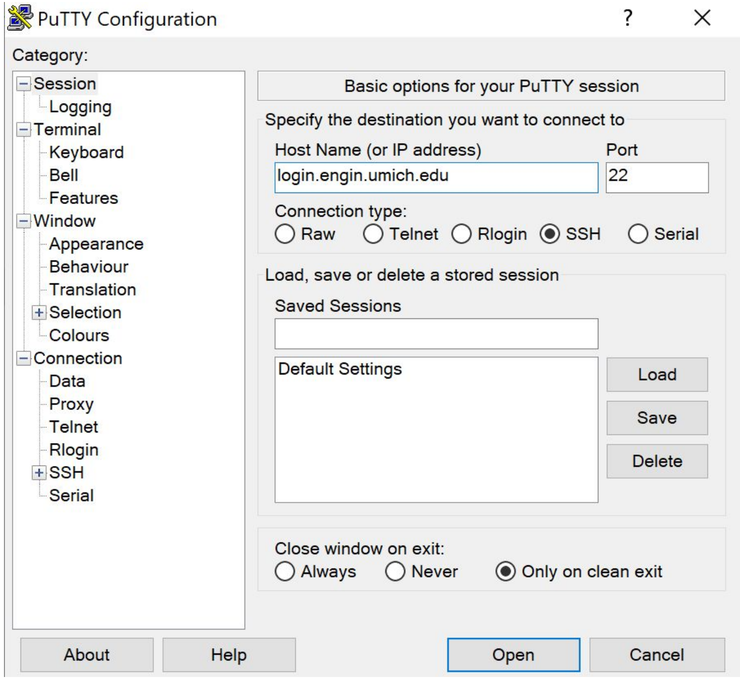
EECS402 Discussion 2!

FileZilla, Compiling on CAEN, Loops, If/else if, Functions

Getting back to CAEN

Open PuTTy

ENGR



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): login.engin.umich.edu

Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

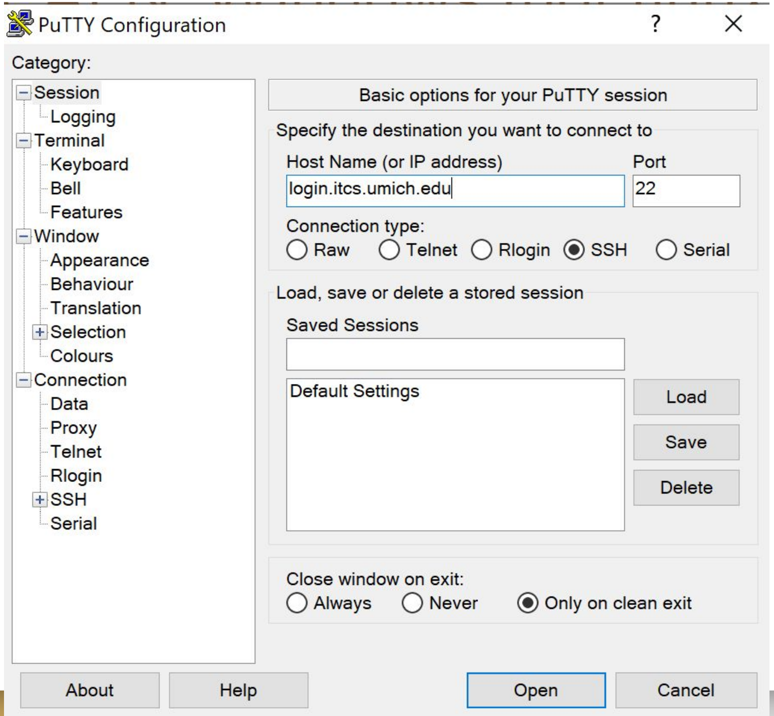
Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit

non-ENGR



Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): login.itcs.umich.edu

Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

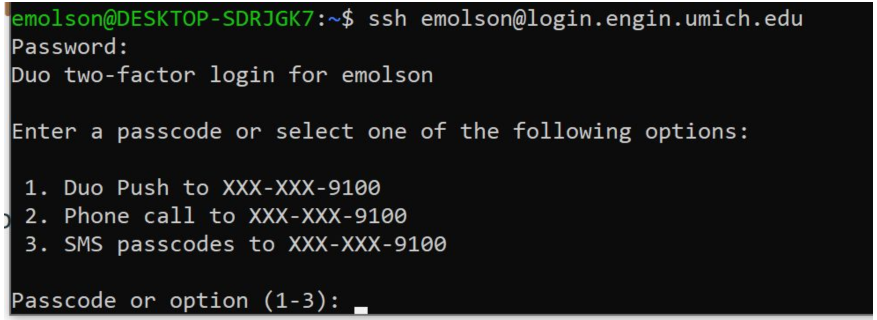
Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit

Getting back to CAEN



```
emolson@DESKTOP-SDRJGK7:~$ ssh emolson@login.engin.umich.edu
Password:
Duo two-factor login for emolson

Enter a passcode or select one of the following options:

 1. Duo Push to XXX-XXX-9100
 2. Phone call to XXX-XXX-9100
 3. SMS passcodes to XXX-XXX-9100

Passcode or option (1-3):
```

Open terminal (Mac) or Command Prompt (Windows)

If in CoE:

```
$ ssh <uniqname>@login.engin.umich.edu
```

If not in CoE:

```
$ ssh <uniqname>@login.itcs.umich.edu
```

Getting Files to CAEN

Recommended: FileZilla (https://filezilla-project.org/download.php?show_all=1)

- Note: Please use the link above. For some reason the “main download page” gave me a virus (yikes!)

Other options: pscp, scp, github, mfile

- Note: name files and folders without spaces to make your life easier

Setting Up FileZilla

1. Download the version that works for your OS
   a. https://filezilla-project.org/download.php?show_all=1 (Use this link, NOT the main link)

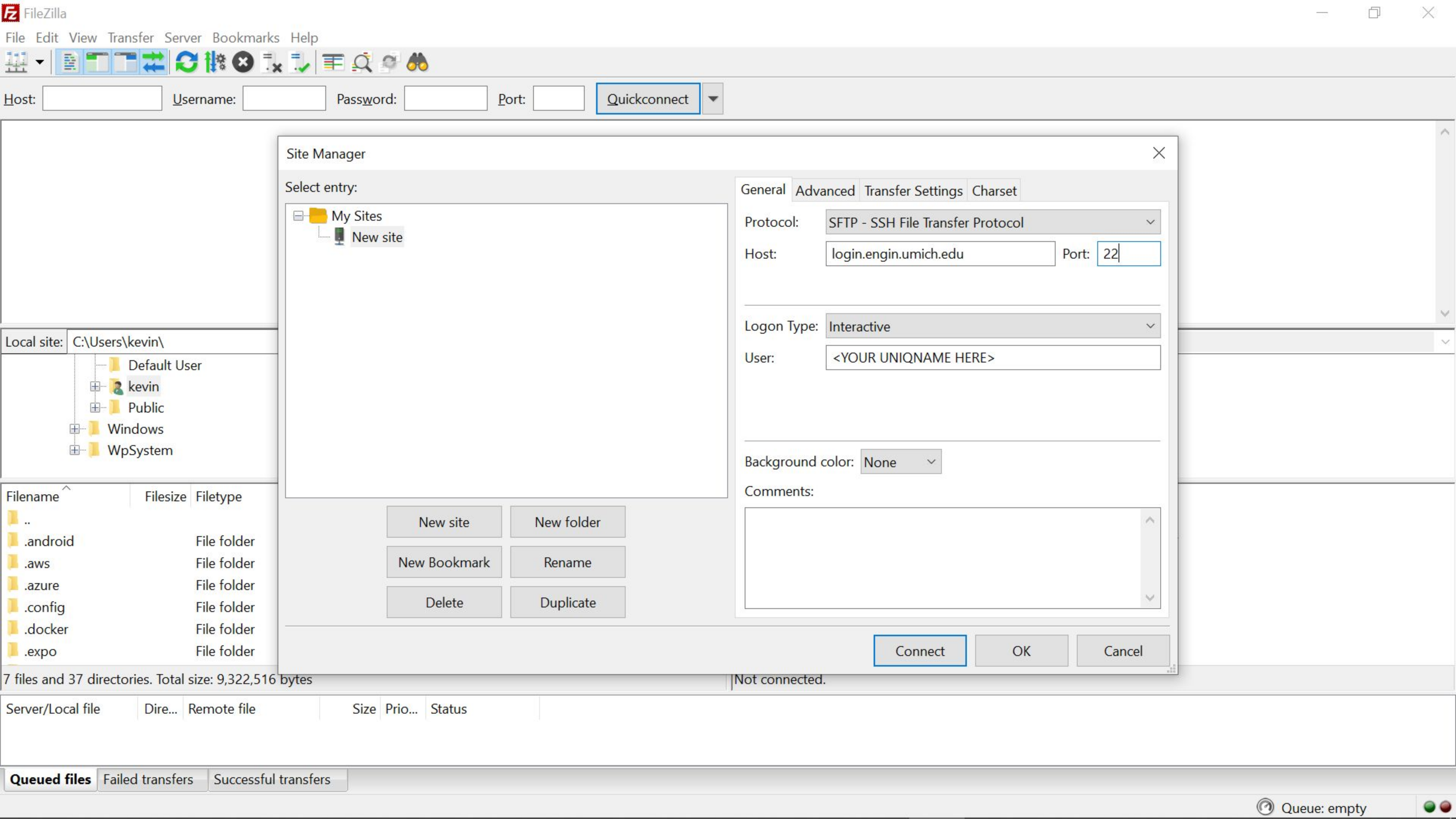
2. Open the Site Manager (File -> Site Manager) and click “New site”

3. Set “Protocol:” to **SFTP**

4. Set “Host:” to **login.engin.umich.edu** (CoE) or **login.itcs.umich.edu** (not CoE)

5. Set “Logon Type:” to **Interactive**

6. Set “User:” to **your uniqname** (Ex. yankevn)



Protocol: SFTP - SSH File Transfer Protocol

Host: login.engin.umich.edu

Port: 22

Logon Type: Interactive

User: <YOUR UNIQNAME HERE>

Background color: None

Comments:

Local site: C:\Users\kevin\

Filename	Filesize	Filetype
..		
.android		File folder
.aws		File folder
.azure		File folder
.config		File folder
.docker		File folder
.expo		File folder

7 files and 37 directories. Total size: 9,322,516 bytes

Not connected.

