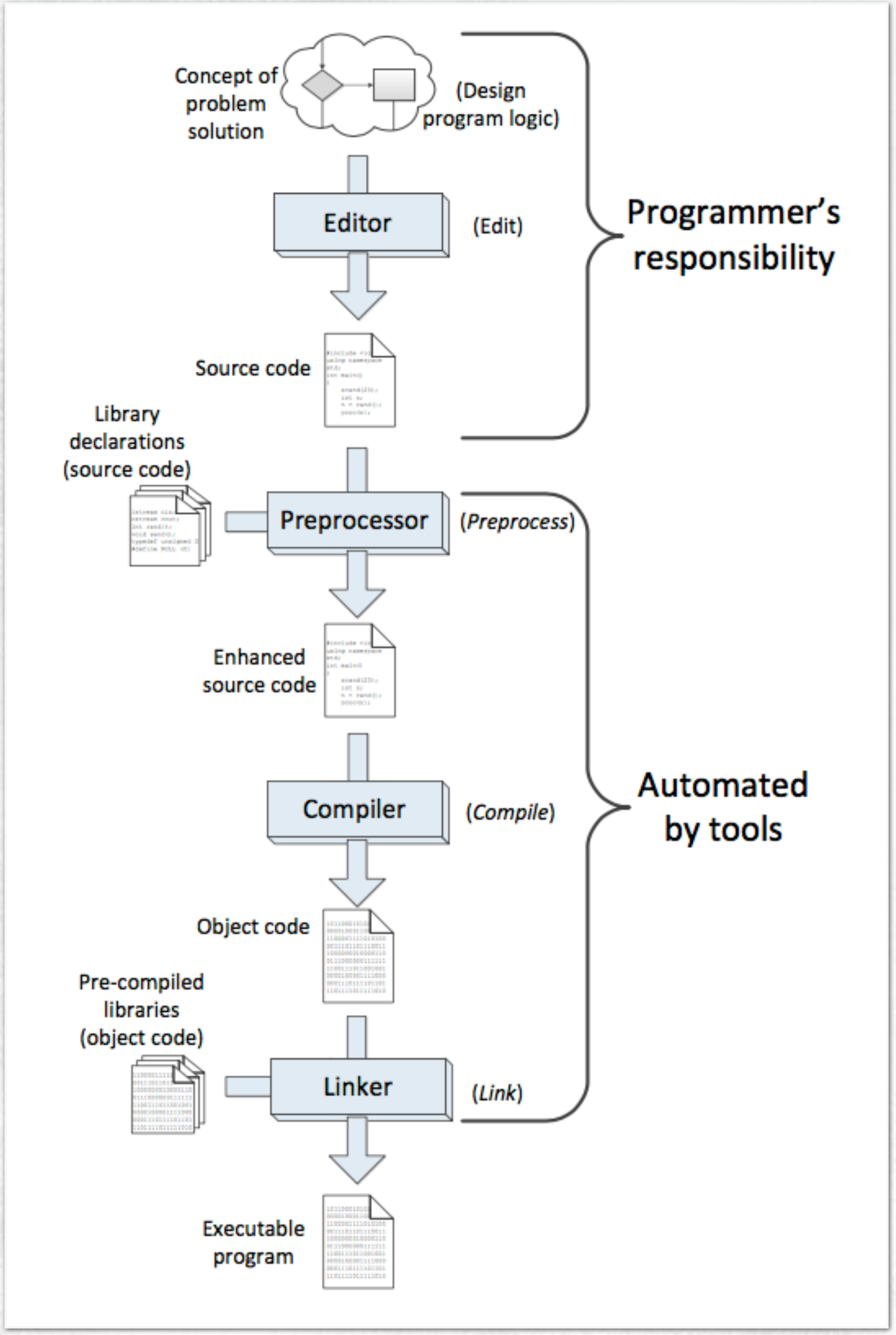
Language Popularity

GitHub



Reference: http://blog.hackerearth.com/2016/11/top-programming-language-2017.html

