# CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

990

1/42

This lecture will be recorded

Please take a moment to fill out the Teacher Evaluations

CSci 127 (Hunter) Lecture 13 30 November 2021 2 / 42

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

CSci 127 (Hunter) Lecture 13 30 November 2021 2 / 42

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

CSci 127 (Hunter) Lecture 13 30 November 2021 2 / 42

- 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

2/42

CSci 127 (Hunter) Lecture 13

- 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

2 / 42

CSci 127 (Hunter) Lecture 13

- 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

2 / 42

CSci 127 (Hunter) Lecture 13

• Final Exam Monday 20 December

- Final Exam Monday 20 December
- Deadline for choosing Early exam is on December 6
   Submit Early Final Exam Option on Gradescope

CSci 127 (Hunter) Lecture 13 30 November 2021 3 / 42

- Final Exam Monday 20 December
- Deadline for choosing Early exam is on December 6
   Submit Early Final Exam Option on Gradescope
   If you don't submit, we will assume you are taking the exam on 20
   December.

3/42

- Final Exam Monday 20 December
- Deadline for choosing Early exam is on December 6
   Submit Early Final Exam Option on Gradescope
   If you don't submit, we will assume you are taking the exam on 20 December.

If you choose the Early Exam, you CANNOT take the exam on 20 December.

- Final Exam Monday 20 December
- Deadline for choosing Early exam is on December 6
   Submit Early Final Exam Option on Gradescope
   If you don't submit, we will assume you are taking the exam on 20 December.
  - If you choose the Early Exam, you CANNOT take the exam on 20 December.
- NextTuesday 7 December we will have an IN-PERSON Mock Exam

- Final Exam Monday 20 December
- Deadline for choosing Early exam is on December 6
   Submit Early Final Exam Option on Gradescope
   If you don't submit, we will assume you are taking the exam on 20 December.
  - If you choose the Early Exam, you CANNOT take the exam on 20 December.
- NextTuesday 7 December we will have an IN-PERSON Mock Exam
  - Room 118 Hunter North (Assembly Hall), ground floor of the North Building

- Final Exam Monday 20 December
- Deadline for choosing Early exam is on December 6 Submit Early Final Exam Option on Gradescope If you don't submit, we will assume you are taking the exam on 20 December.
  - If you choose the Early Exam, you CANNOT take the exam on 20 December.
- NextTuesday 7 December we will have an IN-PERSON Mock Exam
  - ► Room 118 Hunter North (Assembly Hall), ground floor of the North Building
  - ► You will receive an email with your assigned seating by the end of this. It will be the same for the Mock and for the final on December 20.

- Final Exam Monday 20 December
- Deadline for choosing Early exam is on December 6
   Submit Early Final Exam Option on Gradescope
   If you don't submit, we will assume you are taking the exam on 20 December.
  - If you choose the Early Exam, you CANNOT take the exam on 20 December.
- NextTuesday 7 December we will have an IN-PERSON Mock Exam
  - Room 118 Hunter North (Assembly Hall), ground floor of the North Building
  - ► You will receive an email with your assigned seating by the end of this. It will be the same for the Mock and for the final on December 20.
  - ▶ I will send email when the seating assignments become available.

3/42

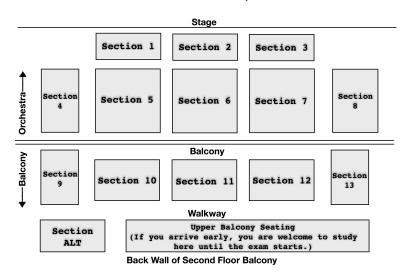
- Final Exam Monday 20 December
- Deadline for choosing Early exam is on December 6
   Submit Early Final Exam Option on Gradescope
   If you don't submit, we will assume you are taking the exam on 20 December.
  - If you choose the Early Exam, you CANNOT take the exam on 20 December.
- NextTuesday 7 December we will have an IN-PERSON Mock Exam
  - Room 118 Hunter North (Assembly Hall), ground floor of the North Building
  - ➤ You will receive an email with your assigned seating by the end of this. It will be the same for the Mock and for the final on December 20.
  - ▶ I will send email when the seating assignments become available.
  - ▶ Only 1.15 hours for the Mock, 2 hours for the real exam.

