

C++ Program Design

-- Introduction

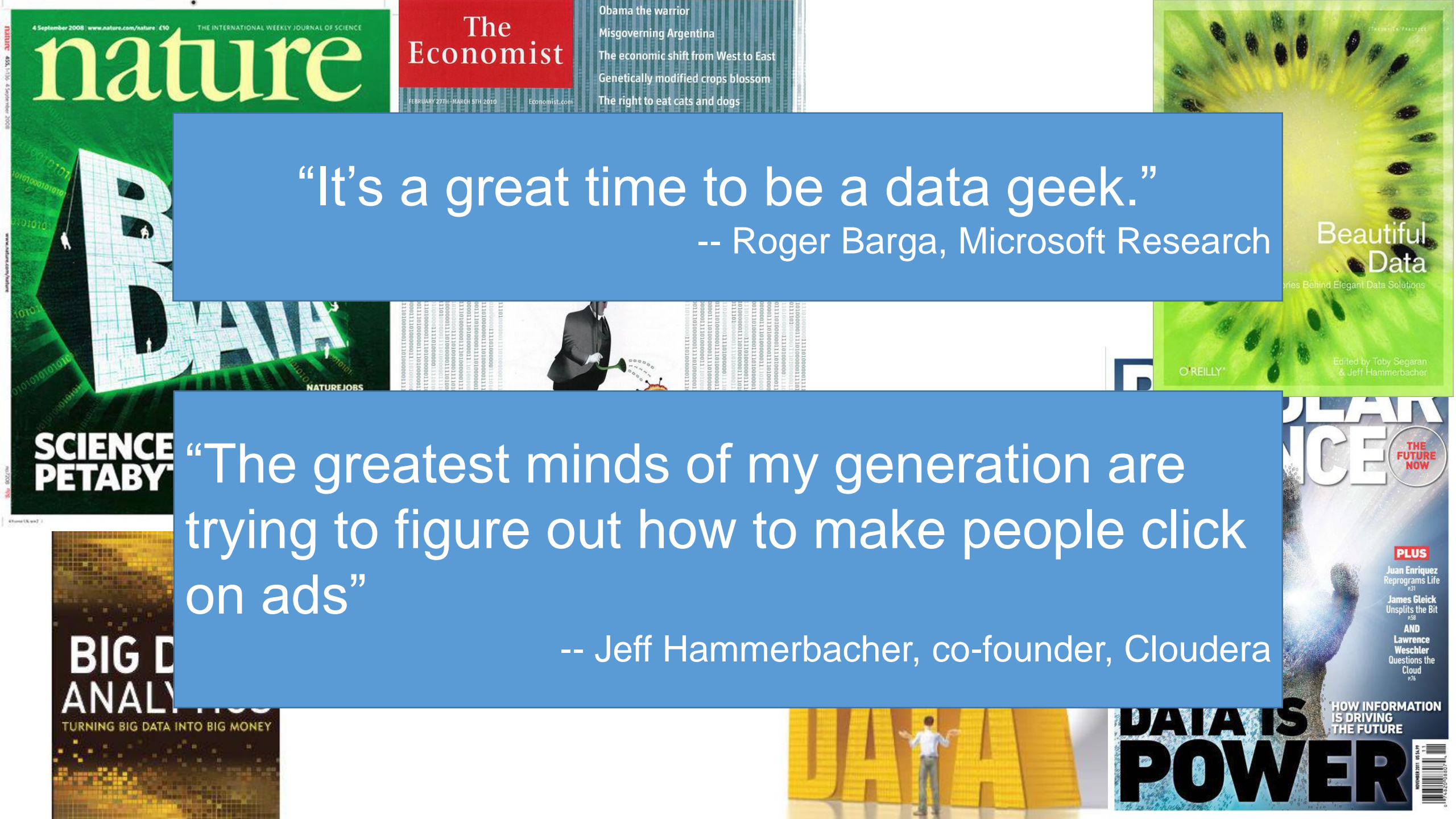


Junjie Cao @ DLUT

Summer 2017

<http://jjcao.github.io/cPlusPlus>

Coding is important



“It’s a great time to be a data geek.”
-- Roger Barga, Microsoft Research

“The greatest minds of my generation are trying to figure out how to make people click on ads”
-- Jeff Hammerbacher, co-founder, Cloudera

