C++ Program Design

-- hello world in details

Junjie Cao @ DLUT

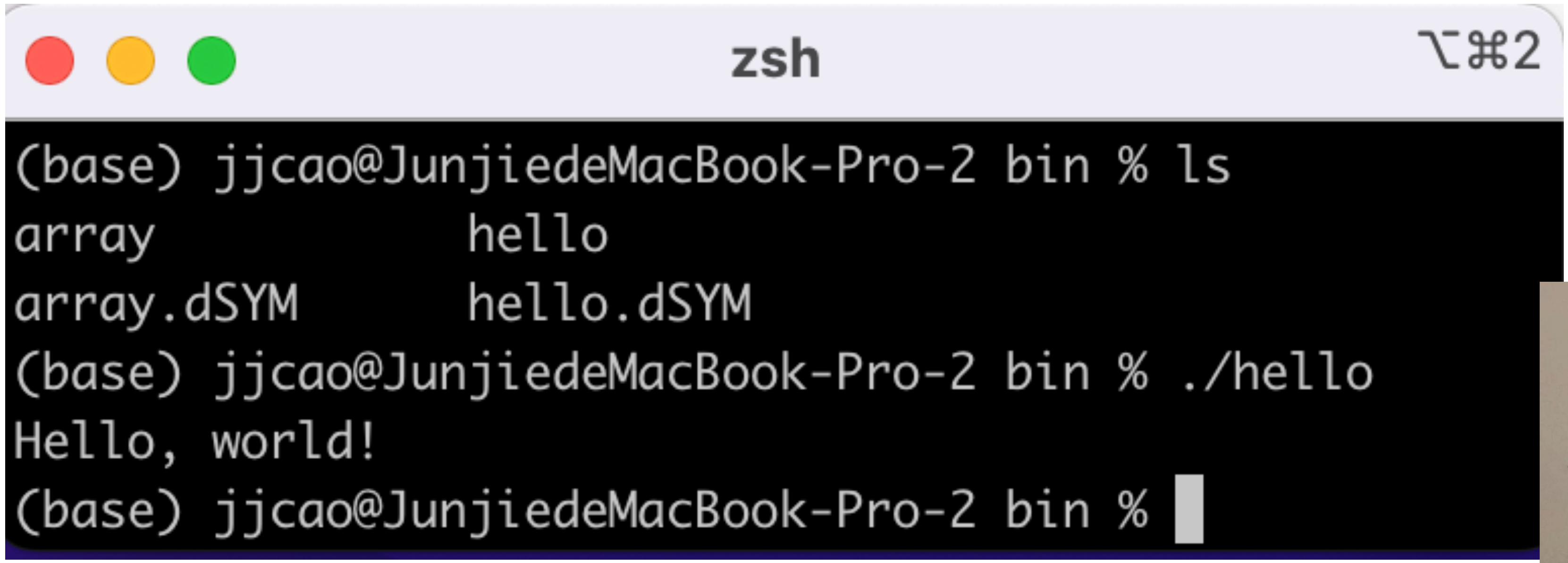
Summer 2022

https://github.com/jjcao-school/c

4 Your 1st Program

控制台程序(Console programs)

远比图形接口程序容易实现和迁移到不同的操作系统



```
(base) jjcao@JunjiedeMacBook-Pro-2 bin % ls
array                hello
array.dSYM           hello.dSYM
(base) jjcao@JunjiedeMacBook-Pro-2 bin % ./hello
Hello, world!
(base) jjcao@JunjiedeMacBook-Pro-2 bin %
```

Hello World

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

Line-By-Line Explanation

- // 注释comment

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
    them: **document** your code well!