3/42

- Final Exam Monday 20 December
- Deadline for choosing Early exam is on December 6
   Submit Early Final Exam Option on Gradescope
   If you don't submit, we will assume you are taking the exam on 20 December.
  - If you choose the Early Exam, you CANNOT take the exam on 20 December.
- NextTuesday 7 December we will have an IN-PERSON Mock Exam
  - Room 118 Hunter North (Assembly Hall), ground floor of the North Building
  - ► You will receive an email with your assigned seating by the end of this. It will be the same for the Mock and for the final on December 20.
  - ▶ I will send email when the seating assignments become available.
  - ▶ 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.

◆□▶ ◆□▶ ◆三▶ ◆三▶ ● めぬ◇

### CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

◆□▶ ◆□▶ ◆三▶ ◆三▶ ● めぬ◇

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

CSci 127 (Hunter) Lecture 13 30 November 2021 5 / 42

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.

5 / 42

- 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.

- 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 how
  - 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 Credit/NoCredit?

5 / 42

- What's the best way to study for the final exam? The final exam problems are variations on the homework, guizzes, 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 Credit/NoCredit? It's fine with us, but check with your advisor to make sure it's accepted for your
  - program of study.

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 Credit/NoCredit?
   It's fine with us, but check with your advisor to make sure it's accepted for your program of study.
- Why do you care about cheating?

CSci 127 (Hunter) Lecture 13 30 November 2021 5 / 42

- 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 Credit/NoCredit?
   It's fine with us, but check with your advisor to make sure it's accepted for your program of study.
- Why do you care about cheating?
   First: it gives unfair advantage & is immoral.

CSci 127 (Hunter) Lecture 13 30 November 2021 5 / 42

- 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 Credit/NoCredit?
   It's fine with us, but check with your advisor to make sure it's accepted for your program of study.
- Why do you care about cheating?
   First: it gives unfair advantage & is immoral.
   Second: it degrades the quality of our students.

- 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 Credit/NoCredit?
- It's fine with us, but check with your advisor to make sure it's accepted for your program of study.
- 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.

- 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 Credit/NoCredit?
   It's fine with us, but check with your advisor to make sure it's accepted for your program of study.
- 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 finctude closterous targ numespace std; int main O {
float kg, lbs; one - "three kg;"; lbs = kg '2.2; cost - end ("lbs; "« lbs « "whn"; return 0; }
```

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

# Today's Topics

```
//Acother (-* program, demonstrating I/O & arithmetic finitude closterous targ namespace std; int main () { floot kg, lbs; cost vc. Toner kg; "; lbs = kg "2.2; cost vc. end! oc. "lbs: " « lbs « "\n\n"; return 0; tetum 0;
```

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

```
//Another C++ program, demonstrating I/O & arithmetic
#include <iostream>
using namespace std;
int main ()
  float kg, lbs;
  cout << "Enter kg: ";
  cin >> ka;
  lbs = kq * 2.2;
  cout << endl << "Lbs: " << lbs << "\n\n";
  return 0;
```

Efficient for systems programming.