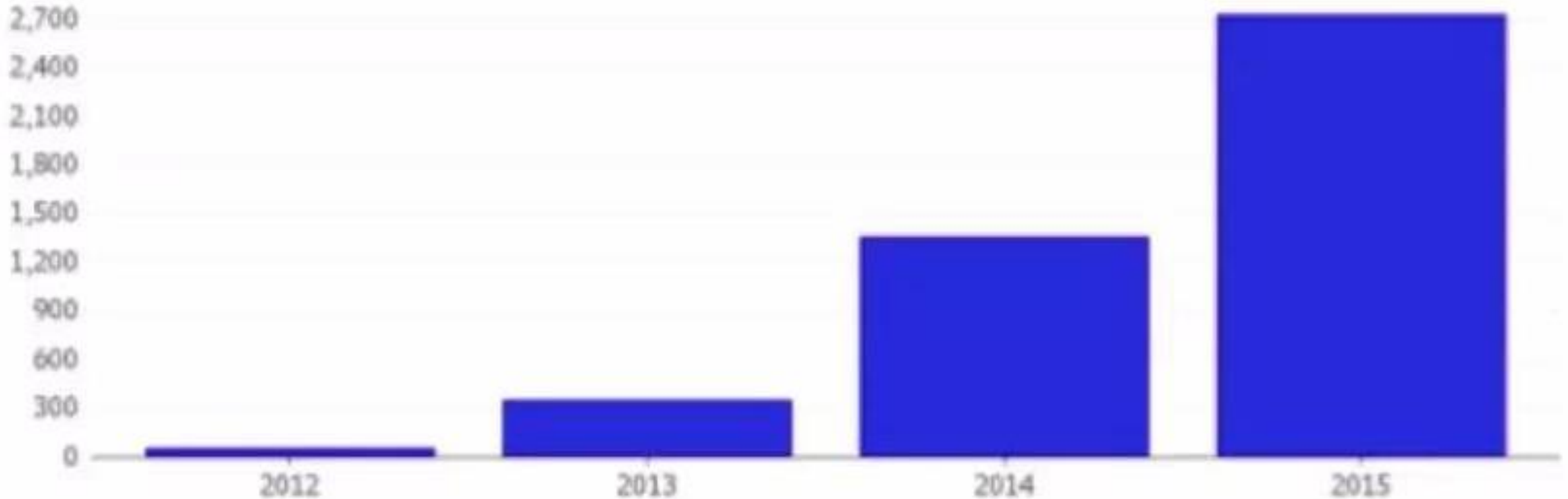


In “Nature” 27 January 2016:

- “DeepMind’s program AlphaGo beat Fan Hui, the European Go champion, five times out of five in tournament conditions...”
- “AlphaGo was not preprogrammed to play Go: rather, it learned using a general-purpose algorithm that allowed it to interpret the game’s patterns.”
- “...AlphaGo program applied **deep learning** in neural networks (convolutional NN) — brain-inspired programs in which connections between layers of simulated neurons are strengthened through examples and experience.”

Artificial intelligence takes off at google

Number of software projects within Google that uses a key AI technology, called Deep Learning.



Source: Google

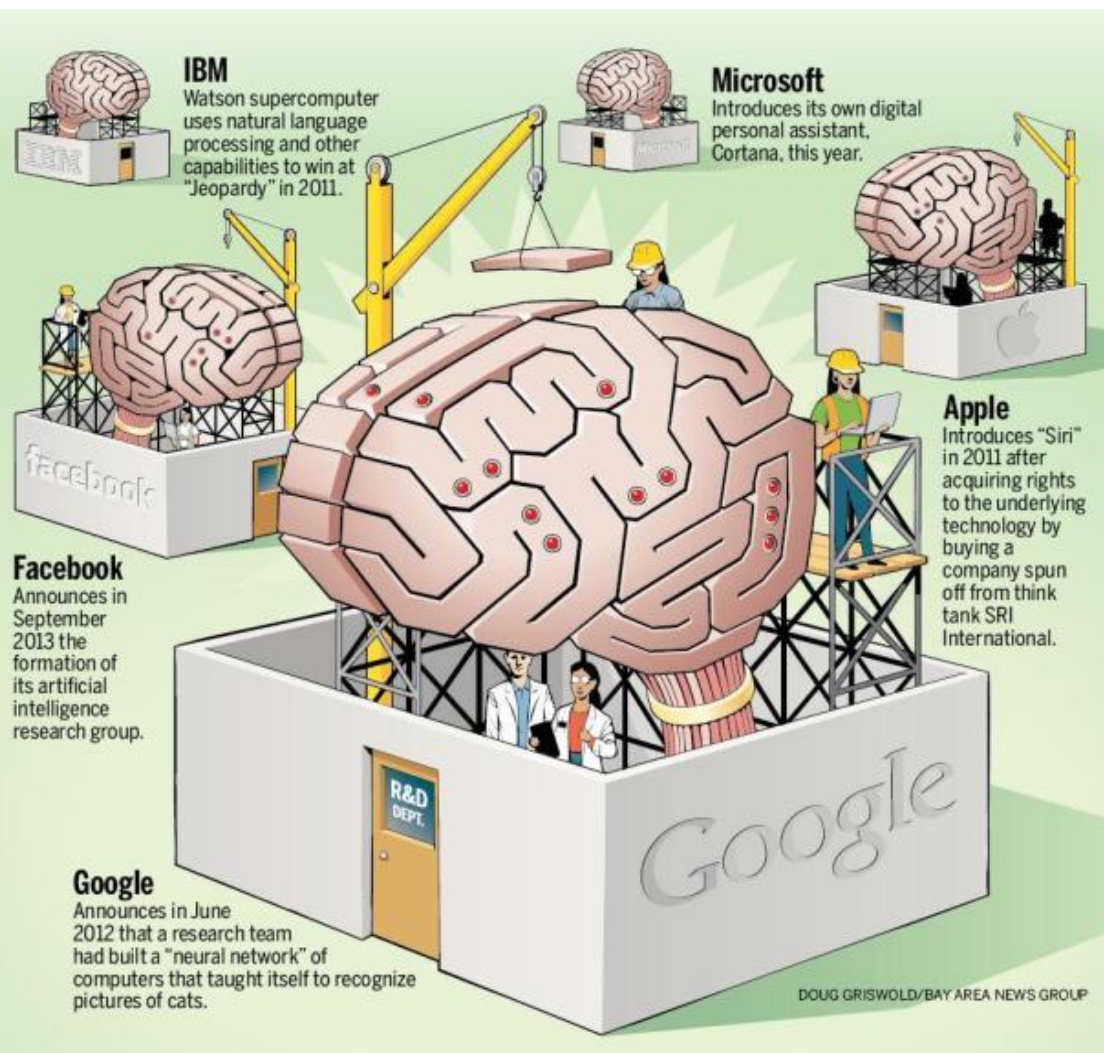
Note: 2015 data does not incorporate data from Q4

