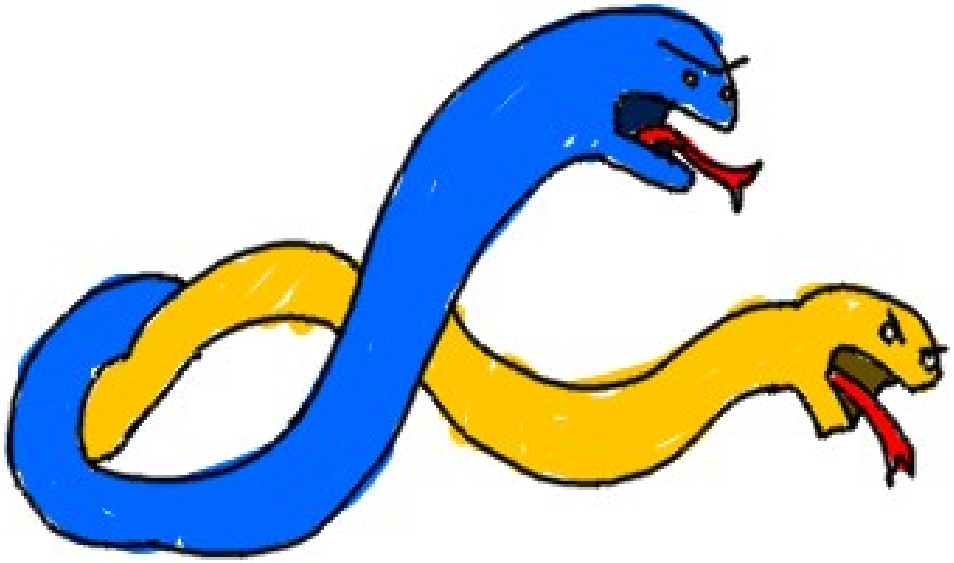
Python vs. C++



Written by Rachel J. Morris, last updated 2013-05-24

Why C++?

- Python and C++ are both pretty widely used, though in different domains.
- C++, Java, and C# are more common for business software development.
- Python may be used on the testing-side. Django is a web framework that uses Python (think PHP but way more awesome)

In Kansas City...

- Cerner has been shifting towards using Python, but also wants Java, C++, and C# developers
  https://www.cerner.com/About_Cerner/Careers/
- Garmin wants C, C++, C#, Java, and ASM
  http://www.garmin.com/en-US/company/careers/
- Perceptive Software wants C++, Java, and C#
  http://www.perceptivesoftware.com/company/careers/north-america
- IBM wants Java
  http://www-03.ibm.com/employment/us/
- *Make sure to get internships while you're in school – it's easier to get your first job after graduation this way!*

Useful Languages...

- Software companies use a lot of Java, C#, and C++.
- Web companies may want PHP, ASP.NET, Django, or Ruby on Rails.
  - Django was developed in Lawrence, KS!
- Game development companies still heavily use C++, though Java and Objective C are more popular due to mobile devices, and C# for Microsoft platforms.
- Embedded platforms tend to use C and Assembly
- There are still jobs for legacy software maintenance – Cobol, Fortran, etc.

What can you do with C++?

- UI programs
  - wxWidgets, qtCreator, MFC, ...
- Network applications
  - POSIX Sockets, Winsock, high level libraries...
- Web Servers
- Game Development
  - SFML, SDL, Allegro, OpenGL, …
- Data Visualization
  - VTK, OpenDX, …
- ...Anything...

Python vs. C++

- Input
- Output
- Variables
- If Statements
- Loops
- Functions
- Classes
- File I/O
- Random Numbers

Syntax

Python

Python statements generally just go on one line.
If statements, loops, and other statements that contain an inner body end with a colon : and on the next line any internal code is indented by one tab.
PYTHON IS SENSITIVE TO WHITESPACE

```
if ( balance < 0 ):
    print( "Your balance is under $0!" )
```

C++

Statements in C++ end with a semicolon ;
If statements, loops, and other statements that contain an inner body do NOT end with a semicolon ; , and their internal code is contained within curly braces { }
C++ IGNORES WHITESPACE – but you should still indent for clean code!

```
if ( balance < 0 )
{
    cout << "Your balance is under $0!" << endl;
}
```

Syntax

Python

Comments begin with a pound:

```
# Verifies the user's choice
```

C++

Single-line comments begin with two forward-slashes:

```
// Verifies the user's choice
```

Multiple-line comments begin with /* and end with */:

```
/* My Program
    Last update 2013-03-02
    This program generates
    fibonacci numbers.
*/
```

Your starting program

Python

Python begins executing at the first line in the file. There is no need to specify a starting point.

Simple.py

```
sum = 4 + 5
```

C++

C++ requires a main function created, **main** is always the entry point in a C++ program.