```
//Another C++ program, demonstrating I/O & arithmetic minclude clostreams using namespose st;
int main ()
{
floot kg, lbs;
cout < "Enter kg: ";
cin >> kg;
lbs = kg * 2.2;
cout << endl << "Lbs: " << lbs << "\n\n";
return 0;
}
```

9 / 42

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

9 / 42

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

```
//Another C++ program, demonstrating I/O & arithmetic finclude clostream-using namespace std; int main () { float kg, lbs: cout <= Enter kg: "; cin >> kg: lbs = kg * 2.2; cout <= entl <= "Lbs: " << lbs <= "\n\n"; return 0; }
```

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

```
//Another C++ program, demonstrating I/O & arithmetic finiclude clostream-
using namespace std;
int main () {
    float kg, lbs;
    cout << "Enter kg:";
    cin >> kg;
    lbs = kg * 2.2;
    cout << end! << "Lbs: " << lbs << "'\n\n";
    return 0;
}
```

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

```
//Another C++ program, demonstrating I/O & arithmetic finitude clostreams using namespace std; 
int main O {
    float kg, lbs; 
    cout <= Enter kg; "; 
    in >> kg; 
    lbs = kg * 2.2; 
    cout <= entl <= "ths: " << 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, ...

```
//Another C++ program, demonstrating I/O & arithmetic
finclude clostream
using namespace std;
int main O {
    float kg, lbs;
    cout << "Enter kg: ";
    cin >> kg;
    lbs = kg " 2.2;
    cout << entl << "there's "</pre>
```

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

```
//Another (++ program, demonstrating I/O & arithmetic
sinclude <lastream
using namespace std;
int main ()
{
    float kg, lbs;
    cout << "Enter kg: ";
    cin > kg;
    classes, 2,2;
    cout << endl << "Lbs: " << lbs << "\n\n";
    return 0;
}</pre>
```

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

```
//Another C++ program, demonstrating I/O & arithmetic
finclude <isotreom
using namespace std;
int main O {
   float kg, lbs;
   cout << "Enter kg: ";
   cin >> kg;
   lbs = kg " 2.2;
   cout << entl << "lbs: " << lbs << "\n\n";
   return 0;</pre>
```

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

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

```
//Another C++ program, demonstrating I/O & arithmetic finclude clostreambusing namespace std; int main O { float kg, lbs; cout << "Enter kg: "; tbs = kg; 2.2; cout << endl << "tbs: " << lbs << "\n\n"; return 6; 's cout << endl << 'tbs: " << lbs << "\n\n"; return 6; 's cout << endl << "tbs: " << lbs << "\n\n"; return 6; 's cout << endl << "tbs: " << lbs << "\n\n"; return 6; 's cout << endl << "tbs: " << lbs << "\n\n"; return 6; 's cout << endl << "tbs: " << lbs << "\n\n"; return 6; 's cout << endl << "tbs: " << lbs << "\n\n"; return 6; 's cout << endl << "tbs: " << lbs << "\n\n"; return 6; 's cout << endl << "tbs: " << lbs << "\n\n"; return 6; 's cout << endl << "tbs: " << lbs << "\n\n"; 's cout << endl << "tbs: " << lbs << "\n\n"; 's cout << endl << "tbs: " << lbs << "\n\n"; 's cout << endl << "tbs: " << lbs << "\n\n"; 's cout << endl << "tbs: " << lbs << "\n\n"; 's cout << endl << "tbs: " << lbs << "\n\n"; 's cout << endl << "tbs: " << lbs << "\n\n"; 's cout << endl << "tbs: " << lbs << "\n\n"; 's cout << endl << "tbs: " << lbs << "\n\n"; 's cout << endl << "tbs: " << endl << "tbs: " << endl << "tbs: " << endl << endl << "tbs: " << endl <<
```

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

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

```
//Another (++ program, demonstrating I/O & arithmetic minclude <iostream using namespace std; int main () { float kg, lbs; cout << "Enter kg: "; cout << enter kg: "; cout <
```

- Efficient for systems programming.
- Programs are organized in functions.
- Must declare variables: int num;
- Many types available: int, float, char, ...
- To print: cout << "Hello!!";</li>
- 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 Finclude 
//Another C++ program, demonstrating I/O & arithmetic using nomespoce std;
int main ()
{
float kg. lbs;
cout << float kg. lbs;
cout
```

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

```
• Efficient for systems programming.
```

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

```
//Another C++ program, demonstrating I/O & arithmetic finclude <tostream using nomespace std; int main O {
    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!!";</li>
- To get input: cin >> num;
- To use those I/O functions: #include <iostream> using namespace std;
- Definite loops:
   for (i = 0; i < 10; i++) {...}</pre>
- Blocks of code uses '{' and '}'.
- Commands generally end in ';'.