2010s

Multi-billion dollar investments in artificial intelligence and robotics in the 2010s

- Amazon (Kiva, 2012),
- Google (Industrial Robotics, Meka, Holomni, Bot & Dolly, DNNresearch, Schaft, Bost, DeepMind, Redwood Robotics, 2013-14),
- IBM (Watson project),
- Microsoft (Project Adam, 2014),
- Apple (Siri, 2011),
- Facebook (DeepFace, 2013),
- Yahoo (LookFlow, 2013),
- Baidu (Deep Learning Institute in Cupertino, 2013; hired Andrew Ng, 2014)

2010年以来，人工智能领域已吸引超过**200亿**美元的投风险投资



- **When human beings acquired language, we learned not just how to listen but how to speak.**
- **When we gained literacy, we learned not just how to read but how to write.**
- **And as we move into an increasingly digital reality, we must learn not just how to use programs but to make them.**
- **In the emerging, highly programmed landscape ahead, you will either create the software or you will be the software.**
- **It's really that simple: Program, or be programmed.**
- **Choose the former, and you gain access to the control panel of civilization. Choose the latter, and it could be the last real choice you get to make.**



Goal by the end of the semester

Given a **data** source and a **problem description**, you can independently write a complete, useful program to **solve the problem**

What This Course is **not**

- A “skills course” in C++
 - ... though you will become proficient in the basics of the C++ programming language
 - ... and you will gain experience with some C++ libraries
- A “project” course
 - the assignments are “real,” but are intended to teach specific programming concepts

**Is Matlab/Python the final weapon
for us?**

Why teaching C++



C
Rulez!

Dennis Ritchie
1969 -- 1973 at [Bell Labs](#)
C99, C11



C++
Rulez!

Bjarne Stroustrup
1979--1983 at [Bell Labs](#)
[C++11](#), [C++14](#), [C++17](#), [C++20](#)

Why teaching C++

1. **Most common for research**, and some areas of **industry**.
2. Java and Python are not suitable for learning algorithms and data structures.
3. Lisp maybe better than C++ for leaning algorithms and data structures. But it is not so common and **limited** in research field.
4. **Matlab** is better for research, but sometimes c & c++ is still a necessary **complement**.
5. The most of **libraries** for science computation are still implemented in C++.
6. Java is not, generally, a **hard** enough programming language that it can be used to discriminate between great programmers and mediocre programmers.

C and C++'s philosophy

- Underlying design philosophy: “**trust the programmer**”
 - Wonderful
 - compiler will not stand in your way if you try to do something unorthodox that makes sense,
 - Dangerous
 - compiler will not stand in your way if you try to do something that could produce unexpected results.
 - That is one of the primary reasons why knowing **what you shouldn't do** in C/C++ is almost **as important as knowing what you should do** -- because there are quite a few pitfalls that new programmers are likely to fall into if caught unaware.



Matlab, Python & C++

- Versatile
 - Python > C++ > Matlab
- Easy to master
 - Python (free) > Matlab (commercial)
 - Python leads to more beautiful & maintainable code
 - Programming courses in famous universities, CMU, MIT, etc.: c, c++ => Python
 - Big data analysis
- Performance
 - C++

Programming language & Thought

- Assembly language
- Computation: Fortran 1954
- System programming: C 1969, **C++** 1979, C# 1999, Objective-C
- Application: Java 1995, Java script, PHP
- Unix shell to everything: Perl, **Python**, Ruby
- Computation: **Matlab**, Mathematics, Mapple, R
- The "concept" of "programming languages" are quite "similar"