Queue: empty

FileZilla Quality of Life Change

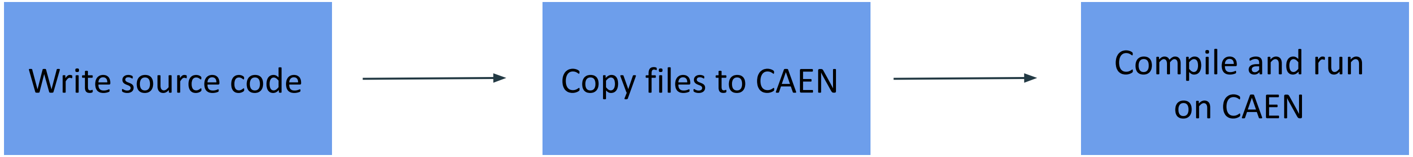
- You may notice FileZilla asks you to reauthenticate with each file transfer/operation
    a. This is because FileZilla uses a new connection for each transfer
- To fix, do the following:
    a. Open Site Manager (File -> Site Manager)
    b. Go to Transfer Settings
    c. Make sure the "Limit number of simultaneous connections" is checked and set the value to 1

Compiling

Getting started

- We write source code
- That is compiled into an executable (we use CAEN g++ compiler)
- The executable (.exe) is run

your projects MUST compile on CAEN's g++ compiler



Write source code

Copy files to CAEN

Compile and run on CAEN

Compiling and running code using CAEN

```
$ g++ -std=c++98 -Wall helloworld.cpp -o helloworld.exe
```

Be careful with naming!!

- g++ (the linux compiler)
- -std=c++98 (the version of c++ we use)
- -Wall (gives feedback for bugs)
- helloworld.cpp (your source code)
- helloworld.exe (the executable)

```
$ ./helloworld.exe
```

- Runs the executable

Super mini project

- Write a program that prints out “Hello World!”
- Compile this on CAEN to make sure you know how
- Transfer file using FileZilla
- Email file to me!
  - Hint: check out slides 10 and 11 in the Lecture 1 slides...

Identifier Types

Types

int: 5, 1, 7, 8, 10

double: 5.4, 4, 7.8, 11.4

bool: true/false

char: ‘a’, ‘4’, ‘g’, ‘9’

C++ Operators

Operators

Basic operators

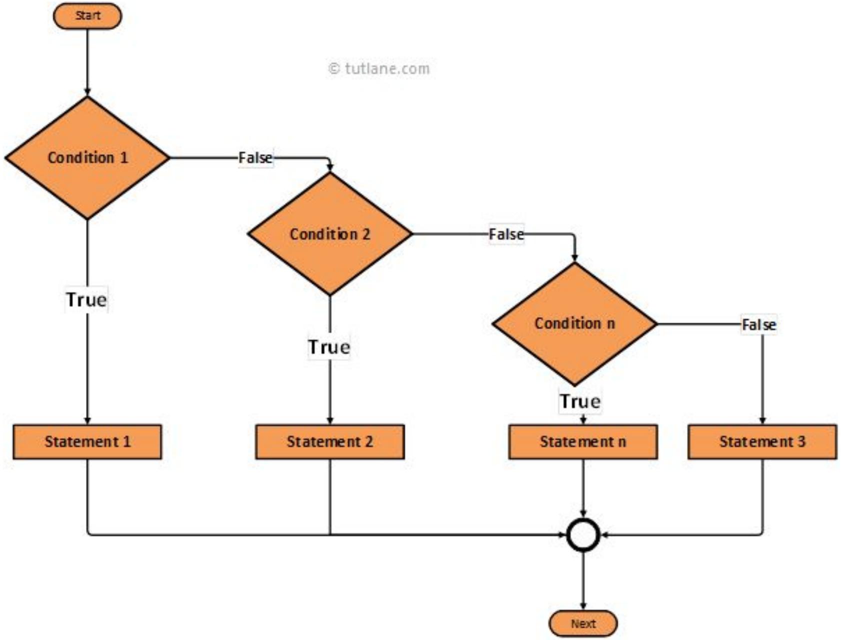
- +, -, *, /
- ==, !=, <, >, <=, >=
- =

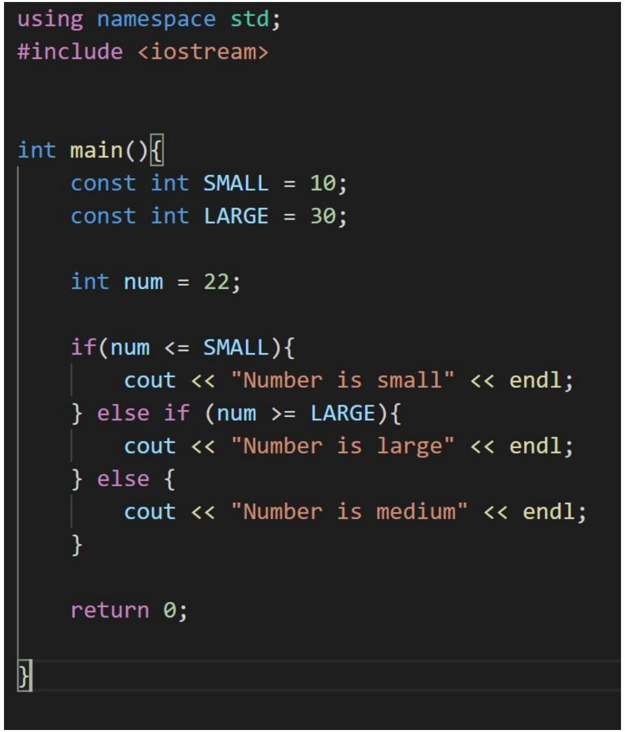
Special operators

- % - modulo
- ++, -- - increment and decrement
- !, ||, && - NOT, OR, and AND

If/Else if

If, else if, else

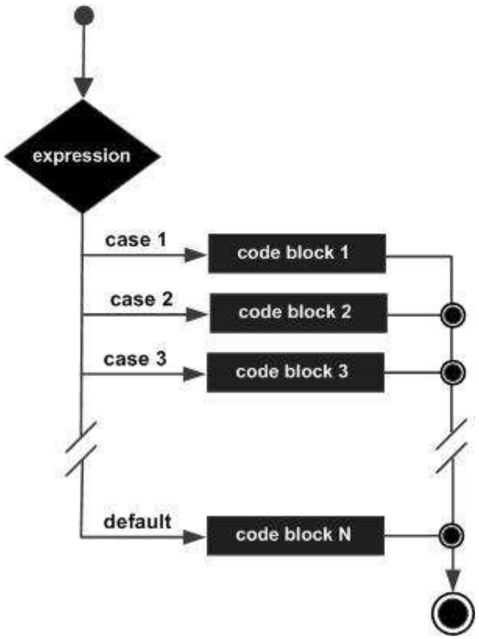


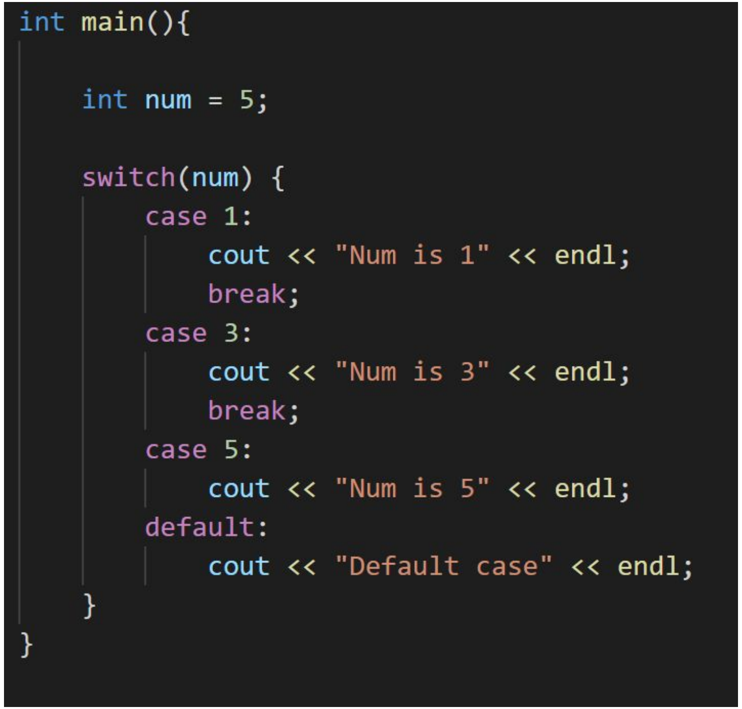


```
using namespace std;
#include <iostream>


int main(){
    const int SMALL = 10;
    const int LARGE = 30;

    int num = 22;

    if(num <= SMALL){
        cout << "Number is small" << endl;
    } else if (num >= LARGE){
        cout << "Number is large" << endl;
    } else {
        cout << "Number is medium" << endl;
    }

    return 0;

}
```