Line-By-Line Explanation

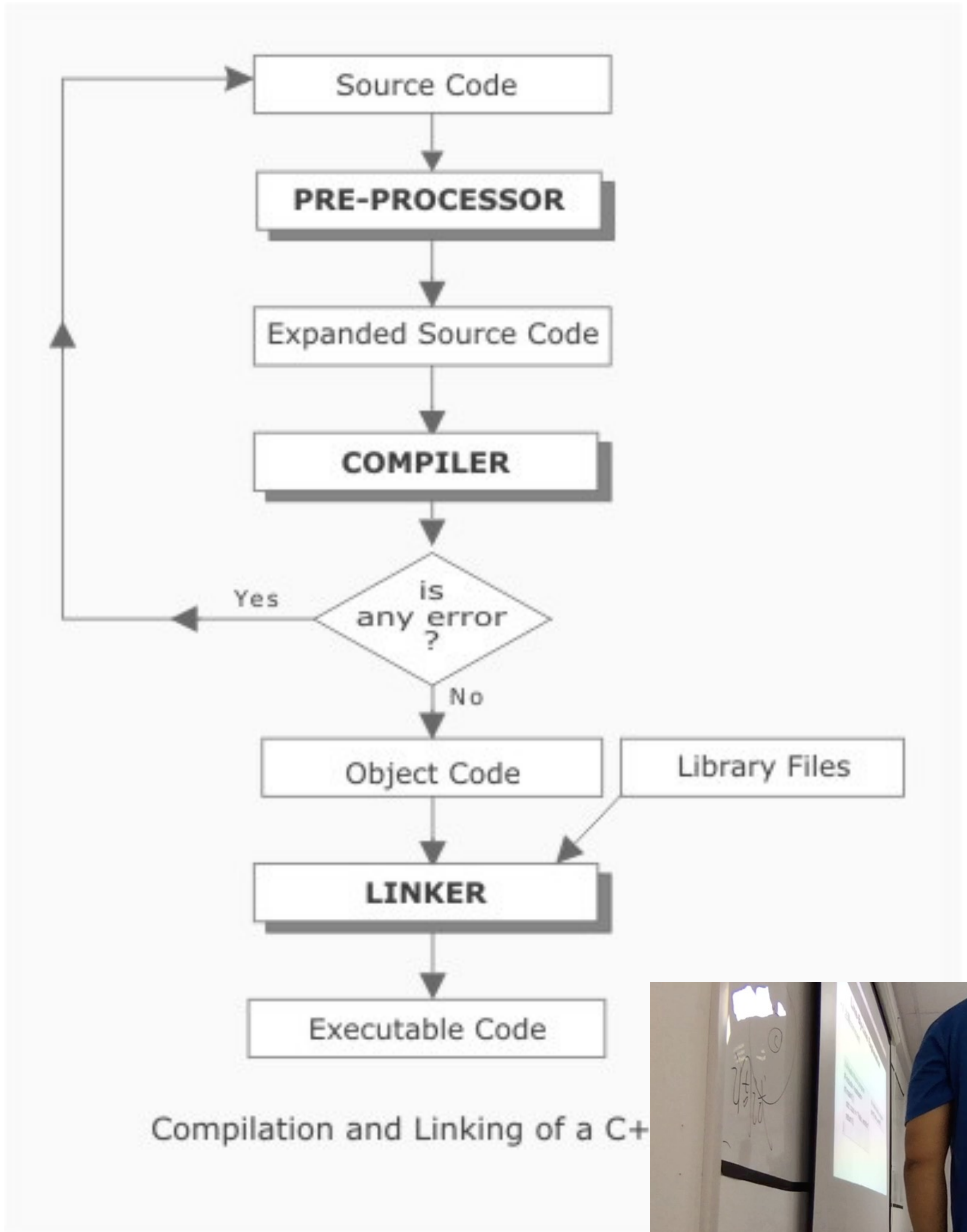
- // 注释comment

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

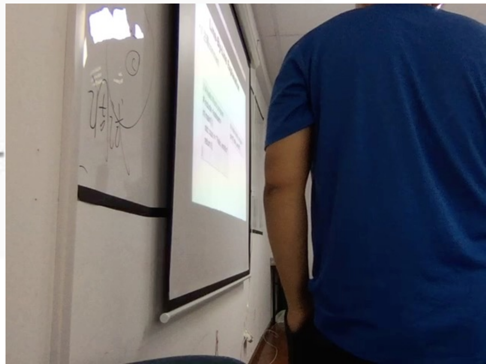
```
# A Hello World program
print("Hello, world!")
```

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

- **# preprocessor commands**
  - 用#开始的行是预处理命令(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.



Compilation and Linking of a C+



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

```
# A Hello World program
def main():
    print("Hello World!")

main()
```

- int main()
  - main 函数名
  - 跟随mian的()说明它是一个函数
  - main()之前的int表明该函数返回一个整数值
  - 当程序被执行（载入内存），main()是第一个被执行的函
    入口）

Why?

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

```
# A Hello World program
def main():
    print("Hello World!")

main()
```

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

```
// not OK
def main(): print("Hello World!") main()
```

- See next page.

