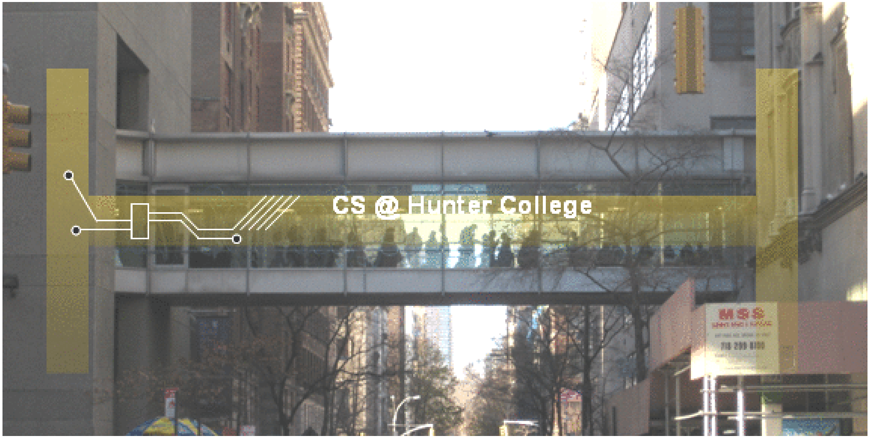
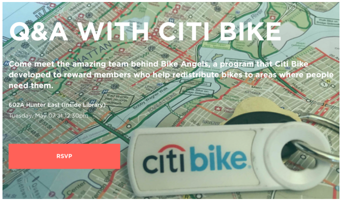
CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

Announcements



Q&A WITH CITI BIKE

Come meet the amazing team behind Bike Angels, a program that Citi Bike developed to reward members who help redistribute bikes to areas where people need them.

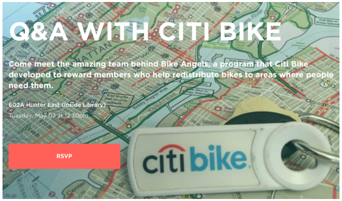
602A Hunter East (Inside Library)

Tuesday, May 07 at 12:30pm

RSVP
citi bike

- Mock exam next week.
  Final exam: 2 weeks!

Announcements



Q&A WITH CITI BIKE

Come meet the amazing team behind Bike Angels, a program that Citi Bike developed to reward members who help redistribute bikes to areas where people need them.

602A Hunter East (Inside Library)

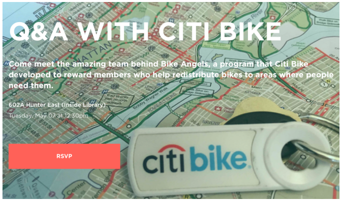
Tuesday, May 07 at 12:30pm

RSVP
citi bike

- Mock exam next week.
  Final exam: 2 weeks!
- We end lecture with a survey of computing research and tech in NYC.

  Today: Citi Bike's Bike Angels: Collin Waldroch

Announcements



Q&A WITH CITI BIKE

Come meet the amazing team behind Bike Angels, a program that Citi Bike developed to reward members who help redistribute bikes to areas where people need them.

602A Hunter East (Inside Library)

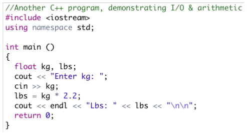
Tuesday, May 07 at 12:30pm

RSVP
citi bike

- Mock exam next week.
  Final exam: 2 weeks!
- We end lecture with a survey of computing research and tech in NYC.

  Today: Citi Bike's Bike Angels: Collin Waldroch
- 12:30pm: Informal Q&A with Collin in 631 Hunter East (inside library).

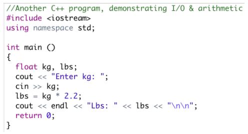
Today's Topics



```
//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;
}
```

- Recap: I/O & Definite Loops in C++
- Conditionals in C++
- Indefinite Loops in C++
- Recap: C++ & Python
- CS Survey

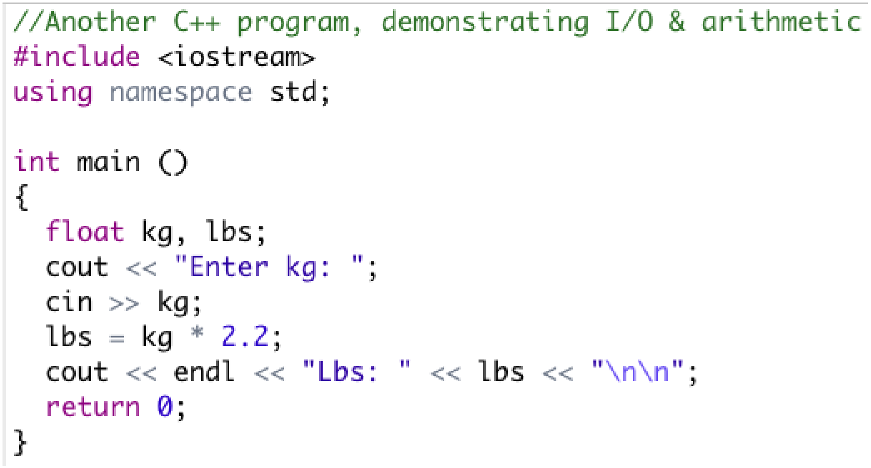
Today's Topics



```
//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;
}
```

- **Recap: I/O & Definite Loops in C++**
- Conditionals in C++
- Indefinite Loops in C++
- Recap: C++ & Python
- CS Survey

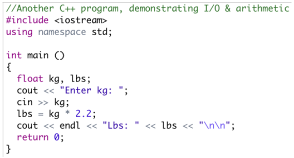
Recap: Basic Form & I/O in C++



```
//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;
}
```

Recap: Basic Form & I/O in C++

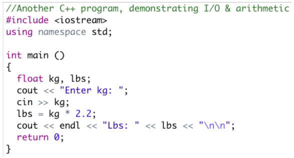
- Efficient for systems programming.



```
//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;
}
```

Recap: Basic Form & I/O in C++

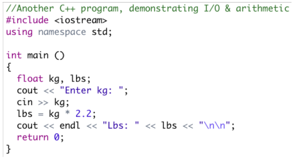
- Efficient for systems programming.
- Programs are organized in functions.



```
//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;
}
```

Recap: Basic Form & I/O in C++

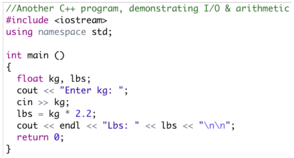
- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables:



```
//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;
}
```

