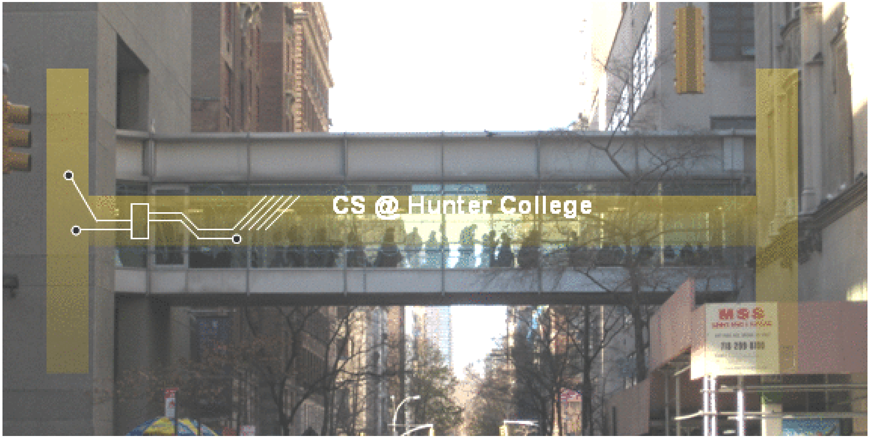
CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

Teacher Evaluation

- Please take a moment to fill out the **Teacher Evaluations**

Teacher Evaluation

- Please take a moment to fill out the **Teacher Evaluations**
- Your chance to provide feedback!

Teacher Evaluation

- Please take a moment to fill out the **Teacher Evaluations**
- Your chance to provide feedback!
- Responses are **completely anonymous.**

Teacher Evaluation

- Please take a moment to fill out the **Teacher Evaluations**
- Your chance to provide feedback!
- Responses are **completely anonymous.**
- Smartphone: www.hunter.cuny.edu/mobilete

Teacher Evaluation

- Please take a moment to fill out the **Teacher Evaluations**
- Your chance to provide feedback!
- Responses are **completely anonymous.**
- Smartphone: www.hunter.cuny.edu/mobilete
- Computer: www.hunter.cuny.edu/te

Teacher Evaluation

- Please take a moment to fill out the **Teacher Evaluations**
- Your chance to provide feedback!
- Responses are **completely anonymous.**
- Smartphone: www.hunter.cuny.edu/mobilete
- Computer: www.hunter.cuny.edu/te

Announcements

- **Final Exam December 19 at 9-11 AM**

Announcements

- **Final Exam December 19 at 9-11 AM**
- **Seating assignment available on Blackboard/My Grades**
  Please check now, if there are any problems, email me ASAP

Announcements

- **Final Exam December 19 at 9-11 AM**
- **Seating assignment available on Blackboard/My Grades**
  Please check now, if there are any problems, email me ASAP
- Next **Tuesday December 13, we will have a Mock Exam**

Announcements

- **Final Exam December 19 at 9-11 AM**
- **Seating assignment available on Blackboard/My Grades**
  Please check now, if there are any problems, email me ASAP
- Next **Tuesday December 13, we will have a Mock Exam**
  - Room 118 Hunter North (Assembly Hall), ground floor of the North Building

Announcements

- **Final Exam December 19 at 9-11 AM**
- **Seating assignment available on Blackboard/My Grades**
  Please check now, if there are any problems, email me ASAP
- Next **Tuesday December 13, we will have a Mock Exam**
  - Room 118 Hunter North (Assembly Hall), ground floor of the North Building
  - Only 1.15 hours for the Mock, 2 hours for the real exam.

Announcements

- **Final Exam December 19 at 9-11 AM**
- **Seating assignment available on Blackboard/My Grades**
  Please check now, if there are any problems, email me ASAP
- Next **Tuesday December 13, we will have a Mock Exam**
  - Room 118 Hunter North (Assembly Hall), ground floor of the North Building
  - Only 1.15 hours for the Mock, 2 hours for the real exam.
  - Just a practice run, this WILL NOT be the same as the real exam, and it will not be graded.

CSci 127: Introduction to Computer Science

Frequently Asked Questions

- What's the best way to study for the final exam?

Frequently Asked Questions

- What's the best way to study for the final exam?
  The final exam problems are variations on the homework, quizzes, lecture examples, and lecture previews.

