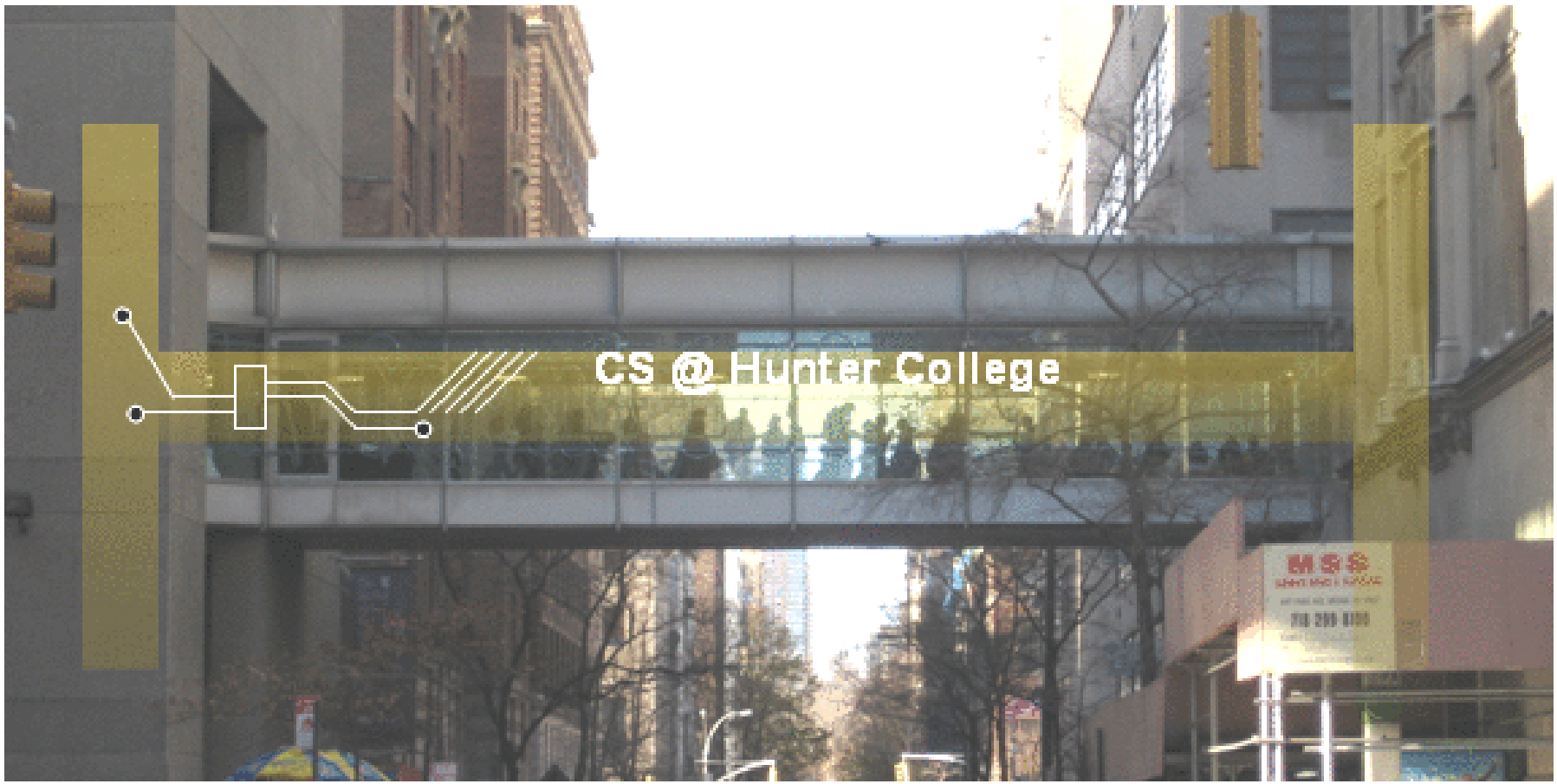
CSci 127: Introduction to Computer Science



Finished the lecture preview?

hunter.cuny.edu/csci

Announcement: Academic Dishonesty

Hunter College regards acts of academic dishonesty (e.g., plagiarism, cheating on examinations, obtaining unfair advantage, and falsification of records and official documents) as serious offenses against the values of intellectual honesty. The College is committed to enforcing the CUNY Policy on Academic Integrity and will pursue cases of academic dishonesty according to the Hunter College Academic Integrity Procedures.

Announcement: Academic Dishonesty

Hunter College regards acts of academic dishonesty (e.g., plagiarism, cheating on examinations, obtaining unfair advantage, and falsification of records and official documents) as serious offenses against the values of intellectual honesty. The College is committed to enforcing the CUNY Policy on Academic Integrity and will pursue cases of academic dishonesty according to the Hunter College Academic Integrity Procedures.

- **All instances of academic dishonesty will be reported to the office of student affairs.**

Announcement: Academic Dishonesty

Hunter College regards acts of academic dishonesty (e.g., plagiarism, cheating on examinations, obtaining unfair advantage, and falsification of records and official documents) as serious offenses against the values of intellectual honesty. The College is committed to enforcing the CUNY Policy on Academic Integrity and will pursue cases of academic dishonesty according to the Hunter College Academic Integrity Procedures.

- **All instances of academic dishonesty will be reported to the office of student affairs.**

Frequently Asked Questions

From email

Frequently Asked Questions

From email

- Can we do more on colors, images, numpy & matplotlib?

Frequently Asked Questions

From email

- Can we do more on colors, images, numpy & matplotlib?
  Yes, we will in Labs 6-9 & Lectures 6-9.
  Today, we'll focus on decisions, and logical expressions & circuits.

Frequently Asked Questions

From email

- Can we do more on colors, images, numpy & matplotlib?
  Yes, we will in Labs 6-9 & Lectures 6-9.
  Today, we'll focus on decisions, and logical expressions & circuits.
- What is pseudocode? Why do we use it?

Frequently Asked Questions

From email

- Can we do more on colors, images, numpy & matplotlib?
  Yes, we will in Labs 6-9 & Lectures 6-9.
  Today, we'll focus on decisions, and logical expressions & circuits.
- What is pseudocode? Why do we use it?
  Pseudocode is the "informal high-level description of the operating principle of a computer program or other algorithm."

Frequently Asked Questions

From email

- Can we do more on colors, images, numpy & matplotlib?
  Yes, we will in Labs 6-9 & Lectures 6-9.
  Today, we'll focus on decisions, and logical expressions & circuits.
- What is pseudocode? Why do we use it?
  Pseudocode is the "informal high-level description of the operating principle of a computer program or other algorithm."
  We use it to write down the ideas, before getting deep into the details.