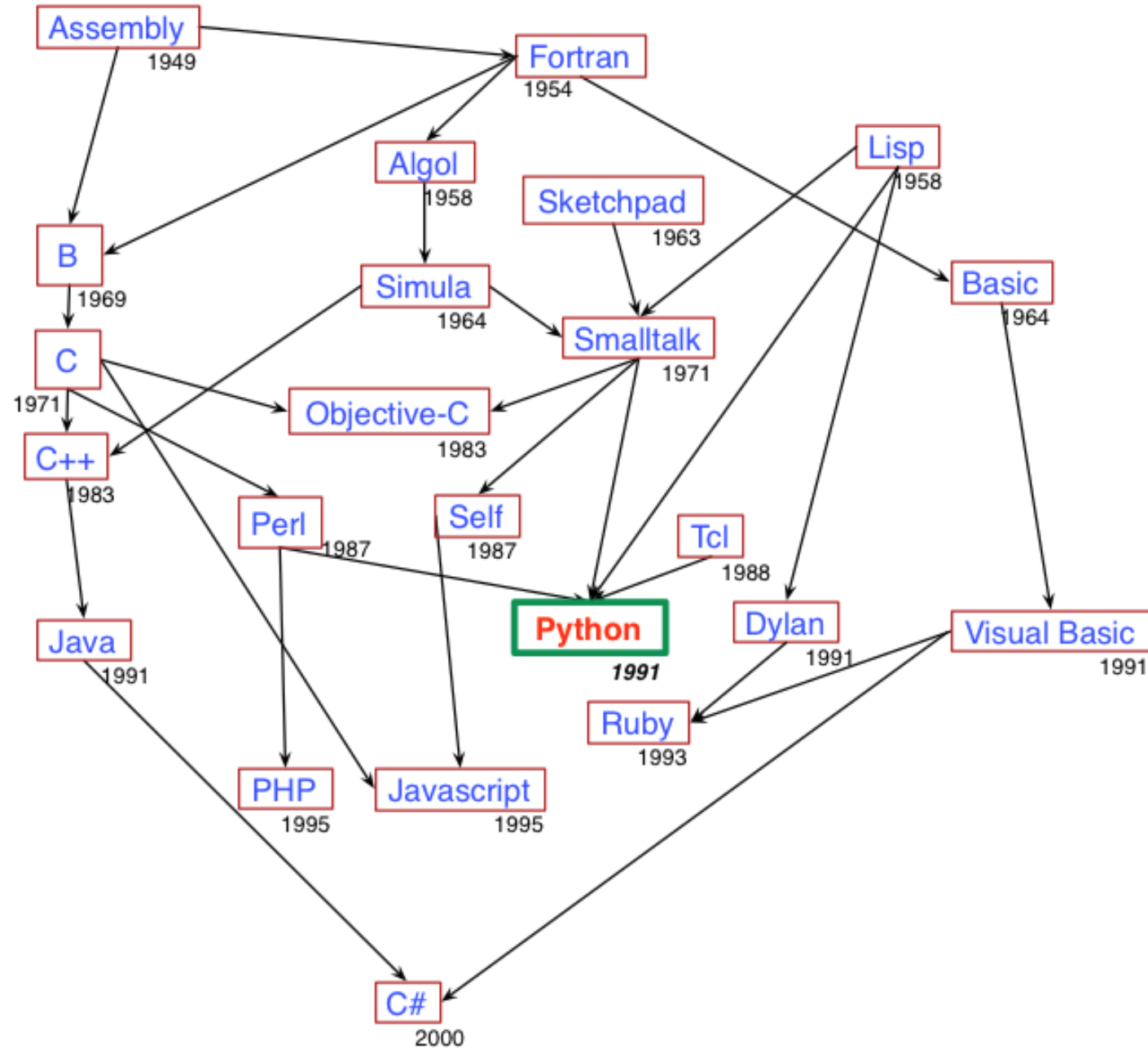
Language is the dress of thought.

~Samuel Johnson

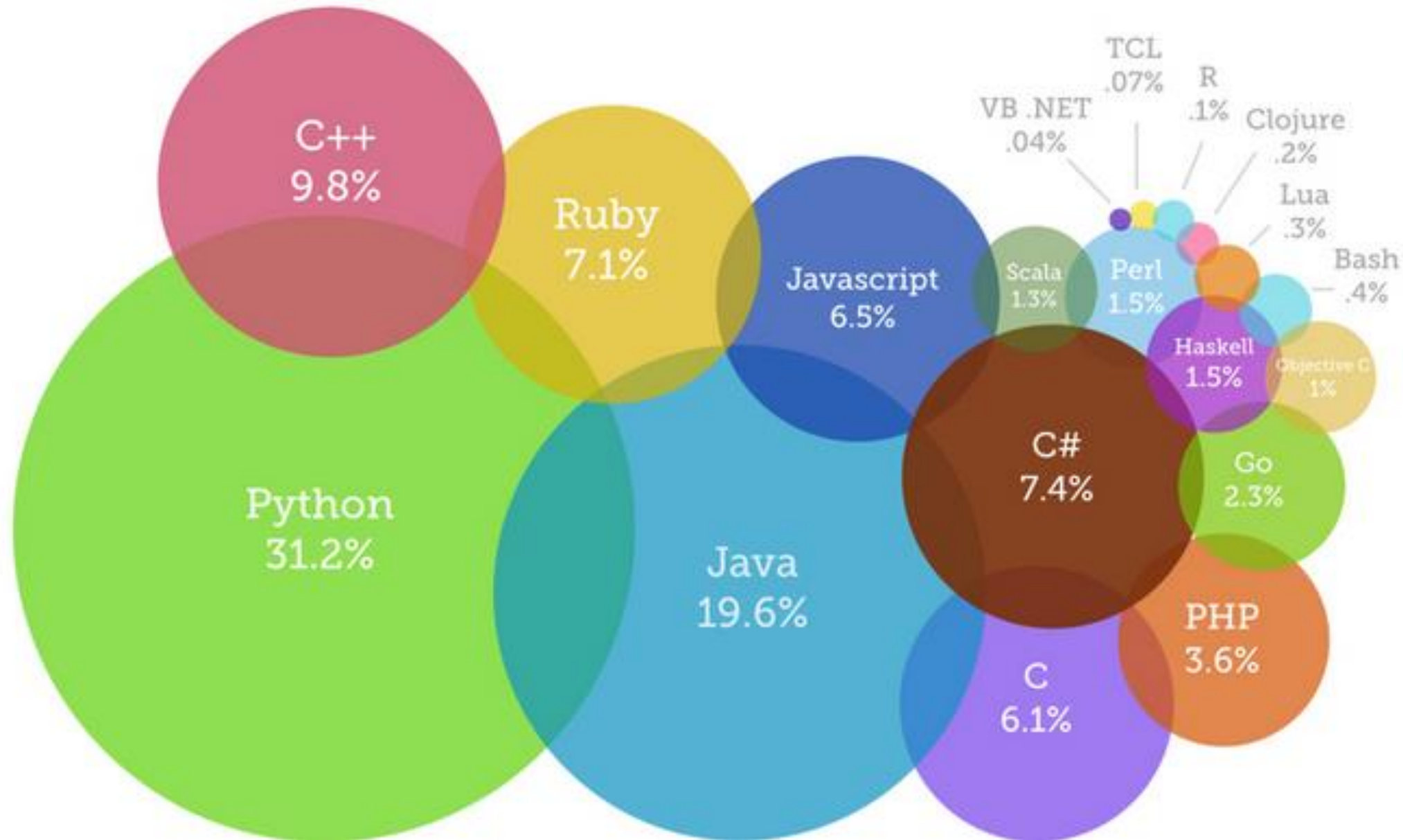
But if thought corrupts language, language can
also corrupt thought.

