# 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

- 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 December 19 at 9-11 AM

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

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

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

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

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

CSci 127: Introduction to Computer Science

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 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 hour (half time) and work through , grade yourself, update 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 on-line. Do 7-10 previous exams: allow 1 hour (half time) and work through , grade yourself, update note sheet, and repeat.
- I'm worried about my grade. Should I do Pass/NoCredit?
   It's fine with us, but check with your advisor to make sure it's accepted for your program of study.

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

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

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

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

Past exams (and answer keys) are on-line. Do 7-10 previous exams: allow 1 hour (half time) and work through , grade yourself, update note sheet, and repeat.

I'm worried about my grade. Should I do Pass/NoCredit? It's fine with us, but check with your advisor to make sure it's accepted for your program of study.

Deadline: Monday, December 5, 2022, 11:59 PM (EST)

Why do you care about cheating?

First: it gives unfair advantage & is immoral.

Second: it degrades the quality of our students.

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

Past exams (and answer keys) are on-line. Do 7-10 previous exams: allow 1 hour (half time) and work through , grade yourself, update note sheet, and repeat.

I'm worried about my grade. Should I do Pass/NoCredit? It's fine with us, but check with your advisor to make sure it's accepted for your program of study.

Deadline: Monday, December 5, 2022, 11:59 PM (EST)

• Why do you care about cheating?

First: it gives unfair advantage & is immoral.

Second: it degrades the quality of our students.

Third: it's a standard question on faculty references.

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

Past exams (and answer keys) are on-line. Do 7-10 previous exams: allow 1 hour (half time) and work through , grade yourself, update note sheet, and repeat.

I'm worried about my grade. Should I do Pass/NoCredit?
 It's fine with us, but check with your advisor to make sure it's accepted for your program of study.

Deadline: Monday, December 5, 2022, 11:59 PM (EST)

• Why do you care about cheating?

First: it gives unfair advantage & is immoral.

Second: it degrades the quality of our students.

Third: it's a standard question on faculty references.

Industry & graduate schools hate it: don't want someone who falsifies work.

5 / 44

CSci 127 (Hunter) Lecture 13 December 6, 2022

# Today's Topics

```
//Amother (-» program, demonstrating I/O & arithmetic Britished constraint of the Imminused Constraint of Immi
```

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

# Today's Topics

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

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

Lecture 13

7 / 44

CSci 127 (Hunter)

```
Recap: Basic Form & I/O in C++
1 //C++ program demonstrating I/O & arithmetic
2 | #include <iostream>
 using namespace std;
4
  int main ()
    float kg, lbs;
    cout << "Enter kg: ";</pre>
    cin >> kg;
    lbs = kg * 2.2;
10
    cout << endl << "Lbs: " << lbs << "\n\n":
11
    return 0;
12
```

Lecture 13

December 6, 2022

CSci 127 (Hunter)

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 Binclude cisotreme using namespace std; int main O { floot kg, lbs; cost < "fnter kg: "; cin > kg; lbs = kg * 2.2; cost < end! < "lbs: " << lbs << "\n\n"; return 0; } }
```

9 / 44

CSci 127 (Hunter) Lecture 13 December 6, 2022

- 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 ka, lbs:
 cout << "Enter kg: ";
 cin >> kg;
 lbs = kg * 2.2;
 cout << endl << "Lbs: " << lbs << "\n\n":
 return 0:
```

9 / 44

- 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 ka, lbs:
 cout << "Enter kg: ";
 cin >> kg;
 lbs = kg * 2.2;
 cout << endl << "Lbs: " << lbs << "\n\n":
 return 0:
```

9 / 44

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

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

9 / 44

CSci 127 (Hunter) Lecture 13 December 6, 2022

- 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 cistremp
using namespace std;
int main O {
   float kg, lbs;
   cout << Enter kg; ";
   cth = 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:

```
//Monther C++ program, demonstrating I/O & arithmetic
finclude cistream
using namespace std;
int main ()
{
    float kg.lb;
    cout <= "Enter kg:";
    cin > kg;
    lb= 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!!";</li>

```
//Monother C++ program, demonstrating I/O & arithmetic
finclude cistremp
using namespace std;
int main O {
   floot kg.lb;
   coot <= "Enter kg: ";
   coot <= "Enter kg: ";
   lb = kg. 2, 2;
   coot <= 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!!";</li>
- To get input:

```
//Another (++ program, demonstrating I/O & orithmetic
finclude <isotremn
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;</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;

```
//Another C++ program, demonstrating I/O & orithmetic finitude clostreams using nomespace std; int moin () {
    float kg, lbs; cout << "Enter kg: "; cb = kg, 2.2; cout << endl << "Elbs: " << lb> << "Vn\n"; return %; " </td>
```

- 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;

```
    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 <id>stream
using nonespace std;
int main ()
{
    cout < "Enter kg: ";
    cout < "Enter kg: ";
    bs = kg: 2.2;
    cout << end! << "tbs: " << 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: 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
#include <iostream>
using namespace std;
int main ()
 float ka, lbs:
 cout << "Enter kg: ";
 cin >> kg;
 lbs = kg * 2.2;
 cout << endl << "Lbs: " << lbs << "\n\n":
```

```
    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  $(i = 0; i < 10; i++) {...}$
- Blocks of code uses '{' and '}'.
- Commands generally end in ';'.

# 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++
- Recap: C++ & Python

### 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 A > 4 B > 4 B >
                                            100 Q
```

December 6, 2022

11 / 44

Lecture 13

```
//C++ program demonstrates conditionals
     #include <iostream>
     using namespace std;
     int main ()
6
         int yearBorn;
         cout << "Enter year born: ";
         cin >> yearBorn;
10
         if (yearBorn < 1946)
11
             cout << "Greatest Generation":</pre>
13
14
         else if (yearBorn <= 1964)
16
             cout << "Baby Boomer":
17
         else if (vearBorn <= 1984)
19
20
             cout << "Generation X";</pre>
21
         else if (vearBorn <= 2004)
23
             cout << "Millennial";</pre>
         else
27
28
             cout << "TBD";
31
         return 0;
32
```

### 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 December 6, 2022 14 / 44

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 December 6, 2022 14 / 44

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

### and (&&)

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

### or (||)

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

not (!)

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

### Lecture Slip

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

CSci 127 (Hunter) Lecture 13 December 6, 2022 15 / 44

# Today's Topics

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

CSci 127 (Hunter)

### Challenge:

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

```
//While Growth example
   #include <iostream>
   using namespace std;
4
   int main ()
5
6
     int population = 100;
7
     int year = 0;
8
     cout << "Year\tPopulation\n";</pre>
9
     while (population < 1000)</pre>
10
11
          cout << year << "\t" << population << "\n";</pre>
12
          population = population * 2;
13
14
     return 0;
15
16
     CSci 127 (Hunter)
                                 Lecture 13
                                                       December 6, 2022
                                                                    17 / 44
```

### C++ Demo

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

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

### Indefinite Loops: while

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

int main () {
   int population = 100;
   int year = 0;
   cout << "year\Population\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 do

```
#include <iostream>
   using namespace std;
3
   int main ()
      int num;
      cout << "Enter an even number: ";</pre>
      cin >> num;
      while (num % 2 != 0)
      {
10
          cout << "\nThat's odd!\n";</pre>
11
          cout << "Enter an even number: ";</pre>
12
          cin >> num;
13
14
      cout << "You entered: " << num << ".\n";</pre>
15
      return 0;
16
17
      CSci 127 (Hunter)
                                   Lecture 13
                                                           December 6, 2022
                                                                         20 / 44
```

### C++ Demo

https://www.onlinegdb.com/rJttLSLgG

CSci 127 (Hunter)

# 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;
4
   int main ()
6
     int num;
7
     do
         cout << "Enter an even number: ";</pre>
10
         cin >> num:
11
     } while (num % 2 != 0);
12
13
     cout << "You entered: " << num << ".\n";</pre>
14
     return 0;
15
16
```

### 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 % Z != 0);

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

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

CSci 127 (Hunter) Lecture 13

24 / 44

### 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 ammespace std; int main () {
    floot kg. lbs; coot x < "finer kg."; coot x < month x < too x < to x < to
```

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

# ${\sf Recap:}\ C++\ {\sf Control\ Structures}$

I/O:

```
//Arother C+s program; Demonstrates loops finclude clostered include clostered inclu
```

CSci 127 (Hunter) Lecture 13 December 6, 2022 27 / 44

I/O: cin >> ...;

```
//Another C++ program; formstrutes loops functioned control minimum care in minimum care in minimum care in the minimum care i
```

CSci 127 (Hunter) Lecture 13 December 6, 2022 27 / 44

• I/O: cin >> ...; & cout << ...;