Frequently Asked Questions

From email

- Can we do more on colors, images, numpy & matplotlib?
  Yes, we will in Labs 6-9 & Lectures 6-9.
  Today, we'll focus on decisions, and logical expressions & circuits.
- What is pseudocode? Why do we use it?
  Pseudocode is the "informal high-level description of the operating principle of a computer program or other algorithm."
  We use it to write down the ideas, before getting deep into the details.
- How can I tell strings from variables?

Frequently Asked Questions

From email

- Can we do more on colors, images, numpy & matplotlib?
  Yes, we will in Labs 6-9 & Lectures 6-9.
  Today, we'll focus on decisions, and logical expressions & circuits.
- What is pseudocode? Why do we use it?
  Pseudocode is the "informal high-level description of the operating principle of a computer program or other algorithm."
  We use it to write down the ideas, before getting deep into the details.
- How can I tell strings from variables?
  Strings are surrounded by quotes (either single or double).

Frequently Asked Questions

From email

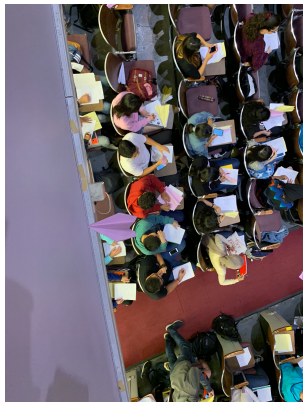
- Can we do more on colors, images, numpy & matplotlib?
  Yes, we will in Labs 6-9 & Lectures 6-9.
  Today, we'll focus on decisions, and logical expressions & circuits.
- What is pseudocode? Why do we use it?
  Pseudocode is the "informal high-level description of the operating principle of a computer program or other algorithm."
  We use it to write down the ideas, before getting deep into the details.
- How can I tell strings from variables?
  Strings are surrounded by quotes (either single or double).
  Variables names (identifiers) for memory locations are not.

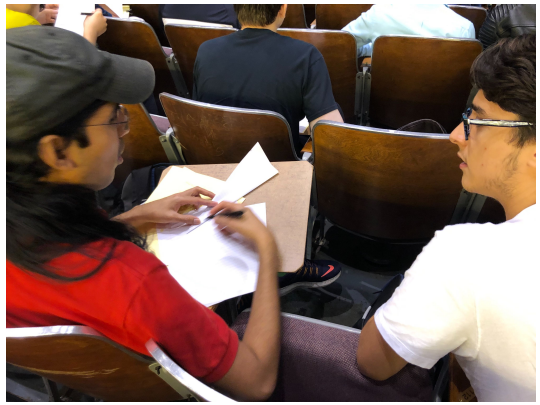
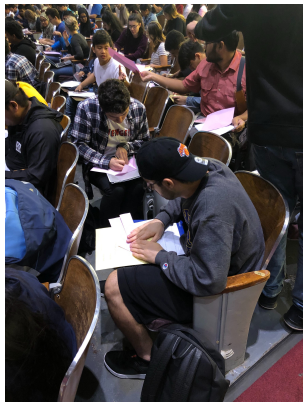
Frequently Asked Questions

From email

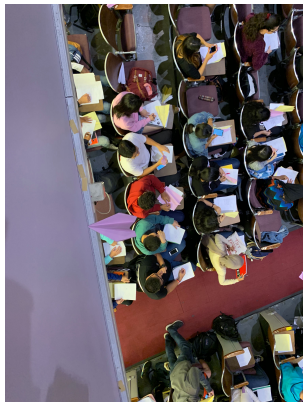
- Can we do more on colors, images, numpy & matplotlib?
  Yes, we will in Labs 6-9 & Lectures 6-9.
  Today, we'll focus on decisions, and logical expressions & circuits.
- What is pseudocode? Why do we use it?
  Pseudocode is the "informal high-level description of the operating principle of a computer program or other algorithm."
  We use it to write down the ideas, before getting deep into the details.
- How can I tell strings from variables?
  Strings are surrounded by quotes (either single or double).
  Variables names (identifiers) for memory locations are not. Ex: `'num'` *vs.* `num`*.*

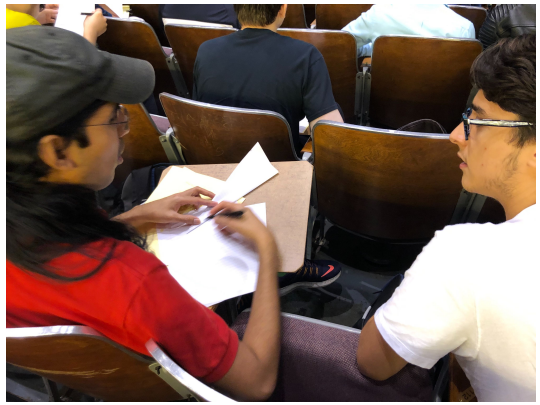
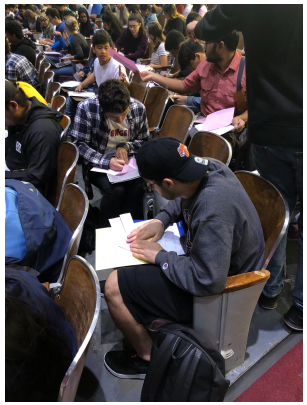
One More FAQ: Why Paper Planes?



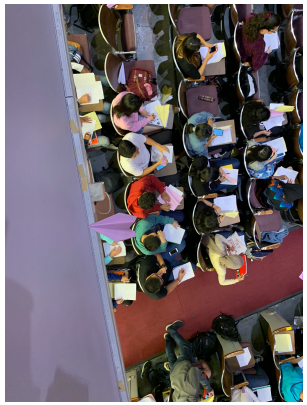
One More FAQ: Why Paper Planes?

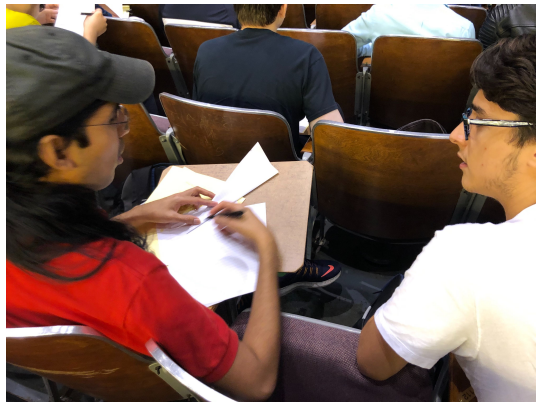
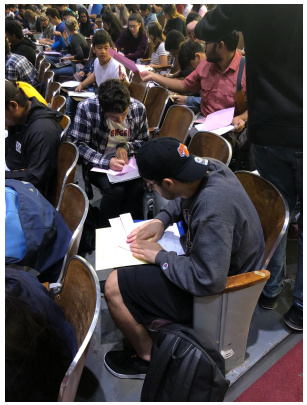


Why paper planes?

