CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

• Please take a moment to fill out the Teacher Evaluations

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

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

- 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

- 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

Final Exam May 22 at 9-11 AM

- Final Exam May 22 at 9-11 AM
- Seating assignment will be available on Blackboard/My Grades by this week
- Next Tuesday, May 16, we will have a Mock Exam

- Final Exam May 22 at 9-11 AM
- Seating assignment will be available on Blackboard/My Grades by this week
- Next Tuesday, May 16, we will have a Mock Exam
 - ► Room 118 Hunter North (Assembly Hall), ground floor of the North Building

- Final Exam May 22 at 9-11 AM
- Seating assignment will be available on Blackboard/My Grades by this week
- Next Tuesday, May 16, we will have a Mock Exam
 - Room 118 Hunter North (Assembly Hall), ground floor of the North Building
 - ▶ Only 1 hr 15 mins for the Mock, 2 hours for the real exam.

- Final Exam May 22 at 9-11 AM
- Seating assignment will be available on Blackboard/My Grades by this week
- Next Tuesday, May 16, we will have a Mock Exam
 - Room 118 Hunter North (Assembly Hall), ground floor of the North Building
 - ▶ Only 1 hr 15 mins for the Mock, 2 hours for the real exam.
 - Just a practice run and it will NOT be graded (answer keys will be posted).
 - However, the mock will have the same logistics and question format as the real final exam.

CSci 127 (Hunter) Lecture 13 Ma

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

• 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.

- 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 online. Do 7-10 previous exams: allow 1 hour (half time) and work through, grade yourself, update your 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 online. Do 7-10 previous exams: allow 1 hour (half time) and work through, grade yourself, update your note sheet, and repeat.
- I'm worried about my grade. Should I do Pass/NoCredit?

- 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 online. Do 7-10 previous exams: allow 1 hour (half time) and work through, grade yourself, update your 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.

- 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 online. Do 7.10 provious exams: allow 1 how.
 - Past exams (and answer keys) are online. Do 7-10 previous exams: allow 1 hour (half time) and work through, grade yourself, update your 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.
- Why do you care about cheating?

- 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.
 Part and a proper level and a proper level are a placed in the province of the province
 - Past exams (and answer keys) are online. Do 7-10 previous exams: allow 1 hour (half time) and work through, grade yourself, update your 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.
- Why do you care about cheating?
 First: it gives unfair advantage & is immoral.

- 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.
 Part and control of the province of the
 - Past exams (and answer keys) are online. Do 7-10 previous exams: allow 1 hour (half time) and work through, grade yourself, update your 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.
- 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 online. Do 7-10 previous exams: allow 1 hour (half time) and work through, grade yourself, update your 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.
- 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.

4 / 27

CSci 127 (Hunter) Lecture 13 May 9, 2023

- 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 online. Do 7-10 previous exams: allow 1 hour (half time) and work through, grade yourself, update your 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.
- 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.

4 / 27

CSci 127 (Hunter) Lecture 13 May 9, 2023

For those of you taking CSCI 135 next semester (Fall 2023):

- On the first day of CSCI 135, you will have a graded quiz on the C++ topics covered in CSCI 127.
- This includes all examples covered in the labs and lectures as well as the last 8 homework problems.
- You will have to remember and review all this material during the Summer so you are prepared for the first day of class.

4 D > 4 B > 4 E > 4 B > 4 C

Today's Topics

```
//Acother (** program, demonstrating L/O & arithmetic film(under clostered unity anneapone std) inter main () {
floot May, lbs; cost . "force kg; "; cost . "cost . "force kg; "; cost . "cost . "force kg; "; cost . "cost . "lbs . "whin"; cost . "lbs . "whin"; cost . "lbs . "whin";
```

- Recap: I/O & Definite Loops in C++
- Conditionals in C++
 - Indefinite Loops in C++
- Guest: Prof. Ahearn, Geography

Today's Topics

```
//Acother C+p program, demonstrating I/O & arithmetic finitudes clostered using namespace std; int main () { | floot kg, lbs; cost v. Sorer kg; "; lbs - kg *2.2; cost v. end (* "lbs: " * lbs * "\n\n"; return (), return (), }
```

- Recap: I/O & Definite Loops in C++
- Conditionals in C++
 - Indefinite Loops in C++
- Guest: Prof. Ahearn, Geography

CSci 127 (Hunter)

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

Efficient for systems programming.

```
//Another C++ program, demostrating I/O & arithmetic sinclude cisotrems using namespace std; int main O { floot kg, lbs; cott <= "Enter kg: "; cit >> kg: "; cit >> kg: "2.2; cott <= end! <= "Lbs: " << "bh <= "h\n"; return 0; } }
```

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

```
//Another C++ program, demonstrating I/O & arithmetic finclude cisotreams using namespace std; int main O { floot kg, lbs; cott < "Enter kg: "; cott < "Enter kg: "; cott <> kg: " cott << end! < "Lbs: " << lbs << "\n\n"; return 0; } kg: cott << end! << "Lbs: " << lbs << "\n\n"; return 0; }
```