Switch Statements

Switch Statements



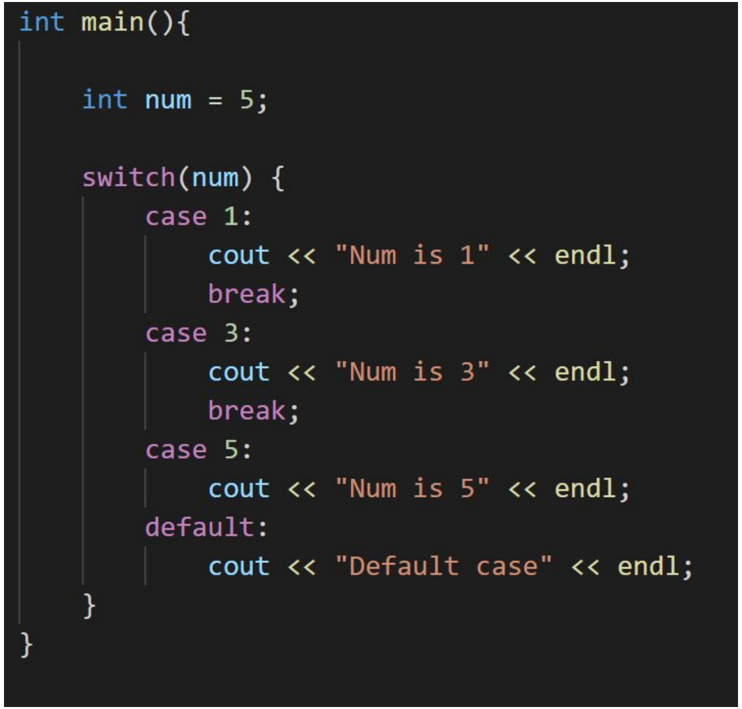


```
int main(){

    int num = 5;

    switch(num) {
        case 1:
            cout << "Num is 1" << endl;
            break;
        case 3:
            cout << "Num is 3" << endl;
            break;
        case 5:
            cout << "Num is 5" << endl;
        default:
            cout << "Default case" << endl;
    }
}
```

Switch Statements

Output: Num is 5

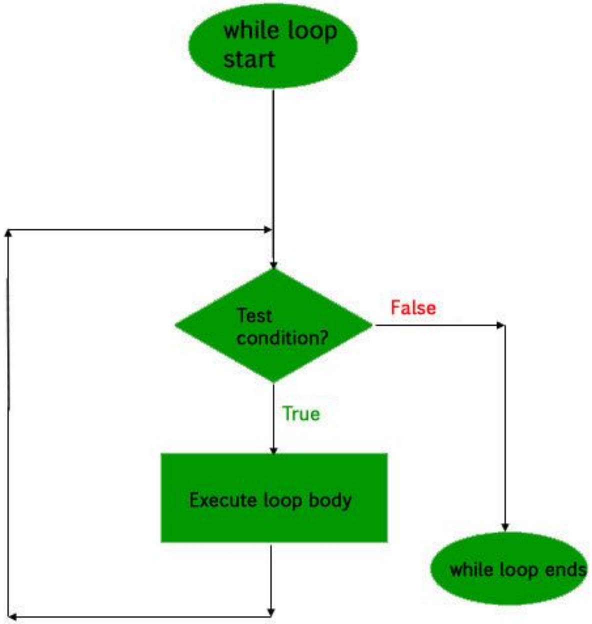
Default case

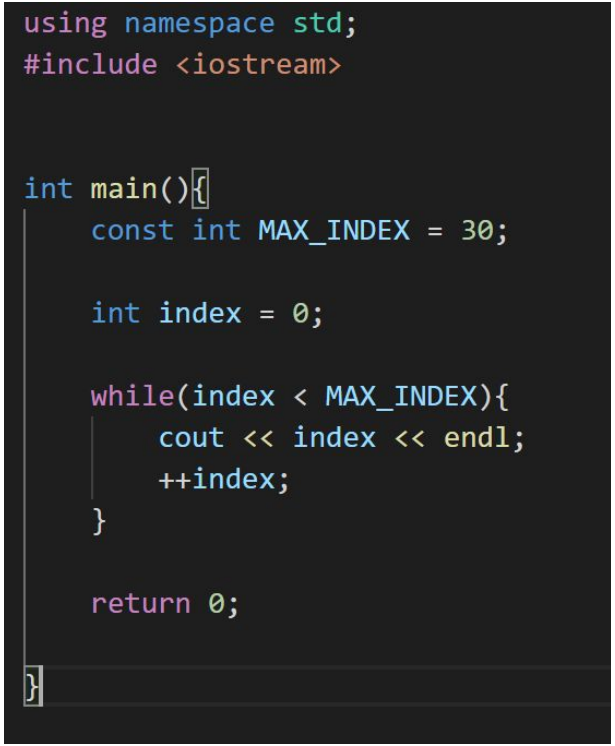


```
int main(){

    int num = 5;

    switch(num) {
        case 1:
            cout << "Num is 1" << endl;
            break;
        case 3:
            cout << "Num is 3" << endl;
            break;
        case 5:
            cout << "Num is 5" << endl;
        default:
            cout << "Default case" << endl;
    }
}
```

While loops

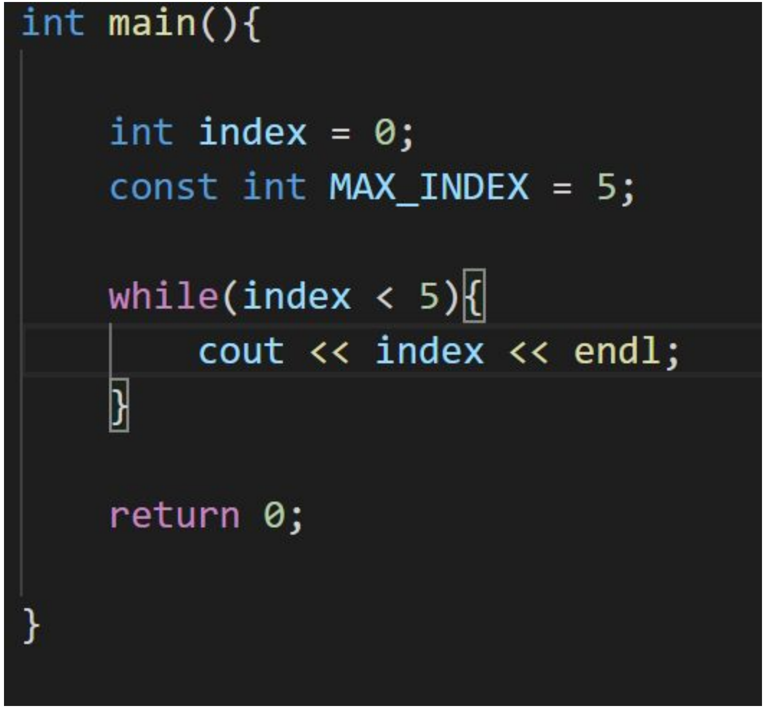
While Loops



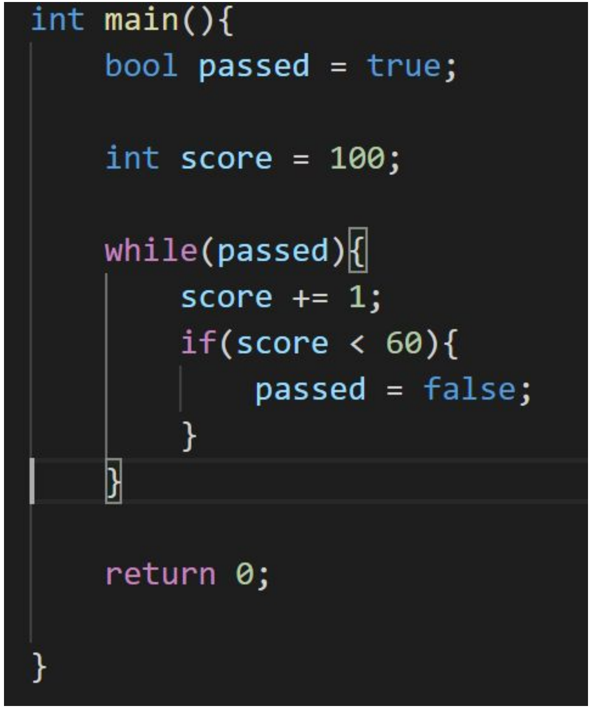


```cpp
using namespace std;
#include <iostream>


int main(){
    const int MAX_INDEX = 30;

    int index = 0;

    while(index < MAX_INDEX){
        cout << index << endl;
        ++index;
    }

    return 0;

}
```

Be Careful of Infinite Loops!!