One More FAQ: Why Paper Planes?

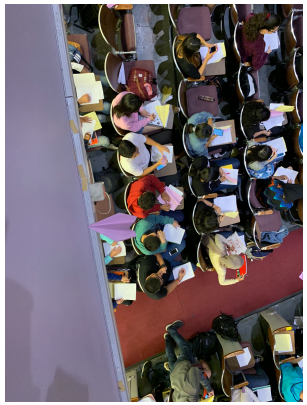


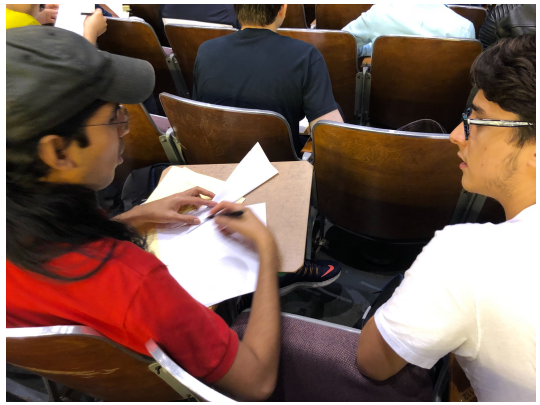
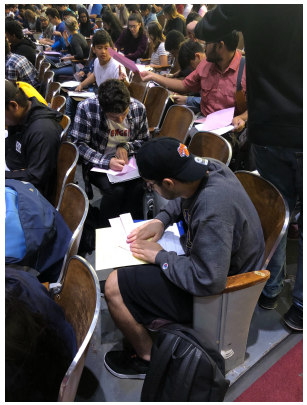



Why paper planes?

- It's a classic design question in introductory programming classes, since

One More FAQ: Why Paper Planes?

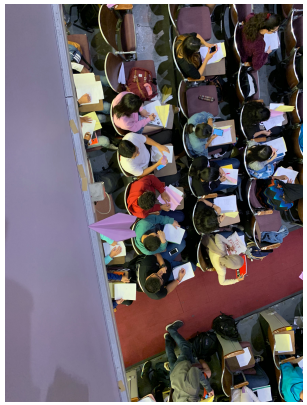


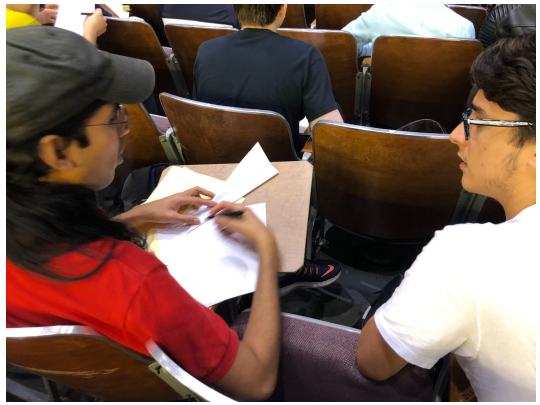
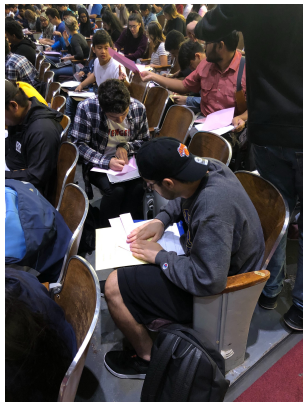



Why paper planes?

- It's a classic design question in introductory programming classes, since
  - Practice writing solutions in plain English (pseudocode) without worrying about syntax.

One More FAQ: Why Paper Planes?

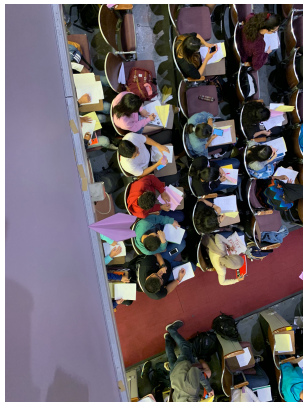


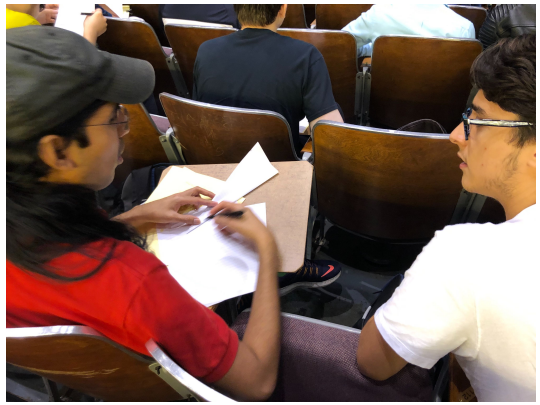
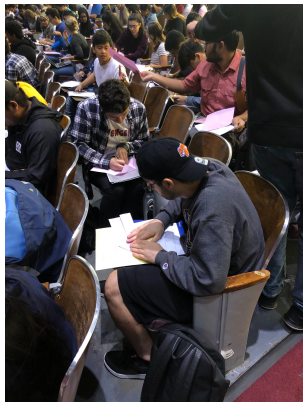



Why paper planes?

- It's a classic design question in introductory programming classes, since
  - Practice writing solutions in plain English (pseudocode) without worrying about syntax.
  - Practice thinking (and writing) precisely.

One More FAQ: Why Paper Planes?

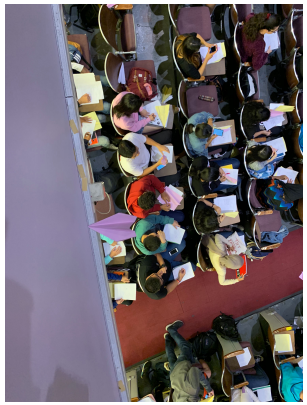


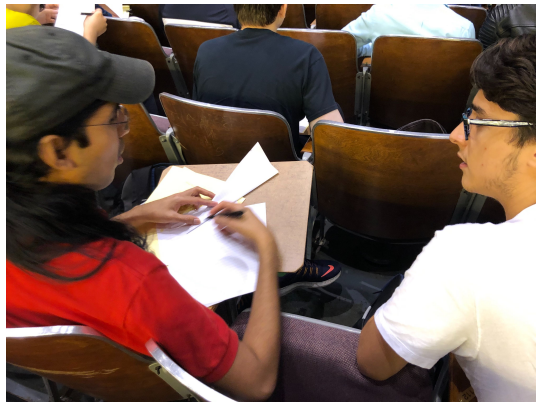
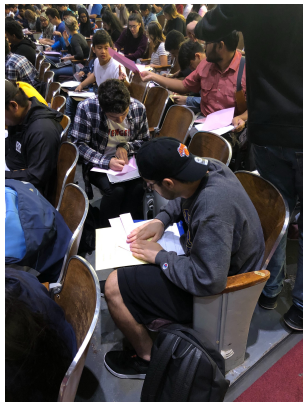



Why paper planes?