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

```python
# A Hello World program
def main():
    print("Hello World!")

main()
```

- C++大括号{}表明main()的函数体
  - {}把多个命令组成一组命令：multiple commands =》a block代码块
  - 每一个命令/声明（command/statement）必须分号结尾
  - More about this syntax in the next few lectures.

- Python uses leading whitespace to mark scope: Tab

```
// 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;
}
```

- std是一个名称空间Namespaces
  - 作用域解析操作符scope resolution operator ::
  - 通知编译器要调用std中的cout，而不是别处jjcao::cout

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.

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

- 字符串String
  - *Hello, world*
  - 像这样显示指定的字符串，叫string literal.字符串字面量

- \n
  - The \n indicates a **newline** character.
  - 转义序列（Escape sequences）: 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;
    std::cout << "Hello, again!\n";
}
```

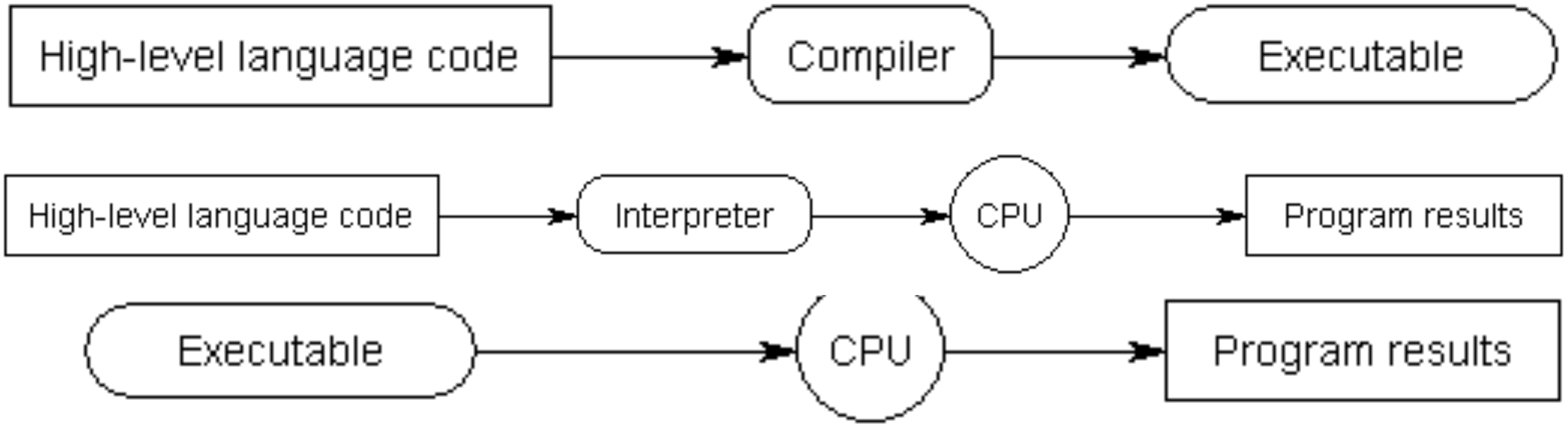
- return 0
  - 通知OS，本程序成功执行完毕。
  - 是main block的最后一行
- 注意
  - 每一个声明需要分号结束（预处理命令和{}除外（如果是定义class的时候，{}也要跟着分号））
  - 忘记分号，是新手常犯错误

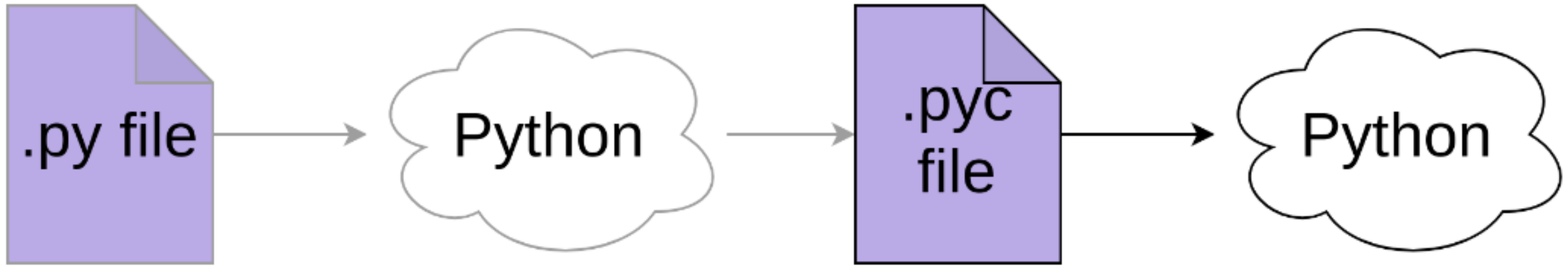
The Compilation Process

Our language v.s. binary language the computer used

C++ is like natural language

Compiler: make computer understand C++

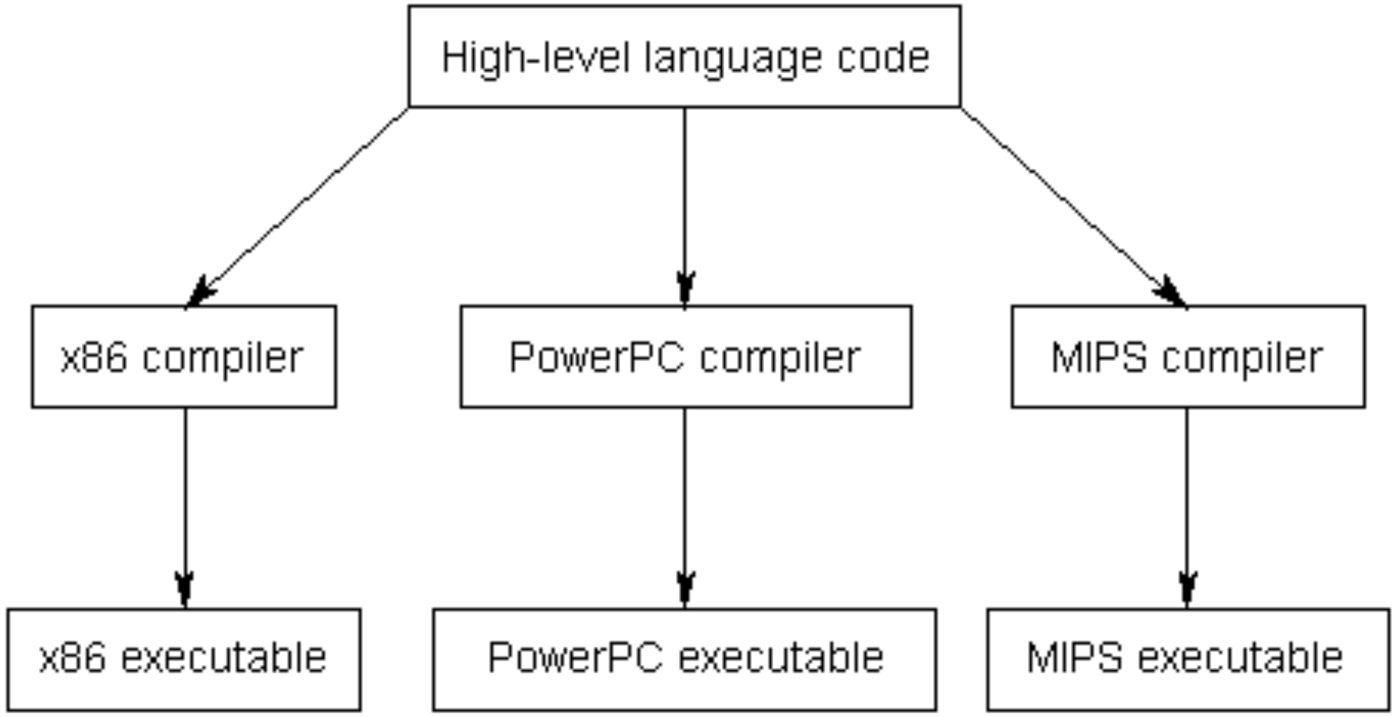


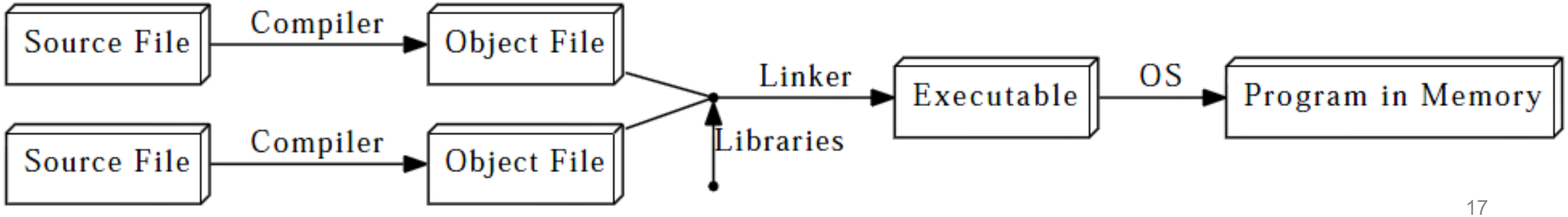


- Python compiles to bytecode instead of native machine code.
- Bytecode is the native instruction code for the Python virtual machine.
- To speed up subsequent runs of your program, Python stores the bytecode in .pyc files:

The Compilation Process

Compiler: make computer understand C++





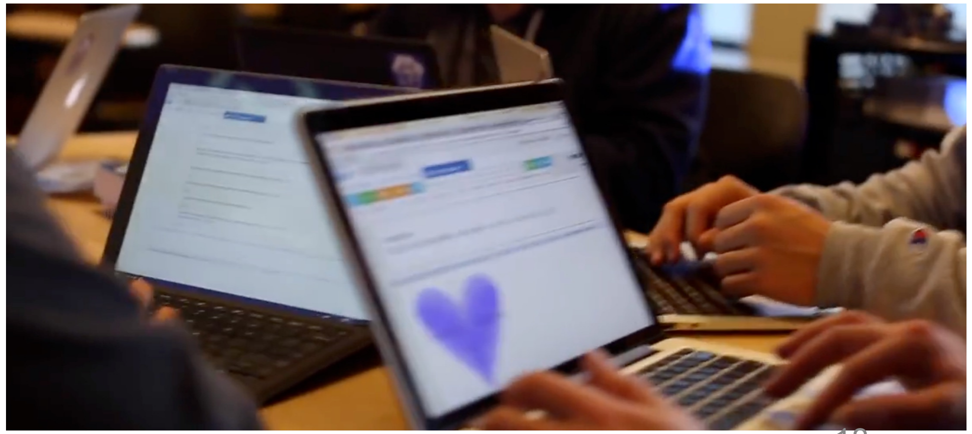
How to construct your virtual world?

- Every creative activity needs tools: a sheet of paper and a pencil?