CSci 127 (Hunter) Lecture 13 December 6, 2022 27 / 44

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

```
//Another C+p program; Demonstrates loops
Binclude -dostream
unity namespace xtd;
int main O;
int i,j;
for (j = 0; i < 4; i++)
{
    cout << "The world turned upside down...\n";
}
for (j = 10; j > 0; j--)
{
    cout << j << "";
}
cout << "Blast offil" << endl;
return 0;</pre>
```

CSci 127 (Hunter)

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

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

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

```
int main ()  \begin{cases} &\text{int } i,j;\\ &\text{int } i,j;\\ &\text{for } (i=0;\ i=4;\ i++)\\ &\text{cout} << \text{'The world turned upside down...} \text{'n'};\\ &\text{for } (j=10;\ j>0;\ j--)\\ &\text{cout} << j << \text{''} \text{''};\\ &\text{cout} << j << \text{''} \text{''};\\ &\text{cout} << \text{'Blast off!}!' << \text{end1};\\ &\text{return } 0;\\ \end{cases}
```

Conditionals:

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

```
//Monther C++ programs; Demonstrates loops
sinclude clostreame
using nomespace std;
int main O)
{
    tit, i;
    for (i = 0; i < 4; i+-)
    {
        cout << "The world turned upside down...\n";
    }
    for (j = 10; j > 0; j--)
    {
        cout << "Blost off!!" << endl;
        return 0;
        return 0;
```

```
Definite loops:
                                                        for (i = 0; i < 10; i++)
                                                                  ...
                                                    Conditionals:
//Another C++ program; Demonstrates loops
#include <iostream>
using namespace std;
                                                        if (logical expression)
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 << " ":
                                                        else
 cout << "Blast off!!" << endl:
 return 0;
```

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

• Indefinite loops:

#### Recap: C++ Control Structures

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

```
Finctude -tostream
using namespose std;

int main C)

{
    tnt i,j: 0; i < 4; i+>
    for (i = 10; j > 0; j ->
    }

    for (j = 10; j > 0; j ->
    cout << 'Time world turned upside down...\n";

} cout << 'Time world turned upside down...\n";

cout << 'Tim
```

//Another C++ program; Demonstrates loops

return 0;

• 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>
```

Rewrite this program in C++:

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

Print("Year is", i)

#include <iostream>
using namespace std;
#rear is namespace std;

29 / 44

CSci 127 (Hunter) Lecture 13 December 6, 2022

Print("Year is", i)

\*\*include <iostream>
\*\*Test in C++:

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

#include <iostream>

using namespace std;

int main()

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()

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;
```

• 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>
```

• 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):
```

30 / 44

• 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 / 44

#### Leap Year

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

- 0
- rectangle represents all the years
- outer red circle means years divided by 4.
- yellow circle means century year, ie, years divided by 100. Every year that is divided by 100 must also be divided by 4, but not vice verse. So yellow circle is completely enclosed in outer red circle.
- inner red circle means years divided by 400. It is completely inside yellow circle.
- red shape: leap year, yellow shape: non-leap year.
- ① Red ring represents (year % 4 == 0) and not (year % 100 == 0).
- ② Inner red circle represents (year % 400 == 0).
- 3 Red shape represents ((year % 4 == 0) and not (year % 100 == 0)) or (year % 400 == 0), same as (year % 4 == 0) and ( not (year % 100 == 0) or (year % 400 == 0) ) by De Morgan's Law.

```
• 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")
```

• 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")
```

```
• 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 \frac{100}{100} == 0) or (year \frac{100}{100} == 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 \frac{100}{100} == 0) or (year \frac{100}{100} == 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")
```

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

Prints: Leap! Year

• 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;
```

• 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()
```

• 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;
```

• 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>
```

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

```
#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
```

```
#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";
```

CSci 127 (Hunter)

• 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";
  return 0;
```

Lecture 13

December 6, 2022

40 / 44

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.

```
9 = 1111
```

• 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)
```

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

```
#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;
```

42 / 44

• 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)
```

```
#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.

Lecture 13

December 6, 2022

42 / 44

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

CSci 127 (Hunter)



Before next lecture, don't forget to:

Work on this week's Online Lab



Before next lecture, don't forget to:

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



Before next lecture, don't forget to:

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



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

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



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

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

# Lecture Slips & Writing Boards



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

44 / 44

CSci 127 (Hunter) Lecture 13