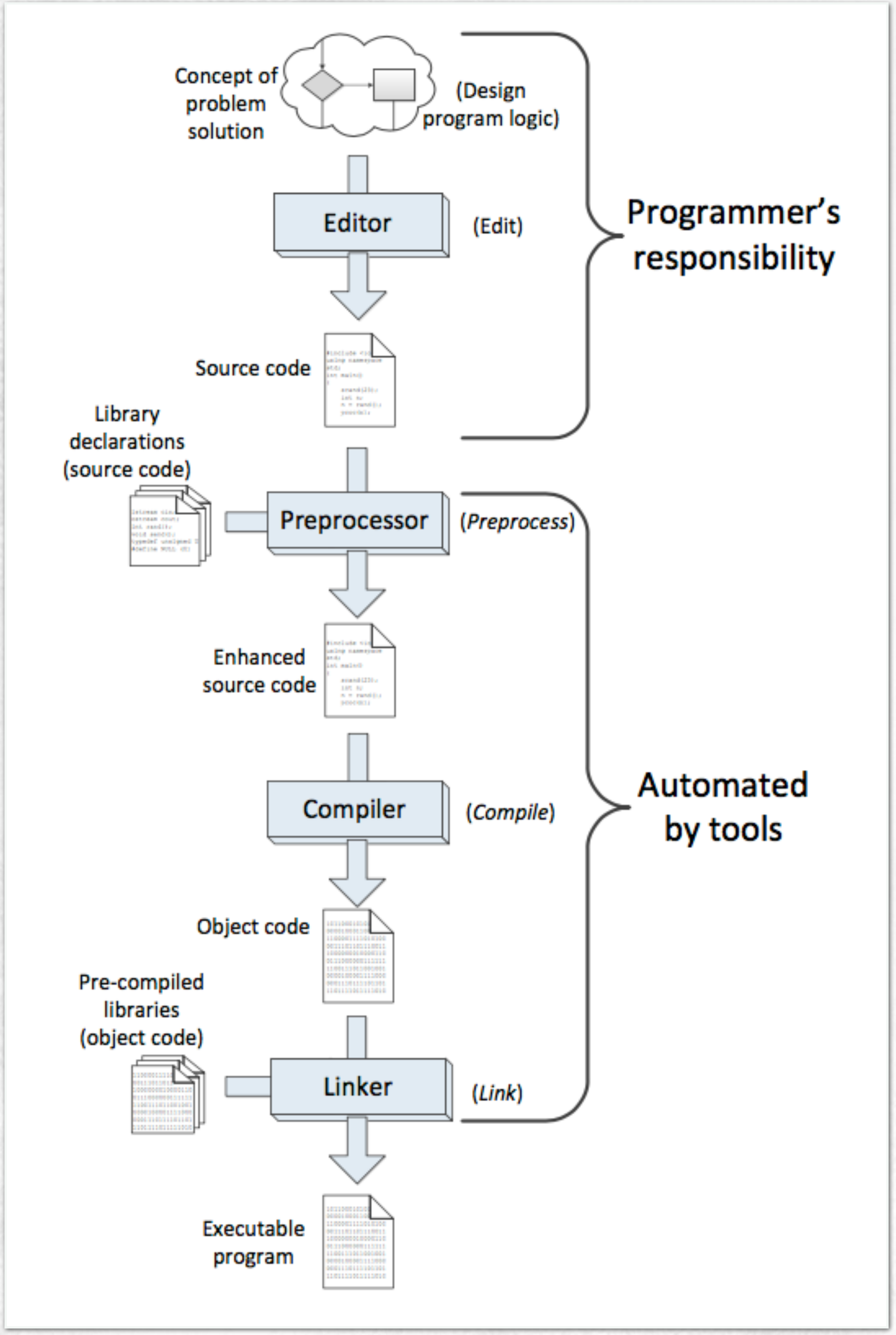
Editor



- An editor allows the user to enter the program source code and save it to files. Most programming editors increase programmer productivity by using colors to highlight language features.
- The syntax of a language refers to the way pieces of the language are arranged to make well-formed sentences. To illustrate, the sentence
- Programmers must follow strict syntax rules to create well-formed computer programs. Only well-formed programs are acceptable and can be compiled and executed. Some syntax-aware editors can use colors or other special annotations to alert programmers of syntax errors before the program is compiled.