Recap: Basic Form & I/O in C++

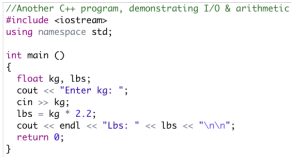
- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: `int num;`



```
//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;
}
```

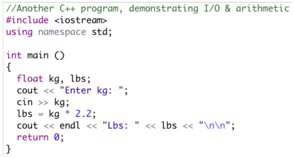
Recap: Basic Form & I/O in C++

- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: `int num;`
- Many types available:



```
//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;
}
```

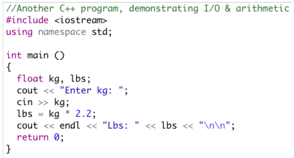
Recap: Basic Form & I/O in C++



```
//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;
}
```

- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: `int num;`
- Many types available:
  `int, float, char, ...`

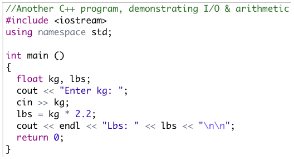
Recap: Basic Form & I/O in C++



```
//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;
}
```

- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: `int num;`
- Many types available:
  `int, float, char, ...`
- To print:

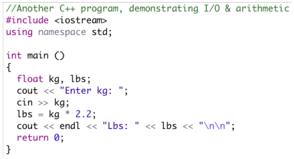
Recap: Basic Form & I/O in C++



```
//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;
}
```

- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: `int num;`
- Many types available:
  `int, float, char, ...`
- To print: `cout << "Hello!!";`

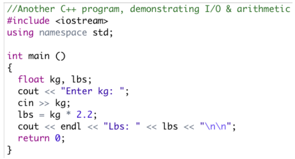
Recap: Basic Form & I/O in C++



```
//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;
}
```

- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: `int num;`
- Many types available:
  `int, float, char, ...`
- To print: `cout << "Hello!!";`
- To get input:

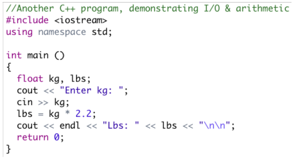
Recap: Basic Form & I/O in C++



```
//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;
}
```

- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: `int num;`
- Many types available:
  `int, float, char, ...`
- To print: `cout << "Hello!!";`
- To get input: `cin >> num;`

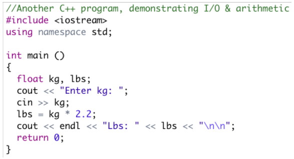
Recap: Basic Form & I/O in C++



```
//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;
}
```

- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: `int num;`
- Many types available:
  `int, float, char, ...`
- To print: `cout << "Hello!!";`
- To get input: `cin >> num;`
- To use those I/O functions:

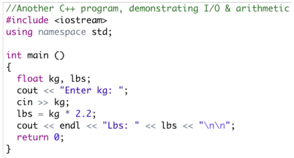
Recap: Basic Form & I/O in C++



```
//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;
}
```

- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: `int num;`
- Many types available: `int, float, char, ...`
- To print: `cout << "Hello!!";`
- To get input: `cin >> num;`
- To use those I/O functions: `#include <iostream>` `using namespace std;`

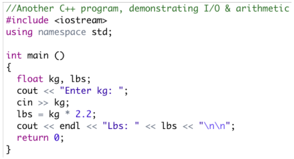
Recap: Basic Form & I/O in C++



```
//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;
}
```

- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: `int num;`
- Many types available:
  `int, float, char, ...`
- To print: `cout << "Hello!!";`
- To get input: `cin >> num;`
- To use those I/O functions:
  `#include <iostream>`
  `using namespace std;`
- Definite loops:

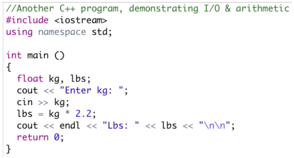
Recap: Basic Form & I/O in C++



```
//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;
}
```

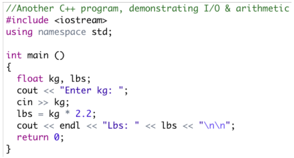
- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: `int num;`
- Many types available:
  `int, float, char, ...`
- To print: `cout << "Hello!!";`
- To get input: `cin >> num;`
- To use those I/O functions:
  `#include <iostream>`
  `using namespace std;`
- Definite loops:
  `for (i = 0; i < 10; i++) {...}`

Recap: Basic Form & I/O in C++


```
//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;
}
```

- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: `int num;`
- Many types available:
  `int, float, char, ...`
- To print: `cout << "Hello!!";`
- To get input: `cin >> num;`
- To use those I/O functions:
  `#include <iostream>`
  `using namespace std;`
- Definite loops:
  `for (i = 0; i < 10; i++) {...}`
- Blocks of code uses '{' and '}'.

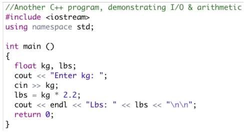
Recap: Basic Form & I/O in C++



```
//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;
}
```

- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: `int num;`
- Many types available:
  `int, float, char, ...`
- To print: `cout << "Hello!!";`
- To get input: `cin >> num;`
- To use those I/O functions:
  `#include <iostream>`
  `using namespace std;`
- Definite loops:
  `for (i = 0; i < 10; i++) {...}`
- Blocks of code uses '{' and '}'.
- Commands generally end in ';'.

Today's Topics

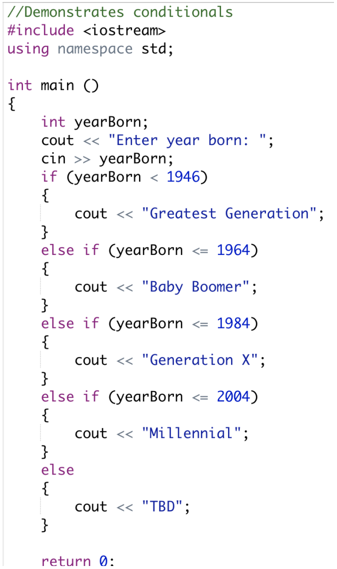


```
//Another C++ program, demonstrating I/O & arithmetic
#include <iostream>
using namespace std;

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

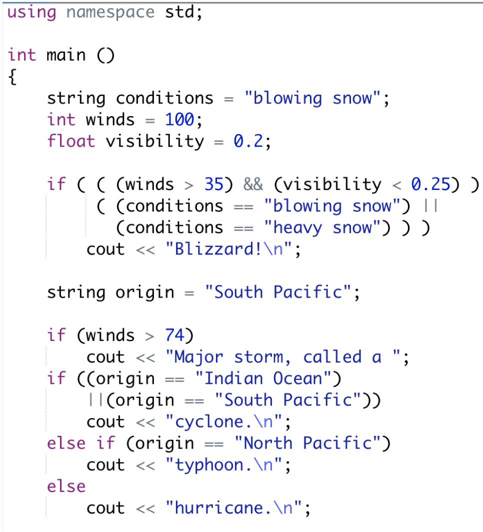
- Recap: I/O & Definite Loops in C++
- **Conditionals in C++**
- Indefinite Loops in C++
- Recap: C++ & Python
- CS Survey

In Pairs or Triples:

Predict what the following pieces of code will do:



```
//Demonstrates conditionals
#include <iostream>
using namespace std;

int main ()
{
    int yearBorn;
    cout << "Enter year born: ";
    cin >> yearBorn;
    if (yearBorn < 1946)
    {
        cout << "Greatest Generation";
    }
    else if (yearBorn <= 1964)
    {
        cout << "Baby Boomer";
    }
    else if (yearBorn <= 1984)
    {
        cout << "Generation X";
    }
    else if (yearBorn <= 2004)
    {
        cout << "Millennial";
    }
    else
    {
        cout << "TBD";
    }

    return 0;
```



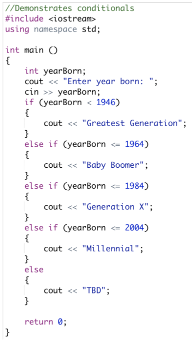
```
using namespace std;

int main ()
{
    string conditions = "blowing snow";
    int winds = 100;
    float visibility = 0.2;

    if ( ( (winds > 35) && (visibility < 0.25) )
         ( (conditions == "blowing snow") ||
           (conditions == "heavy snow") ) )
        cout << "Blizzard!\n";

    string origin = "South Pacific";

    if (winds > 74)
        cout << "Major storm, called a ";
    if ((origin == "Indian Ocean")
        ||(origin == "South Pacific"))
        cout << "cyclone.\n";
    else if (origin == "North Pacific")
        cout << "typhoon.\n";
    else
        cout << "hurricane.\n";
```

C++ Demo

```
//Demonstrates conditionals
#include <iostream>
using namespace std;

int main ()
{
    int yearBorn;
    cout << "Enter year born: ";
    cin >> yearBorn;
    if (yearBorn < 1946)
    {
        cout << "Greatest Generation";
    }
    else if (yearBorn <= 1964)
    {
        cout << "Baby Boomer";
    }
    else if (yearBorn <= 1984)
    {
        cout << "Generation X";
    }
    else if (yearBorn <= 2004)
    {
        cout << "Millennial";
    }
    else
    {
        cout << "TBD";
    }

    return 0;
}
```

(Demo with `onlinegdb`)

Conditionals



```
//Demonstrates conditionals
#include <iostream>
using namespace std;

int main ()
{
    int yearBorn;
    cout << "Enter year born: ";
    cin >> yearBorn;
    if (yearBorn < 1946)
    {
        cout << "Greatest Generation";
    }
    else if (yearBorn <= 1964)
    {
        cout << "Baby Boomer";
    }
    else if (yearBorn <= 1984)
    {
        cout << "Generation X";
    }
    else if (yearBorn <= 2004)
    {
        cout << "Millennial";
    }
    else
    {
        cout << "TBD";
    }

    return 0;
}
```

General format:

```
if ( logical expression )
{
      command1;
      ...
}
else if ( logical expression )
{
      command1;
      ...
}
else
{
      command1;
      ...
}
```

Logical Operators in C++

Very similar, just different names: `&&`, `||`, and `!`:

Logical Operators in C++

Very similar, just different names: `&&`, `||`, and `!`:

and (`&&`)

in1		in2	*returns:*
False	&&	False	False
False	&&	True	False
True	&&	False	False
True	&&	True	True

Logical Operators in C++

Very similar, just different names: `&&`, `||`, and `!`:

and (`&&`)

`in1`		`in2`	*returns:*
`False`	`&&`	`False`	`False`
`False`	`&&`	`True`	`False`
`True`	`&&`	`False`	`False`
`True`	`&&`	`True`	`True`

or (`||`)

`in1`		`in2`	*returns:*
`False`	`||`	`False`	`False`
`False`	`||`	`True`	`True`
`True`	`||`	`False`	`True`
`True`	`||`	`True`	`True`

Logical Operators in C++

Very similar, just different names: `&&`, `||`, and `!`:

and (`&&`)

`in1`		`in2`	*returns:*
`False`	`&&`	`False`	`False`
`False`	`&&`	`True`	`False`
`True`	`&&`	`False`	`False`
`True`	`&&`	`True`	`True`

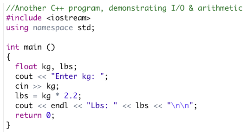
or (`||`)

`in1`		`in2`	*returns:*
`False`	`||`	`False`	`False`
`False`	`||`	`True`	`True`
`True`	`||`	`False`	`True`
`True`	`||`	`True`	`True`

not (`!`)

	`in1`	*returns:*
`!`	`False`	`True`
`!`	`True`	`False`

Today's Topics

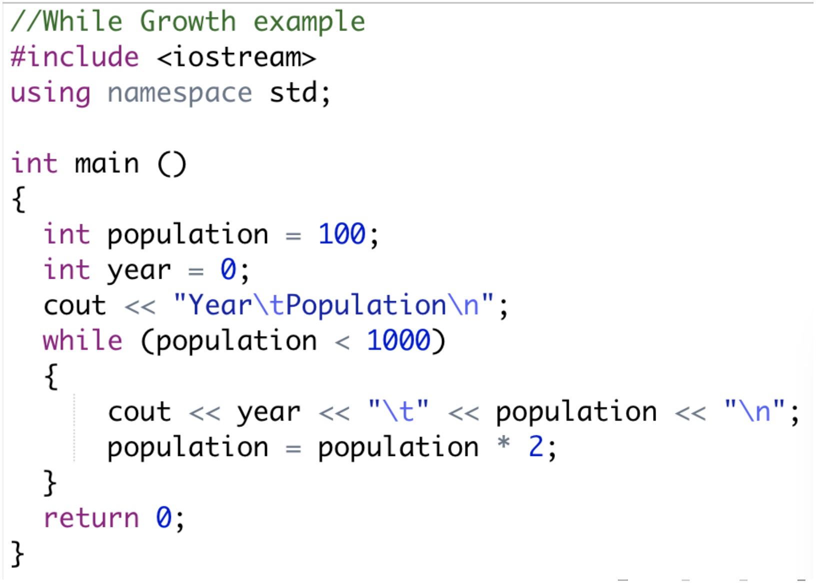


```
//Another C++ program, demonstrating I/O & arithmetic
#include <iostream>
using namespace std;

