Python:

def sum\_of\_evens(n):

return sum(i for i in range(1, n+1) if i % 2 == 0)

print(sum\_of\_evens(10))

C++:

#include <iostream>

int sumOfEvens(int n) {

int sum = 0;

for (int i = 1; i <= n; i++) {

if (i % 2 == 0) {

sum += i;

}

}

return sum;

}

int main() {

std::cout << sumOfEvens(10) << std::endl;

return 0;

}

A good program for comparing Python and C++ could involve writing similar algorithms or functions in both languages and then analyzing the differences in syntax, performance, and other aspects. Here's a simple example comparing a basic function in both languages:

### Example: Finding the Sum of Even Numbers

#### Python

```python

def sum\_of\_evens(n):

return sum(i for i in range(1, n+1) if i % 2 == 0)

print(sum\_of\_evens(10))

```

#### C++

```cpp

#include <iostream>

int sumOfEvens(int n) {

int sum = 0;

for (int i = 1; i <= n; i++) {

if (i % 2 == 0) {

sum += i;

}

}

return sum;

}

int main() {

std::cout << sumOfEvens(10) << std::endl;

return 0;

}

In this example, you can compare how the same logic is expressed in both languages. You can also consider comparing aspects such as:

Syntax differences: e.g., C++ uses semicolons at the end of statements, while Python uses indentation for block structure.

Performance differences: C++ is typically faster than Python due to being a compiled language, whereas Python is interpreted.

Memory management: C++ requires explicit memory management, while Python has automatic memory management (garbage collection).