~George Orwell

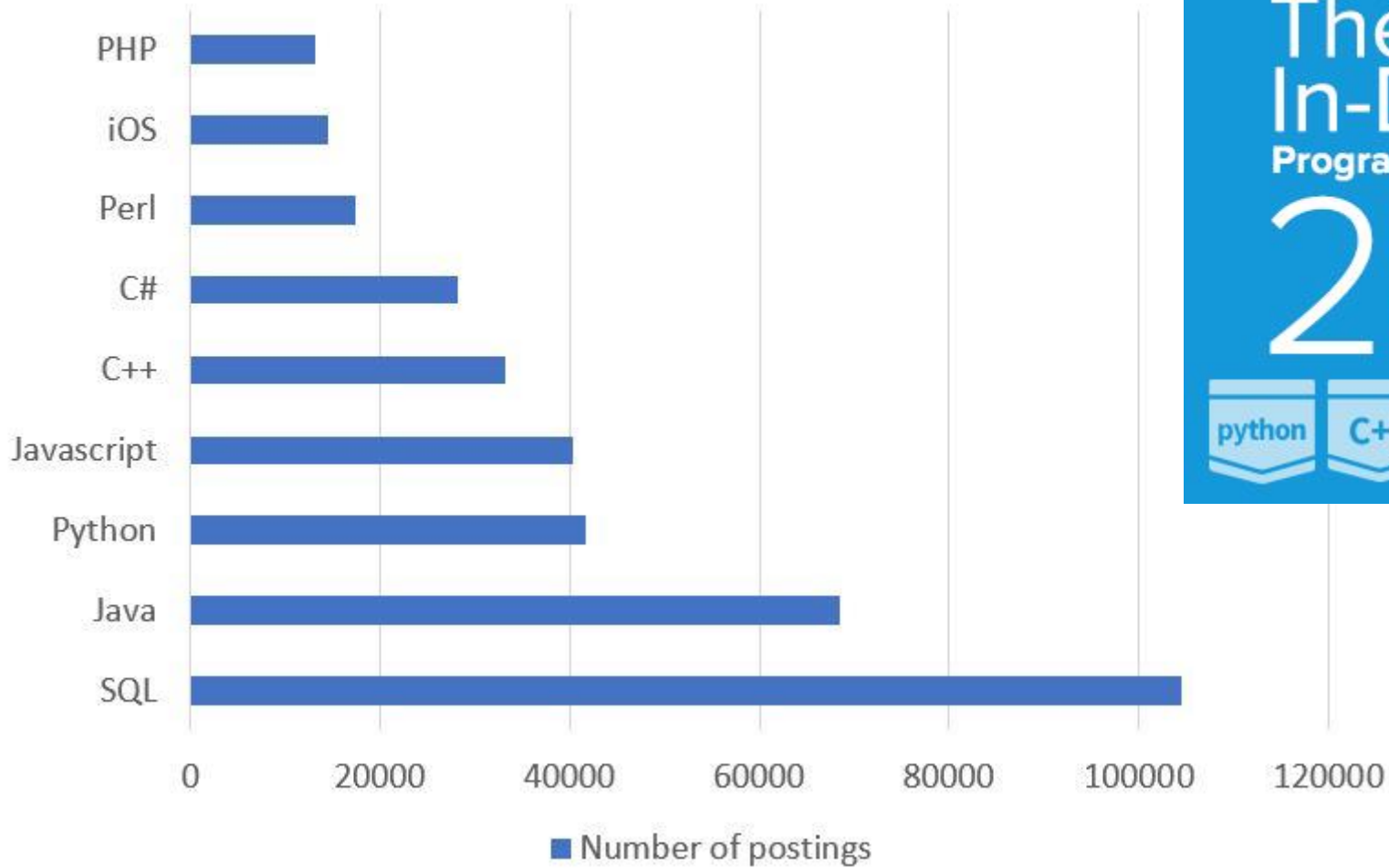
Evolution of Programming Languages



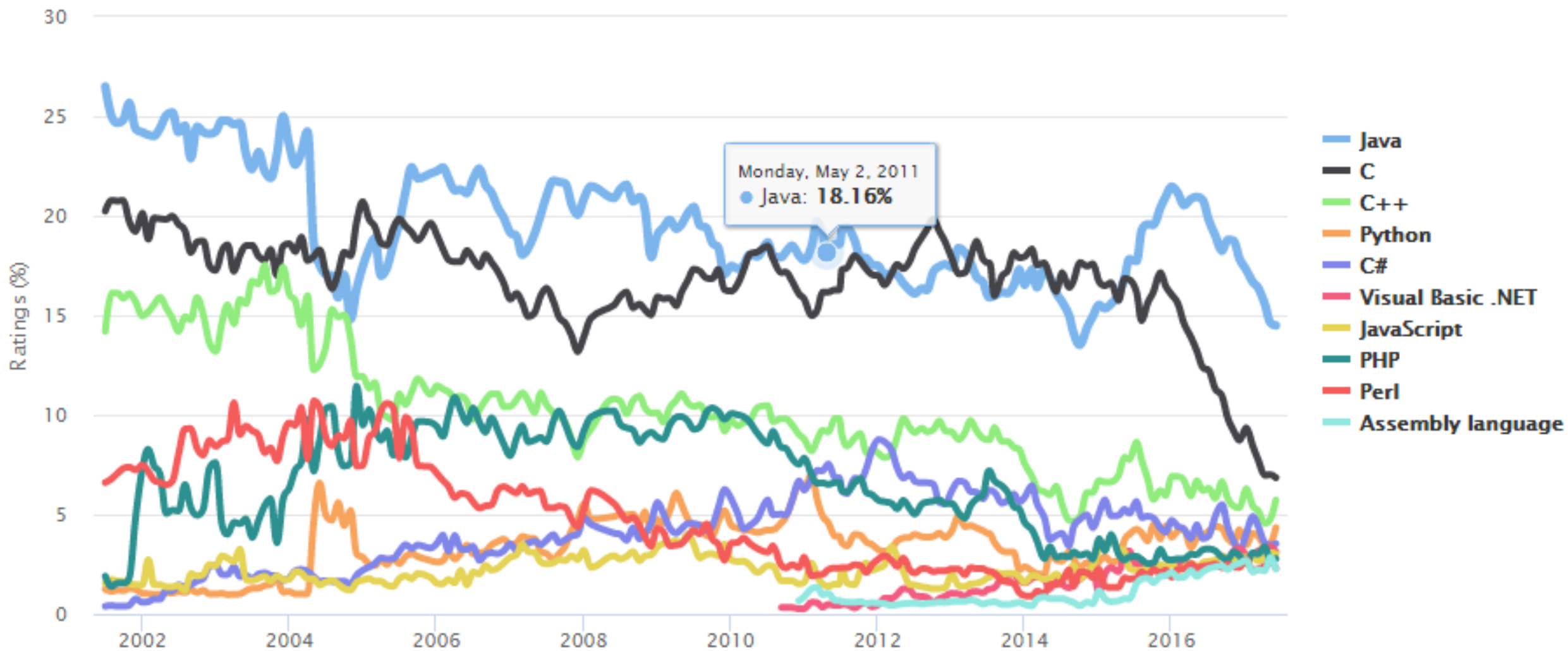
Most Popular Coding Languages of 2015



Number of Indeed Job Postings by Programming Language



TIOBE Index for June 2017



Course Logistics

- Staff
 - [Junjie Cao](http://jjcao.github.io), <http://jjcao.github.io>
 - jjcao@dlut.edu.cn
- Website: <http://jjcao.github.io/cPlusPlus/>