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

- Recap: I/O & Definite Loops in C++
- Conditionals in C++
- **Indefinite Loops in C++**
- Recap: C++ & Python
- CS Survey

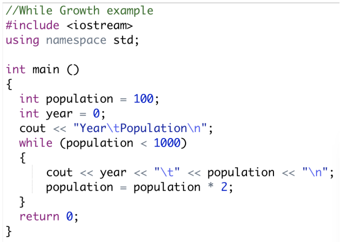
In Pairs or Triples:

Predict what the following pieces of code will do:



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

int main ()
{
  int population = 100;
  int year = 0;
  cout << "Year\tPopulation\n";
  while (population < 1000)
  {
      cout << year << "\t" << population << "\n";
      population = population * 2;
  }
  return 0;
}
```

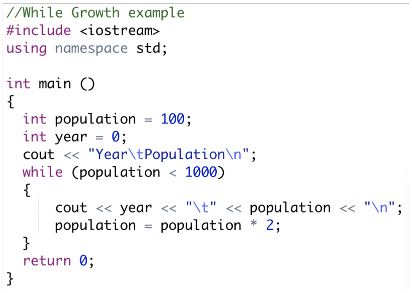
C++ Demo



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

int main ()
{
  int population = 100;
  int year = 0;
  cout << "Year\tPopulation\n";
  while (population < 1000)
  {
      cout << year << "\t" << population << "\n";
      population = population * 2;
  }
  return 0;
}
```

(Demo with `onlinegdb`)

Indefinite Loops: `while`



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

int main ()
{
  int population = 100;
  int year = 0;
  cout << "Year\tPopulation\n";
  while (population < 1000)
  {
      cout << year << "\t" << population << "\n";
      population = population * 2;
  }
  return 0;
}
```

General format:

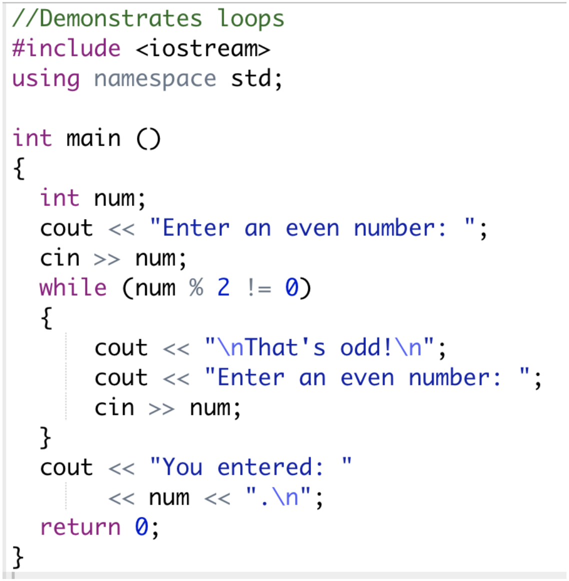
`while` (*logical expression*)
{

command1;
command2;
command3;
...

}

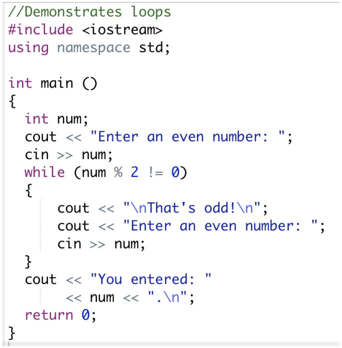
In Pairs or Triples:

Predict what the following piece of code will do:



```
//Demonstrates loops
#include <iostream>
using namespace std;

int main ()
{
  int num;
  cout << "Enter an even number: ";
  cin >> num;
  while (num % 2 != 0)
  {
      cout << "\nThat's odd!\n";
      cout << "Enter an even number: ";
      cin >> num;
  }
  cout << "You entered: "
       << num << ".\n";
  return 0;
}
```

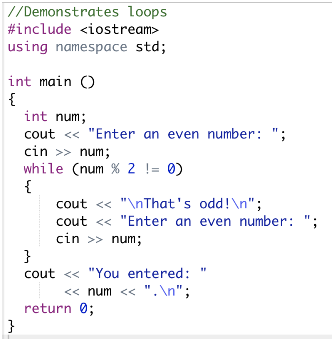
C++ Demo



```
//Demonstrates loops
#include <iostream>
using namespace std;

int main ()
{
  int num;
  cout << "Enter an even number: ";
  cin >> num;
  while (num % 2 != 0)
  {
      cout << "\nThat's odd!\n";
      cout << "Enter an even number: ";
      cin >> num;
  }
  cout << "You entered: "
       << num << ".\n";
  return 0;
}
```

(Demo with `onlinegdb`)

Indefinite Loops: `while`



```cpp
//Demonstrates loops
#include <iostream>
using namespace std;

int main ()
{
  int num;
  cout << "Enter an even number: ";
  cin >> num;
  while (num % 2 != 0)
  {
      cout << "\nThat's odd!\n";
      cout << "Enter an even number: ";
      cin >> num;
  }
  cout << "You entered: "
       << num << ".\n";
  return 0;
}
```

General format:

`while` (*logical expression*)
{

command1;
command2;
command3;
...

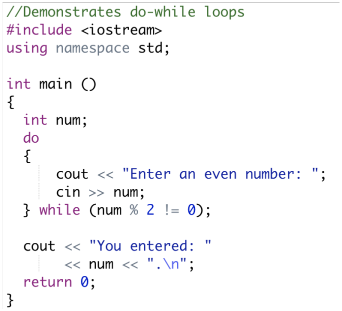
}

In Pairs or Triples:

Predict what the following pieces of code will do:

```
//Demonstrates do-while loops
#include <iostream>
using namespace std;

int main ()
{
  int num;
  do
  {
      cout << "Enter an even number: ";
      cin >> num;
  } while (num % 2 != 0);

  cout << "You entered: "
       << num << ".\n";
  return 0;
}
```

C++ Demo



```
//Demonstrates do-while loops
#include <iostream>
using namespace std;

int main ()
{
  int num;
  do
  {
      cout << "Enter an even number: ";
      cin >> num;
  } while (num % 2 != 0);

  cout << "You entered: "
       << num << ".\n";
  return 0;
}
```

(Demo with `onlinegdb`)

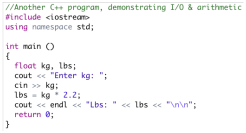
Indefinite Loops: `do-while`