- It's a classic design question in introductory programming classes, since
  - Practice writing solutions in plain English (pseudocode) without worrying about syntax.
  - Practice thinking (and writing) precisely.
  - Practice testing and revising your algorithm.

One More FAQ: Why Paper Planes?

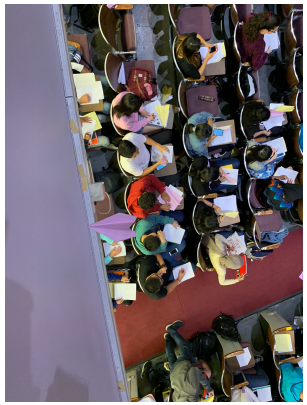


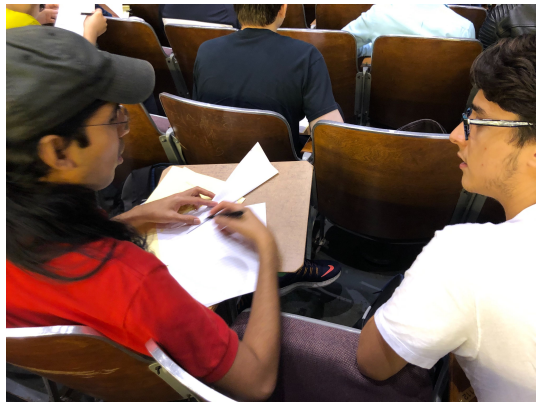
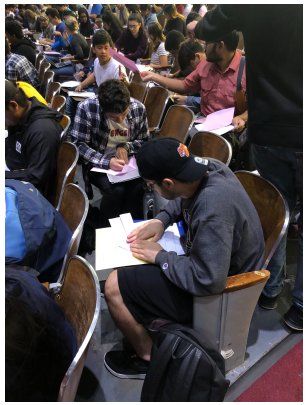



Why paper planes?

- It's a classic design question in introductory programming classes, since
  - Practice writing solutions in plain English (pseudocode) without worrying about syntax.
  - Practice thinking (and writing) precisely.
  - Practice testing and revising your algorithm.
- Why in groups?

One More FAQ: Why Paper Planes?

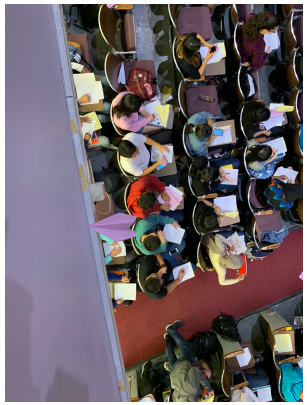


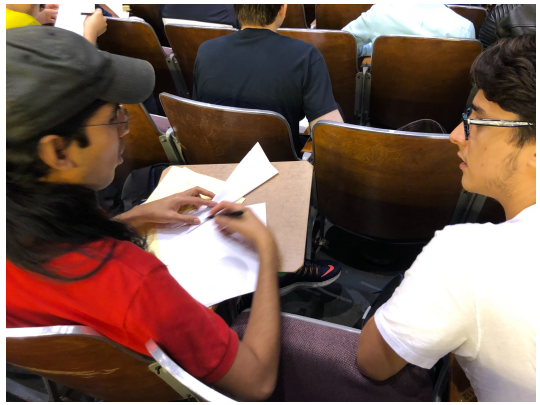
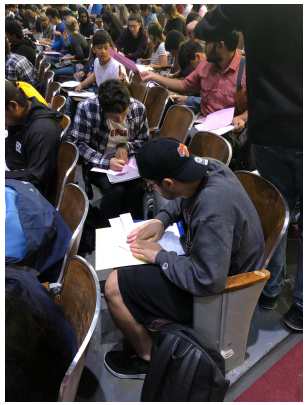



Why paper planes?

- It's a classic design question in introductory programming classes, since
  - Practice writing solutions in plain English (pseudocode) without worrying about syntax.
  - Practice thinking (and writing) precisely.
  - Practice testing and revising your algorithm.
- Why in groups?
  - Improves mastery of material.

One More FAQ: Why Paper Planes?

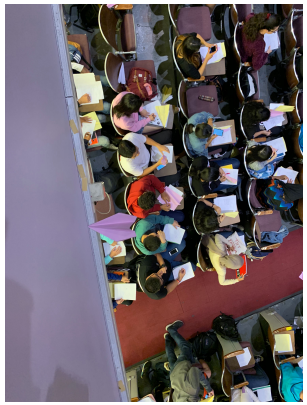


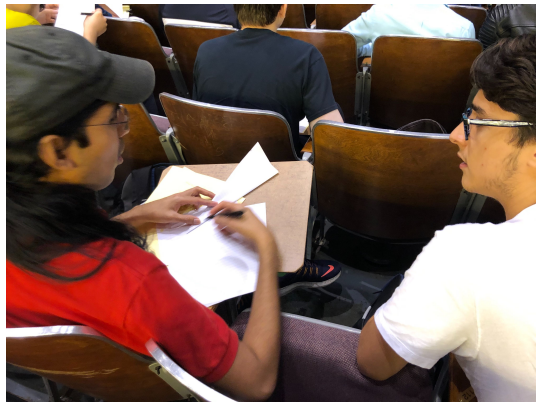
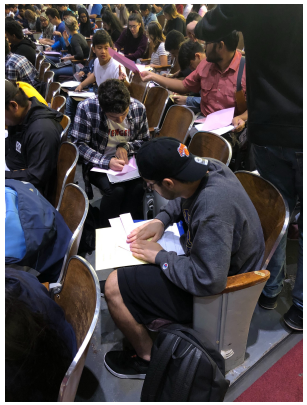



Why paper planes?

- It's a classic design question in introductory programming classes, since
  - Practice writing solutions in plain English (pseudocode) without worrying about syntax.
  - Practice thinking (and writing) precisely.
  - Practice testing and revising your algorithm.
- Why in groups?
  - Improves mastery of material.
  - Our industry partners want strong communication skills:

One More FAQ: Why Paper Planes?