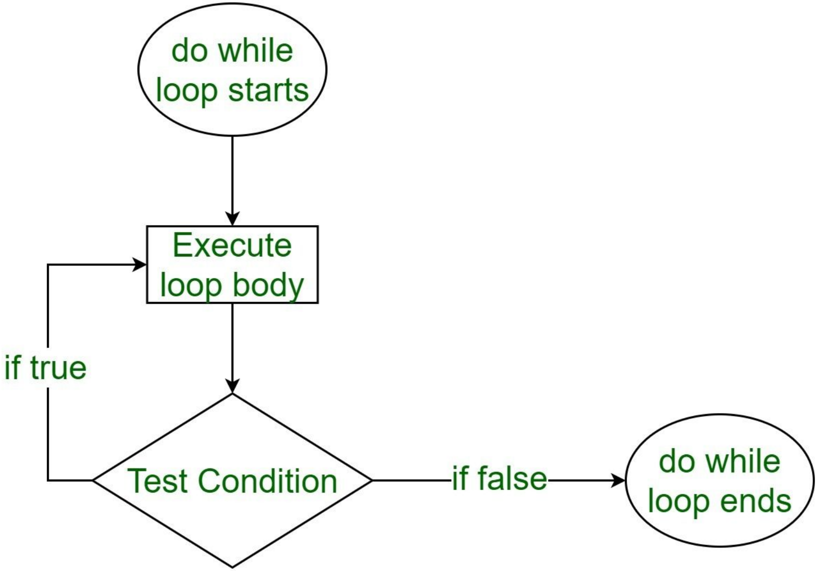


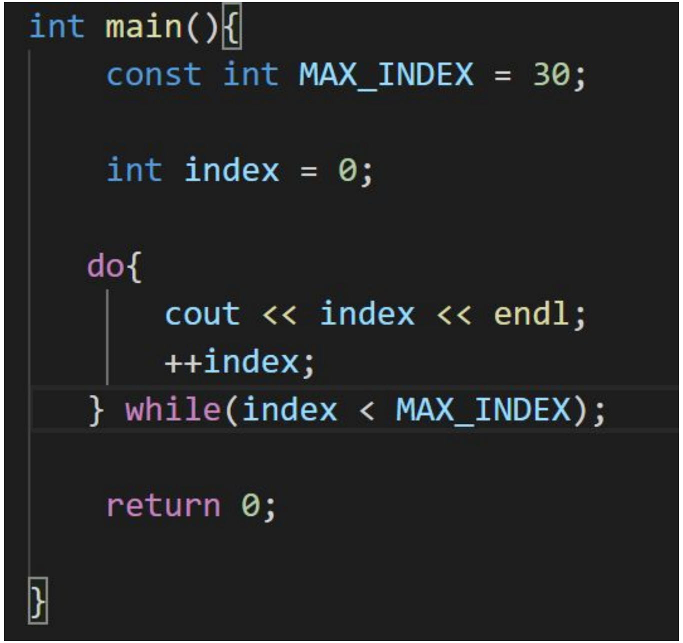
```cpp
int main(){

    int index = 0;
    const int MAX_INDEX = 5;

    while(index < 5){
        cout << index << endl;
    }

    return 0;

}
```



```cpp
int main(){
    bool passed = true;

    int score = 100;

    while(passed){
        score += 1;
        if(score < 60){
            passed = false;
        }
    }

    return 0;

}
```

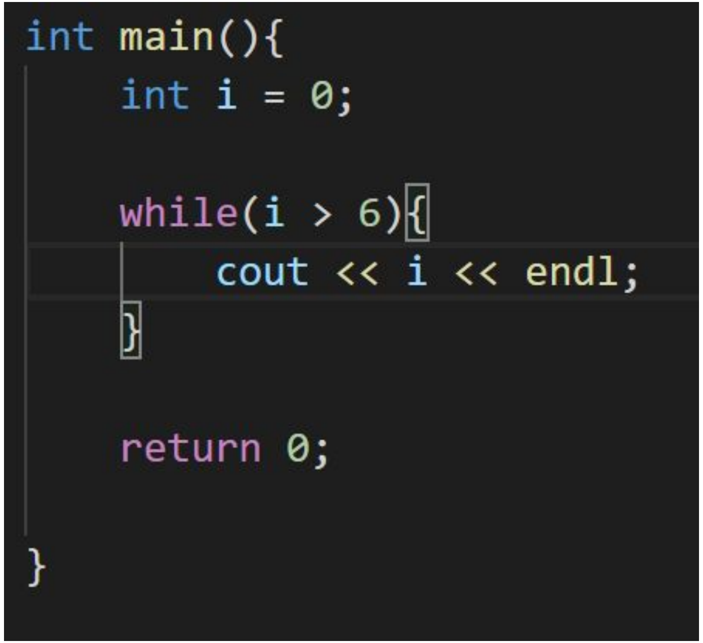
Do While Loop



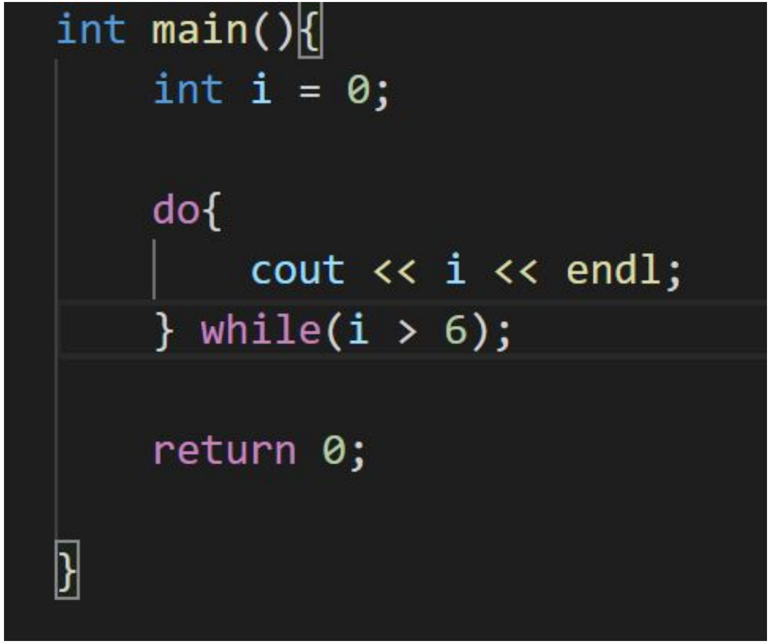


```
int main(){
    const int MAX_INDEX = 30;

    int index = 0;

   do{
       cout << index << endl;
       ++index;
   } while(index < MAX_INDEX);

    return 0;

}
```

So what's the difference?

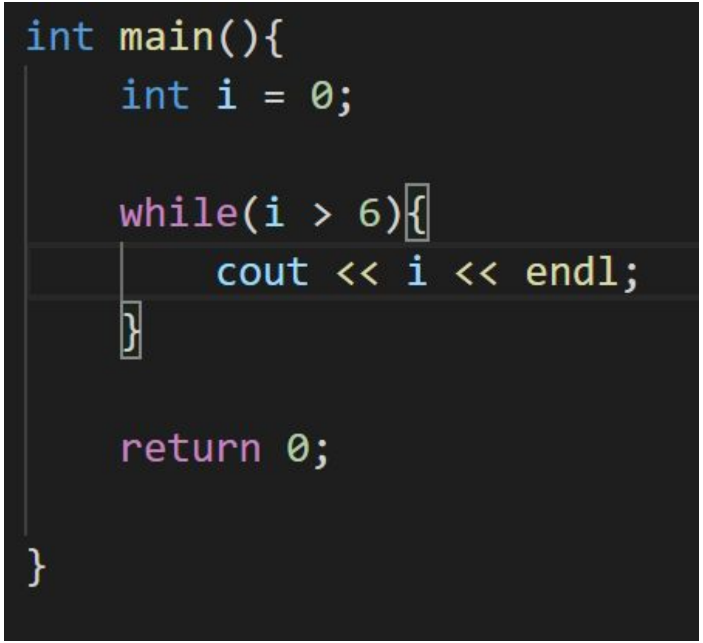


```
int main(){
    int i = 0;

    while(i > 6){
        cout << i << endl;
    }

    return 0;

}
```



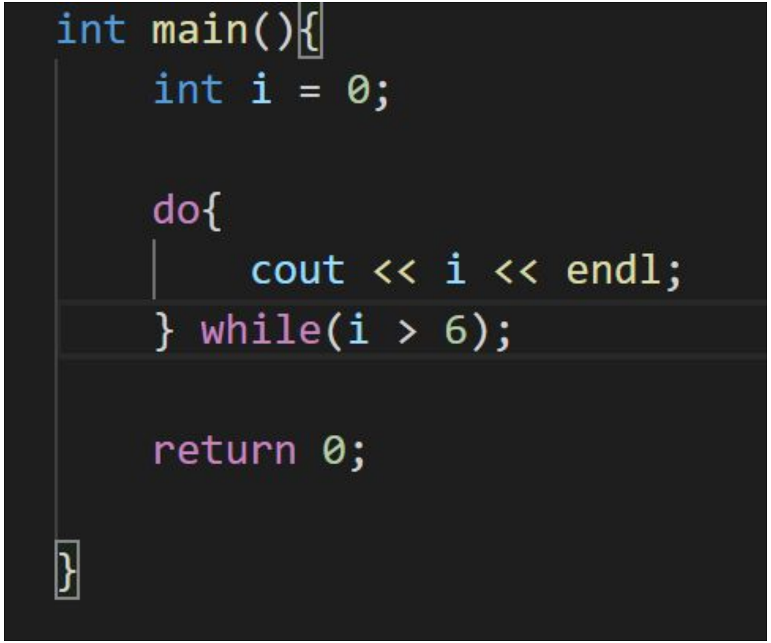
```
int main(){
    int i = 0;

    do{
        cout << i << endl;
    } while(i > 6);

    return 0;

}
```

So what’s the difference?



```
int main(){
    int i = 0;

    while(i > 6){
        cout << i << endl;
    }

    return 0;

}
```

Output: nothing



```
int main(){
    int i = 0;

    do{
        cout << i << endl;
    } while(i > 6);

    return 0;

}
```

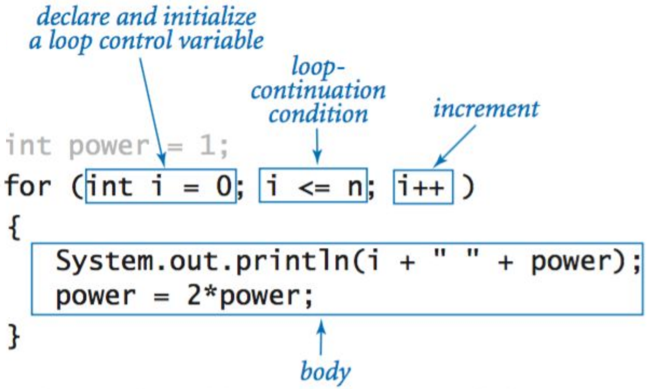
Output: 0

So what’s the difference?