# Today's Topics

```
//Acother (** program, demonstrating I/O & arithmetic finitude closteros tdg) int main () { finot kg, lbs; cost « Triter kg; "; cost « Triter kg; "; lbs » kg *2.2; cost « end! « "lbs: " « lbs « "\whi\n"; return 0; return 0; return 0;
```

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

CSci 127 (Hunter)

#### Challenge:

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";</pre>
    else if (yearBorn <= 1964)
        cout << "Baby Boomer":
    else if (yearBorn <= 1984)
        cout << "Generation X";</pre>
    else if (vearBorn <= 2004)
        cout << "Millennial":
    else
        cout << "TBD":
    return 0:
   CSci 127 (Hunter)
```

```
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":</pre>
    string origin = "South Pacific";
    if (winds > 74)
        cout << "Major storm, called a ";</pre>
    if ((origin == "Indian Ocean")
        |/(origin == "South Pacific"))
        cout << "cyclone.\n";</pre>
    else if (origin == "North Pacific")
        cout << "typhoon.\n";</pre>
    else
        cout << "hurricane.\n";</pre>
              4 D > 4 A > 4 B > 4 B >
                                            90 Q
```

30 November 2021

11 / 42

Lecture 13

#### C++ Demo

```
//Demonstrates conditionals
#include <iostream>
usina namespace std:
int main ()
    int vearBorn:
    cout << "Enter year born: ";</pre>
    cin >> yearBorn;
    if (yearBorn < 1946)
        cout << "Greatest Generation";
    else if (yearBorn <= 1964)
        cout << "Baby Boomer";
                                              (Demo with onlinegdb)
    else if (yearBorn <= 1984)
        cout << "Generation X";</pre>
    else if (yearBorn <= 2004)
        cout << "Millennial";</pre>
    else
        cout << "TBD":
    return 0;
```

#### Conditionals

#### General format:

```
if ( logical expression )
//Demonstrates conditionals
#include <iostream>
using namespace std;
                                                command1;
int main ()
   int yearBorn:
                                                ...
   cout << "Enter year born: ";
   cin >> yearBorn;
   if (yearBorn < 1946)
      cout << "Greatest Generation";
                                      else if ( logical expression )
   else if (yearBorn <= 1964)
      cout << "Baby Boomer";
   else if (yearBorn <= 1984)
                                                command1;
      cout << "Generation X":
   else if (yearBorn <= 2004)
      cout << "Millennial":
   else
                                      else
      cout << "TBD":
   return 0;
                                                command1;
                                                ...
```

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

CSci 127 (Hunter) Lecture 13 30 November 2021 14 / 42

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

#### and (&&)

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

CSci 127 (Hunter) Lecture 13 30 November 2021 14 / 42

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	11	False	False
False	11	True	True
True	11	False	True
True	11	True	True

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	11	True	True
True	11	False	True
True	-11	True	True

## not (!)

in1		returns:	
<u>!</u>	False	True	
!	True	False	

#### Lecture Quiz

- Log-in to Gradescope
- Find LECTURE 13 Quiz
- Take the quiz
- You have 3 minutes

15 / 42

CSci 127 (Hunter) Lecture 13 30 November 2021

# Today's Topics

