CSci 127: Introduction to Computer Science

hunter.cuny.edu/csci

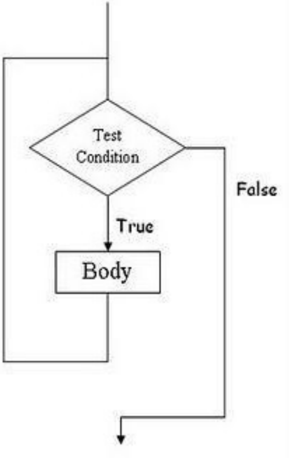
Today’s Topics

- Loops
- Examples

LOOPS

Loop

- **Loop**: a coding construct that allows for some code to be run repeatedly.
- A loop, like a branch:
  - Also has a condition (**loop condition**).
  - Can execute a chunk of code (the **loop body**) if the condition is true.



What does this look like in C++?

The `for` Loop

Useful for counter-controlled loop

General Format:

```
for(initialization; test; update)
     statement; // or block in { }
```

No semicolon after the `update` expression or after the `)`

`for` Loop - Mechanics

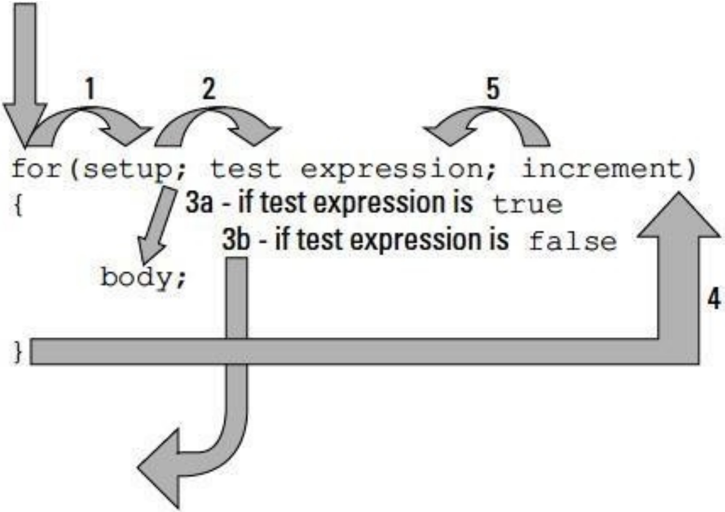
```
for(initialization; test; update)
          statement; // or block in { }
```

1) Perform *`initialization`*
2) Evaluate *`test`* expression

   If `true`, execute *`statement`*

   If `false`, terminate loop execution
3) Execute *`update`*, then re-evaluate *`test`* expression

`for` loop parts!



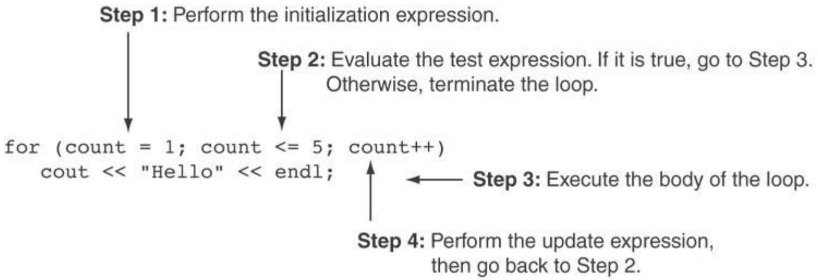
```
for(setup; test expression; increment)
{
    body;

}
```

1
2
5
3a - if test expression is true
3b - if test expression is false
4

`for` Loop - Example

```
int count;

for (count = 1; count <= 5; count++)
   cout << "Hello" << endl;
```

A Closer Look at the Previous Example



Step 1: Perform the initialization expression.