- **Running** the code is the **only method** of finding out whether it’s correct.
  - a computer equipped with some additional tools.

- A standard text editor and command-line compiler tools? May talk this later.

- An **IDE** is better.
- Or On-line tools?



IDE (Integrated Development Environment)

- A software: a code **editor, a compiler, a debugger**, and a graphical user interface (GUI) builder.
  - consume a lot of resources and, frankly speaking, you probably don't need most of the functions they can perform.

- If using on-line tools: an Internet browser + Internet access. But …

- Choose the one that's more convenient for you.

eclipse

Visual Studio

Code::Blocks

NetBeans

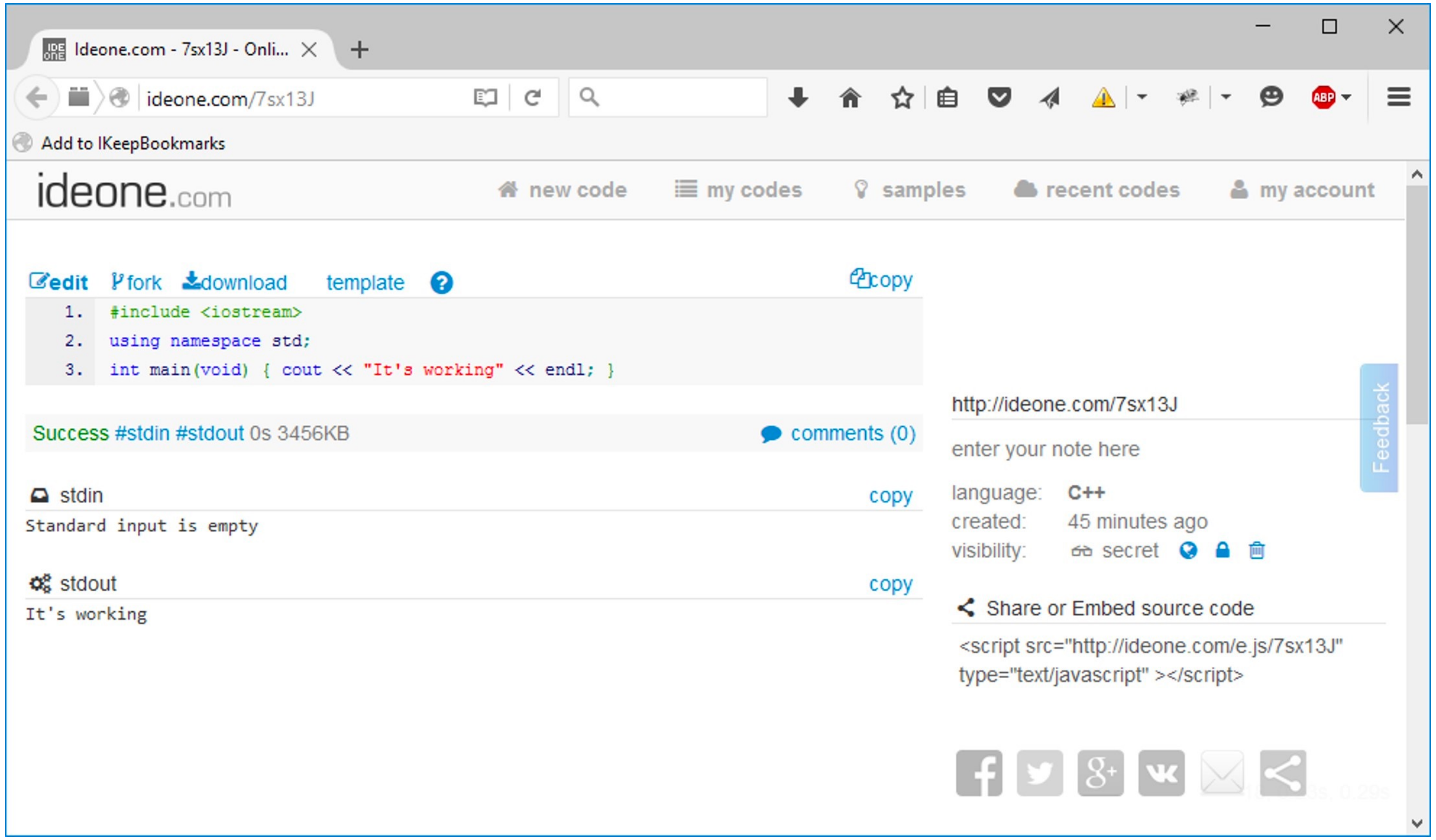
XCODE

Visual Studio Code

[Installing an Integrated Development Environment (IDE)]()

On-line tools: ideone

- http://ideone.com
- http://cpp.sh