How to Succeed?

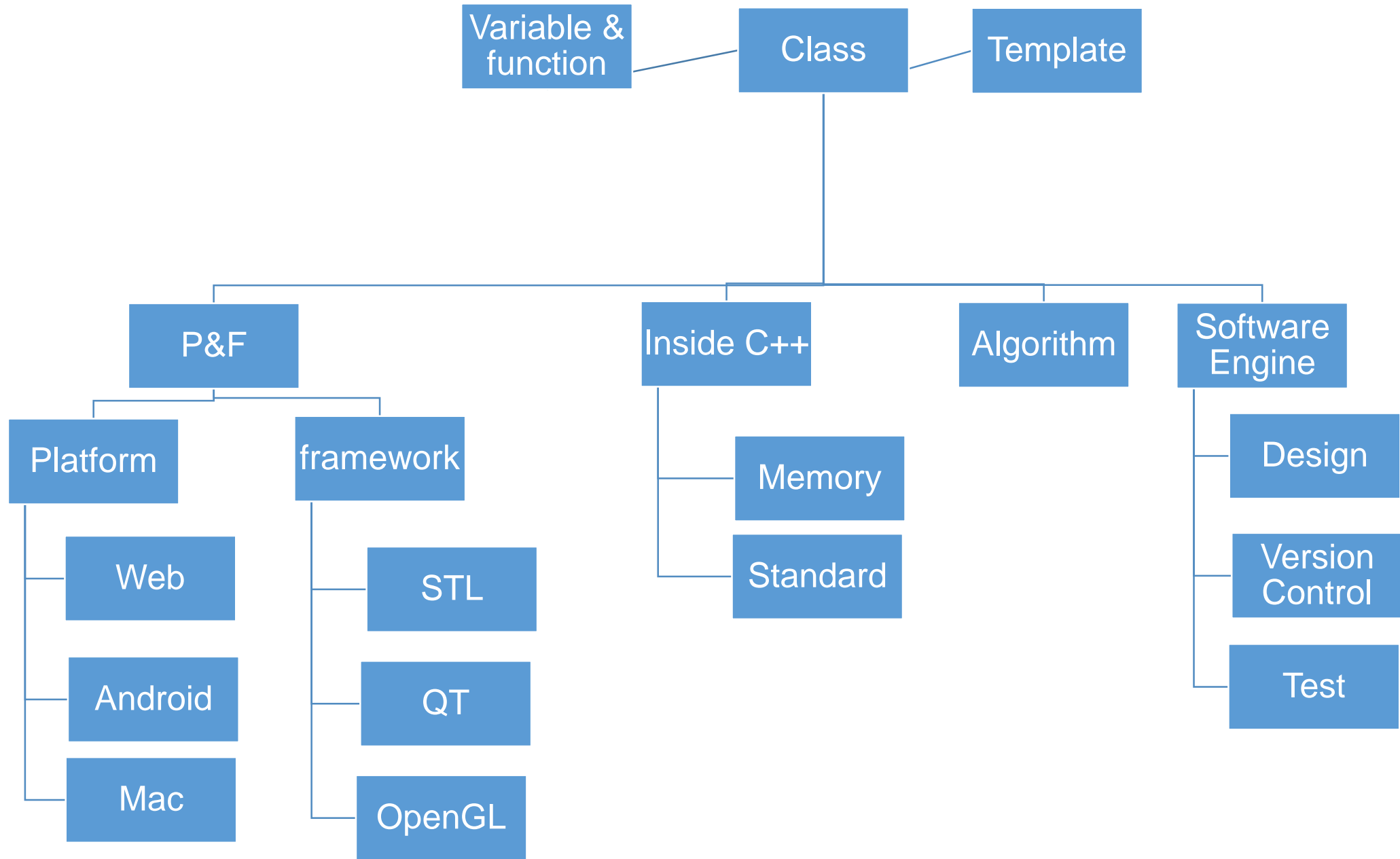
- An introductory course
- 56 hours (32 talks + 24 practices) in 4 weeks
- Every one of you can succeed
 - Little, even without programming background is acceptable
 - There is no such thing as a “**born programmer**”
 - **Work hard**
 - Follow directions
 - Be methodical: Think before you act
 - Try on your own, then ask for help, **Google!!!**
 - **Start early**