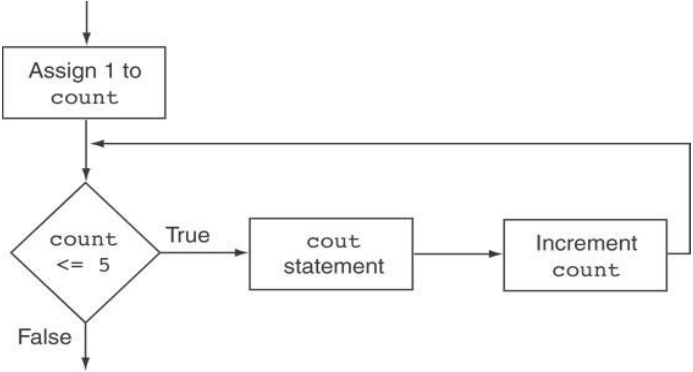
Step 2: Evaluate the test expression. If it is true, go to Step 3. Otherwise, terminate the loop.

```
for (count = 1; count <= 5; count++)
   cout << "Hello" << endl;
```

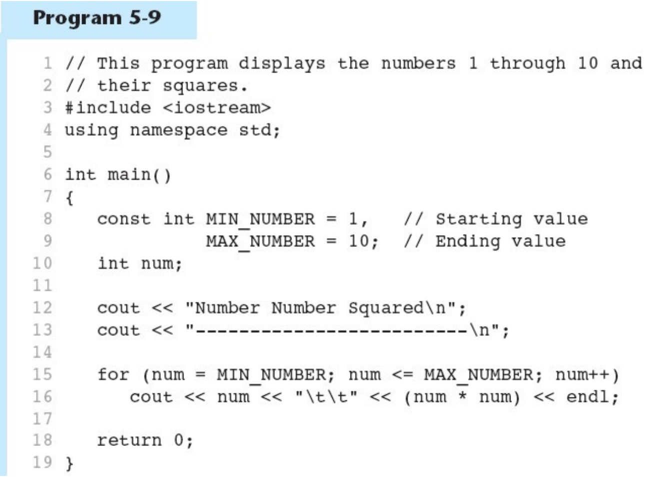
Step 3: Execute the body of the loop.

Step 4: Perform the update expression, then go back to Step 2.

Flowchart for the Previous Example



A `for` Loop example

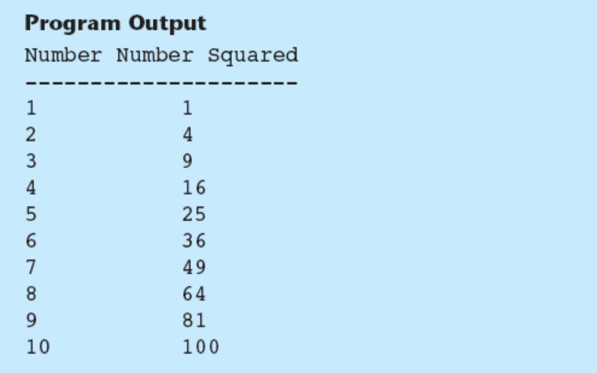


Program 5-9

```
// This program displays the numbers 1 through 10 and
// their squares.
#include <iostream>
using namespace std;

int main()
{
   const int MIN_NUMBER = 1,   // Starting value
             MAX_NUMBER = 10;  // Ending value
   int num;

   cout << "Number Number Squared\n";
   cout << "-------------------------\n";

   for (num = MIN_NUMBER; num <= MAX_NUMBER; num++)
      cout << num << "\t\t" << (num * num) << endl;

   return 0;
}
```

Continued…

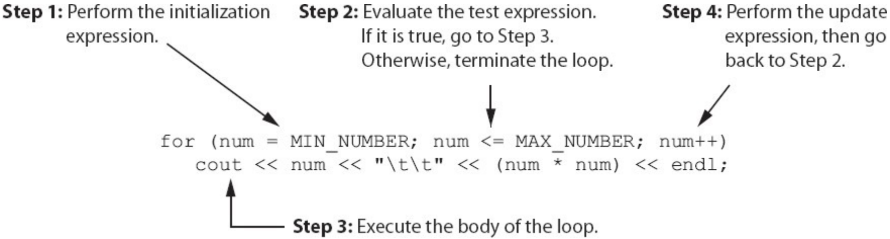
A `for` Loop example



Program Output

```
Number Number Squared
----------------------
1             1
2             4
3             9
4             16
5             25
6             36
7             49
8             64
9             81
10            100
```

A Closer Look at Lines 15 through 16 example



Step 1: Perform the initialization expression.