```
#include <iostream>
using namespace std;
int main(void) { cout << "It's working" << endl; }
```

Success #stdin #stdout 0s 3456KB   comments (0)

stdin

```
Standard input is empty
```

stdout

```
It's working
```

http://ideone.com/7sx13J

enter your note here

language: C++
created: 45 minutes ago
visibility: secret

Share or Embed source code

<script src="http://ideone.com/e.js/7sx13J" type="text/javascript" ></script>

编译你的第一个程序

- lab01_IDE_vscode_helloworld.pptx
- lab01_IDE_VC_Win32ConsoleApplication.pptx

- LearnCpp.com

Errors

1. Syntax errors
2. Run-time errors

Syntax errors

- Run the code:

```
print("Hello, world)
```

- The code won’t run! IDLE displays:

```
EOL while scanning string literal.
```

- EOL stands for End Of Line, so this message tells you that Python
- read all the way to the end of the line without finding the end of something called a string literal 字符串字面量.

- A string literal is text contained in-between two double quotation marks. The text "Hello, world" is an example of a string literal.
- For brevity, string literals are often referred to as strings

Run-time errors

- Run the code:

```
print(Hello, world)
```

- What do you think happens when you run the script? Try it out & see!

```
EOL while scanning string literal.
```

- What happened?

```
Traceback (most recent call last):
File "/home/hello_world.py", line 1, in <module>
print(Hello, world)
NameError: name 'Hello' is not defined
```

Run-time errors - continued

```
print(Hello, world)
```

- What happened?

```
Traceback (most recent call last):
File "/home/hello_world.py", line 1, in <module>
print(Hello, world)
NameError: name 'Hello' is not defined
```

Create a Variable 变量

```
phrase = "Hello, world"
print(phrase)
```

```
std::string phase = "Hello world";
std::cout << phase << std::endl;
```

- variables are names that can be assigned a value and used to reference that value throughout your code.

- =: The Assignment Operator. Try this & compare the error info with :

```
#phrase = "Hello, world"
print(phrase)
```

```
print("Hello, world)
```

Variable names are case-sensitive

```
>>> phrase = "Hello, world"
>>> print(Phrase)
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
NameError: name 'Phrase' is not defined
```