- While loops- doesn’t always run once
- Do… while loops- always runs at least once

For loops

For Loops



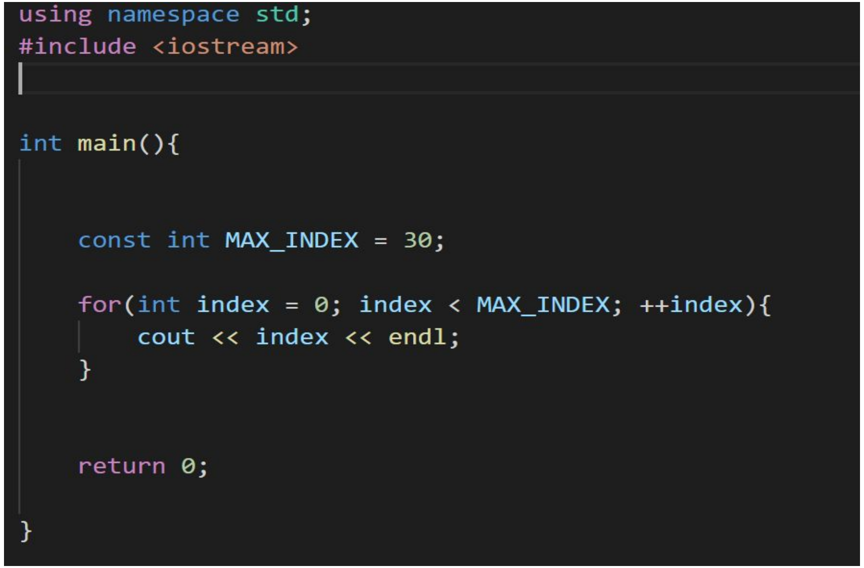
declare and initialize a loop control variable

loop-continuation condition

increment

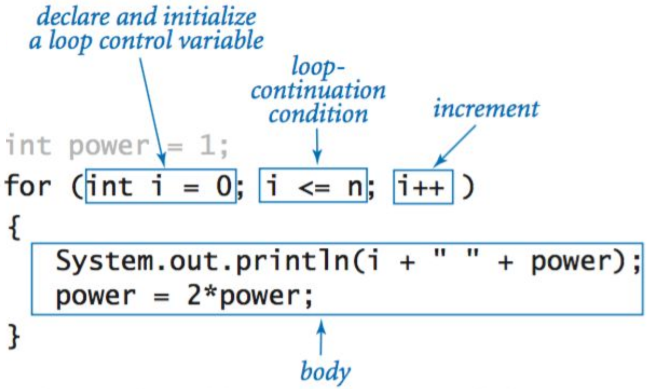
```
int power = 1;
for (int i = 0; i <= n; i++ )
{
    System.out.println(i + " " + power);
    power = 2*power;
}
```

body



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


int main(){


    const int MAX_INDEX = 30;

    for(int index = 0; index < MAX_INDEX; ++index){
        cout << index << endl;
    }


    return 0;

}
```

You don’t have to declare and initialize in the loop



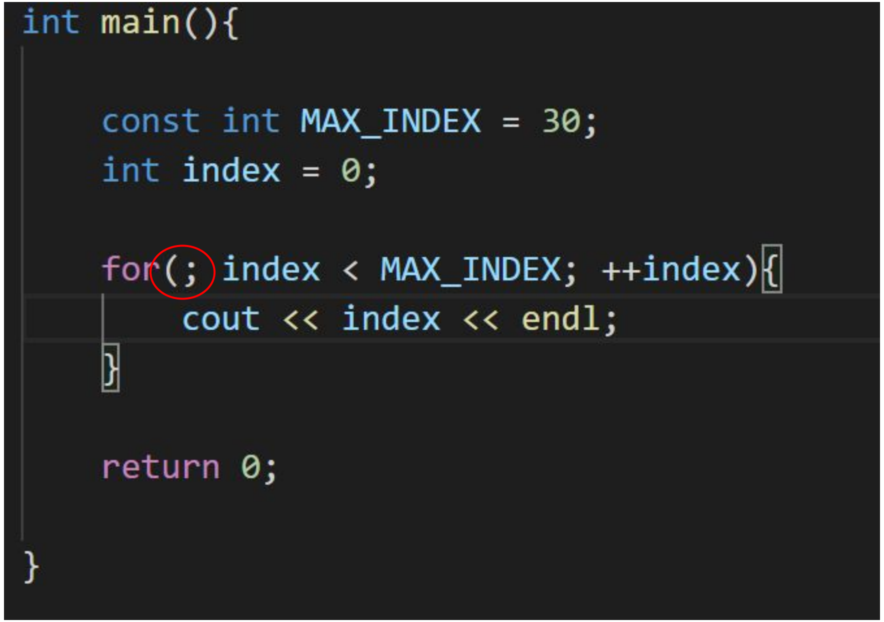
declare and initialize a loop control variable

loop-continuation condition

increment

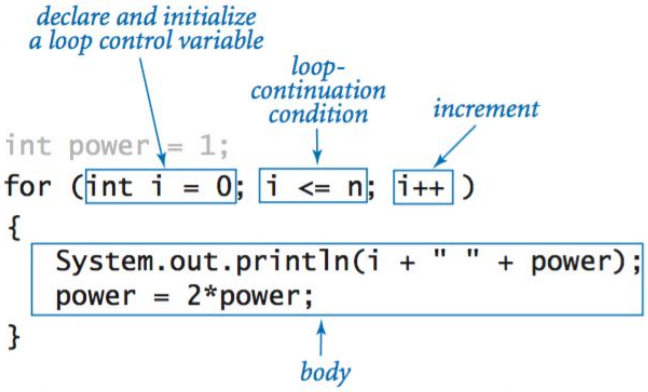
```
int power = 1;
for (int i = 0; i <= n; i++ )
{
    System.out.println(i + " " + power);
    power = 2*power;
}
```

body



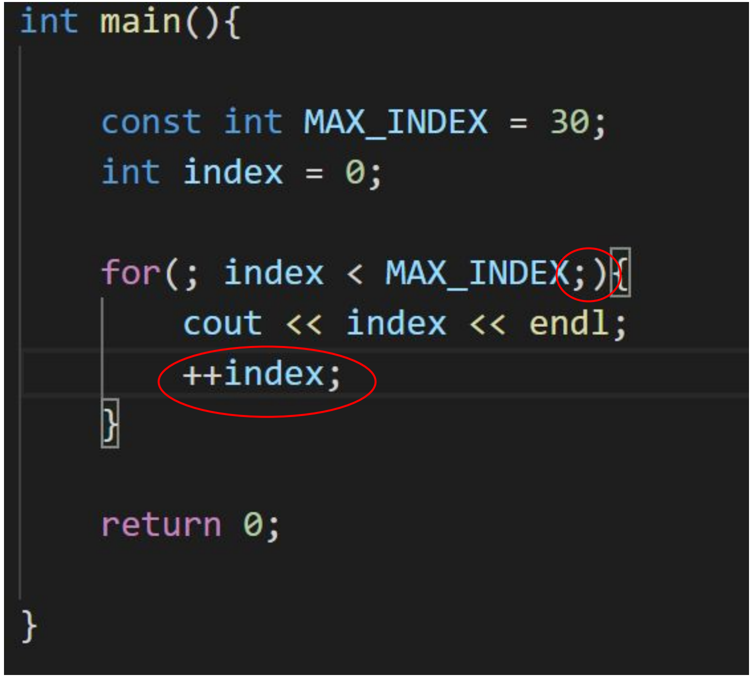
```
int main(){

    const int MAX_INDEX = 30;
    int index = 0;

    for(; index < MAX_INDEX; ++index){
        cout << index << endl;
    }

    return 0;

}
```

You don’t have to increment in the loop either



```
int power = 1;
for (int i = 0; i <= n; i++ )
{
    System.out.println(i + " " + power);
    power = 2*power;
}
```

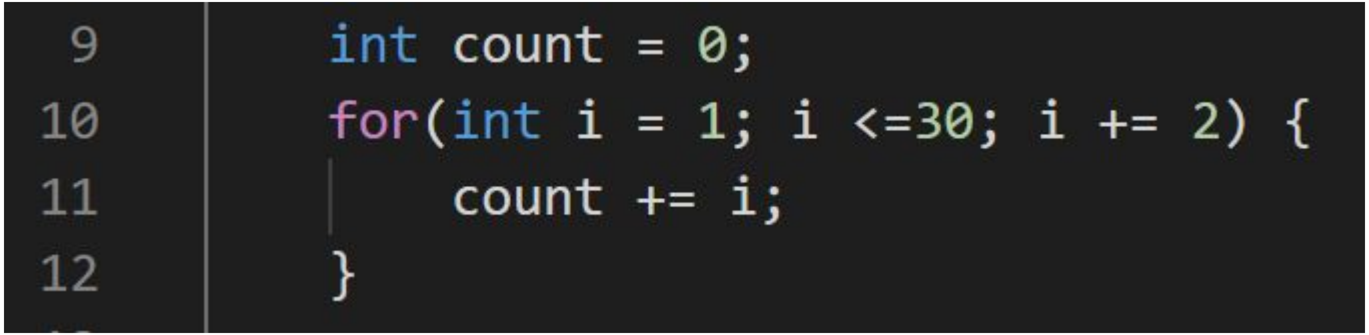
declare and initialize a loop control variable
loop-continuation condition
increment
body