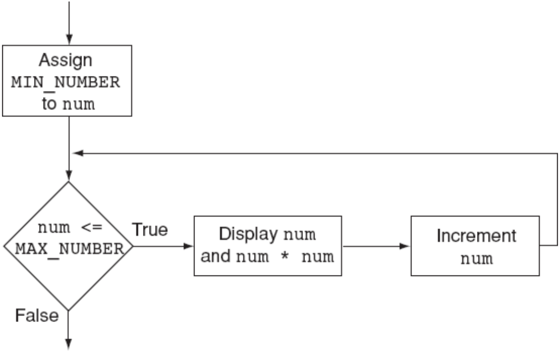
Step 2: Evaluate the test expression. If it is true, go to Step 3. Otherwise, terminate the loop.

Step 4: Perform the update expression, then go back to Step 2.

```
for (num = MIN_NUMBER; num <= MAX_NUMBER; num++)
   cout << num << "\t\t" << (num * num) << endl;
```

Step 3: Execute the body of the loop.

Flowchart for Lines 15 through 16 in example



When to Use the `for` Loop

In any situation that clearly requires

1. an initialization
2. a false condition to stop the loop
3. an update to occur at the end of each iteration

The `for` Loop is a Pretest Loop

The for loop tests its test expression before each iteration, so it is a pretest loop.
The following loop will never iterate:

```
for (count = 11; count <= 10; count++)
   cout << "Hello" << endl;
```

The `while` Loop

Loop: a control structure that causes a statement or statements to repeat

General format of the `while` loop:

```
while (expression)
   statement;
```

statement; can also be a block of statements enclosed in `{ }`

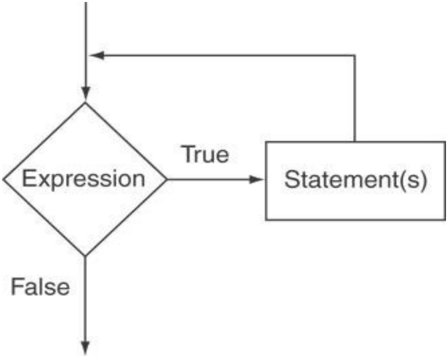
The `while` Loop – How It Works

```
while (expression)
     statement;
```

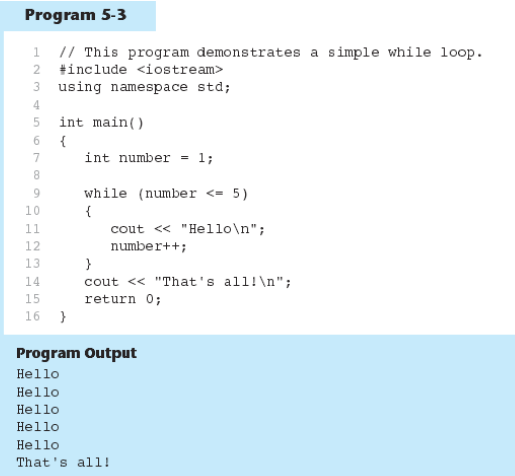
`expression` is evaluated

- if `true`, then *`statement`* is executed, and *`expression`* is evaluated again
- if `false`, then the loop is finished and program statements following *`statement`* execute