Software Build Process & Tools

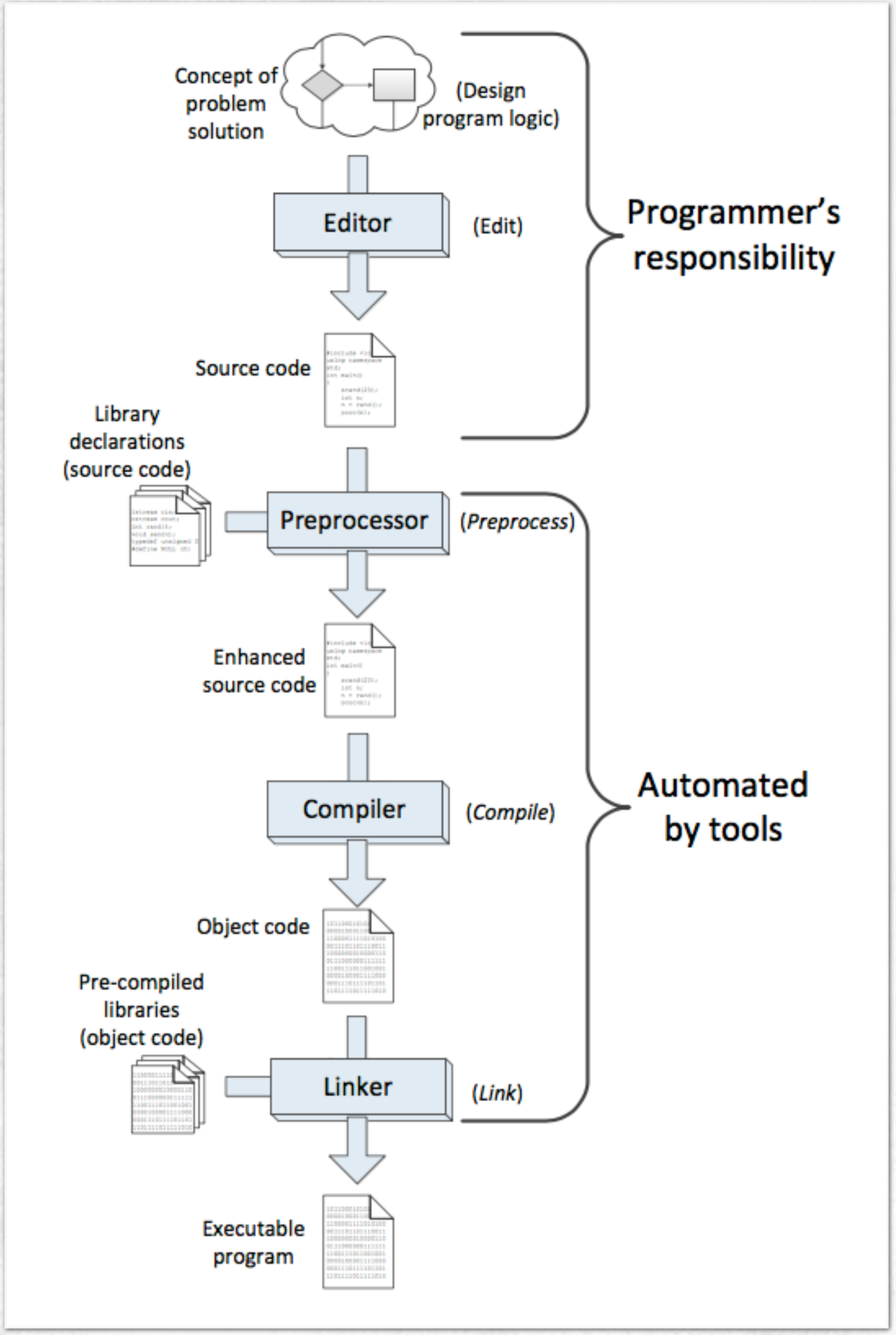
Pre-processor



- Preprocessor adds to or modifies the contents of the source file before the compiler begins processing the code.
- We use the services of the preprocessor mainly to #include information about library routines our programs use.