Research and Interview Oriented

- Scientist
(thesis + **research**)
- IT, even other **job**
(interview + work)





Objectives and Philosophy



Example driven



Question based

Examination

item	ratio
Attendance & Quizzes	30%
Exam	70%

Video

- [The birth of the computer](#), George Dyson
- [SageMath – Open source is ready to compete with Mathematica for use in the classroom](#), William Stein

程序员 vs 程序猿



General ideas about C++

- A computer is a **processor** with some **memory**, capable of running tiny **instructions** like “store 5 in memory locations 23459.”
- C++: **more abstract, easy:**
 - Conciseness: 1 = many
 - Maintainability: easier to modify
 - Portability: suitable for different types of processor
- C++ is a **high-level** language, **compiled** language, strong **types**, **case sensitive**.

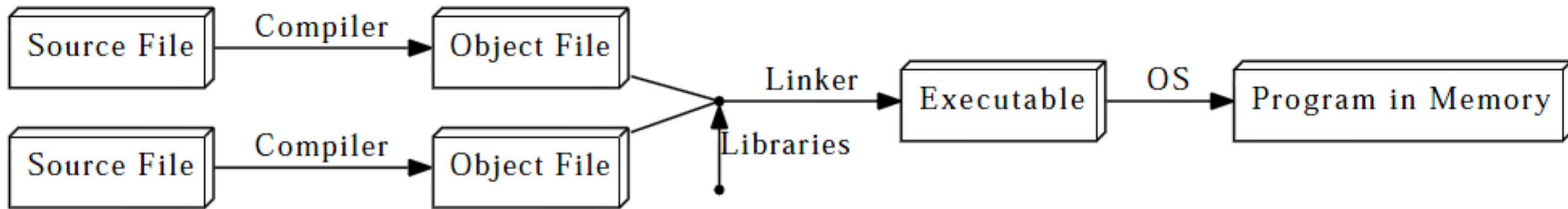
```
int main(){  
    return 0;  
}
```

The Compilation Process

Our language v.s. binary language the computer used

C++ is like natural language

Compiler: make computer understand C++



More

- Created in 1979 by (Extensions to C)



Bjarne Stroustrup
[bijani sdʒəʊsdʒup]



- **Console** programs is much easy and more portable than graphical programs

Hello World

```
// A Hello World program
# include <iostream>
int main()
{
    std::cout << "Hello, world!\n";
    return 0;
}
```

Line-By-Line Explanation

- `//`

indicates that everything following it until the end of the line is a **comment**: it is ignored by the compiler.

- `/* and */`

- (e.g. `x = 1 + /*sneaky comment here*/ 1;`)
- multiple lines;

```
// A Hello World program
# include <iostream>
int main() {
    std::cout << "Hello, world!\n";
    return 0;
}
```

- Usages

- Comments exist to **explain non-obvious** things going on in the code. Use them: **document** your code well!

```
// A Hello World program
# include <iostream>
int main() {
    std::cout << "Hello, world!\n";
    return 0;
}
```

- **# preprocessor commands**

- Lines beginning with **#** are preprocessor commands, which usually change what code is actually being compiled.
- **#include** tells the **preprocessor** to dump in the contents of another file, here the `iostream` file, which defines the procedures for input/output.

```
// A Hello World program
# include <iostream>
int main() {
    std::cout << "Hello, world!\n";
    return 0;
}
```

- **int main()**

- main is a function name
- Brackets () with main tells that main() is a function
- int before main() indicates integer values is being returned by main()
- When program is loaded in the memory, the control is handed over to function main () and it is the **first** function to be executed.

```
// A Hello World program
# include <iostream>
int main() {
    std::cout << "Hello, world!\n";
    return 0;
}
```

- **Curly bracket and body of function main()**

- A C++ program starts with function called main().
- The body of the function is enclosed between curly braces. They represent grouping of multiple commands into a block.
- Each commands/statement must end by a semicolon.
- More about this syntax in the next few lectures.

```
// A Hello World program
# include <iostream>
int main() {
    std::cout << "Hello, world!\n";
    return 0;
}
```