```
//Another C+program, demonstrating I/O & arithmetic Binclude closterous using namespace std; int main O {
floot kg, lbs; coat ~ "Inter kg:";
lbs = kg " 2.2;
coat ~ end! ~ "lbs: " ~ lbs ~ "whn";
return 0;
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:

```
///While Growth Example
#include <iostream>
using namespace std;
int main ()
  int population = 100;
  int year = 0;
  cout << "Year\tPopulation\n";</pre>
  while(population < 1000)</pre>
    cout << year << "\t\t" << population << "\n";
    population = population * 2;
    year++;
  return 0;
```

900

#### C++ Demo

```
///Mhile Growth Example
#include ciostream>
using namespace std;
int main ()
{
   int population = 100;
   int year = 0;
   cout <= "Year+Tepulation\n";
   while(population < 1000)
   {
      cout << year <= "\t\t\" << population <= '\n";
      population = population * 2;
      year++;
   }
   return 0;
}</pre>
```

(Demo with onlinegdb)

## Indefinite Loops: while

```
///white Growth Example
#include <iostream>
using namespace std;
int main ()
{
  int population = 100;
  int year = 0;
  cout << "Year\fopulation\n";
  while(population < 1000)
{
   cout << year << "\t\t" << population << "\n";
   population = population * 2;
   year++;
}
return 0;
}</pre>
```

```
General format:
while ( logical expression )
{
    command1;
    command2;
    command3;
    ...
```

#### Challenge:

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 != \emptyset)
  {
      cout << "\nThat's odd!\n";</pre>
      cout << "Enter an even number: ";</pre>
      cin >> num;
  cout << "You entered: "</pre>
        << num << ".\n";
  return 0;
```

#### C++ Demo

```
//Demonstrates loops
#include <iostream>
using namespace std;
int main ()
  int num;
  cout << "Enter an even number: ";</pre>
  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)

CSci 127 (Hunter)

# Indefinite Loops: while

```
General format:
while ( logical expression )
{
    command1;
    command2;
    command3;
    ...
}
```

## Challenge:

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: ";</pre>
      cin >> num;
  } while (num % 2 != 0);
  cout << "You entered: "
       << num << ".\n";
  return 0;
```

30 November 2021

#### C++ Demo

## Indefinite Loops: do-while

```
General format:

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

# Today's Topics

```
//Acother (-» program, demonstrating I/O & arithmetic finitude closterose std; int main () { | floot kg, lbs; coat < % There kg; "; coat < % There kg; "; lbs = kg; "2.2; coat < endi < "lbs: " < lbs < "\whin"; return 0; return 0; return 0; return 0;
```

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

CSci 127 (Hunter) Lecture 13 30 November 2021 27 / 42

 $I/0: cin >> \ldots;$ 

```
//Another C++ program; Demonstrates loops
Binclude -dostream
using namespace std;
int min O
{
   int i,j;
   for (i = 0; i < 4; i+>)
   {
      cout < "The world turned upside down...\n";
   }
   for (j = 10; j > 0; j--)
   {
      cout < " o " ";
   }
   cout < " o" "!
}
cout < " o" "list off!!" << end!;
   return 0;
}</pre>
```

CSci 127 (Hunter) Lecture 13 30 November 2021 27 / 42

• I/O: cin >> ...; & cout << ...;

27 / 42

CSci 127 (Hunter) Lecture 13 30 November 2021

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

```
//Another C++ program; Demonstrates loops
###Include clostrates
using nomespace std;
int main O;
for (1 = 0; i < 4; i++)
{
    cout << "The world turned upside down...\n";
}
for (3 = 10; j > 0; j--)
{
    cout << "Tile world turned upside down...\n";
}
cout << "Tile world turned upside down...\n";
}
for (3 = 10; j > 0; j--)
{
    cout << "Tile world turned upside down...\n";
}
return 0;
}</pre>
```

27 / 42

CSci 127 (Hunter) Lecture 13 30 November 2021

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

//Another C++ program; Demonstrates loops #include <iostreams using namespace std;

```
int main () {
    Int i,j or (i=0;\ i<4;\ i++)
    Cout << "The world turned upside down...\n";
    for (j=18;\ j>0;\ j--)
    cout << j<< "";
    cout << j and < "";
    cout < < "Blast off!!" << endl;
    return 0;
```