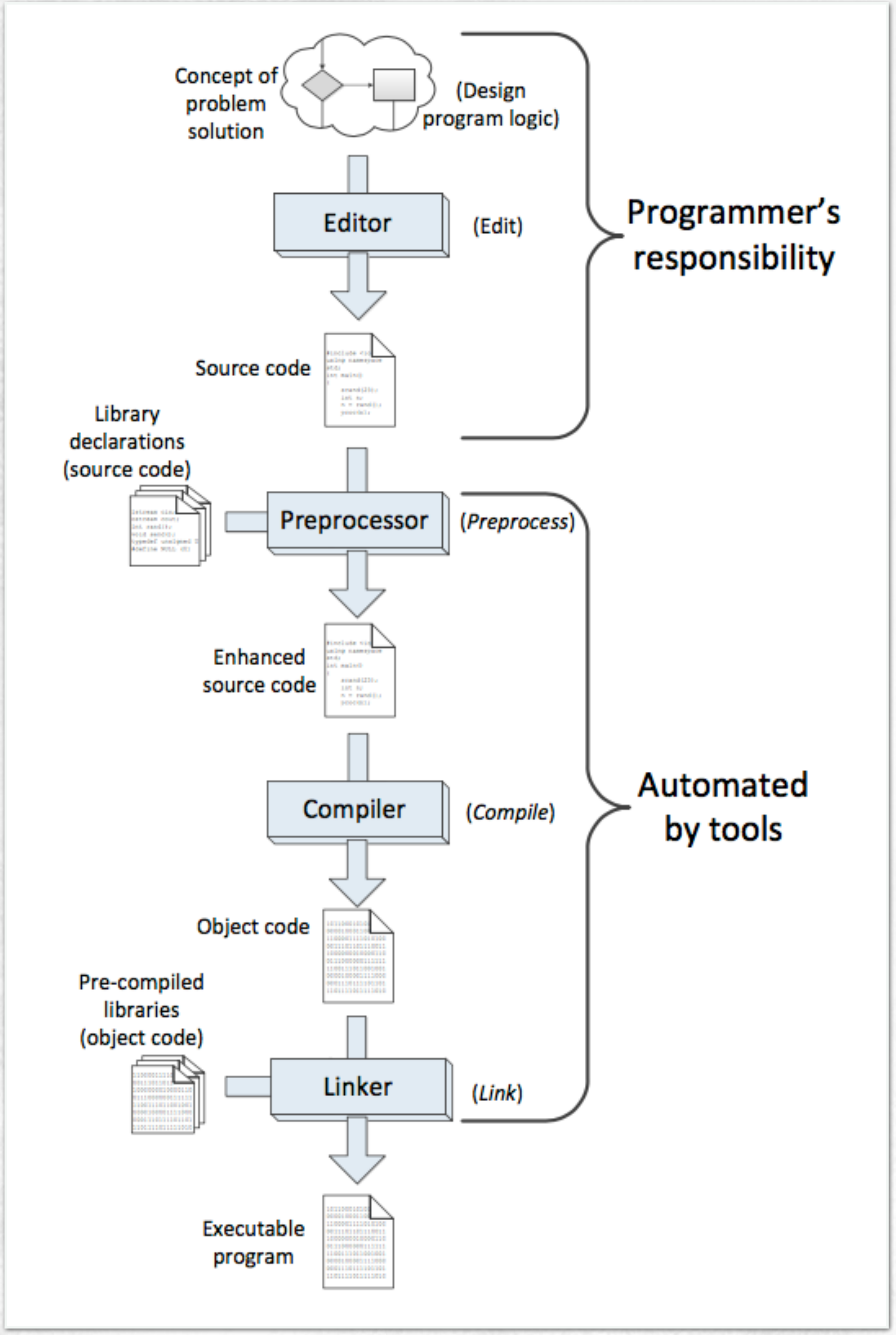
Software Build Process & Tools

Compiler



- A compiler translates the source code to target code.
- The target code may be the machine language for a particular platform or embedded device.
- The target code could be another source language; for example, the earliest C++ compiler translated C++ into C, another higher-level language.
- The resulting C code was then processed by a C compiler to produce an executable program.
- C++ compilers today translate C++ directly into machine language.

Linker



- Linker combines the compiler-generated machine code with precompiled library code or compiled code from other sources to make a complete executable program.
- Most compiled C++ code is incapable of running by itself and needs some additional machine code to make a complete executable program.
- The missing machine code has been precompiled and stored in a repository of code called a library.
- A program called a linker combines the programmer's compiled code and the library code to make a complete program.

Software Build Process & Tools

Debugger

- A debugger allows a programmer to more easily trace a program’s execution in order to locate and correct errors in the program’s implementation.
- With a debugger, a developer can simultaneously run a program and see which line in the source code is responsible for the program’s current actions.
- The programmer can watch the values of variables and other program elements to see if their values change as expected.
- Debuggers are valuable for locating errors (also called bugs) and repairing programs that contain errors. (See Section 4.6 in Textbook for more information about programming errors.)

Profiler

- A profiler collects statistics about a program's execution allowing developers to tune appropriate parts of the program to improve its overall performance.
- A profiler indicates how many times a portion of a program is executed during a particular run, and how long that portion takes to execute.
- Profilers also can be used for testing purposes to ensure all the code in a program is actually being used somewhere during testing - this is known as coverage.
- It is common for software to fail after its release because users exercise some part of the program that was not executed anytime during testing.
- The main purpose of profiling is to find the parts of a program that can be improved to make the program run faster.