- `cout <<`
- This is the syntax for outputting some piece of text to the screen.


```
// A Hello World program
# include <iostream>
int main() {
    std::cout << "Hello, world!\n";
    return 0;
}
```

- **:: Namespaces**

- In C++, identifiers can be defined within a context – sort of a directory of names – called a namespace.
- When we want to access an identifier defined in a namespace, we tell the compiler to look for it in that namespace using the scope resolution operator (::).
- Here, we're telling the compiler to look for cout in the std namespace, in which many standard C++ identifiers are defined.
- A cleaner alternative is to add the following line below line 2:

using namespace std;

- This line tells the compiler that it should look in the std namespace for any identifier we haven't defined.
- If we do this, we can omit the std:: prefix when writing cout. This is the recommended practice.

```
// A Hello World program
# include <iostream>
int main() {
    std::cout << "Hello, world!\n";
    return 0;
}
```

- Strings

- A sequence of characters such as *Hello, world* is known as a string.
- A string that is specified explicitly in a program is a **string literal**. 字符串字面量

- \n

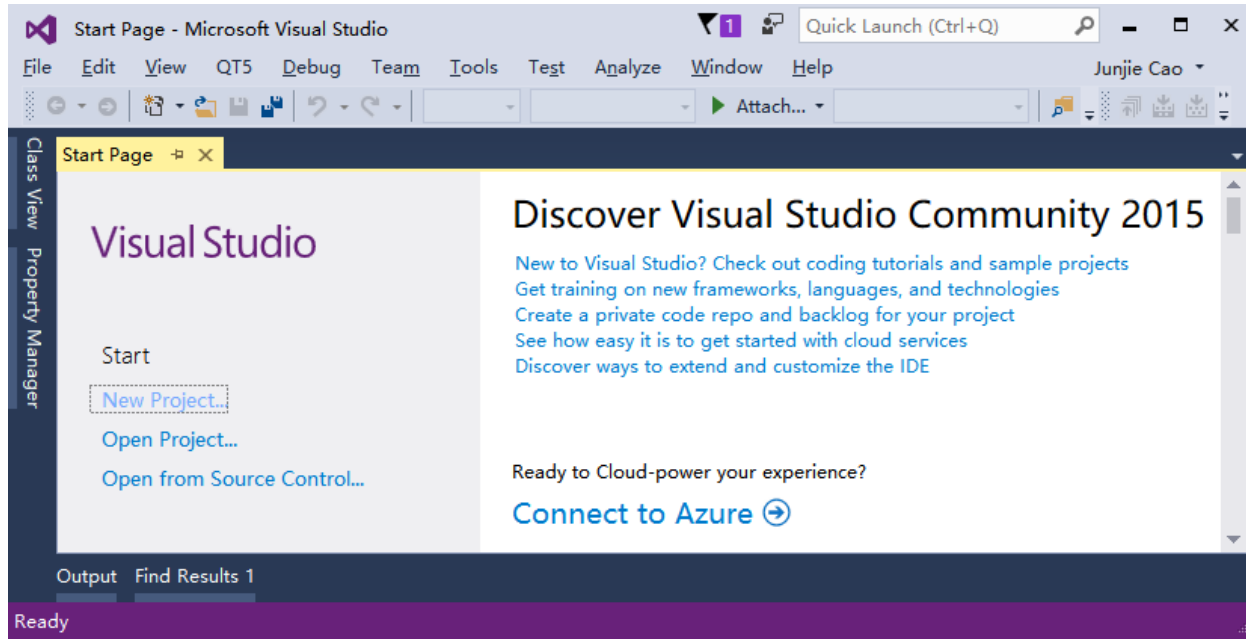
- Escape sequences: The \n indicates a newline character. It is an example of an escape sequence – a symbol used to represent a special character in a **text literal**.

```
// A Hello World program
# include <iostream>
int main() {
    std::cout << "Hello, world!\n";
    return 0;
}
```

- **return 0**
 - Indicates that the program should tell OS it has completed successfully.
 - it as the last line in the main block.
- Note that every statement ends with a **semicolon** (except **preprocessor commands** & blocks using **{}**).
- **Forgetting** these semicolons is a common mistake among new C++ programmers.

Integrated Development Environment

- Visual C++: Windows
- Code::Blocks: Linux
- Xcode, Eclipse: Mac



- **CodeChef: Web based**
 - Web-based compilers are fine for dabbling and simple exercises. However, they are generally quite limited in functionality -- many won't allow you to save projects, create executables, or effectively debug your programs. You'll want to migrate to a full IDE when you can.



- [Installing an Integrated Development Environment \(IDE\)](#)

Compiling your first program

- [lab01_IDE_VC_Win32ConsoleApplication.pptx](#)
- [LearnCpp.com](#)

A few common C++ problems

- LearnCpp.com

Reference Courses

- [cpp for school](#)
 - simpler and with assignments, projects, quiz and papers.
- [LearnCpp.com](#)
 - more detail explanations than cpp for school

Reference Books

1. C++ Primer

2. The C++ Programming Language. (more advance than 1)
3. The C++ Standard Library – A Tutorial and Reference
4. Teach Yourself C++ in One Hour a Day
5. Code complete 2nd
6. Clean Code A Handbook of Agile Software Craftsmanship

Useful Links

- <http://www.cplusplus.com>

Academic Integrity

- Honest work is required of a scientist or engineer.
- Integrity is the key for everything!!!
- Discussion is permitted.
- Everything you turn in must be your own work.
- Cite your sources, explain any unconventional action.
- If you have a question, ask.