The Logic of a `while` Loop



The `while` loop in Program 5-3



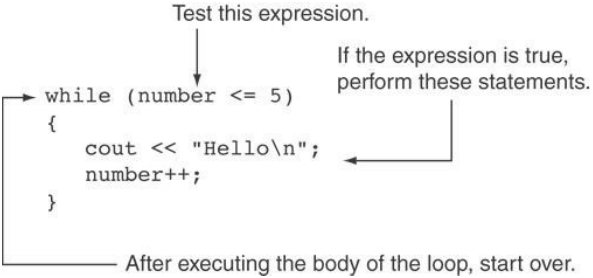
Program 5-3

```
// This program demonstrates a simple while loop.
#include <iostream>
using namespace std;

int main()
{
   int number = 1;

   while (number <= 5)
   {
      cout << "Hello\n";
      number++;
   }
   cout << "That's all!\n";
   return 0;
}
```

Program Output

```
Hello
Hello
Hello
Hello
Hello
That's all!
```

How the `while` Loop in example Lines 9 through 13 Works



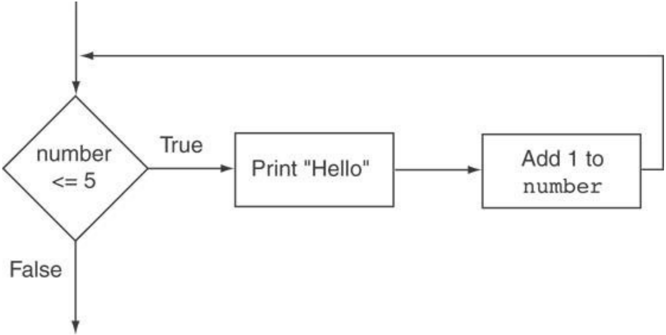
Test this expression.

If the expression is true, perform these statements.

```
while (number <= 5)
{
   cout << "Hello\n";
   number++;
}
```

After executing the body of the loop, start over.

Flowchart of the `while` Loop in Program 5-3



The `while` Loop is a Pretest Loop

`expression` is evaluated *before* the loop executes. The following loop will never execute:

```
int number = 6;
while (number <= 5)
{
    cout << "Hello\n";
    number++;
}
```

Watch Out for Infinite Loops

The loop must contain code to make `expression` become `false`

Otherwise, the loop will have no way of stopping

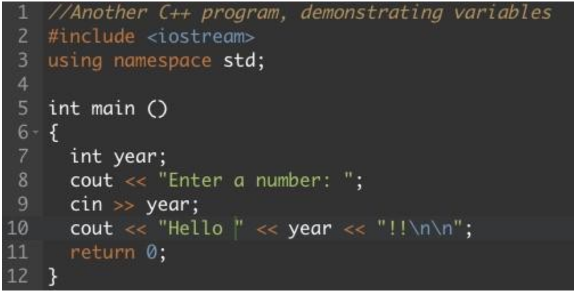
Such a loop is called an *infinite loop*, because it will repeat an infinite number of times

Example of an Infinite Loop

```
int number = 1;
while (number <= 5)
{
   cout << "Hello\n";
}
```

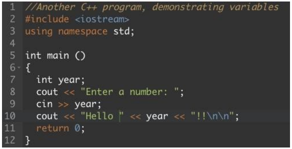
In Pairs or Triples:

- *Predict what the C++ code will do:*



```
//Another C++ program, demonstrating variables
#include <iostream>
using namespace std;

int main ()
{
  int year;
  cout << "Enter a number: ";
  cin >> year;
  cout << "Hello " << year << "!!\n\n";
  return 0;
}
```