Famous Free IDEs and Compiler

- IDE: Microsoft Visual C++ 2005 Express Edition [support MS-Windows]

  https://www.microsoft.com/korea/msdn/vstudio/express/visualc/

- IDE: Microsoft Visual Studio Community [MS-Windows/Mac]

  https://www.visualstudio.com/ko/vs/visual-studio-express/

- IDE: Xcode [Mac]

  https://developer.apple.com/xcode/

- IDE: Qt Creator [MS-Windows/Mac/Linux]

  http://doc.qt.io/qtcreator/

- Compiler: GCC, the GNU C/C++ Compiler [MS-Windows/Mac/Linux]

  https://gcc.gnu.org/

Famous Free Editors

- Editor: Microsoft Visual Studio Code

  https://code.visualstudio.com/

- Editor: ATOM

  https://atom.io/

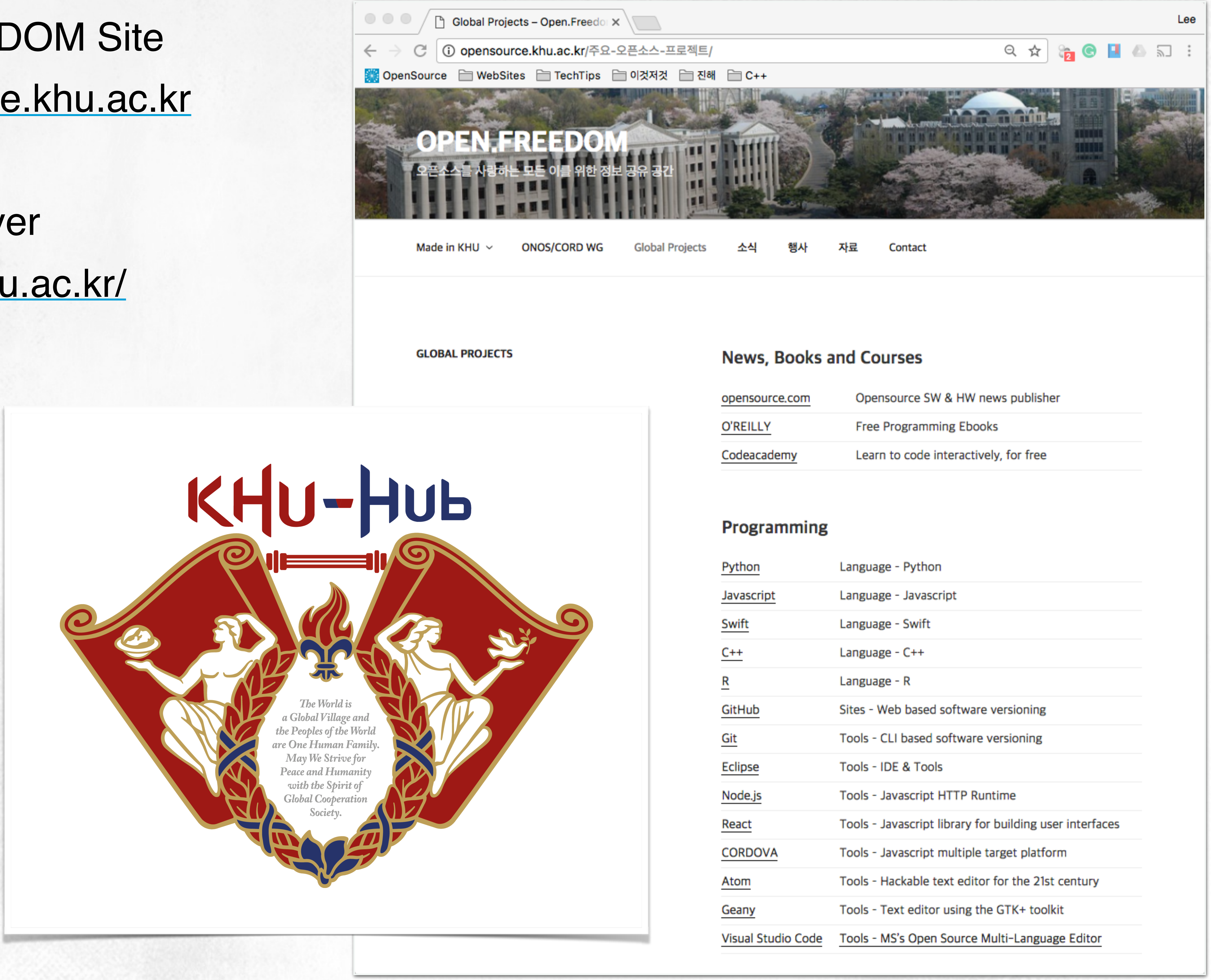
KHU Open Sources

- KHU OPEN.FREEDOM Site

  http://opensource.khu.ac.kr

- KHU-HUB Git Server

  http://khuhub.khu.ac.kr/


Global Projects – Open.Freedo
Lee
opensource.khu.ac.kr/주요-오픈소스-프로젝트/
OpenSource
WebSites
TechTips
이것저것
진해
C++
OPEN.FREEDOM
오픈소스를 사랑하는 모든 이를 위한 정보 공유 공간
Made in KHU
ONOS/CORD WG
Global Projects
소식
행사
자료
Contact
KHU-Hub
The World is a Global Village and the Peoples of the World are One Human Family. May We Strive for Peace and Humanity with the Spirit of Global Cooperation Society.

GLOBAL PROJECTS

News, Books and Courses

opensource.com	Opensource SW & HW news publisher
O'REILLY	Free Programming Ebooks
Codeacademy	Learn to code interactively, for free

Programming

Python	Language - Python
Javascript	Language - Javascript
Swift	Language - Swift
C++	Language - C++
R	Language - R
GitHub	Sites - Web based software versioning
Git	Tools - CLI based software versioning
Eclipse	Tools - IDE & Tools
Node.js	Tools - Javascript HTTP Runtime
React	Tools - Javascript library for building user interfaces
CORDOVA	Tools - Javascript multiple target platform
Atom	Tools - Hackable text editor for the 21st century
Geany	Tools - Text editor using the GTK+ toolkit
Visual Studio Code	Tools - MS's Open Source Multi-Language Editor

Example Program

Empty

```
int main()
{
}
```

Example Program

Empty

- Line 01:
  - This specifies the real beginning of our program. Here we are declaring a function named main.
  - All C++ programs must contain this function to be executable. Details about the meaning of int and the parentheses will appear in Week 2. More general information about functions appear in Week 3 and 4.
  - The opening curly brace at the end of the line marks the beginning of the body of a function. The body of a function contains the statements the function is to execute.

```
01: int main()
02: {
03: }
```

Example Program
Empty

- Line 02:
  - The opening curly brace at the end of the line marks the beginning of the body of a function.
  - The body of a function contains the statements the function is to execute.

```
01: int main()
02: {
03: }
```