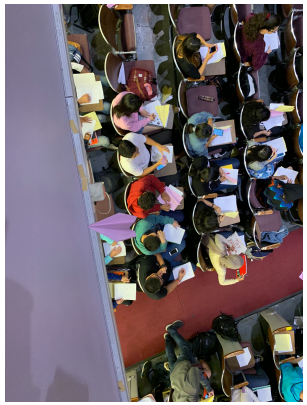


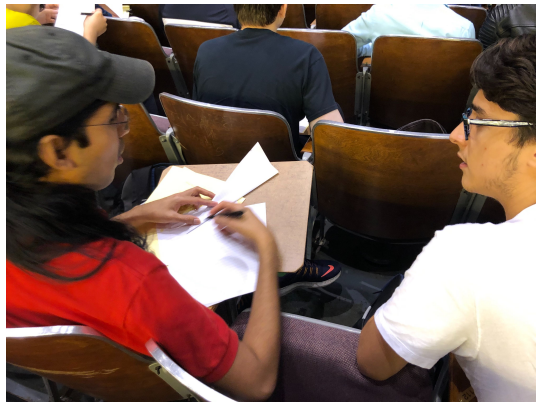
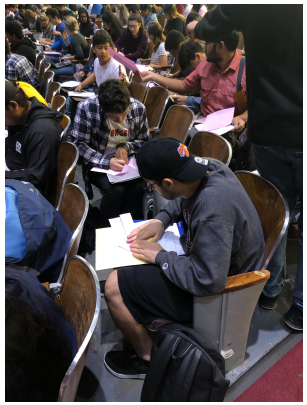



Why paper planes?

- It's a classic design question in introductory programming classes, since
  - Practice writing solutions in plain English (pseudocode) without worrying about syntax.
  - Practice thinking (and writing) precisely.
  - Practice testing and revising your algorithm.
- Why in groups?
  - Improves mastery of material.
  - Our industry partners want strong communication skills:
    - ★ communicating technical ideas precisely, and

One More FAQ: Why Paper Planes?



Why paper planes?

- It's a classic design question in introductory programming classes, since
  - Practice writing solutions in plain English (pseudocode) without worrying about syntax.
  - Practice thinking (and writing) precisely.
  - Practice testing and revising your algorithm.
- Why in groups?
  - Improves mastery of material.
  - Our industry partners want strong communication skills:
    - ★ communicating technical ideas precisely, and
    - ★ communicating and working in teams.

Plane Winners

Come claim your prizes after lecture:

Team:
Ahmed
Bahri Berk Gulay + 1
Caroline, Bryan, Shuming
Iftekar Hossain
Richard Ng
(unmatched)

Today's Topics

- Recap: Decisions
- Logical Expressions
- Circuits
- Binary Numbers

Today's Topics

- **Recap: Decisions**
- Logical Expressions
- Circuits
- Binary Numbers

Challenge

Some challenges with types & decisions:

```
#What are the types:
y1 = 2017
y2 = "2018"
print(type(y1))
print(type("y1"))
print(type(2017))
print(type("2017"))
print(type(y2))
print(type(y1/4.0))

x = int(y2) - y1
if x < 0:
    print(y2)
else:
    print(y1)
```

```
cents = 432
dollars = cents // 100
change = cents % 100
if dollars > 0:
    print('$'+str(dollars))
if change > 0:
    quarters = change // 25
    pennies = change % 25
    print(quarters, "quarters")
    print("and", pennies, "pennies")
```

Python Tutor

```
#What are the types:
y1 = 2017
y2 = "2018"
print(type(y1))
print(type("y1"))
print(type(2017))
print(type("2017"))
print(type(y2))
print(type(y1/4.0))

x = int(y2) - y1
if x < 0:
    print(y2)
else:
    print(y1)
```

(Demo with pythonTutor)

