

CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

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++
- Review: Design & Final Questions

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++

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

```
//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:
`int, float, char, ...`
- To print:

```
//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:
int, float, char, ...
- To print: cout << "Hello!!";

```
//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:
int, float, char, ...
- To print: cout << "Hello!!";
- To get input:

```
//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:
`int, float, char, ...`
- To print: `cout << "Hello!!";`
- To get input: `cin >> 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++

```
//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++

- 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;`

```
//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:
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:

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

```
//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:
`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 '}'.

```
//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:
`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 ';'.

```
//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;
}
```

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++
- Review: Design & Final Questions

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 [onlinedgdb](#))

Conditionals

General format:

```
//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;
}
```

```
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	<i>returns:</i>
False	<code>&&</code>	False	False
False	<code>&&</code>	True	False
True	<code>&&</code>	False	False
True	<code>&&</code>	True	True

Logical Operators in C++

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

and (`&&`)

in1		in2	<i>returns:</i>
False	<code>&&</code>	False	False
False	<code>&&</code>	True	False
True	<code>&&</code>	False	False
True	<code>&&</code>	True	True

or (`||`)

in1		in2	<i>returns:</i>
False	<code> </code>	False	False
False	<code> </code>	True	True
True	<code> </code>	False	True
True	<code> </code>	True	True

Logical Operators in C++

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

and (`&&`)

in1		in2	<i>returns:</i>
False	<code>&&</code>	False	False
False	<code>&&</code>	True	False
True	<code>&&</code>	False	False
True	<code>&&</code>	True	True

or (`||`)

in1		in2	<i>returns:</i>
False	<code> </code>	False	False
False	<code> </code>	True	True
True	<code> </code>	False	True
True	<code> </code>	True	True

not (`!`)

	in1	<i>returns:</i>
!	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++**
- Review: Design & Final Questions

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

```
//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++
- Review: Design & Final Questions

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

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

- Definite loops:

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

- Conditionals:

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

- Conditionals:

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

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

- Conditionals:

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

```
//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;  
}
```

- Indefinite loops:

Recap: C++ Control Structures

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

- Definite loops:

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

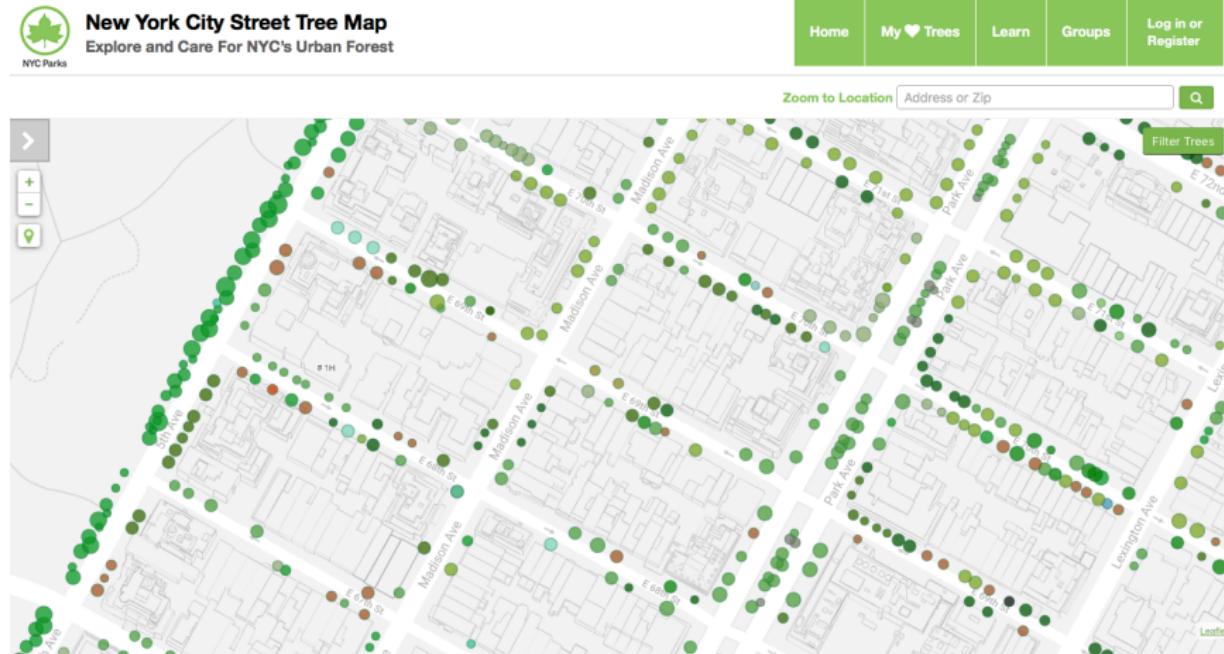
- Conditionals:

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

- Indefinite loops:

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

Design Question: Street Trees



Design an algorithm that prints out all the street trees in your zip code (using NYC Urban Forest OpenData).

Final Review Questions

```
#Name: your name here
#Date: October 2017
#This program, uses functions,
#      says hello to the world!

def main():
    print("Hello, World!")

if __name__ == "__main__":
    main()
```

Plan: Alternate between working in pairs and sketching solutions (until time runs out):

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

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

Final Review Questions

```
#Name: your name here
#Date: October 2017
#This program, uses functions,
#      says hello to the world!

def main():
    print("Hello, World!")

if __name__ == "__main__":
    main()
```

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

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

Plan: Alternate between working in pairs and sketching solutions (until time runs out):

• Definite Loops in Python & C++

Final Review Questions

```
#Name: your name here
#Date: October 2017
#This program, uses functions,
#      says hello to the world!

def main():
    print("Hello, World!")

if __name__ == "__main__":
    main()
```

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

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

Plan: Alternate between working in pairs and sketching solutions (until time runs out):

- Definite Loops in Python & C++
- Conditionals in Python & C++

Final Review Questions

```
#Name: your name here
#Date: October 2017
#This program, uses functions,
#      says hello to the world!

def main():
    print("Hello, World!")

if __name__ == "__main__":
    main()
```

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

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

Plan: Alternate between working in pairs and sketching solutions (until time runs out):

- Definite Loops in Python & C++
- Conditionals in Python & C++
- Indefinite Loops in Python & C++

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

Complete as many as possible:

- *Python: what is the output?*

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

- *C++: what is the output?*

```
int i;
for (i = 2017; i > 2000; i = i - 2)
    cout << "Year is " << i << endl)
```

- In Python, write a complete program that prints out 1 to 100.
- In C++, write a complete program that prints out 1 to 100.

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

- *Python: what is the output?*

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

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

- C++: what is the output?

```
int i;  
for (i = 2017; i > 2000; i = i - 2)  
    cout << "Year is " << i << endl)
```

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

- In Python, write a complete program that prints out 1 to 100.

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

- In C++, write a complete program that prints out 1 to 100.

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!!")
```

- *C++: what is the output?*

```
int i = 13;
if ((i % 2 == 1) && (i % 3 != 0) && !(i % 5 == 0))
    cout << i << " is not divisible!" << endl;
```

- *Write Python code that asks for the hour (24-hour time) and greets them with “Good Morning” if before 12, “Good Afternoon” for 12 but not yet 17, and “Good Evening” otherwise.*
- *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!!")
```

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

- C++: what is the output?

```
int i = 13;  
if ((i % 2 == 1) && (i % 3 != 0) && !(i % 5 == 0))  
    cout << i << " is not divisible!" << endl;
```

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

- Write Python code that asks for the hour (24-hour time) and greets them with “Good Morning” if before 12, “Good Afternoon” for 12 but not yet 17, and “Good Evening” otherwise.

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

- 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: Indefinite Loops in Python & C++

Complete as many as possible:

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

Complete as many as possible:

- *Python: what is the output?*

```
bal = 100
while bal < 200:
    print("Balance", bal)
    bal = bal + 0.1*bal
```

- *C++: what is the output?*

```
int n = 10;
do {
    if (n % 2 == 0)
        n = n / 2;
    else
        n = 3*n + 1;
    cout << "n is " << endl;
} while (n > 1);
```

- 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++

- *Python: what is the output?*

```
bal = 100
while bal < 200:
    print("Balance", bal)
    bal = bal + 0.1*bal
```

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

- C++: what is the output?

```
int n = 10;
do {
    if ( n % 2 == 0)
        n = n / 2;
    else
        n = 3*n + 1;
    cout << "n is " << endl;
} while (n > 1);
```

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

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



- Return writing boards as you leave...