Frequently Asked Questions

- What's the best way to study for the final exam?
  The final exam problems are variations on the homework, quizzes, lecture examples, and lecture previews.
  Past exams (and answer keys) are on-line. Do 7-10 previous exams: allow 1 hour (half time) and work through , grade yourself, update note sheet, and repeat.

Frequently Asked Questions

- What's the best way to study for the final exam?
  The final exam problems are variations on the homework, quizzes, lecture examples, and lecture previews.
  Past exams (and answer keys) are on-line. Do 7-10 previous exams: allow 1 hour (half time) and work through , grade yourself, update note sheet, and repeat.
- I'm worried about my grade. Should I do Pass/NoCredit?

Frequently Asked Questions

- What's the best way to study for the final exam?
  The final exam problems are variations on the homework, quizzes, lecture examples, and lecture previews.
  Past exams (and answer keys) are on-line. Do 7-10 previous exams: allow 1 hour (half time) and work through , grade yourself, update note sheet, and repeat.
- I'm worried about my grade. Should I do Pass/NoCredit?
  It's fine with us, but check with your advisor to make sure it's accepted for your program of study.

Frequently Asked Questions

- What's the best way to study for the final exam?
  The final exam problems are variations on the homework, quizzes, lecture examples, and lecture previews.
  Past exams (and answer keys) are on-line. Do 7-10 previous exams: allow 1 hour (half time) and work through , grade yourself, update note sheet, and repeat.
- I'm worried about my grade. Should I do Pass/NoCredit?
  It's fine with us, but check with your advisor to make sure it's accepted for your program of study.
  Deadline: Monday, December 5, 2022, 11:59 PM (EST)

Frequently Asked Questions

- What's the best way to study for the final exam?
  The final exam problems are variations on the homework, quizzes, lecture examples, and lecture previews.
  Past exams (and answer keys) are on-line. Do 7-10 previous exams: allow 1 hour (half time) and work through , grade yourself, update note sheet, and repeat.
- I'm worried about my grade. Should I do Pass/NoCredit?
  It's fine with us, but check with your advisor to make sure it's accepted for your program of study.
  Deadline: Monday, December 5, 2022, 11:59 PM (EST)
- Why do you care about cheating?

Frequently Asked Questions

- What's the best way to study for the final exam?
  The final exam problems are variations on the homework, quizzes, lecture examples, and lecture previews.
  Past exams (and answer keys) are on-line. Do 7-10 previous exams: allow 1 hour (half time) and work through , grade yourself, update note sheet, and repeat.
- I'm worried about my grade. Should I do Pass/NoCredit?
  It's fine with us, but check with your advisor to make sure it's accepted for your program of study.
  Deadline: Monday, December 5, 2022, 11:59 PM (EST)
- Why do you care about cheating?
  First: it gives unfair advantage & is immoral.

Frequently Asked Questions

- What's the best way to study for the final exam?
  The final exam problems are variations on the homework, quizzes, lecture examples, and lecture previews.
  Past exams (and answer keys) are on-line. Do 7-10 previous exams: allow 1 hour (half time) and work through , grade yourself, update note sheet, and repeat.
- I'm worried about my grade. Should I do Pass/NoCredit?
  It's fine with us, but check with your advisor to make sure it's accepted for your program of study.
  Deadline: Monday, December 5, 2022, 11:59 PM (EST)
- Why do you care about cheating?
  First: it gives unfair advantage & is immoral.
  Second: it degrades the quality of our students.

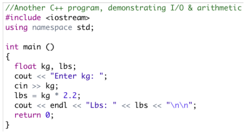
Frequently Asked Questions

- What's the best way to study for the final exam?
  The final exam problems are variations on the homework, quizzes, lecture examples, and lecture previews.
  Past exams (and answer keys) are on-line. Do 7-10 previous exams: allow 1 hour (half time) and work through , grade yourself, update note sheet, and repeat.
- I'm worried about my grade. Should I do Pass/NoCredit?
  It's fine with us, but check with your advisor to make sure it's accepted for your program of study.
  Deadline: Monday, December 5, 2022, 11:59 PM (EST)
- Why do you care about cheating?
  First: it gives unfair advantage & is immoral.
  Second: it degrades the quality of our students.
  Third: it's a standard question on faculty references.