Decisions

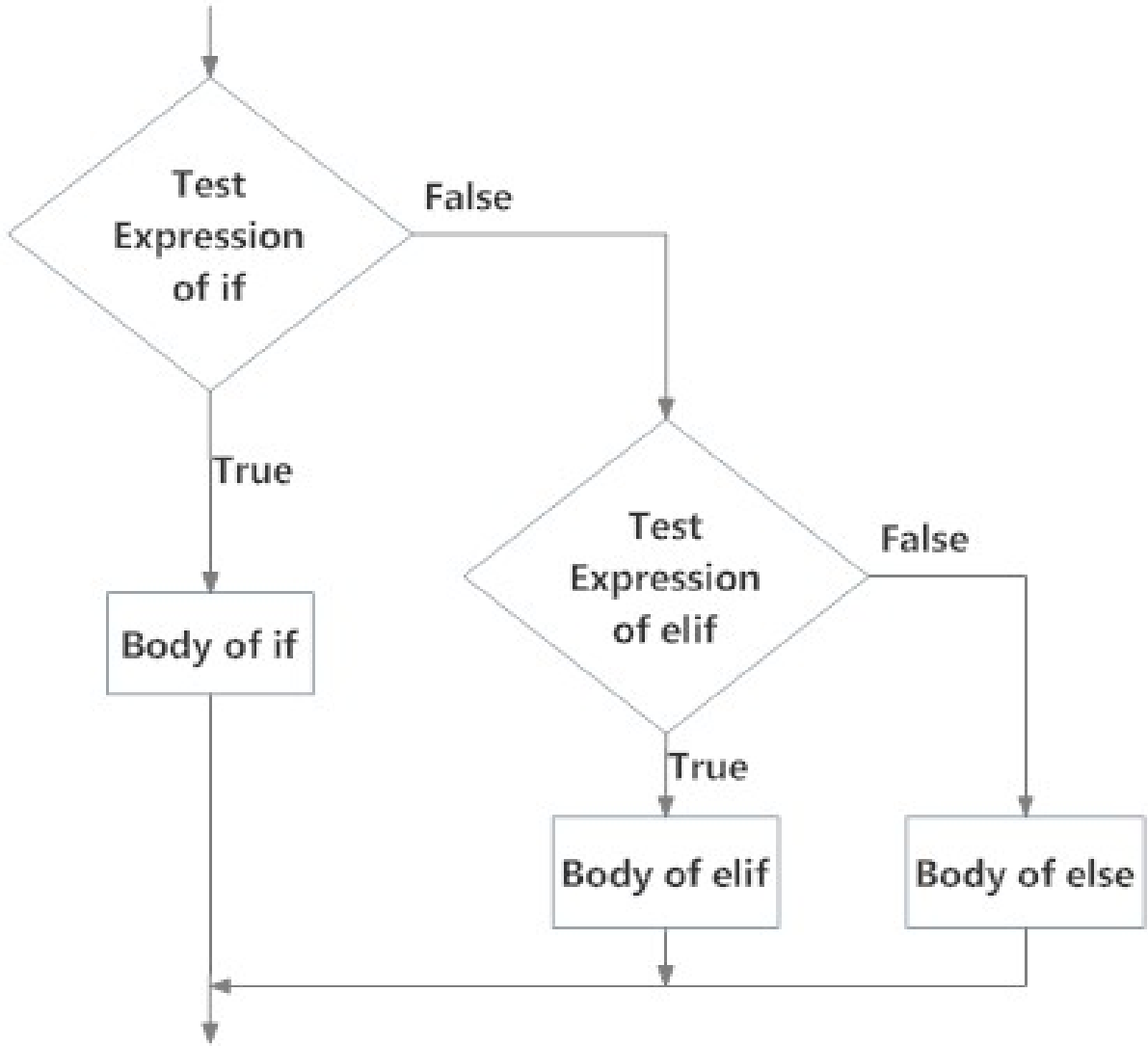
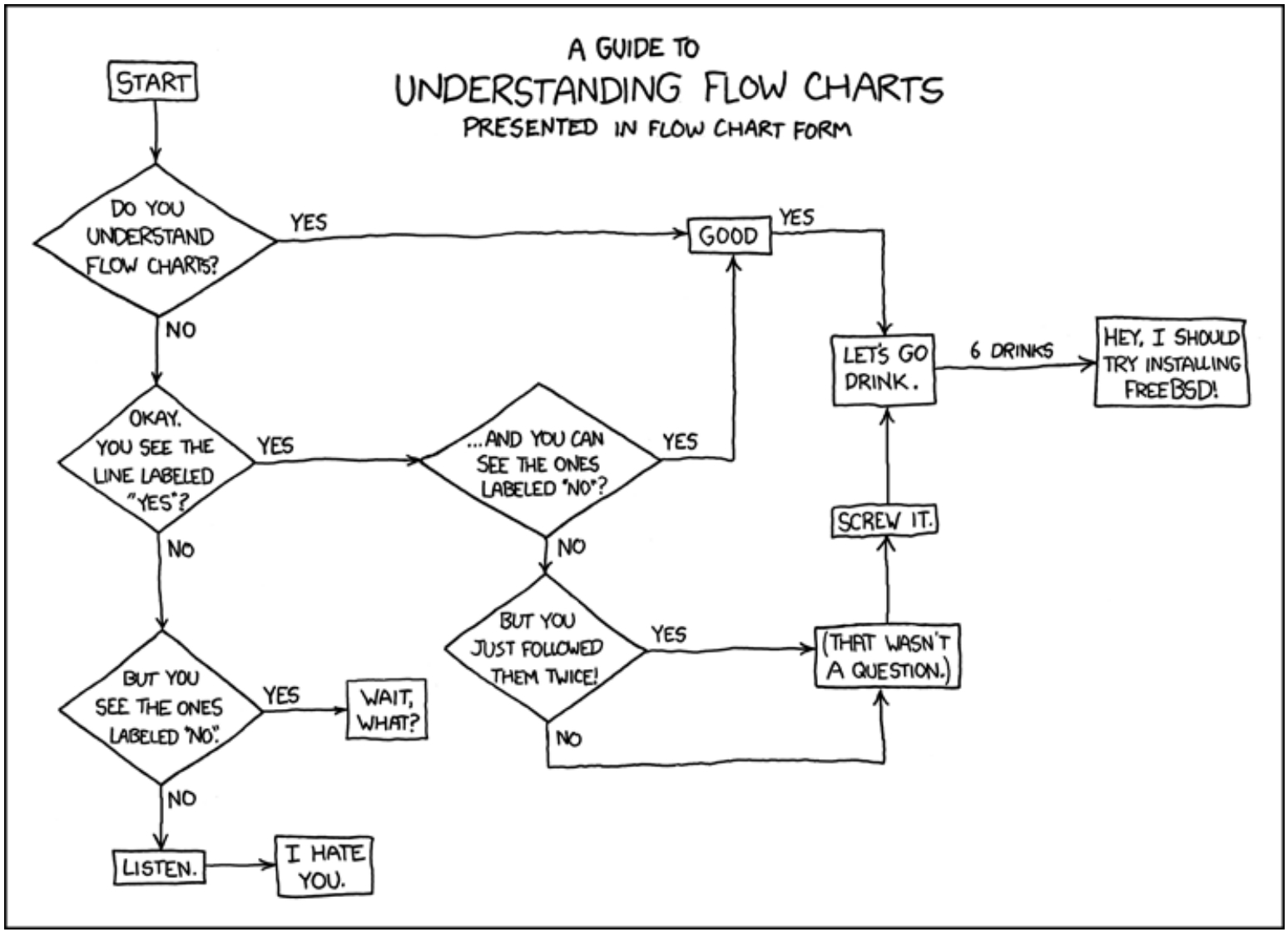


Fig: Operation of if...elif...else statement

(programiz)

Side Note: Reading Flow Charts



(xkcd/518)

Today's Topics

- Recap: Decisions
- **Logical Expressions**
- Circuits
- Binary Numbers

Challenge

Predict what the code will do:

```
origin = "Indian Ocean"
winds = 100
if (winds > 74):
    print("Major storm, called a ", end="")
    if origin == "Indian Ocean" or origin == "South Pacific":
        print("cyclone.")
    elif origin == "North Pacific":
        print("typhoon.")
    else:
        print("hurricane.")

visibility = 0.2
winds = 40
conditions = "blowing snow"
if (winds > 35) and (visibility < 0.25) and \
      (conditions == "blowing snow" or conditions == "heavy snow"):
    print("Blizzard!")
```

Python Tutor

```
origin = "Indian Ocean"
winds = 100
if (winds > 74):
    print("Major storm, called a ", end="")
    if origin == "Indian Ocean" or origin == "South Pacific":
        print("cyclone.")
    elif origin == "North Pacific":
        print("typhoon.")
    else:
        print("hurricane.")

visibility = 0.2
winds = 40
conditions = "blowing snow"
if (winds > 35) and (visibility < 0.25) and \
     (conditions == "blowing snow" or conditions == "heavy snow"):
    print("Blizzard!")
```

(Demo with `pythonTutor`)

Logical Operators

and

in1		in2	*returns:*
`False`	`and`	`False`	`False`
`False`	`and`	`True`	`False`
`True`	`and`	`False`	`False`
`True`	`and`	`True`	`True`

Logical Operators

and

in1		in2	*returns:*
`False`	`and`	`False`	`False`
`False`	`and`	`True`	`False`
`True`	`and`	`False`	`False`
`True`	`and`	`True`	`True`

or

in1		in2	*returns:*
`False`	`or`	`False`	`False`
`False`	`or`	`True`	`True`
`True`	`or`	`False`	`True`
`True`	`or`	`True`	`True`