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

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

```
//Another C++ program, demonstrating I/O & arithmetic finitude clostreams using nomespace std; int main () { float (kg. lbs; cout << "Enter kg: "; cin >> kg; lbs = kg * 2.2; cout << endl << "Lbs: " << "kh\n"; return 0; }
```

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

- 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 cistremb
using namespace std;
int main C)
{
   float kg, lbs;
   cout << "Enter kg: ";
   cin kg;
   cin kg;
   cout << end| << "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:

```
//Another (++ program, demonstrating I/O & arithmetic sinclude <lastrambusing namespace std; 
int main () {
   float kg. 'bs; 
   cot << "Enter kg: "; 
   lb = kg " 2.2; 
   cout << enter could << "I'lbs; " << lb << "\n\n"; 
   return 0; 
}
```

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

```
//Another (++ program, demonstrating I/O & arithmetic
finclude softeem
using namespace std;
int main O
{
    float kg, lbs;
    cout << "Enter kg: ";
    cin >> kg;
    lbs = kg " 2.2;
    cout << end! << "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!!";
- To get input:

```
//Monther C++ program, demonstrating I/O & arithmetic
finclude cistream
using namespace std;
int main O {
   float kg, lbs;
   cout << "Enter kg: ";
   cin >> kg;
   lbs = kg * 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!!";
- To get input: cin >> num;

```
//Another C++ program, demonstrating I/O & orithmetic finclude 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; return 0; 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:

- 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;

- 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:

- 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++) {...}</pre>

9 / 27

```
    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++) {...}</pre>
- Blocks of code uses '{' and '}'.

```
• 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++) {...}</pre>
- Blocks of code uses '{' and '}'.
- Commands generally end in ';'.

Today's Topics

```
//Acother (** program, demonstrating L/O & arithmetic film(tude clostered using namespace std; int main () {

{
flost lag. lbs; coot ** offere sig."; coot ** offere sig."; coo ** offere sig."; coo ** offere sig."; coot *
```

- Recap: I/O & Definite Loops in C++
- Conditionals in C++
 - Indefinite Loops in C++
- Guest: Prof. Ahearn, Geography

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":</pre>
    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 D > 4 D > 4 D >
```

May 9, 2023

11 / 27

Lecture 13

Conditionals

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

General format:

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

return 0;

12 / 27

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

< ロ ト ∢ @ ト ∢ 差 ト → 差 → りへ(^-)

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

and (&&)

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

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

and (&&)

turns:
lse
lse
lse
ue

or (||)

in1		in2	returns:
False	11	False	False
False	\Box	True	True
True	\Box	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	11	False	False
False	11	True	True
True	11	False	True
True	11	True	True

not (!)

	in1	returns:
!	False	True
!	True	False

Lecture Slip

- Write a complete C++ program that prompts the user to enter a time (in 24-hour format) and prints the time of day: morning, afternoon, or evening.
- Assume that afternoon is any time after 12 P.M. (1200), and that the evening is any time after 6 P.M. (1800).

Today's Topics

```
//Another C+program, demonstrating L/O & arithmetic finctude closterose td; int main () {
    flow lay, lbs; cont < first refers kg; '; cot < first refers kg; ' < first refers kg; '; cot < red < "lbs < kg; L2; cot < red < "lbs; ' < lbs < "unin"; return 0; '
```

- Recap: I/O & Definite Loops in C++
- Conditionals in C++
- Indefinite Loops in C++
- Guest: Prof. Ahearn, Geography

Challenge: predict what the code will do

```
#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" << population << "\n";</pre>
     population = population * 2;
     year++;
 return 0;
```

16 / 27

C++ Demo

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

int main ()
{
    int population = 100;
    int year = 0;
    cout << "Year \Population\n";
    white(population < 1000)
    {
        cout << year << "\t\t" << population << "\n";
        population = population * 2;
        year++;
    }
    return 0;
}</pre>
```

(Demo with onlinegdb)

CSci 127 (Hunter)

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;
}</pre>
```

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

Challenge: predict what the code does

```
#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";</pre>
      cout << "Enter an even number: ";</pre>
      cin >> num;
  cout << "You entered: " << num << ".\n";</pre>
 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";</pre>
      cout << "Enter an even number: ":
      cin >> num;
  cout << "You entered: "
      << num << ".\n";
  return 0;
```

(Demo with onlinegdb)

20 / 27

CSci 127 (Hunter) Lecture 13 May 9, 2023

Indefinite Loops: while

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

C++ Demo:

Indefinite Loops: do-while

```
General format:

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

Today's Topics

```
//Acother C+program, demonstrating L/O & arithmetic finitude dosterous unity amenages std; int main () {
from kys. bb; cost < "former kgs"; cost < "lost < "los
```

- Recap: I/O & Definite Loops in C++
- Conditionals in C++
- Indefinite Loops in C++
- Guest: Prof. Ahearn, Geography

CSci 127 (Hunter)



Before the next lecture, don't forget to:

Work on this week's Online Lab



Before the 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



Before the 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 programming assignments



Before the 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 programming assignments
- If you need help, schedule an appointment for Tutoring in lab 1001G



Before the 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 programming assignments
- If you need help, schedule an appointment for Tutoring in lab 1001G
- 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.