Frequently Asked Questions

- What's the best way to study for the final exam?
  The final exam problems are variations on the homework, quizzes, lecture examples, and lecture previews.
  Past exams (and answer keys) are on-line. Do 7-10 previous exams: allow 1 hour (half time) and work through , grade yourself, update note sheet, and repeat.
- I'm worried about my grade. Should I do Pass/NoCredit?
  It's fine with us, but check with your advisor to make sure it's accepted for your program of study.
  Deadline: Monday, December 5, 2022, 11:59 PM (EST)
- Why do you care about cheating?
  First: it gives unfair advantage & is immoral.
  Second: it degrades the quality of our students.
  Third: it's a standard question on faculty references.
  Industry & graduate schools hate it: don't want someone who falsifies work.

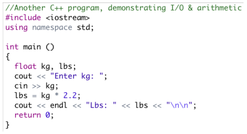
Today's Topics



```
//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

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;
}
```

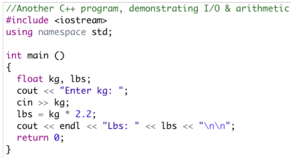
- **Recap: I/O & Definite Loops in C++**
- Conditionals in C++
- Indefinite Loops in C++
- Recap: C++ & Python

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

```
//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++

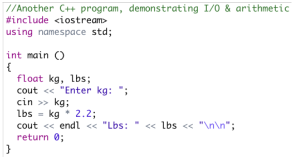
- Efficient for systems programming.



```
//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++

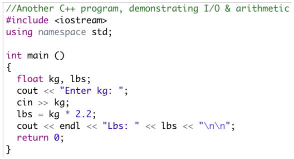
- Efficient for systems programming.
- Programs are organized in functions.



```
//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++

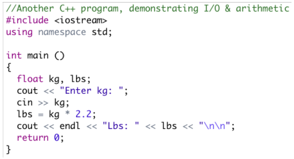
- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables:



```cpp
//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++

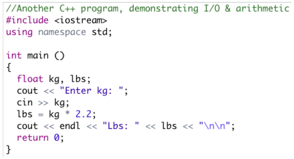
- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: `int num;`



```
//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;
}
```

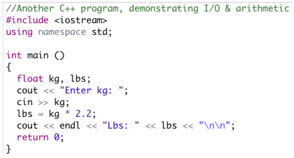
Recap: Basic Form & I/O in C++

- 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;
}
```

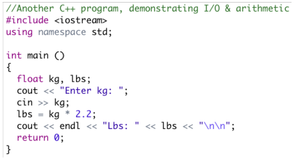
Recap: Basic Form & I/O in C++



```
//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, ...`

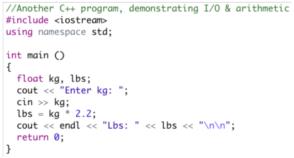
Recap: Basic Form & I/O in C++



```
//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:

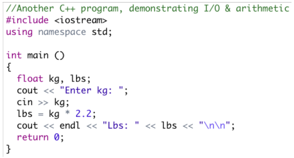
Recap: Basic Form & I/O in C++



```
//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!!";`

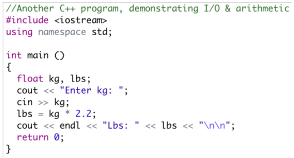
Recap: Basic Form & I/O in C++



```
//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:

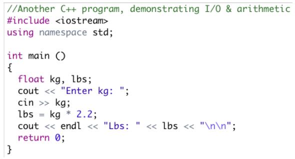
Recap: Basic Form & I/O in C++



```
//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;`

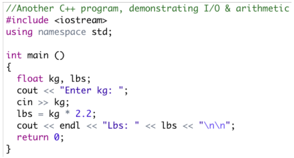
Recap: Basic Form & I/O in C++



```
//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:

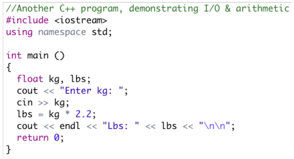
Recap: Basic Form & I/O in C++



```
//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;`

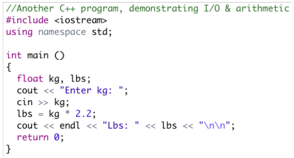
Recap: Basic Form & I/O in C++



```
//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:

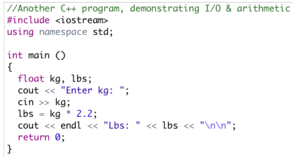
Recap: Basic Form & I/O in C++