```
//Demonstrates do-while loops
#include <iostream>
using namespace std;

int main ()
{
  int num;
  do
  {
      cout << "Enter an even number: ";
      cin >> num;
  } while (num % 2 != 0);

  cout << "You entered: "
       << num << ".\n";
  return 0;
}
```

General format:

`do`
{
    command1;
    command2;
    command3;
    ...
} `while` (*logical expression*);

Today's Topics

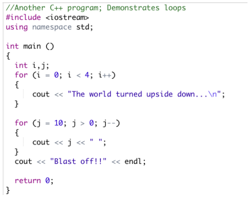


```
//Another C++ program, demonstrating I/O & arithmetic
#include <iostream>
using namespace std;

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

- Recap: I/O & Definite Loops in C++
- Conditionals in C++
- Indefinite Loops in C++
- **Recap: C++ & Python**
- CS Survey

Recap: C++ Control Structures

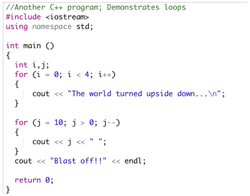
- I/O:



```
//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;
}
```

Recap: C++ Control Structures

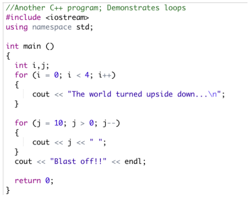
- I/O: `cin >> ...;`



```
//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;
}
```

Recap: C++ Control Structures

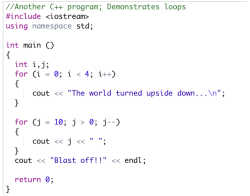
- I/O: `cin >> ...;` & `cout << ...;`



```
//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;
}
```

Recap: C++ Control Structures

- I/O: `cin >> ...;` & `cout << ...;`
- Definite loops:

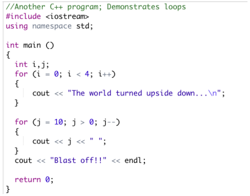


```
//Another C++ program; Demonstrates loops
#include <iostream>
using namespace std;

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;
}
```

Recap: C++ Control Structures

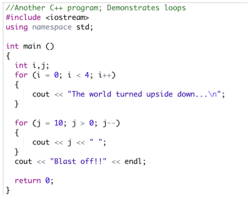
- I/O: `cin >> ...;` & `cout << ...;`
- Definite loops:

```
for (i = 0; i < 10; i++)
{
        ...
}
```



```
//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;
}
```

Recap: C++ Control Structures

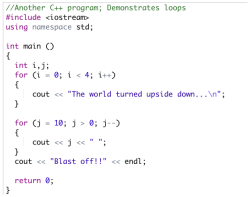


```
//Another C++ program; Demonstrates loops
#include <iostream>
using namespace std;

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;
}
```

- I/O: `cin >> ...;` & `cout << ...;`
- Definite loops:

  ```
  for (i = 0; i < 10; i++)
  {
        ...
  }
  ```
- Conditionals:

Recap: C++ Control Structures

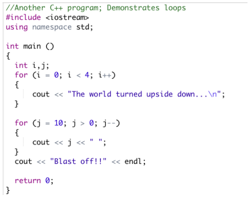


```
//Another C++ program; Demonstrates loops
#include <iostream>
using namespace std;

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;
}
```

- I/O: `cin >> ...;` & `cout << ...;`
- Definite loops:
  ```
  for (i = 0; i < 10; i++)
  {
          ...
  }
  ```
- Conditionals:

  `if` (*logical expression*)
  ```
  {
          ...
  }
  else
  {
          ...
  }
  ```

Recap: C++ Control Structures

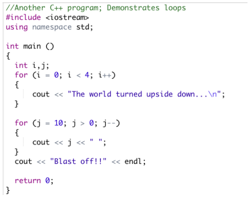


```
//Another C++ program; Demonstrates loops
#include <iostream>
using namespace std;

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;
}
```

- I/O: `cin >> ...;` & `cout << ...;`
- Definite loops:
  ```
  for (i = 0; i < 10; i++)
  {
        ...
  }
  ```
- Conditionals:

  `if` (*logical expression*)
  ```
  {
        ...
  }
  else
  {
        ...
  }
  ```
- Indefinite loops:

Recap: C++ Control Structures



```
//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;
}
```

- I/O: `cin >> ...;` & `cout << ...;`
- Definite loops:

```
for (i = 0; i < 10; i++)
{
      ...
}
```

- Conditionals:

```
if (logical expression)
{
      ...
}
else
{
      ...
}
```

- Indefinite loops:

```
while (logical expression)
{
      ...
}
```

In Pairs or Triples: Definite Loops in Python & C++

- *Rewrite this program in C++:*

```
for i in range(2017, 2000, -2):
    print("Year is", i)
```

- *Rewrite this program in Python:*

```
#include <iostream>
using namespace std;
int main()
{
  for (int i = 1; i < 50; i++)
  {
    cout << i << endl;
  }
  return 0;
}
```

In Pairs or Triples: Definite Loops in Python & C++

- *Rewrite this program in C++:*

```
for i in range(2017, 2000, -2):
    print("Year is", i)
```

In Pairs or Triples: Definite Loops in Python & C++

- *Rewrite this program in C++:*

```
for i in range(2017, 2000, -2):
    print("Year is", i)
```

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

In Pairs or Triples: Definite Loops in Python & C++

- *Rewrite this program in C++:*

```
for i in range(2017, 2000, -2):
    print("Year is", i)
```

```
#include <iostream>
using namespace std;
int main()
```

In Pairs or Triples: Definite Loops in Python & C++

- *Rewrite this program in C++:*

```
for i in range(2017, 2000, -2):
    print("Year is", i)
```

```
#include <iostream>
using namespace std;
int main()
{
```

In Pairs or Triples: Definite Loops in Python & C++

- *Rewrite this program in C++:*

```
for i in range(2017, 2000, -2):
    print("Year is", i)
```

```
#include <iostream>
using namespace std;
int main()
{
  for (int i = 2017; i >= 2000; i=i-2)
```

In Pairs or Triples: Definite Loops in Python & C++

- *Rewrite this program in C++:*

```
for i in range(2017, 2000, -2):
    print("Year is", i)
```

```
#include <iostream>
using namespace std;
int main()
{
  for (int i = 2017; i >= 2000; i=i-2)
  {
    cout << "Year is" << i << endl;
```

In Pairs or Triples: Definite Loops in Python & C++

- *Rewrite this program in C++:*

```
for i in range(2017, 2000, -2):
    print("Year is", i)
```

```
#include <iostream>
using namespace std;
int main()
{
  for (int i = 2017; i >= 2000; i=i-2)
  {
    cout << "Year is" << i << endl;
  }
  return 0;
}
```

In Pairs or Triples: Definite Loops in Python & C++

- *Rewrite this program in Python:*

```
#include <iostream>
using namespace std;
int main()
{
  for (int i = 1; i < 50; i++)
  {
    cout << i << endl;
  }
  return 0;
}
```