Logical Operators

and

in1		in2	*returns:*
False	and	False	False
False	and	True	False
True	and	False	False
True	and	True	True

or

in1		in2	*returns:*
False	or	False	False
False	or	True	True
True	or	False	True
True	or	True	True

not

	in1	*returns:*
not	False	True
not	True	False

Challenge

Predict what the code will do:

```
semHours = 18
reqHours = 120
if semHours >= 12:
     print('Full Time')
else:
     print('Part Time')

pace = reqHours // semHours
if reqHours % semHours != 0:
     pace = pace + 1
print('At this pace, you will graduate in', pace, 'semesters,')
yrs = pace / 2
print('(or', yrs, 'years).')


for i in range(1,20):
     if (i > 10) and (i % 2 == 1):
          print('oddly large')
     else:
          print(i)
```

Python Tutor

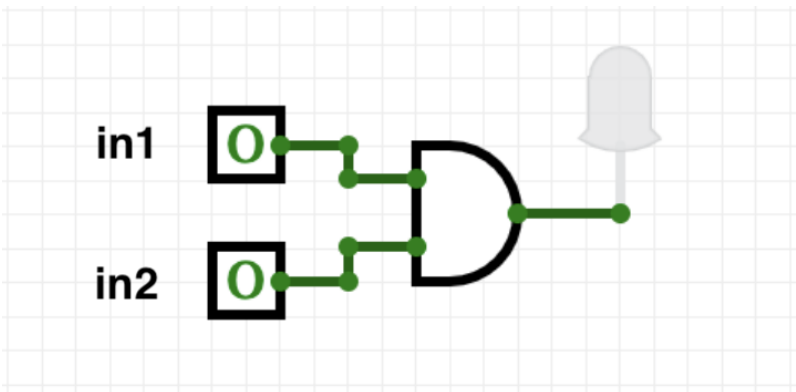
```
semHours = 18
reqHours = 120
if semHours >= 12:
     print('Full Time')
else:
     print('Part Time')

pace = reqHours // semHours
if reqHours % semHours != 0:
     pace = pace + 1
print('At this pace, you will graduate in', pace, 'semesters,')
yrs = pace / 2
print('(or', yrs, 'years).')


for i in range(1,20):
     if (i > 10) and (i % 2 == 1):
          print('oddly large')
     else:
          print(i)
```

(Demo with pythonTutor)

Today's Topics

- Recap: Decisions
- Logical Expressions
- **Circuits**
- Binary Numbers

Circuit Demo

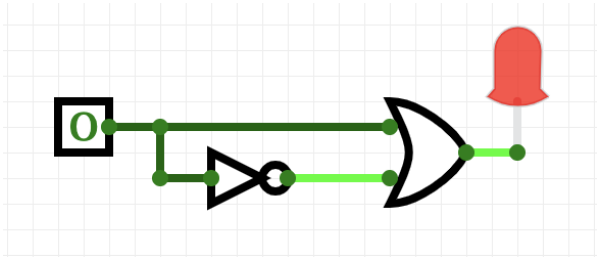


(Demo with `circuitverse`)

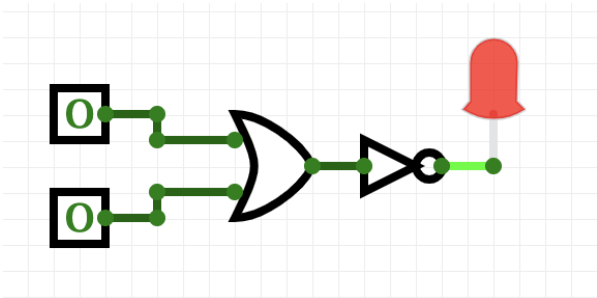
Challenge

Predict when these expressions are true:

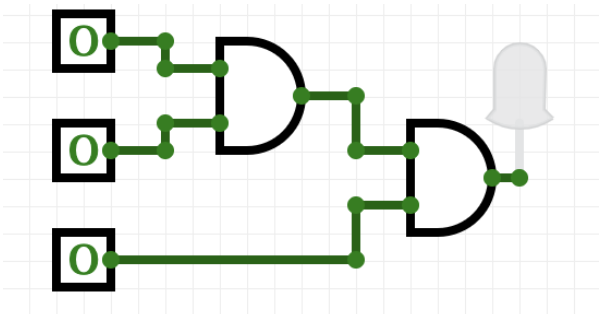
- `in1 or not in1`:



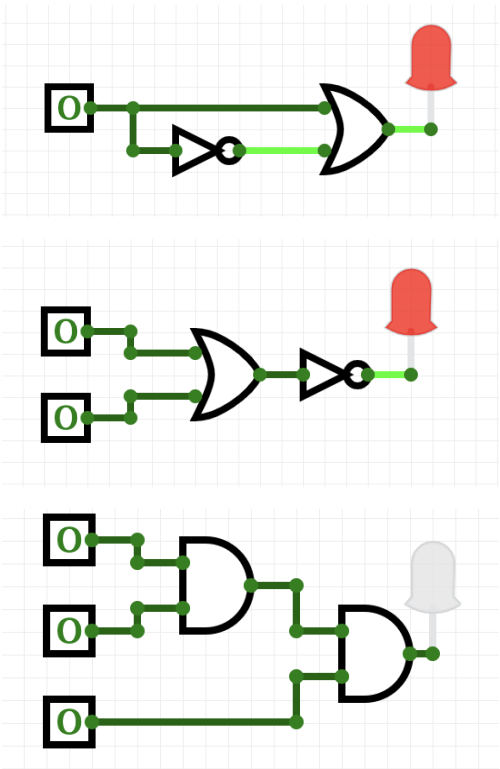
- `not(in1 or in2)`:



- `(in1 and in2) and in3`:

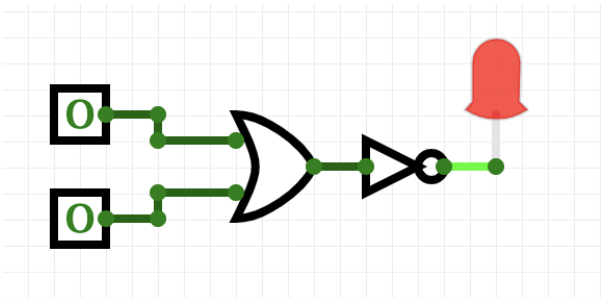


Circuit Demo



(Demo with `circuitverse`)