```
//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++) {...}`

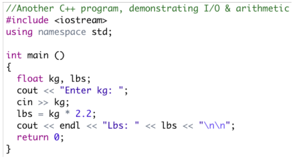
Recap: Basic Form & I/O in C++



```
//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 '}'.

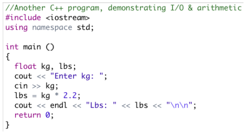
Recap: Basic Form & I/O in C++



```
//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

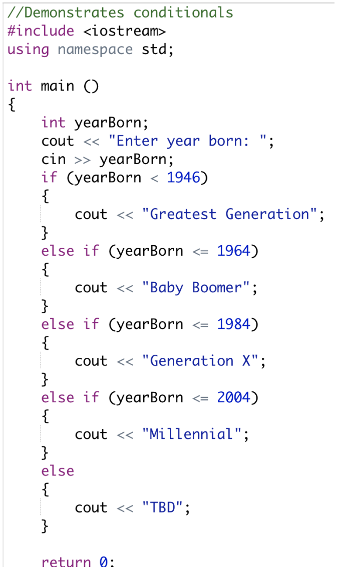


```
//Another C++ program, demonstrating I/O & arithmetic
#include <iostream>
using namespace std;

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

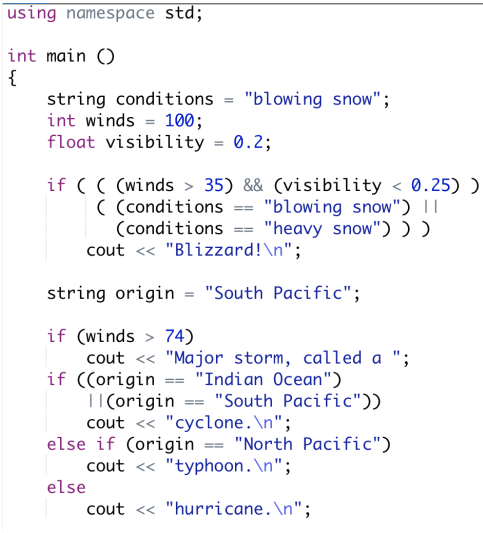
- Recap: I/O & Definite Loops in C++
- **Conditionals in C++**
- Indefinite Loops in C++
- Recap: C++ & Python

Challenge:

Predict what the following pieces of code will do:



```cpp
//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;
```

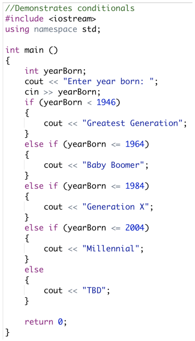


```cpp
using namespace std;

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, https://www.onlinegdb.com/Hk_q0V8xf

```
//C++ program 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;
}
```

Conditionals



```cpp
//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

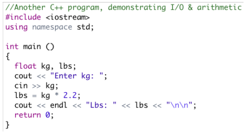
not (`!`)

	in1	*returns:*
!	False	True
!	True	False

Lecture Slip

- Write a C++ program that will ask for the time in 24 hour format and, knowing it is morning before 12pm and evening after 6pm (18), it will print out Morning, Afternoon or Evening.

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;
}
```

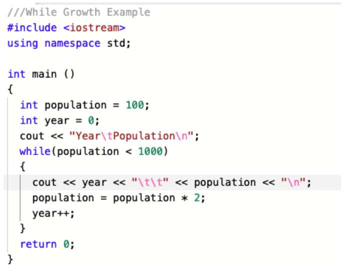
- Recap: I/O & Definite Loops in C++
- Conditionals in C++
- **Indefinite Loops in C++**
- Recap: C++ & Python

Challenge:

Predict what the code will do: https://www.onlinegdb.com/rk86urUlf

```
//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

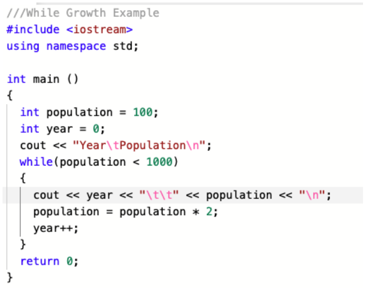


```
///While Growth Example
#include <iostream>
using namespace std;