In Pairs or Triples: Definite Loops in Python & C++

- *Rewrite this program in Python:*

```
#include <iostream>
using namespace std;
int main()
{
  for (int i = 1; i < 50; i++)
  {
    cout << i << endl;
  }
  return 0;
}
```

```
for i in range(1, 50):
```

In Pairs or Triples: Definite Loops in Python & C++

- *Rewrite this program in Python:*

```
#include <iostream>
using namespace std;
int main()
{
  for (int i = 1; i < 50; i++)
  {
    cout << i << endl;
  }
  return 0;
}
```

```
for i in range(1, 50):
    print(i)
```

In Pairs or Triples: Conditionals in Python & C++

- *Python: what is the output?*

```
year = 2016
if year % 4 == 0 and \
    (not (year % 100 == 0) or (year % 400 == 0)):
     print("Leap!!")
print("Year")
```

- *Write a C++ program that asks the user the number of times they plan to ride transit this week. Your program should then print if it is cheaper to buy single ride metro cards or 7-day unlimited card.*
  (The 7-day card is $31.00, and the cost of single ride, with bonus, is $2.48).

In Pairs or Triples: Conditionals in Python & C++

- *Python: what is the output?*

```
year = 2016
if year % 4 == 0 and \
    (not (year % 100 == 0) or (year % 400 == 0)):
     print("Leap!!")
print("Year")
```

In Pairs or Triples: Conditionals in Python & C++

- *Python: what is the output?*

```
year = 2016
if year % 4 == 0 and \
    (not (year % 100 == 0) or (year % 400 == 0)):
     print("Leap!!")
print("Year")
```

Working out the arithmetic and logic:

In Pairs or Triples: Conditionals in Python & C++

- *Python: what is the output?*

```
year = 2016
if year % 4 == 0 and \
    (not (year % 100 == 0) or (year % 400 == 0)):
     print("Leap!!")
print("Year")
```

Working out the arithmetic and logic:

- `year % 4` *is* 504.

In Pairs or Triples: Conditionals in Python & C++

- *Python: what is the output?*

```
year = 2016
if year % 4 == 0 and \
    (not (year % 100 == 0) or (year % 400 == 0)):
     print("Leap!!")
print("Year")
```

Working out the arithmetic and logic:

- `year % 4` *is* 504.
- 504 $\neq$ 0 *so the first part of the* `and` *is* `False`.

In Pairs or Triples: Conditionals in Python & C++

- *Python: what is the output?*

```
year = 2016
if year % 4 == 0 and \
    (not (year % 100 == 0) or (year % 400 == 0)):
     print("Leap!!")
print("Year")
```

Working out the arithmetic and logic:

- `year % 4` *is* 504.
- 504 $\neq$ 0 *so the first part of the* `and` *is* `False`.
- *Since (*`False and` *anything) is* `False`*, the expression is* `False`.

In Pairs or Triples: Conditionals in Python & C++

- *Python: what is the output?*

```
year = 2016
if year % 4 == 0 and \
    (not (year % 100 == 0) or (year % 400 == 0)):
     print("Leap!!")
print("Year")
```

Working out the arithmetic and logic:

- `year % 4` *is* 504.
- 504 $\neq$ 0 *so the first part of the* `and` *is* `False`.
- *Since (*`False and` *anything) is* `False`*, the expression is* `False`*.* *(There's no need to figure out the rest of the expression.)*

In Pairs or Triples: Conditionals in Python & C++

- *Python: what is the output?*

```
year = 2016
if year % 4 == 0 and \
    (not (year % 100 == 0) or (year % 400 == 0)):
     print("Leap!!")
print("Year")
```

Working out the arithmetic and logic:

- `year % 4` *is* 504.
- 504 $\neq$ 0 *so the first part of the* `and` *is* `False`.
- *Since (*`False and` *anything) is* `False`, *the expression is* `False`. *(There's no need to figure out the rest of the expression.)*
- *Never enter the* `if`*-clause and go to the next line.*

In Pairs or Triples: Conditionals in Python & C++

- *Python: what is the output?*

```
year = 2016
if year % 4 == 0 and \
    (not (year % 100 == 0) or (year % 400 == 0)):
     print("Leap!!")
print("Year")
```

Working out the arithmetic and logic:

- `year % 4` *is* 504.
- 504 $\neq$ 0 *so the first part of the* `and` *is* `False`.
- *Since (*`False and` *anything) is* `False`*, the expression is* `False`*. (There's no need to figure out the rest of the expression.)*
- *Never enter the* `if`*-clause and go to the next line.*
- *Only thing printed is:* `Year`

In Pairs or Triples: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.48 per ride) or 7-day unlimited card ($31.00).*

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

In Pairs or Triples: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.48 per ride) or 7-day unlimited card ($31.00).*

```
#include <iostream>
using namespace std;
int main()
```

In Pairs or Triples: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.48 per ride) or 7-day unlimited card ($31.00).*

```
#include <iostream>
using namespace std;
int main()
{
  int rides;
```

In Pairs or Triples: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.48 per ride) or 7-day unlimited card ($31.00).*

```
#include <iostream>
using namespace std;
int main()
{
  int rides;
  cout << "Enter number of rides:";
```

In Pairs or Triples: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.48 per ride) or 7-day unlimited card ($31.00).*

```
#include <iostream>
using namespace std;
int main()
{
  int rides;
  cout << "Enter number of rides:";
  cin >> rides;
```

In Pairs or Triples: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.48 per ride) or 7-day unlimited card ($31.00).*

```
#include <iostream>
using namespace std;
int main()
{
  int rides;
  cout << "Enter number of rides:";
  cin >> rides;
  if (2.48 * rides < 31.00)
```

In Pairs or Triples: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.48 per ride) or 7-day unlimited card ($31.00).*

```
#include <iostream>
using namespace std;
int main()
{
  int rides;
  cout << "Enter number of rides:";
  cin >> rides;
  if (2.48 * rides < 31.00)
  {
    cout << "Cheaper to buy single ride metro cards.\n";
  }
```

In Pairs or Triples: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.48 per ride) or 7-day unlimited card ($31.00).*

```
#include <iostream>
using namespace std;
int main()
{
  int rides;
  cout << "Enter number of rides:";
  cin >> rides;
  if (2.48 * rides < 31.00)
  {
    cout << "Cheaper to buy single ride metro cards.\n";
  }
  else
```

In Pairs or Triples: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.48 per ride) or 7-day unlimited card ($31.00).*

```
#include <iostream>
using namespace std;
int main()
{
  int rides;
  cout << "Enter number of rides:";
  cin >> rides;
  if (2.48 * rides < 31.00)
  {
    cout << "Cheaper to buy single ride metro cards.\n";
  }
  else
  {
    cout << "Cheaper to buy 7-day unlimited card.\n";
  }
```

In Pairs or Triples: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.48 per ride) or 7-day unlimited card ($31.00).*

```
#include <iostream>
using namespace std;
int main()
{
  int rides;
  cout << "Enter number of rides:";
  cin >> rides;
  if (2.48 * rides < 31.00)
  {
    cout << "Cheaper to buy single ride metro cards.\n";
  }
  else
  {
    cout << "Cheaper to buy 7-day unlimited card.\n";
  }
  return 0;
}
```

In Pairs or Triples: Indefinite Loops in Python & C++

- Write Python code that repeatedly prompts for a non-empty string.

