

Differences between C++ and Python

1. C++ is a compiled language. All source files of the program must be compiled and linked into a binary executable before running the program. Python is an interpreter so Python source code is executed as the source files are interpreted. Imported Python modules may be converted to .pyc files. These aren't the same as binary executables. These are simply lower-level meta-language files that are optimized for execution. These make Python a faster and more efficient interpreter compared to other interpreter languages. Both .py and .pyc files are platform indecent but C++ executables are not platform independent. The C++ source code must be re-compiled for each platform when needed.
2. Due to properties described in first item, it is easier to distribute and release C++ executable files compared to Python source. Typically there are far fewer file dependencies when releasing C++ programs. However, execution of Python programs requires distribution of the interpreter and all the modules. There are tools to convert Python programs to binary platform dependent executables, but these files tend to be very large. In many cases this is considered wasteful.
3. Python doesn't require variables to be defined and declared prior to use as is the case in C++ programs. Python programs do not need to manage memory allocation as may be required in equivalent C++ program.
4. Python's syntax depends on indentation to define scope. It also doesn't require semi-colon at the end of each statement nor does it require curly brackets to define scope.

Consequence of Item 1. is that C++ program execution is faster than equivalent Python program. Consequence of Item 3. and Item 4. makes programming in Python more productive.