onlinegdb demo

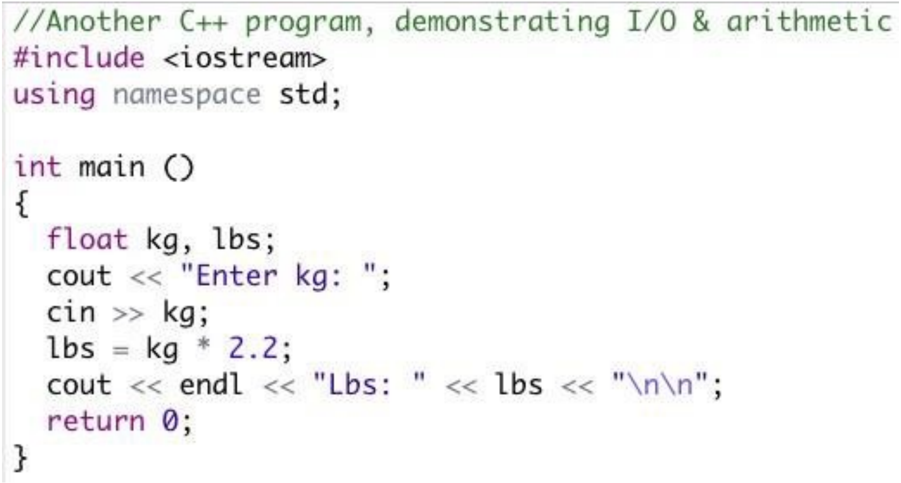


```
//Another C++ program, demonstrating variables
#include <iostream>
using namespace std;

int main ()
{
  int year;
  cout << "Enter a number: ";
  cin >> year;
  cout << "Hello " << year << "!!\n\n";
  return 0;
}
```

(Demo with `onlinegdb`)

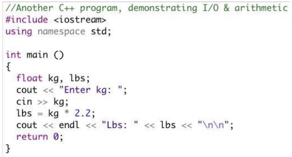
In Pairs or Triples:

Predict what the following pieces of code will do:



```
//Another C++ program, demonstrating I/O & arithmetic
#include <iostream>
using namespace std;

int main ()
{
  float kg, lbs;
  cout << "Enter kg: ";
  cin >> kg;
  lbs = kg * 2.2;
  cout << endl << "Lbs: " << lbs << "\n\n";
  return 0;
}
```

C++ Demo

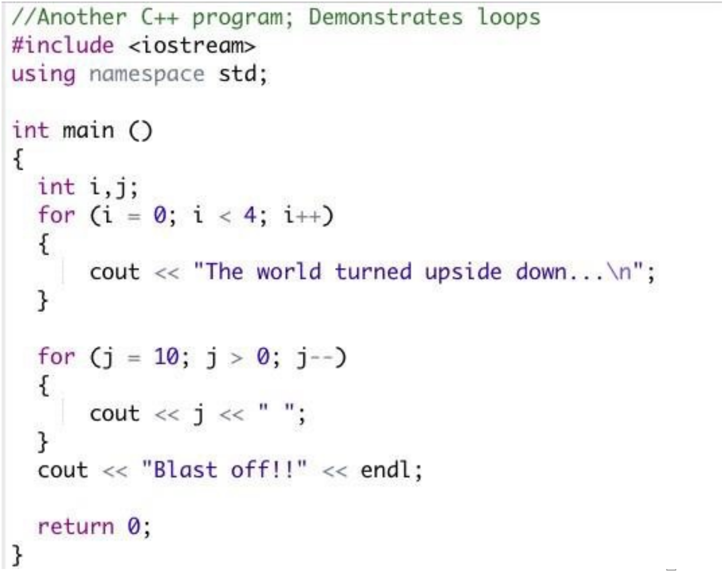


```
//Another C++ program, demonstrating I/O & arithmetic
#include <iostream>
using namespace std;

int main ()
{
  float kg, lbs;
  cout << "Enter kg: ";
  cin >> kg;
  lbs = kg * 2.2;
  cout << endl << "Lbs: " << lbs << "\n\n";
  return 0;
}
```

(Demo with `onlinegdb`)

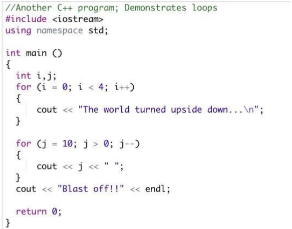
In Pairs or Triples:

Predict what the following pieces of code will do:



```
//Another C++ program; Demonstrates loops
#include <iostream>
using namespace std;

int main ()
{
  int i,j;
  for (i = 0; i < 4; i++)
  {
    cout << "The world turned upside down...\n";
  }

  for (j = 10; j > 0; j--)
  {
    cout << j << " ";
  }
  cout << "Blast off!!" << endl;

  return 0;
}
```

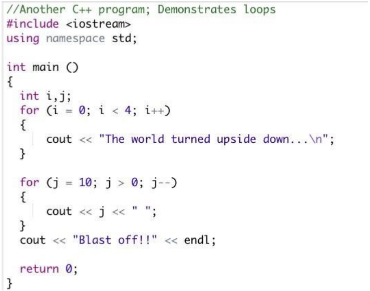
C++ Demo



```
//Another C++ program; Demonstrates loops
#include <iostream>
using namespace std;

int main ()
{
  int i,j;
  for (i = 0; i < 4; i++)
  {
      cout << "The world turned upside down...\n";
  }

  for (j = 10; j > 0; j--)
  {
      cout << j << " ";
  }
  cout << "Blast off!!" << endl;

  return 0;
}
```

(Demo with `onlinegdb`)

Definite loops



```
//Another C++ program; Demonstrates loops
#include <iostream>
using namespace std;

int main ()
{
  int i,j;
  for (i = 0; i < 4; i++)
  {
      cout << "The world turned upside down...\n";
  }

  for (j = 10; j > 0; j--)
  {
      cout << j << " ";
  }
  cout << "Blast off!!" << endl;

  return 0;
}
```

General format:

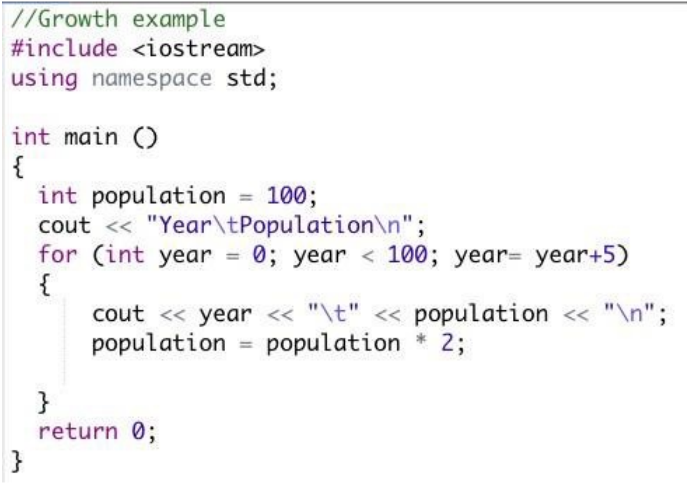
`for` (*initialization* ; *test* ; *updateAction*)
{

command1;
command2;
command3;
...

}

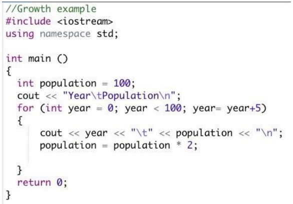
In Pairs or Triples:

Predict what the following pieces of code will do:



```
//Growth example
#include <iostream>
using namespace std;

int main ()
{
  int population = 100;
  cout << "Year\tPopulation\n";
  for (int year = 0; year < 100; year= year+5)
  {
      cout << year << "\t" << population << "\n";
      population = population * 2;

  }
  return 0;
}
```