```
int main(){

    const int MAX_INDEX = 30;
    int index = 0;

    for(; index < MAX_INDEX;){
        cout << index << endl;
        ++index;
    }

    return 0;

}
```

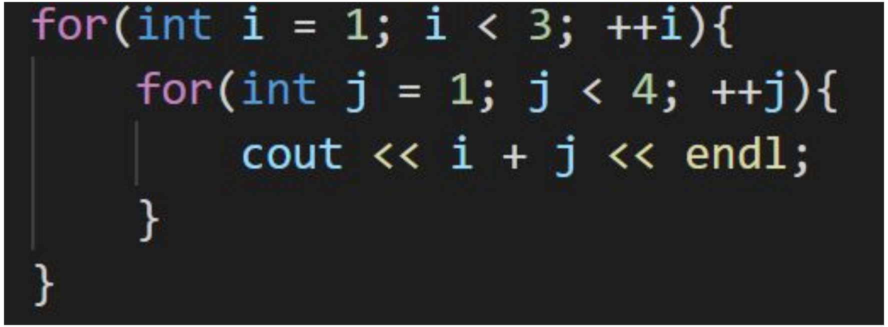
Example: Add all the odd numbers 1 to 30 using a for loop

Example: Add all the odd numbers 1 to 30 using a for loop

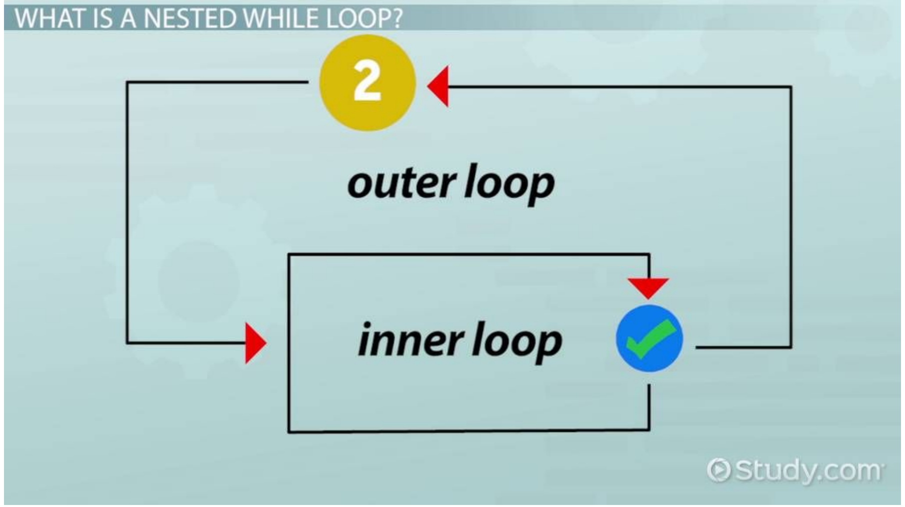


```
int count = 0;
for(int i = 1; i <=30; i += 2) {
    count += i;
}
```

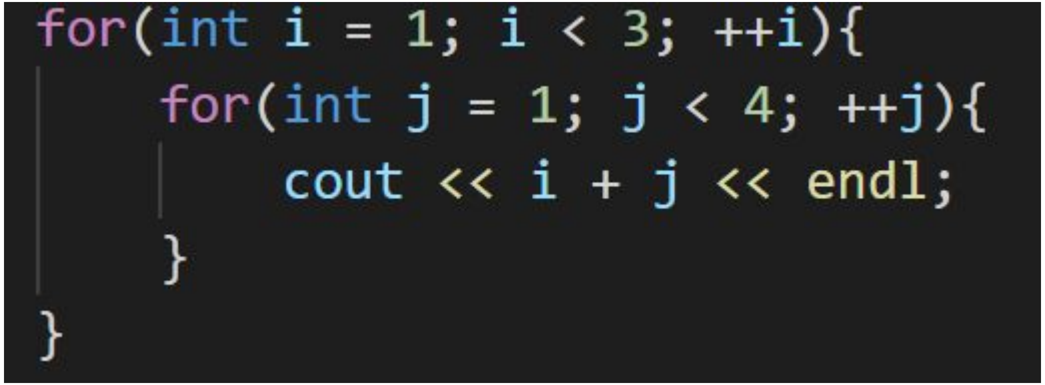
Nested For Loops



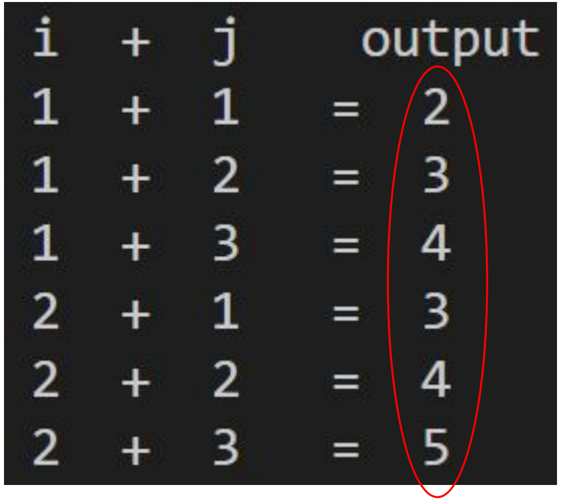
```
for(int i = 1; i < 3; ++i){
    for(int j = 1; j < 4; ++j){
        cout << i + j << endl;
    }
}
```



Nested For Loops



```
for(int i = 1; i < 3; ++i){
    for(int j = 1; j < 4; ++j){
        cout << i + j << endl;
    }
}
```

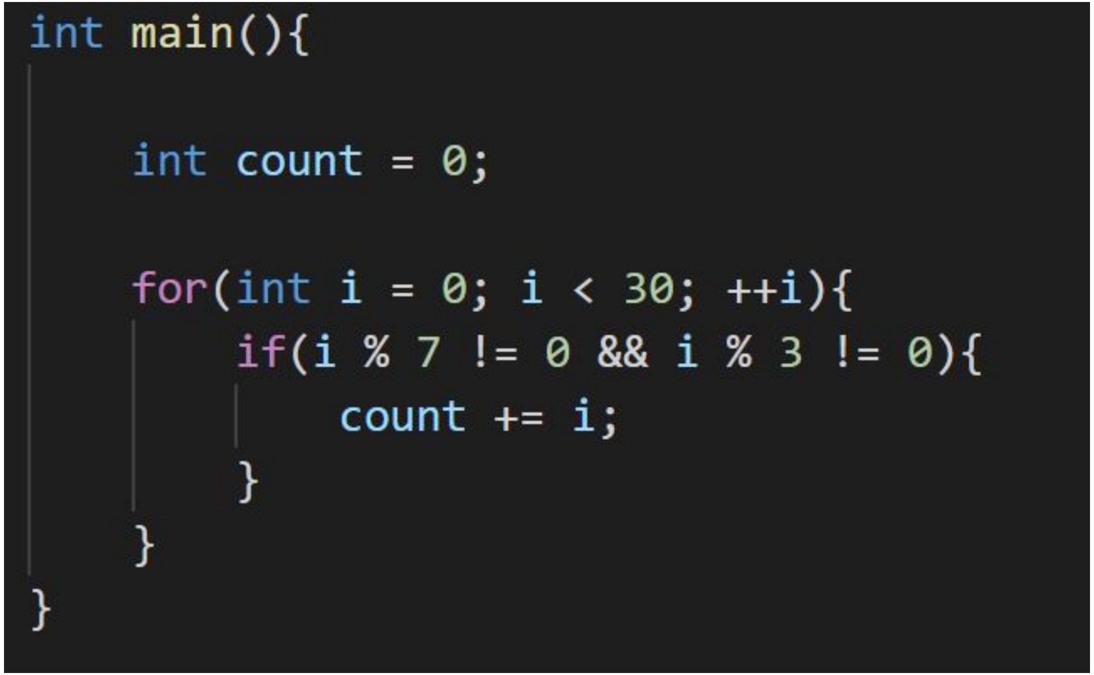


```
i  +  j     output
1  +  1  =  2
1  +  2  =  3
1  +  3  =  4
2  +  1  =  3
2  +  2  =  4
2  +  3  =  5
```

Example: Putting it together

Sum all numbers 0 to 30 that aren’t divisible by 3 or 7

Solution:



```
int main(){

    int count = 0;

    for(int i = 0; i < 30; ++i){
        if(i % 7 != 0 && i % 3 != 0){
            count += i;
        }
    }
}
```

“For” or “While”?

- Calculating the factorial of a number
- Read in user input until a valid input is given
- Add consecutive numbers starting at N until you hit a perfect square
- Add multiples of 10 from 0 to 100

“For” or “While”?

- Calculating the factorial of a number for
- Read in user input until a valid input is given
- Add consecutive numbers starting at N until you hit a perfect square
- Add multiples of 10 from 0 to 100

“For” or “While”?

- Calculating the factorial of a number	for
- Read in user input until a valid input is given	while
- Add consecutive numbers starting at N until you hit a perfect square	
- Add multiples of 10 from 0 to 100	

“For” or “While”?

- Calculating the factorial of a number	for
- Read in user input until a valid input is given	while
- Add consecutive numbers starting at N until you hit a perfect square	while
- Add multiples of 10 from 0 to 100	

“For” or “While”?

- Calculating the factorial of a number	for
- Read in user input until a valid input is given	while
- Add consecutive numbers starting at N until you hit a perfect square	while
- Add multiples of 10 from 0 to 100	for

Common Errors with Loops

- Incrementing one farther than you wanted (off-by-one error)
    - Often caused by using ‘<’ or ‘<=’ interchangeably or logic errors
- Forgetting to update the counter (in while loops) or double updating (in for loops)
- Infinite loops
    - Make sure your condition will fail at some point

Num++ vs ++num

Quick note: ++num and num++ are different

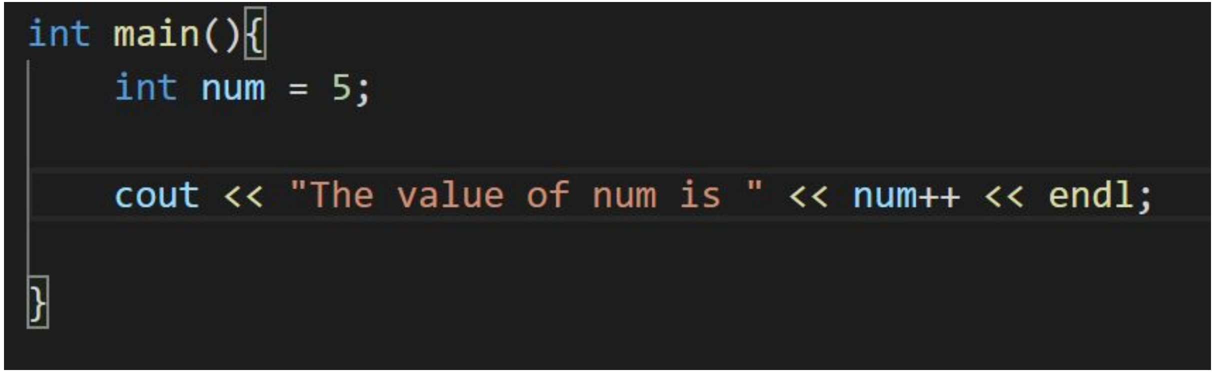
num++ happens after num does its job

Call the post-increment operators

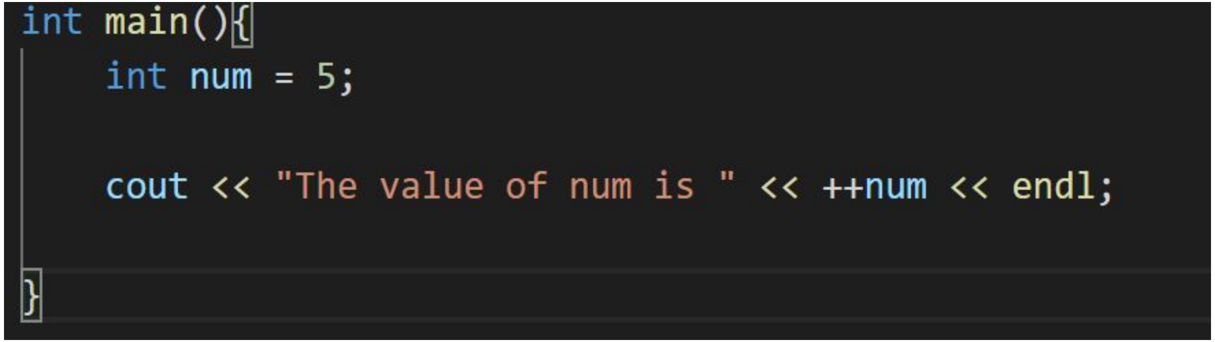
++num happens before num does its job

Called the pre-increment operator

Quick note: ++num and num++ are different

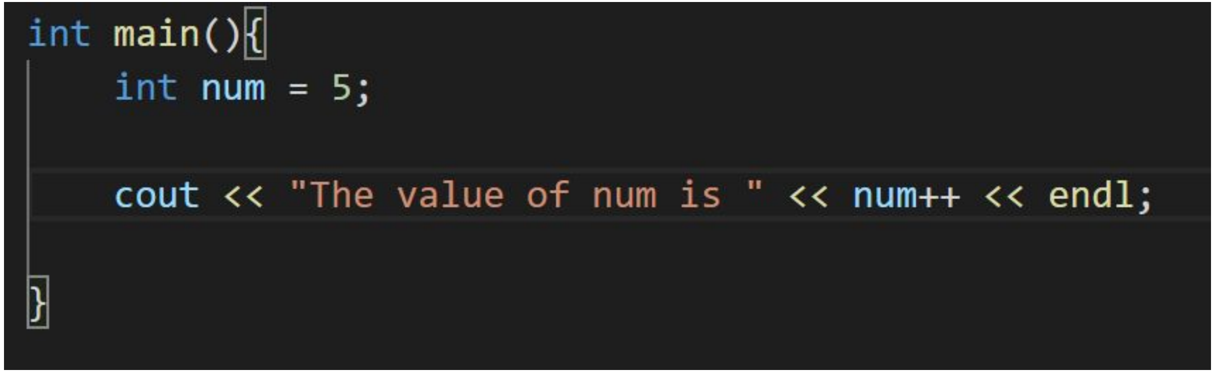


```
int main(){
    int num = 5;

    cout << "The value of num is " << num++ << endl;

}
```



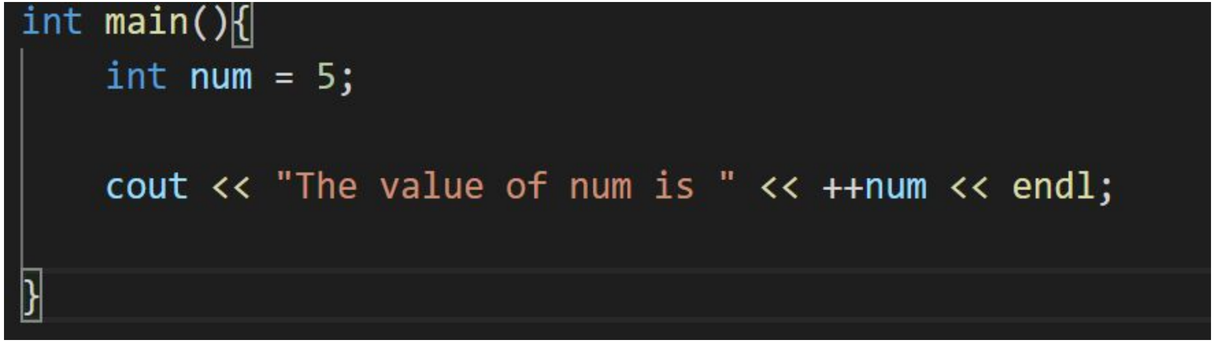
```
int main(){
    int num = 5;

    cout << "The value of num is " << ++num << endl;

}
```

Quick note: ++num and num++ are different



```
int main(){
    int num = 5;

    cout << "The value of num is " << num++ << endl;

}
```

Output: The value of num is 5



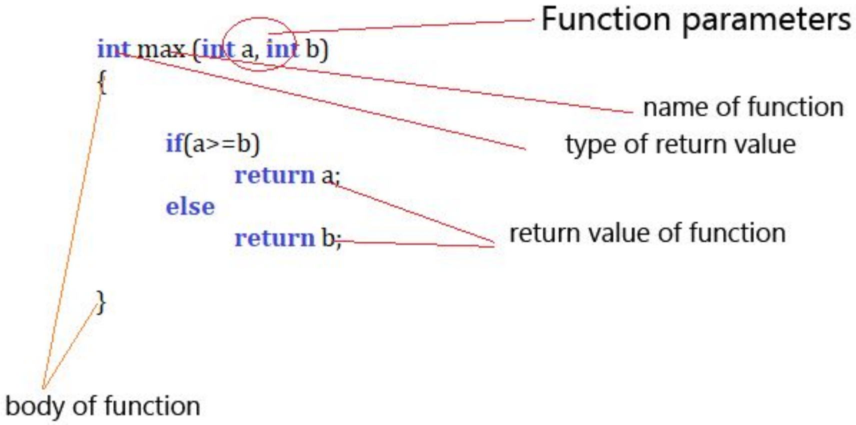
```
int main(){
    int num = 5;

    cout << "The value of num is " << ++num << endl;

}
```

Output: The value of num is 6

Functions

Functions

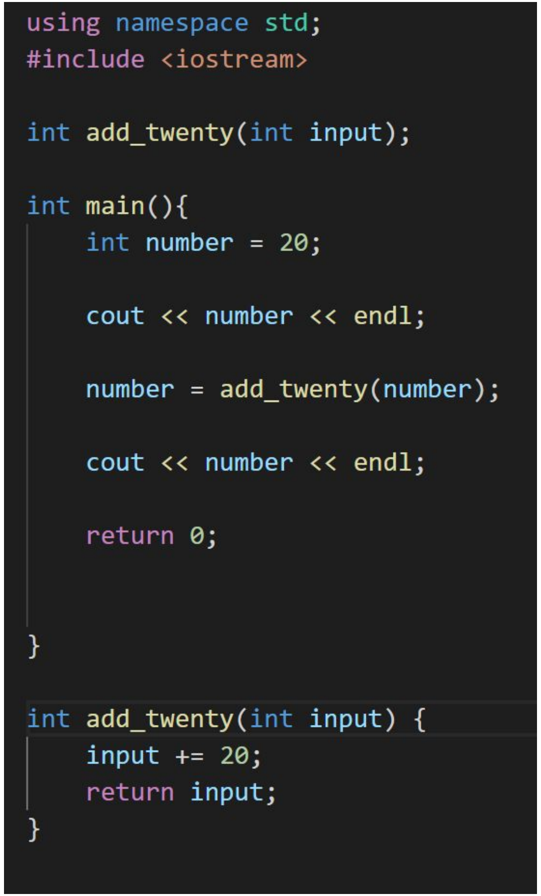
- Functions must be declared before use
- Can take any number of inputs, but can only return up to 1 value