Simple.cpp

```
int main()
{
    int sum = 4 + 5;
    return 0;
}
```

Your starting program

C++

- C++ requires that you "**return 0**" at the end of the program.
- This is a throwback to an old way of checking for errors – return 0 for zero errors, and return any other number to specify different error codes.

```
int main()
{
     int status = InitializeProgram();

     if ( status == 1 )
     {
          LogError( "Error starting program" );
          return 10;
     }
     else if ( status == 2 )
     {
          LogError( "Error loading in images" );
          return 20;
     }

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

Importing Libraries

Python

In Python, you can import additional functionality with:

```
import random
# or
from random import randint
```

C++

In C++, you can import additional functionality with:

```
// Input/output library:
#include <iostream>

// String library:
#include <string>
```

Importing Libraries

- In C++, the Input/Output library is part of the std (standard) namespace.
  - Using namespace require a prefix when using its members...

```
#include <iostream>

int main()
{
   // std::cout (console-out)
   // std::endl (end-line)

   std::cout << "Hello, World!" << std::endl;

   return 0;
}
```

Importing Libraries

- OR, if you use the **using namespace std;** statement, you can use items from the **std** namespace without the prefixes.
- We'll cover these more in-depth later in the semester. For now, just remember to use **using namespace std;** in your programs.

```
#include <iostream>
using namespace std;

int main()
{
   // cout (console-out)
   // endl (end-line)

   cout << "Hello, World!" << endl;

   return 0;
}
```

Output

Python

In Python, we can output with the **print** function:

```
print( "hello!" )
```

C++

C++ requires that you include the **iostream** (input-output stream) library, and then you can use **cout** (console-out) and **endl** (end-line) to write to the screen.

```
#include <iostream>
using namespace std;

int main()
{
    cout << "hello!" << endl;
}
```

Output

Python

You can link together multiple strings and variables like this:

```
name = "Rebekah"
level = 20

print( "Hello, " + name )
print( "You are now level " + str( level ) )
```

We can freely **concatenate** strings together in Python, but if we want to print the number stored in the **level** variable after writing a string, we must **cast** it to a string.

Output

C++

You can link together multiple strings and variables like this:

```
#include <iostream>
#include <string>
using namespace std;

int main()
{
    string name = "Rebekah";
    int level = 20;

    // You can continue a statement over multiple lines
    cout << "Hello, " << name
        << endl << "You are now level " << level << endl;
    return 0;
}
```

Here, we don't have to cast our variable **level** to a string. C++ knows exactly what **level** is already and can stream it out to the console (**cout**).

Output

- C++ doesn't automatically end the line. You can use the escape character **\n** inside of a string, or **endl** as part of the stream to go to a new line.

```
string name;
cout << "What is your name? ";
cin >> name;

cout << endl << "Why hello, " << name << endl;
cout << "\n How are you?";
```

```
What is your name? Jim

Why hello, Jim

How are you?
```

Input

- To get input from the user, use **cin**.
- Notice that stream “flow” is in a different order:
  - cout << “hello, world!” << endl;
  - cin >> variable_name;
- For **cout**, think of your message “streaming” out to the console.
- For **cin**, think of the input from the console “streaming” into the variable.

Input

- In C++, whitespace received is automatically handled as the **delimiter**.
- This means that when it hits a space, tab, or new-line, it will consider the current item being read as finished.
- If you enter data after a space when getting data through **cin** >> variable_name, if there is another **cin** immediately afterward it will pass the extra data to the next cin.

Input

Sample Program

```
string fname, lname;
cout << "What is your name?" << endl;
Cout << "? ";    // Let user know they're being prompted
cin >> fname;
Cout << "? ";    // Let user know they're being prompted
cin >> lname;
cout << "\n Hello " << fname << " " << lname << endl;
```

Output one name at a time

```
What is your name?
? William
? Shatner

Hello William Shatner
```

Input multiple names

```
What is your name?
? William Shatner
?
Hello William Shatner
```

I/O

Input

```
// iostream - I/O
// string - use string datatype
#include <iostream>
#include <string>
using namespace std;

int main()
{
     string name;
     int age;
     float money;

     cin >> name;
     cin >> age;
     cin >> money;

     return 0;
}
```

Output

```
// iostream - I/O
// string - use string datatype
#include <iostream>
#include <string>
using namespace std;

int main()
{
     string name = "LeChuck";
     int age = 100;
     float money = 25.49;

     cout << "Name: " << name;
     cout << "\t Age: " << age;
     cout << "\t Money: $" << money << endl;

     return 0;
}
```

Variables

Python

To create a variable in Python, you simply start using it.

```
myString = "Good afternoon!"
myInteger = 50
myFloat = 9.99
myBoolean = True
```

C++

C++ requires that you **declare** a variable.

A declaration must specify the variable's **data-type**.

```
string myString = "Good afternoon!";
int myInteger = 50;
float myFloat = 9.99;
bool myBoolean = true;
```

Variables

- Once a variable is declared as a certain data-type in C++, it cannot be changed.
- You can **cast** variables to another data-type, but it must be stored in a variable with the correct **data-type**.

```
float tileWidth = 25.5;

// Implicit cast – drops the decimal in 25.5
int screenWidth = tileWidth * 10;

// Explicit C style cast:
int screenWidth = (int)tileWidth * 10;

// Explicit C++ style cast:
int screenWidth = int(tileWidth) * 10;
```

Input, Output, & Variables

Additional Reading

- http://www.cplusplus.com/doc/tutorial/variables/
- http://www.tutorialspoint.com/cplusplus/cpp_variable_types.htm