Conditionals:

```
• I/O: cin >> ...; & cout << ...;
Definite loops:
  for (i = 0; i < 10; i++)
  Conditionals:
  if (logical expression)
  else
       ...
```

```
\label{eq:controller} \begin{split} & //Ancother (++) \operatorname{programs} \operatorname{Demonstrates} \operatorname{loops} \\ & \operatorname{Bincluide} \cdot \operatorname{dostrate} \\ & \operatorname{dist} \operatorname{gammapure} \operatorname{Adf} \\ & \operatorname{int} \operatorname{sin} (n) \\ & \operatorname{int} \operatorname{sin} (1) \\ & \operatorname{for} (1=0 \ \mathrm{i} \ \times 4 \ \mathrm{i} \ \mathrm{i} + \mathrm{i} + \mathrm{i}) \\ & \left\{ & \operatorname{cout} \times \text{"The world turned upside down,...\n";} \right. \\ & \operatorname{for} \left( 3 = 18 \ \mathrm{i} \ \mathrm{j} \ > 0 \ \mathrm{j} - \mathrm{i} \right) \\ & \operatorname{for} \left( 3 = 18 \ \mathrm{i} \ \mathrm{j} \ > 0 \ \mathrm{i} \right) \\ & \operatorname{cout} \times \text{"Bisst offf!"} \times \operatorname{endl}; \\ & \operatorname{return} \theta; \end{split}
```

```
• I/O: cin >> ...; & cout << ...;
Definite loops:
  for (i = 0; i < 10; i++)
  Conditionals:
  if (logical expression)
  else
       ...
```

Indefinite loops:

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

for (j = 10; j > 0; j--) { | cout << j << " ":

cout << "Blast off!!" << endl;
return 0;</pre>

cout << "The world turned upside down...\n";

int main ()
{
 int i,j;
 for (i = 0; i < 4; i++)

```
• I/O: cin >> ...; & cout << ...;
Definite loops:
  for (i = 0; i < 10; i++)
  Conditionals:
  if (logical expression)
  else
• Indefinite loops:
  while (logical expression)
```

Fincture risk from the strip of the solution of the solution

//Another C++ program; Demonstrates loops

4 D > 4 A > 4 B > 4 B >

• 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;
}</pre>
```

28 / 42

• Rewrite this program in C++:

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

CSci 127 (Hunter) Lecture 13 30 November 2021 29 / 42

Prewrite this program in C++:

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

#include <iostream>
using namespace std;

CSci 127 (Hunter) Lecture 13 30 November 2021 29 / 42

Prewrite this program in C++:

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

#include <iostream>
using namespace std;

int main()

29 / 42

Print("Year is", i)

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

Rewrite this program in C++:

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

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

• 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)
```

• 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;</pre>
```

• 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;</pre>
  return 0;
```

29 / 42

• Rewrite this program in Python:

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

CSci 127 (Hunter)

• Rewrite this program in Python:

```
#include <iostream>
using namespace std;
int main()
  for (int i = 1; i < 50; i++)
    cout << i << endl;</pre>
 return 0;
for i in range(1, 50):
```

• Rewrite this program in Python:

```
#include <iostream>
using namespace std;
int main()
  for (int i = 1; i < 50; i++)
    cout << i << endl:</pre>
 return 0;
for i in range(1, 50):
    print(i)
```

30 / 42

```
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).

