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 independent 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 group statements into a code block and define scope. It also doesn't require semi-colin at the end of each statement nor does it require curly brackets to group a block of statements.

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.

Example C++

Example below from this link:
https://www.programiz.com/cpp-programming/examples/prime-number

Exercise: Translate this C++ program to Python.

Hints: In Python, *cout* translates to the *print* function and *cin* translates to the *input* function. The first two lines need no translation. In C++, *<iostream>* is a library needed for input/output statements that are mostly built-in in Python. The second line, *using*

namespace std; allows prefix *std::* to be omitted in *cin* and *cout* statements. Namespace rules as applied to Python will be discussed.

In Python, the block of code under the C++ function call: *int main()*, should be contained under the statement: *if __name__ == "__main__":*

Example: Check Prime Number

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

int main()
{
  int n, i;
  bool isPrime = true;

  cout << "Enter a positive integer: ";
  cin >> n;

  for(i = 2; i <= n / 2; ++i)
  {
       if(n % i == 0)
       {
            isPrime = false;
            break;
       }
  }
  if (isPrime)
       cout << "This is a prime number";
  else
       cout << "This is not a prime number";

  return 0;
}
```

Output

```
Enter a positive integer: 29
This is a prime number.
```