Example Program

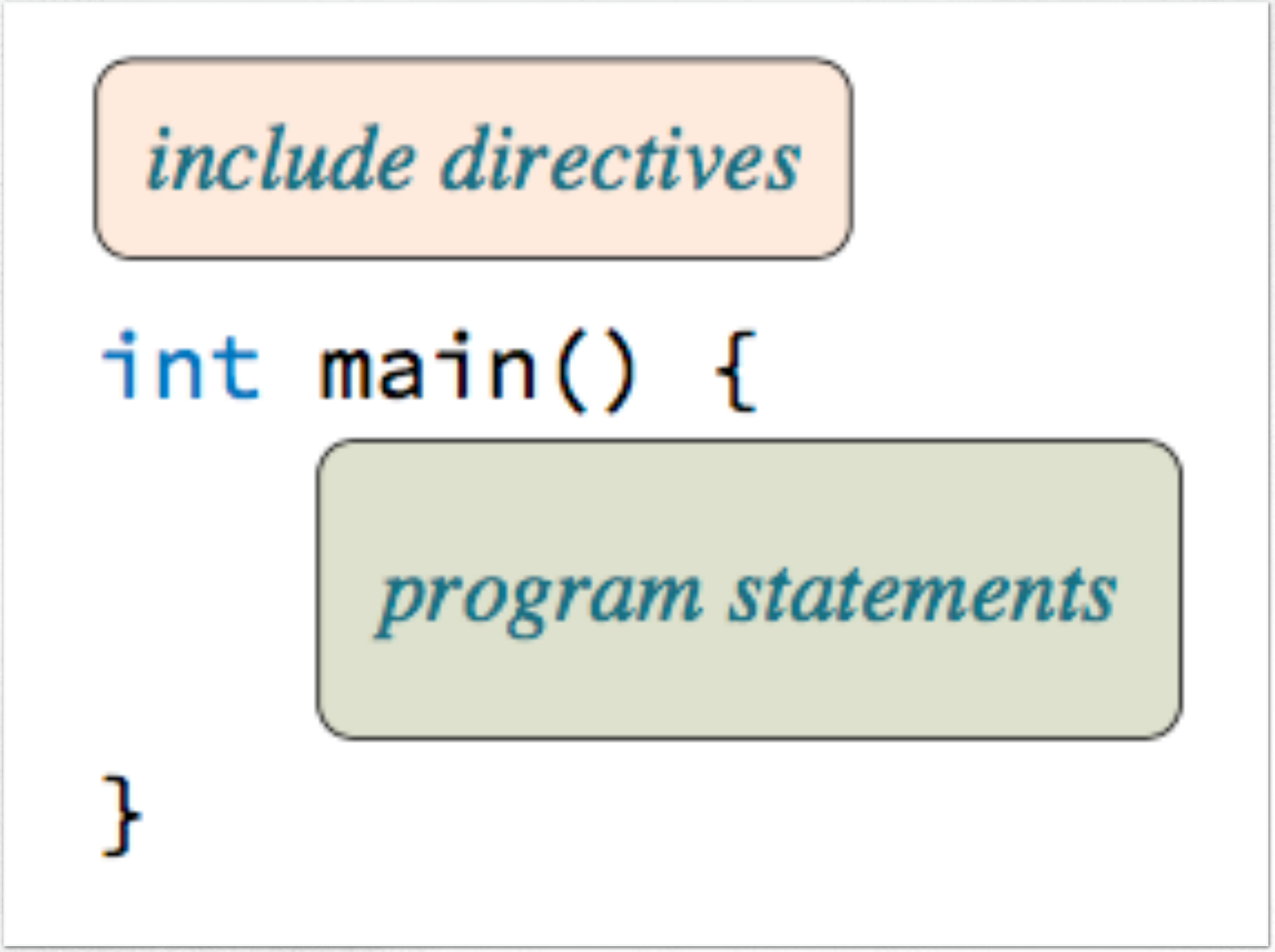
Empty

- Line 03:
  - The closing curly brace marks the end of the body of a function.
  - Both the open curly brace and close curly brace are required for every function definition.

```
01: int main()
02: {
03: }
```

General Structure of C++ Program

Simple Case

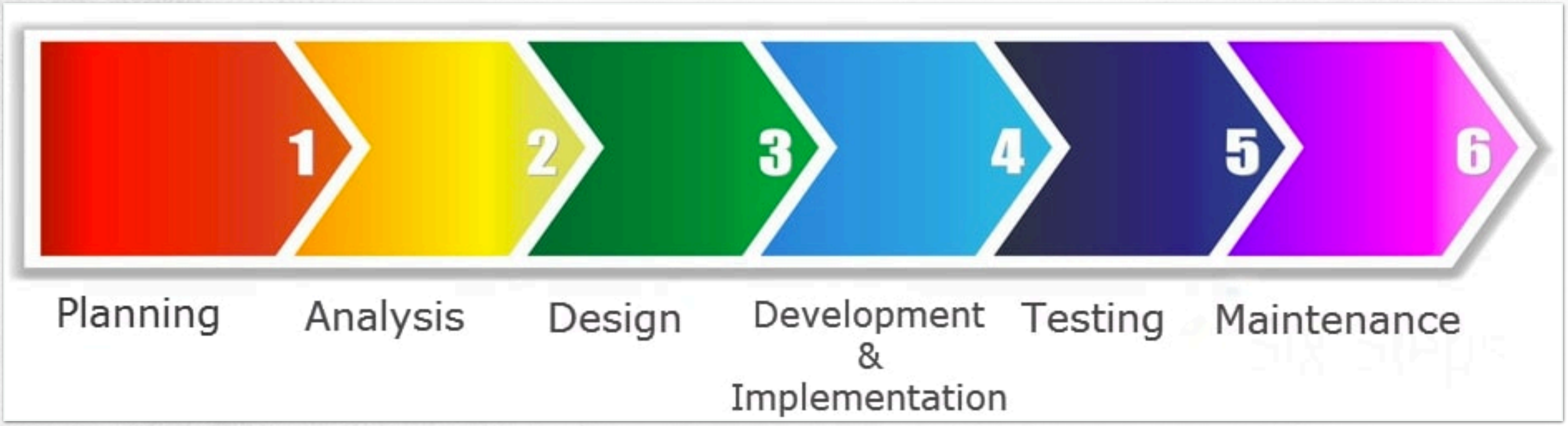


```
include directives

int main() {
        program statements
}
```