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

(Demo with `onlinegdb`)

link: `https://www.onlinegdb.com/rk86urUlf`

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\t" << population << "\n";
    population = population * 2;
    year++;
  }
  return 0;
}
```

General format:

`while` (*logical expression*)
{

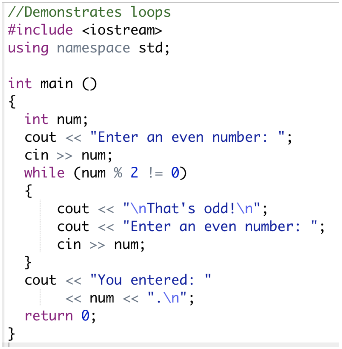
command1;
command2;
command3;
...

}

Challenge: predict what the code do

```
#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

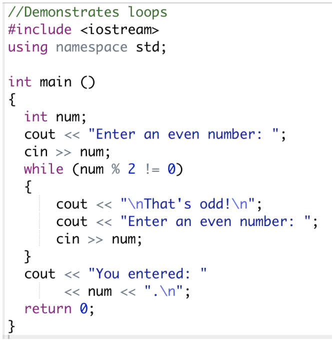


```
//Demonstrates loops
#include <iostream>
using namespace std;

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`)

https://www.onlinegdb.com/rJttLSLgG

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*)
{

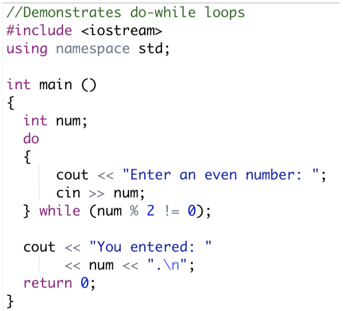
command1;
command2;
command3;
...

}

Challenge: predict what the 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`)

link: https://www.onlinegdb.com/Bkn8DB8eG

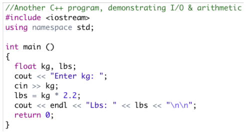
Indefinite Loops: `do-while`

```
//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*);

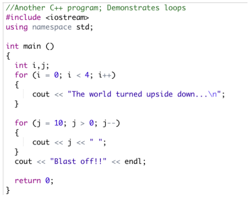
Today's Topics


```
//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**

Recap: C++ Control Structures

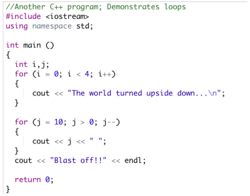
- I/O:



```
//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

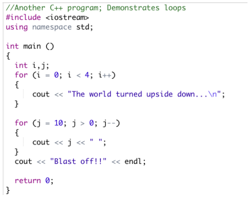
- I/O: `cin >> ...;`



```
//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

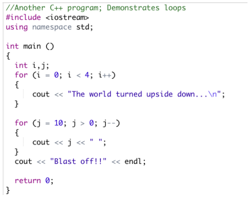
- I/O: `cin >> ...;` & `cout << ...;`



```
//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:

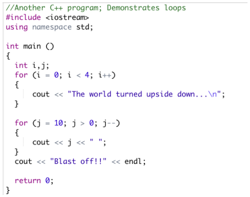


```
//Another C++ program; Demonstrates loops
#include <iostream>
using namespace std;

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

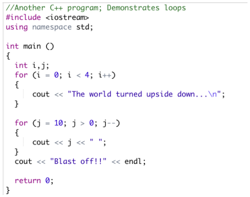
- I/O: `cin >> ...;` & `cout << ...;`
- Definite loops:

```
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;
}
```

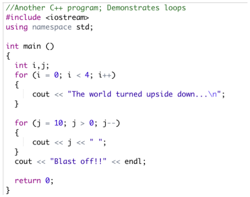
Recap: C++ Control Structures



```
//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:

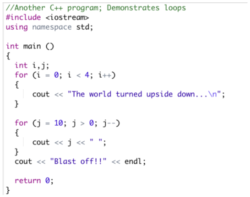
Recap: C++ Control Structures



```
//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
  {
        ...
  }
  ```

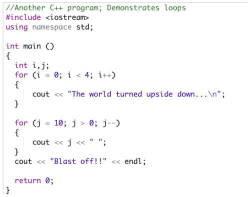
Recap: C++ Control Structures



```
//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*)
  ```
  {
        ...
  }
  ```

Challenge: 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;
}
```

Challenge: Definite Loops in Python & C++

- *Rewrite this program in C++:*

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

Challenge: 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;
```

Challenge: 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()
```

Challenge: 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()
{
```

Challenge: 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)
```

Challenge: 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;
```

Challenge: 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;
}
```

Challenge: 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;
}
```

Challenge: 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):
```

Challenge: Definite Loops in Python & C++

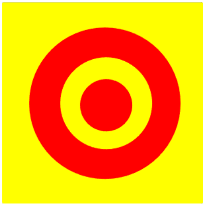
- *Rewrite this program in Python:*

```
#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)
```

Leap Year

A year is a leap year if it is divisible by 4, but century years are not leap years unless they are divisible by 400.