- Write C++ code that repeatedly prompts until an odd number is entered.

In Pairs or Triples: Indefinite Loops in Python & C++

- Write Python code that repeatedly prompts for a non-empty string.

In Pairs or Triples: Indefinite Loops in Python & C++

- Write Python code that repeatedly prompts for a non-empty string.

```
s = ""
```

In Pairs or Triples: Indefinite Loops in Python & C++

- Write Python code that repeatedly prompts for a non-empty string.

```
s = ""
while s == "":
```

In Pairs or Triples: Indefinite Loops in Python & C++

- Write Python code that repeatedly prompts for a non-empty string.

```
s = ""
while s == "":
  s = input("Enter a non-empty string:  ")
```

In Pairs or Triples: Indefinite Loops in Python & C++

- Write Python code that repeatedly prompts for a non-empty string.

```
s = ""
while s == "":
  s = input("Enter a non-empty string:  ")
print("You entered:  ", s)
```

In Pairs or Triples: Indefinite Loops in Python & C++

- Write Python code that repeatedly prompts for a non-empty string.

```
s = ""
while s == "":
  s = input("Enter a non-empty string:  ")
print("You entered:  ", s)
```

- Write C++ code that repeatedly prompts until an odd number is entered.

In Pairs or Triples: Indefinite Loops in Python & C++

- Write Python code that repeatedly prompts for a non-empty string.

```
s = ""
while s == "":
  s = input("Enter a non-empty string:  ")
print("You entered:  ", s)
```

- Write C++ code that repeatedly prompts until an odd number is entered.

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

In Pairs or Triples: Indefinite Loops in Python & C++

- Write Python code that repeatedly prompts for a non-empty string.

```
s = ""
while s == "":
  s = input("Enter a non-empty string:  ")
print("You entered:  ", s)
```

- Write C++ code that repeatedly prompts until an odd number is entered.

```
#include <iostream>
using namespace std;
int main()
```

In Pairs or Triples: Indefinite Loops in Python & C++

- Write Python code that repeatedly prompts for a non-empty string.

```
s = ""
while s == "":
  s = input("Enter a non-empty string:  ")
print("You entered:  ", s)
```

- Write C++ code that repeatedly prompts until an odd number is entered.

```
#include <iostream>
using namespace std;
int main()
{
  int num = 0;
```

In Pairs or Triples: Indefinite Loops in Python & C++

- Write Python code that repeatedly prompts for a non-empty string.

```
s = ""
while s == "":
  s = input("Enter a non-empty string:  ")
print("You entered:  ", s)
```

- Write C++ code that repeatedly prompts until an odd number is entered.

```
#include <iostream>
using namespace std;
int main()
{
  int num = 0;
  while (num % 2 == 0)
```

In Pairs or Triples: Indefinite Loops in Python & C++

- Write Python code that repeatedly prompts for a non-empty string.

```
s = ""
while s == "":
  s = input("Enter a non-empty string:  ")
print("You entered:  ", s)
```

- Write C++ code that repeatedly prompts until an odd number is entered.

```
#include <iostream>
using namespace std;
int main()
{
  int num = 0;
  while (num % 2 == 0)
  {
    cout << "Enter an odd number:";
```

In Pairs or Triples: Indefinite Loops in Python & C++

- Write Python code that repeatedly prompts for a non-empty string.

```
s = ""
while s == "":
  s = input("Enter a non-empty string:  ")
print("You entered:  ", s)
```

- Write C++ code that repeatedly prompts until an odd number is entered.

```
#include <iostream>
using namespace std;
int main()
{
  int num = 0;
  while (num % 2 == 0)
  {
    cout << "Enter an odd number:";
    cin >> num;
```

In Pairs or Triples: Indefinite Loops in Python & C++

- Write Python code that repeatedly prompts for a non-empty string.

```
s = ""
while s == "":
  s = input("Enter a non-empty string:  ")
print("You entered:  ", s)
```

- Write C++ code that repeatedly prompts until an odd number is entered.

```
#include <iostream>
using namespace std;
int main()
{
  int num = 0;
  while (num % 2 == 0)
  {
    cout << "Enter an odd number:";
    cin >> num;
  }
```

In Pairs or Triples: Indefinite Loops in Python & C++

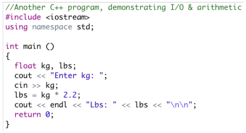
- Write Python code that repeatedly prompts for a non-empty string.

```
s = ""
while s == "":
  s = input("Enter a non-empty string:  ")
print("You entered:  ", s)
```

- Write C++ code that repeatedly prompts until an odd number is entered.

```
#include <iostream>
using namespace std;
int main()
{
  int num = 0;
  while (num % 2 == 0)
  {
    cout << "Enter an odd number:";
    cin >> num;
  }
  return 0;
}
```

Today's Topics



```
//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;
}
```

- Recap: I/O & Definite Loops in C++
- Conditionals in C++
- Indefinite Loops in C++
- Recap: C++ & Python
- **CS Survey**

CS Surveys Talk: CitiBike BikeAngels

Collin Waldoch

(Image from *New Yorker*)

CS Surveys Talk: CitiBike BikeAngels

Collin Waldoch

(Image from *New Yorker*)

- Brief overview of Citi Bike & Bike Angels

CS Surveys Talk: CitiBike BikeAngels

Collin Waldoch

(Image from *New Yorker*)

- Brief overview of Citi Bike & Bike Angels
- What Collin does and loves about Bike Angels.

CS Surveys Talk: CitiBike BikeAngels

Collin Waldoch

(Image from *New Yorker*)

- Brief overview of Citi Bike & Bike Angels
- What Collin does and loves about Bike Angels.
- Design challenge: work in pairs or triples with Bike Angels & UTAs.