```
int max (int a, int b)
{

        if(a>=b)
                return a;
        else
                return b;

}
```

Functions

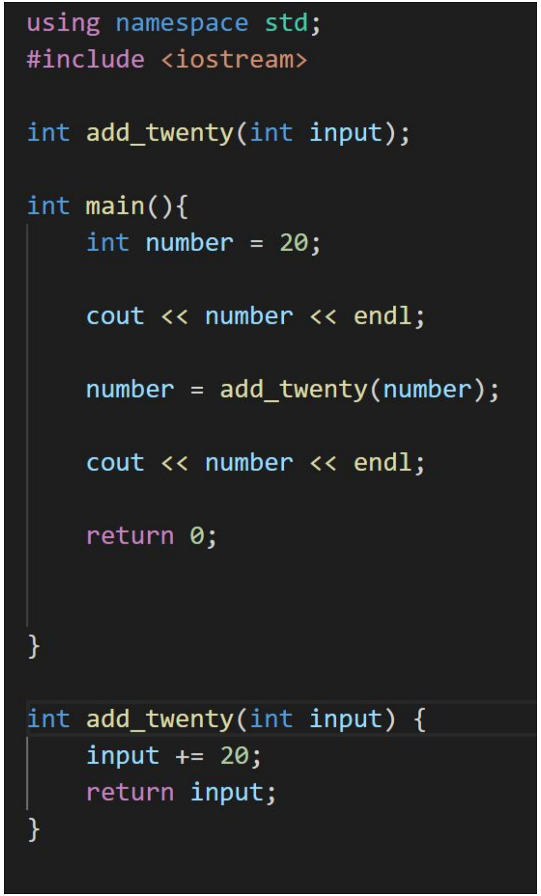
- Functions must be declared before use
- Can take any number of inputs, but can only return up to 1 value



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

int add_twenty(int input);

int main(){
    int number = 20;

    cout << number << endl;

    number = add_twenty(number);

    cout << number << endl;

    return 0;


}

int add_twenty(int input) {
    input += 20;
    return input;
}
```

A note on scope

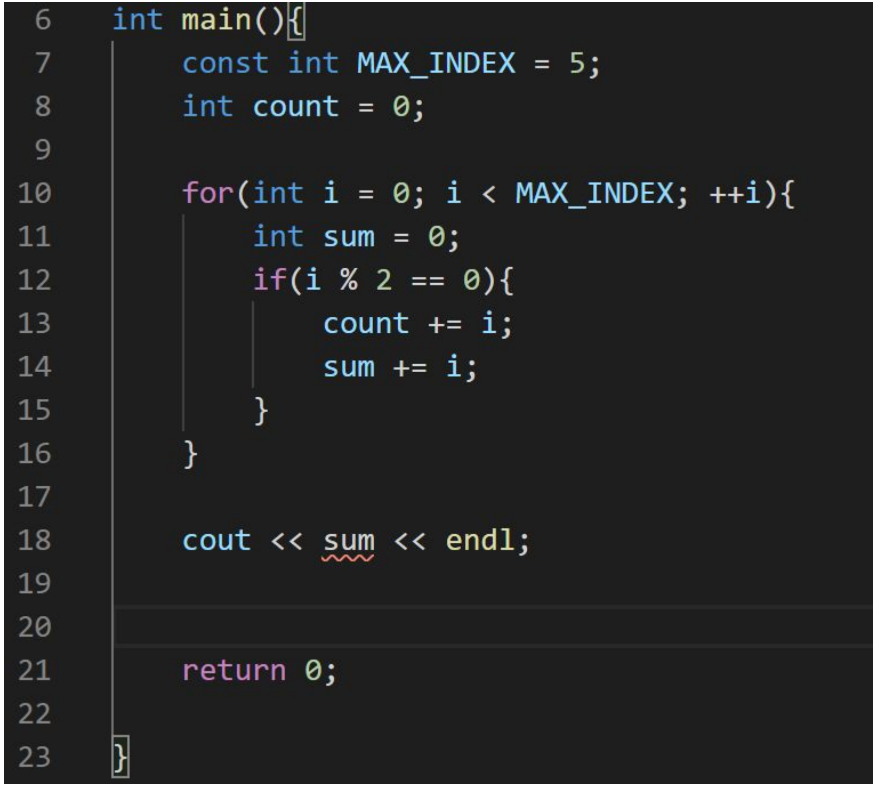
- The variable “input” does not exist in main and therefore cannot be used
- It’s scope is in the “add_twenty” function



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

int add_twenty(int input);

int main(){
    int number = 20;

    cout << number << endl;

    number = add_twenty(number);

    cout << number << endl;

    return 0;


}

int add_twenty(int input) {
    input += 20;
    return input;
}
```

A note on scope

- sum is declared inside the for loop
- sum cannot be used outside of its scope



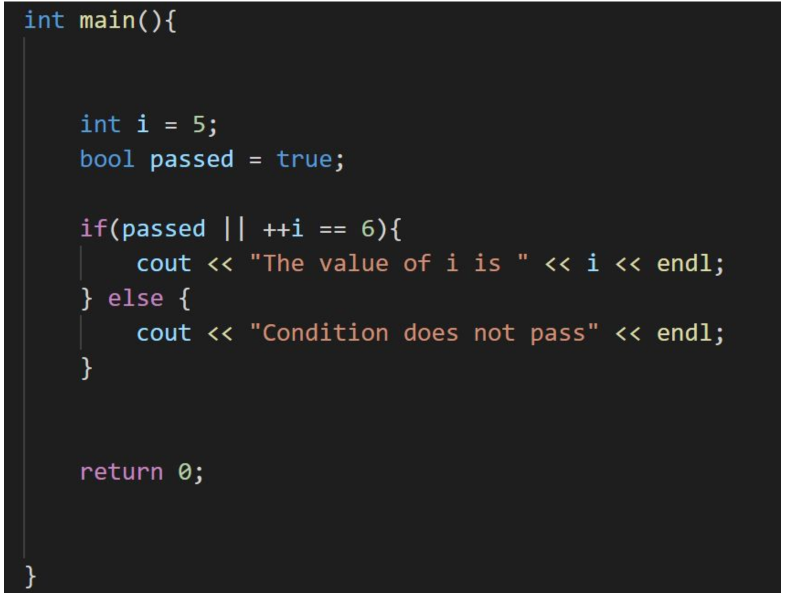
```
int main(){
    const int MAX_INDEX = 5;
    int count = 0;

    for(int i = 0; i < MAX_INDEX; ++i){
        int sum = 0;
        if(i % 2 == 0){
            count += i;
            sum += i;
        }
    }

    cout << sum << endl;


    return 0;

}
```

Function Practice!

Write a function that returns two integers added together!

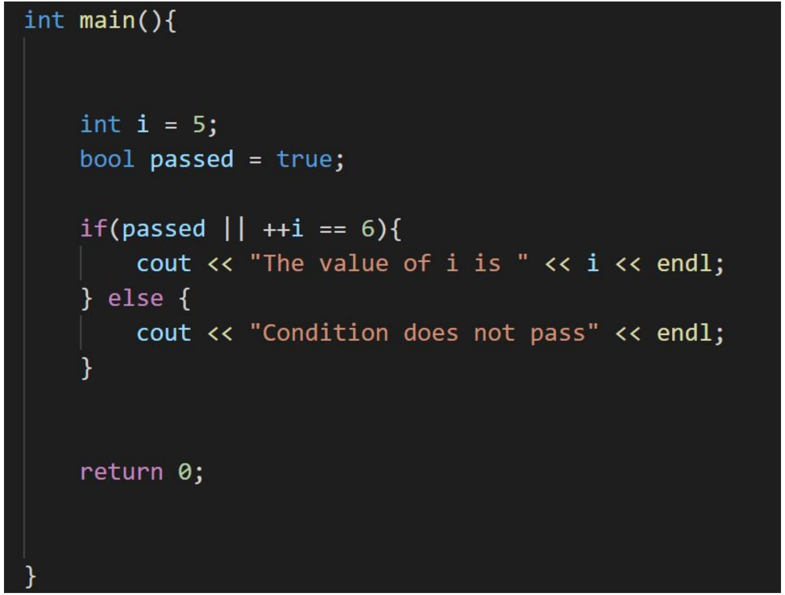
Extra Practice

Practice Problem: What would this output?



```
int main(){


    int i = 5;
    bool passed = true;

    if(passed || ++i == 6){
        cout << "The value of i is " << i << endl;
    } else {
        cout << "Condition does not pass" << endl;
    }


    return 0;


}
```

Practice Problem: What would this output?



```
int main(){


    int i = 5;
    bool passed = true;

    if(passed || ++i == 6){
        cout << "The value of i is " << i << endl;
    } else {
        cout << "Condition does not pass" << endl;
    }


    return 0;


}
```

Answer: The value of i is 5