Software Development Stage

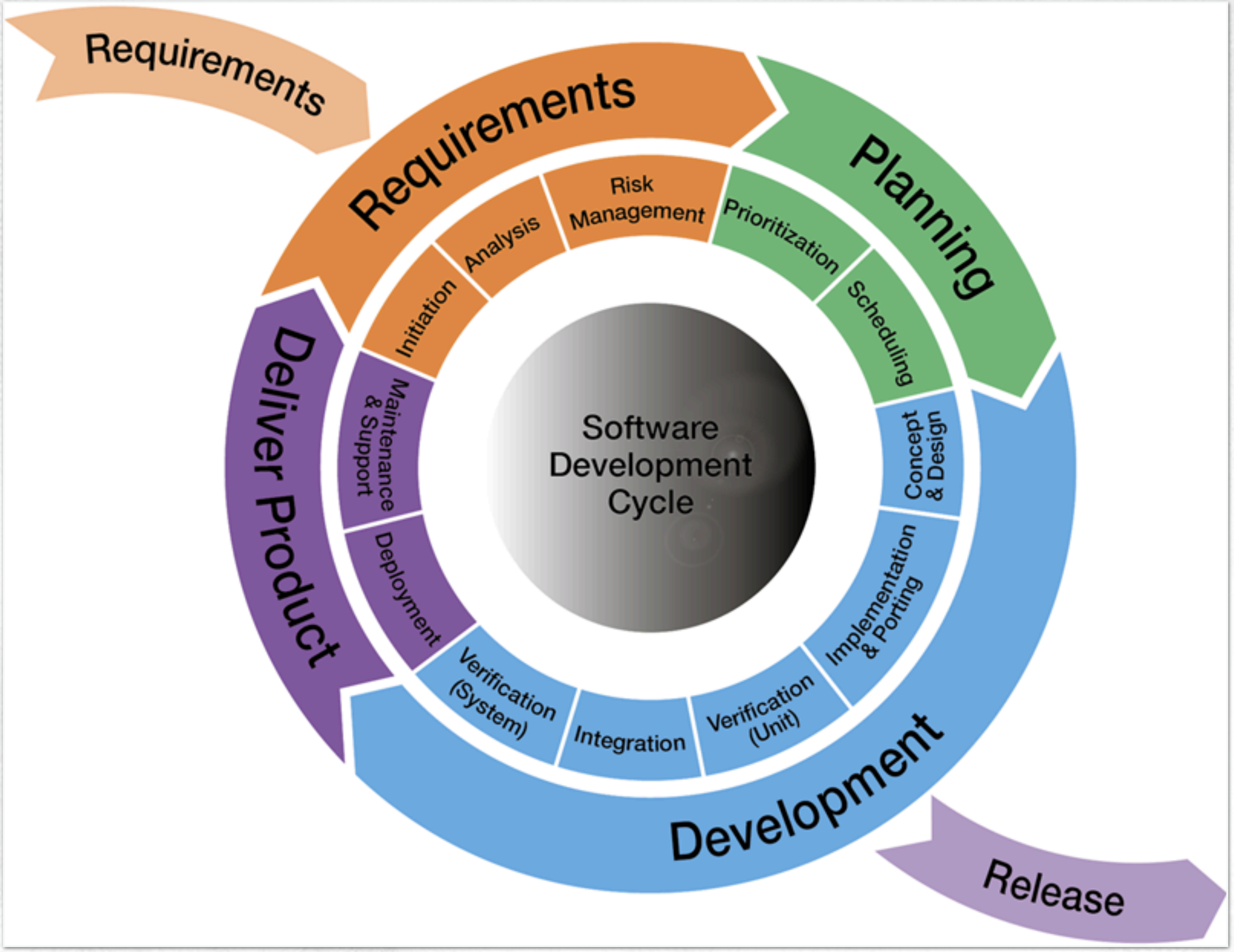
Waterfall (for traditional big & large scale projects)



Reference: https://www.synapseindia.com/6-stages-of-software-development-process/141

Software Development Stage

Agile (Spiral approach; for fast service oriented projects)



Reference: https://www.pinterest.co.kr/pin/356488126733127761/

Object Oriented Programming by C++

Sungwon Lee / Professor

Email: drsungwon@khu.ac.kr
Web: http://mobilelab.khu.ac.kr/