CS Surveys Talk: CitiBike BikeAngels

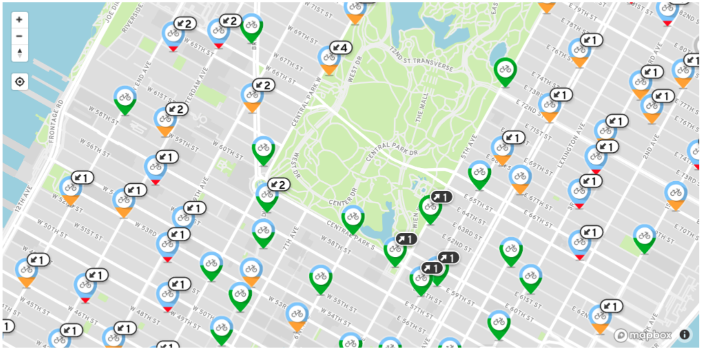
Collin Waldoch

(Image from *New Yorker*)

- Brief overview of Citi Bike & Bike Angels
- What Collin does and loves about Bike Angels.
- Design challenge: work in pairs or triples with Bike Angels & UTAs.
- 12:30pm: Informal Q&A in 631 Hunter East (inside library).

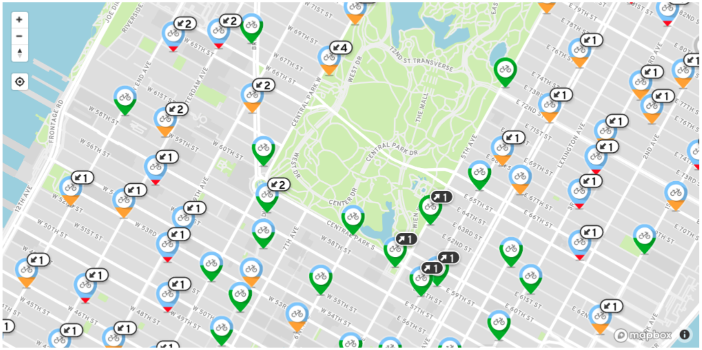
Lecture Slip

Map



Lecture Slip

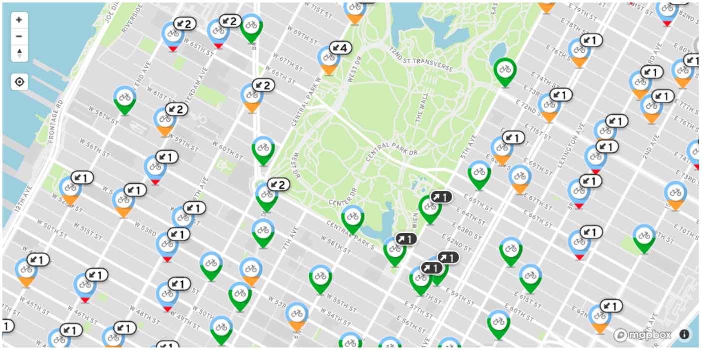
Map



- Design an algorithm to find mostly full stations.

Lecture Slip

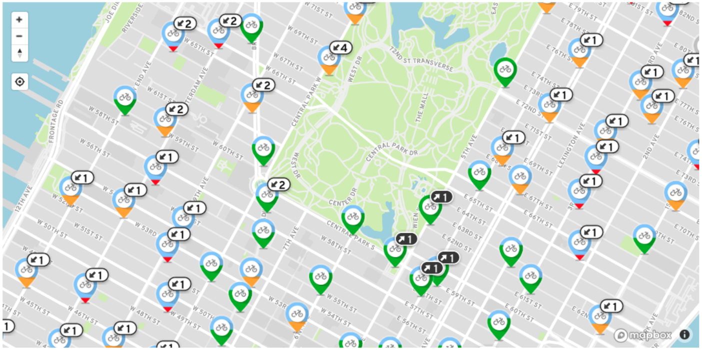
Map



- Design an algorithm to find mostly full stations.
- Design an algorithm to maximize points earned.

Lecture Slip

Map



- Design an algorithm to find mostly full stations.
- Design an algorithm to maximize points earned.
- Note: map and photo form on back of lecture slip.

Practice Quiz & Final Questions

- Lightning rounds:

Practice Quiz & Final Questions

- Lightning rounds:
  - write as much you can for 60 seconds;

Practice Quiz & Final Questions

- Lightning rounds:
  - write as much you can for 60 seconds;
  - followed by answer; and

Practice Quiz & Final Questions

- Lightning rounds:
  - write as much you can for 60 seconds;
  - followed by answer; and
  - repeat.

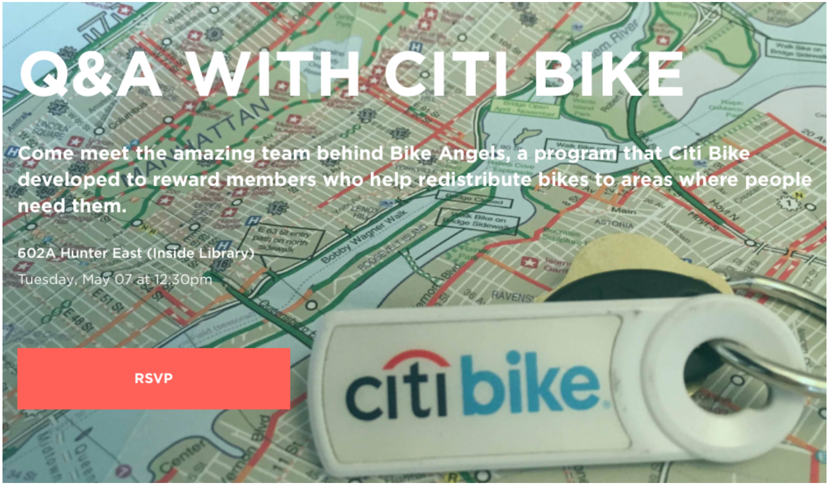
Practice Quiz & Final Questions

- Lightning rounds:
  - write as much you can for 60 seconds;
  - followed by answer; and
  - repeat.
- Past exams are on the webpage (under Final Exam Information).

Practice Quiz & Final Questions

- Lightning rounds:
  - write as much you can for 60 seconds;
  - followed by answer; and
  - repeat.
- Past exams are on the webpage (under Final Exam Information).
- We'll start with: Fall 17, Version 3.

Writing Boards



Q&A WITH CITI BIKE

Come meet the amazing team behind Bike Angels, a program that Citi Bike developed to reward members who help redistribute bikes to areas where people need them.

602A Hunter East (Inside Library)

Tuesday, May 07 at 12:30pm

RSVP

- Return writing boards as you leave...