```
Python: what is the output?
year = 2016
if year % 4 == 0 and \
    (not (year % 100 == 0) or (year % 400 == 0)):
    print("Leap!!")
print("Year")
```

32 / 42

CSci 127 (Hunter) Lecture 13 30 November 2021

```
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")
```

```
Python: what is the output?
year = 2016
if year % 4 == 0 and \
    (not (year % 100 == 0) or (year % 400 == 0)):
    print("Leap!!")
print("Year")
```

```
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")
```

```
Python: what is the output?
year = 2016
if year % 4 == 0 and \
    (not (year % 100 == 0) or (year % 400 == 0)):
    print("Leap!!")
print("Year")
```

```
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")
```

```
Python: what is the output?
year = 2016
if year % 4 == 0 and \
    (not (year % 100 == 0) or (year % 400 == 0)):
    print("Leap!!")
print("Year")
```

35 / 42

```
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")
```

```
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")
```

```
Python: what is the output?
  year = 2016
  if year % 4 == 0 and \setminus
      (not (year \% 100 == 0) or (year \% 400 == 0)):
      print("Leap!!")
  print("Year")
  year = 2016
  if TRUE and \
      (TRUE):
      print("Leap!!")
  print("Year")
```

```
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")
```

38 / 42

```
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

38 / 42

• 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;
```

CSci 127 (Hunter) Lecture 13 30 November 2021 39 / 42

• 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()
```

CSci 127 (Hunter) Lecture 13 30 November 2021 39 / 42

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

• 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:";</pre>
```

CSci 127 (Hunter) Lecture 13 30 November 2021 39 / 42

• 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;
```

39 / 42

CSci 127 (Hunter) Lecture 13 30 November 2021

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

```
#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";
   }</pre>
```

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

39 / 42

## 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:";</pre>
  cin >> rides;
  if (2.75 * rides < 33.00)
    cout << "Cheaper to buy single ride metro cards.\n";</pre>
  else
    cout << "Cheaper to buy 7-day unlimited card.\n";</pre>
  return 0;
```

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

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

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

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

s = ""

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

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

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

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

• 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 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.

• 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)
```

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

• 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)
```

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

• 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;
```

41 / 42

• 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)
```

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

• 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)
```

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

• 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)
```

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

• 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;
}
```

41 / 42

• 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:";</pre>
    cin >> num;
  return 0;
```

30 November 2021



Before next lecture, don't forget to:

Work on this week's Online Lab

42 / 42

CSci 127 (Hunter) Lecture 13 30 November 2021



#### Before next lecture, don't forget to:

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

42 / 42

CSci 127 (Hunter) Lecture 13 30 November 2021



### Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001E Hunter North
- If you haven't already, schedule an appointment to take the Code Review (one every two weeks) in lab 1001E Hunter North



### Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001E Hunter North
- If you haven't already, schedule an appointment to take the Code Review (one every two weeks) in lab 1001E Hunter North
- Submit this week's 5 programming assignments (programs 56-60)

42 / 42

CSci 127 (Hunter) Lecture 13 30 November 2021



### Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001E Hunter North
- If you haven't already, schedule an appointment to take the Code Review (one every two weeks) in lab 1001E Hunter North
- Submit this week's 5 programming assignments (programs 56-60)
- If you need help, schedule an appointment for Tutoring in lab 1001E 11am-5pm

42 / 42

CSci 127 (Hunter) Lecture 13 30 November 2021



### Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001E Hunter North
- If you haven't already, schedule an appointment to take the Code Review (one every two weeks) in lab 1001E Hunter North
- Submit this week's 5 programming assignments (programs 56-60)
- If you need help, schedule an appointment for Tutoring in lab 1001E 11am-5pm
- Take the Lecture Preview on Blackboard on Monday (or no later than 10am on Tuesday)



#### Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001E Hunter North
- If you haven't already, schedule an appointment to take the Code Review (one every two weeks) in lab 1001E Hunter North
- Submit this week's 5 programming assignments (programs 56-60)
- If you need help, schedule an appointment for Tutoring in lab 1001E 11am-5pm
- Take the Lecture Preview on Blackboard on Monday (or no later than 10am on Tuesday)
  - Happy Thanksgiving!