C++ Demo

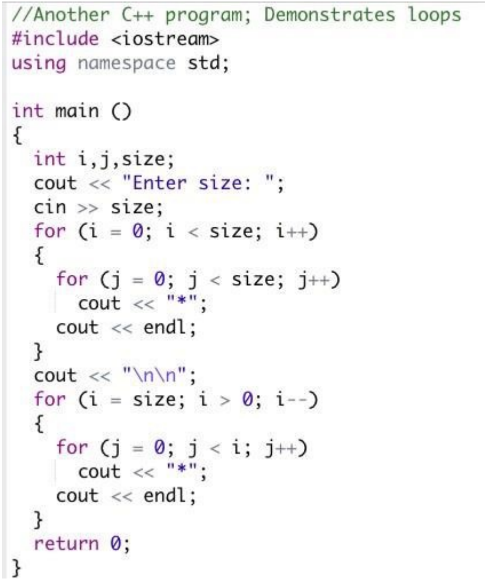


```
//Growth example
#include <iostream>
using namespace std;

int main ()
{
  int population = 100;
  cout << "Year\tPopulation\n";
  for (int year = 0; year < 100; year= year+5)
  {
      cout << year << "\t" << population << "\n";
      population = population * 2;

  }
  return 0;
}
```

(Demo with C++)

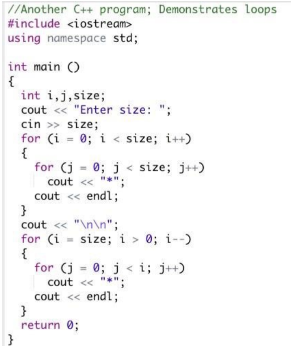
In Pairs or Triples:

Predict what the following pieces of code will do:



```
//Another C++ program; Demonstrates loops
#include <iostream>
using namespace std;

int main ()
{
  int i,j,size;
  cout << "Enter size: ";
  cin >> size;
  for (i = 0; i < size; i++)
  {
    for (j = 0; j < size; j++)
      cout << "*";
    cout << endl;
  }
  cout << "\n\n";
  for (i = size; i > 0; i--)
  {
    for (j = 0; j < i; j++)
      cout << "*";
    cout << endl;
  }
  return 0;
}
```

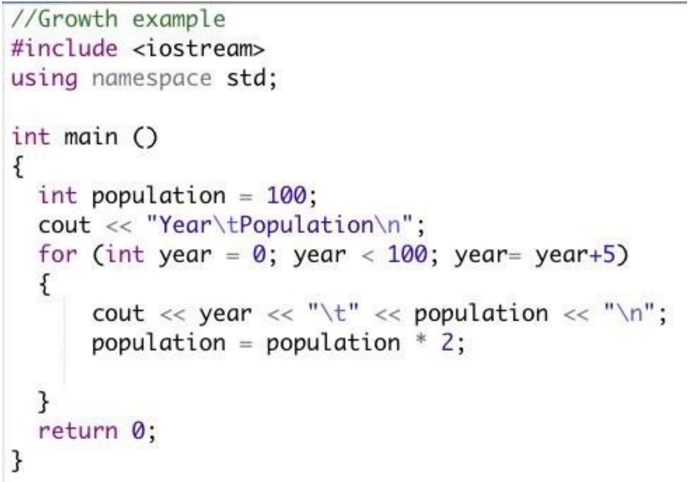
C++ Demo



```
//Another C++ program; Demonstrates loops
#include <iostream>
using namespace std;

int main ()
{
  int i,j,size;
  cout << "Enter size: ";
  cin >> size;
  for (i = 0; i < size; i++)
  {
    for (j = 0; j < size; j++)
      cout << "*";
    cout << endl;
  }
  cout << "\n\n";
  for (i = size; i > 0; i--)
  {
    for (j = 0; j < i; j++)
      cout << "*";
    cout << endl;
  }
  return 0;
}
```

(Demo with C++)

C++



```
//Growth example
#include <iostream>
using namespace std;

int main ()
{
  int population = 100;
  cout << "Year\tPopulation\n";
  for (int year = 0; year < 100; year= year+5)
  {
      cout << year << "\t" << population << "\n";
      population = population * 2;

  }
  return 0;
}
```