- rectangle represents all the years
- outer red circle means years divided by 4.
- yellow circle means century year, ie, years divided by 100. Every year that is divided by 100 must also be divided by 4, but not vice verse. So yellow circle is completely enclosed in outer red circle.
- inner red circle means years divided by 400. It is completely inside yellow circle.
- red shape: leap year, yellow shape: non-leap year.

1. Red ring represents (year % 4 == 0) and not (year % 100 == 0).
2. Inner red circle represents (year % 400 == 0).
3. Red shape represents ((year % 4 == 0) and not (year % 100 == 0)) or (year % 400 == 0), same as (year % 4 == 0) and (not (year % 100 == 0) or (year % 400 == 0)) by De Morgan's Law.

Challenge: 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 $33.00, and the cost of single ride, with bonus, is $2.75).

Challenge: 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")
```

Challenge: 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")  year = 2016
```

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

Challenge: 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")
```

Challenge: 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")
```

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

Challenge: 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")
```

Challenge: 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")
```

```
year = 2016
if TRUE and \
    (TRUE or (year % 400 == 0)):
     print("Leap!!")
print("Year")
```

Challenge: 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")
```

Challenge: 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")
```

```
year = 2016
if TRUE and \
    (TRUE or FALSE):
     print("Leap!!")
print("Year")
```

Challenge: 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")
```

```
year = 2016
if TRUE and \
    (TRUE or FALSE):
     print("Leap!!")
print("Year")
```

Challenge: 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")
```

```
year = 2016
if TRUE and \
    (TRUE):
     print("Leap!!")
print("Year")
```

Challenge: 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")
```

```
year = 2016
if TRUE:
     print("Leap!!")
print("Year")
```

Challenge: 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")
```

```
year = 2016
if TRUE:
     print("Leap!!")
print("Year")
```

Prints: `Leap!`
`Year`

Challenge: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.75 per ride) or 7-day unlimited card ($33.00).*

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

Challenge: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.75 per ride) or 7-day unlimited card ($33.00).*

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

Challenge: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.75 per ride) or 7-day unlimited card ($33.00).*

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

Challenge: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.75 per ride) or 7-day unlimited card ($33.00).*

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

Challenge: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.75 per ride) or 7-day unlimited card ($33.00).*

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

Challenge: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.75 per ride) or 7-day unlimited card ($33.00).*

```
#include <iostream>
using namespace std;
int main()
{
  int rides;
  cout << "Enter number of rides:";
  cin >> rides;
  if (2.75 * rides < 33.00)
```

Challenge: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.75 per ride) or 7-day unlimited card ($33.00).*

```
#include <iostream>
using namespace std;
int main()
{
  int rides;
  cout << "Enter number of rides:";
  cin >> rides;
  if (2.75 * rides < 33.00)
  {
    cout << "Cheaper to buy single ride metro cards.\n";
  }
```

Challenge: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.75 per ride) or 7-day unlimited card ($33.00).*

```
#include <iostream>
using namespace std;
int main()
{
  int rides;
  cout << "Enter number of rides:";
  cin >> rides;
  if (2.75 * rides < 33.00)
  {
    cout << "Cheaper to buy single ride metro cards.\n";
  }
  else
```

Challenge: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.75 per ride) or 7-day unlimited card ($33.00).*

```
#include <iostream>
using namespace std;
int main()
{
  int rides;
  cout << "Enter number of rides:";
  cin >> rides;
  if (2.75 * rides < 33.00)
  {
    cout << "Cheaper to buy single ride metro cards.\n";
  }
  else
  {
    cout << "Cheaper to buy 7-day unlimited card.\n";
  }
```

Challenge: Conditionals in Python & C++

- *Your program should then print if it is cheaper to buy single ride metro cards ($2.75 per ride) or 7-day unlimited card ($33.00).*

```
#include <iostream>
using namespace std;
int main()
{
  int rides;
  cout << "Enter number of rides:";
  cin >> rides;
  if (2.75 * rides < 33.00)
  {
    cout << "Cheaper to buy single ride metro cards.\n";
  }
  else
  {
    cout << "Cheaper to buy 7-day unlimited card.\n";
  }
  return 0;
}
```

Challenge: 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.

Challenge: Indefinite Loops in Python & C++

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

Challenge: Indefinite Loops in Python & C++

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

```
s = ""
```

Challenge: Indefinite Loops in Python & C++

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

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

Challenge: 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:  ")
```

Challenge: 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)
```

Challenge: 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.

Challenge: 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;
```

Challenge: 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()
```

Challenge: 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;
```

Challenge: 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)
```

Challenge: 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:";
```

Challenge: 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;
```

Challenge: 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;
  }
```

Challenge: Indefinite Loops in Python & C++

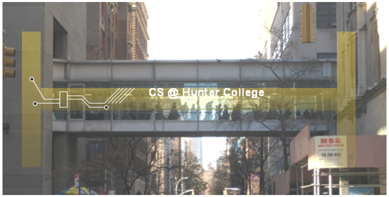
- Write Python code that repeatedly prompts for a non-empty string.

```
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;
}
```

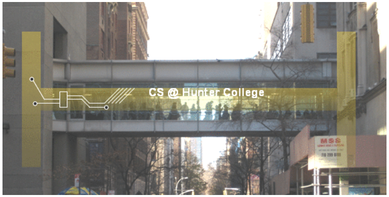
Weekly Reminders!



Before next lecture, don't forget to:

- Work on this week's Online Lab

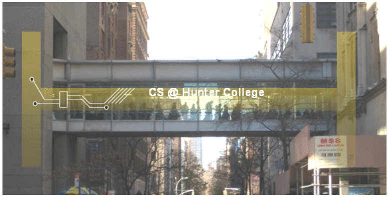
Weekly Reminders!



Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001G Hunter North

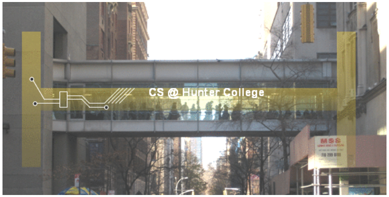
Weekly Reminders!



Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001G Hunter North
- Submit this week's 5 programming assignments **(programs 55-60)**

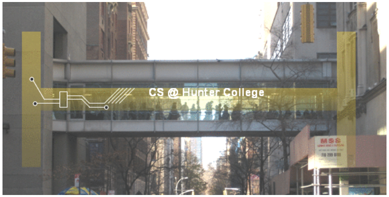
Weekly Reminders!



Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001G Hunter North
- Submit this week's 5 programming assignments **(programs 55-60)**
- If you need help, schedule an appointment for Tutoring in lab 1001G 11:30am-5pm

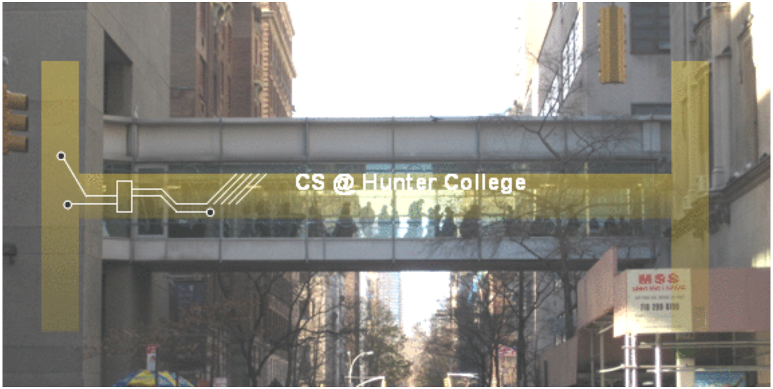
Weekly Reminders!



Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001G Hunter North
- Submit this week's 5 programming assignments **(programs 55-60)**
- If you need help, schedule an appointment for Tutoring in lab 1001G 11:30am-5pm
- Take the Lecture Preview on Blackboard on Monday (or no later than 10:15am on Tuesday)

Lecture Slips & Writing Boards



- Hand your lecture slip to a UTA.
- Return writing boards as you leave.