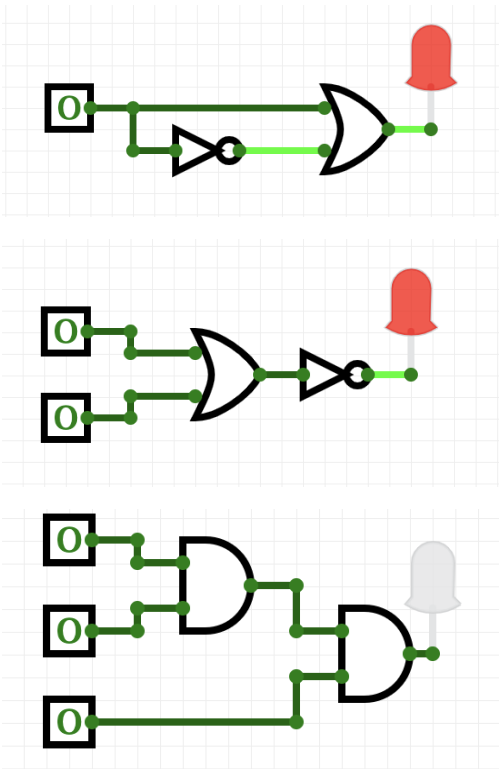
Challenge



Draw a circuit that corresponds to each logical expression:

- `in1 or in2`
- `(in1 or in2) and (in1 or in3)`
- `(not(in1 and not in2)) or (in1 and (in2 and in3))`

Circuit Demo



(Demo with `circuitverse`)

Today's Topics

- Recap: Decisions
- Logical Expressions
- Circuits
- **Binary Numbers**

Binary Numbers

- Logic $\rightarrow$ Circuits $\rightarrow$ Numbers

Binary Numbers

- Logic $\rightarrow$ Circuits $\rightarrow$ Numbers
- Digital logic design allows for two states:

Binary Numbers

- Logic $\rightarrow$ Circuits $\rightarrow$ Numbers
- Digital logic design allows for two states:
  - True / False

Binary Numbers

- Logic $\rightarrow$ Circuits $\rightarrow$ Numbers
- Digital logic design allows for two states:
  - True / False
  - On / Off (two voltage levels)

Binary Numbers

- Logic $\rightarrow$ Circuits $\rightarrow$ Numbers
- Digital logic design allows for two states:
  - True / False
  - On / Off (two voltage levels)
  - 1 / 0

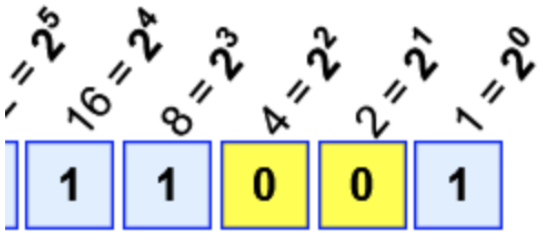
Binary Numbers

- Logic $\rightarrow$ Circuits $\rightarrow$ Numbers
- Digital logic design allows for two states:
  - True / False
  - On / Off (two voltage levels)
  - 1 / 0
- Computers store numbers using the Binary system (base 2)

Binary Numbers

- Logic $\rightarrow$ Circuits $\rightarrow$ Numbers
- Digital logic design allows for two states:
  - True / False
  - On / Off (two voltage levels)
  - 1 / 0
- Computers store numbers using the Binary system (base 2)
- A **bit** (binary digit) being 1 (on) or 0 (off)

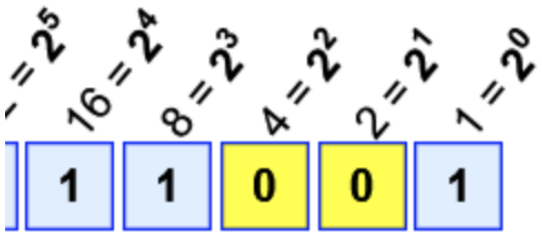
Binary Numbers



Example: 1×16 + 1×8 + 1×1 = 16+8+1 = 25

- Two digits: **0** and **1**

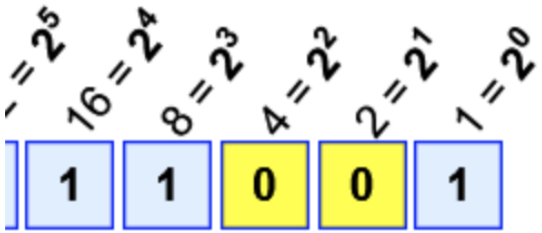
Binary Numbers



Example: 1×16 + 1×8 + 1×1 = 16+8+1 = 25

- Two digits: **0** and **1**
- Each position is a power of two

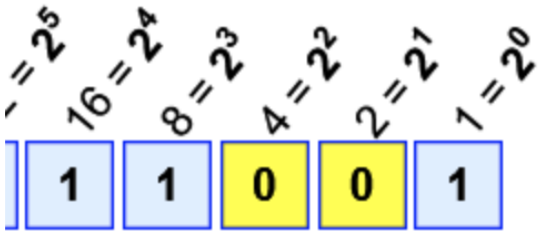
Binary Numbers



Example: 1×16 + 1×8 + 1×1 = 16+8+1 = 25

- Two digits: **0** and **1**
- Each position is a power of two
  - Decimal: the "ones", "tens", "hundreds" and so on (powers of 10)

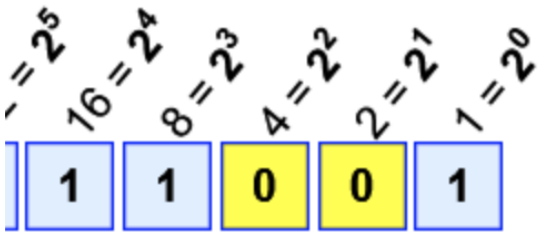
Binary Numbers



Example: 1×16 + 1×8 + 1×1 = 16+8+1 = 25

- Two digits: **0** and **1**
- Each position is a power of two
  - Decimal: the "ones", "tens", "hundreds" and so on (powers of 10)
  - Binary: the "ones", "twos", "fours", "sixteens" and so on (powers of 2)

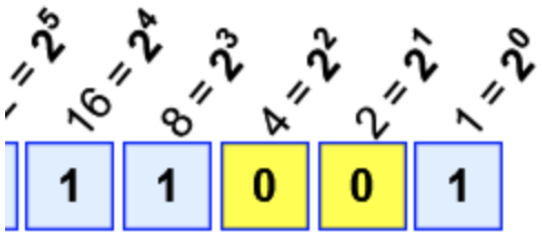
Binary Numbers



Example: 1×16 + 1×8 + 1×1 = 16+8+1 = 25

- Two digits: **0** and **1**
- Each position is a power of two
  - Decimal: the "ones", "tens", "hundreds" and so on (powers of 10)
  - Binary: the "ones", "twos", "fours", "sixteens" and so on (powers of 2)
- In each position the digit is either 0 or 1, so given a binary number we can obtain the decimal equivalent as follows:

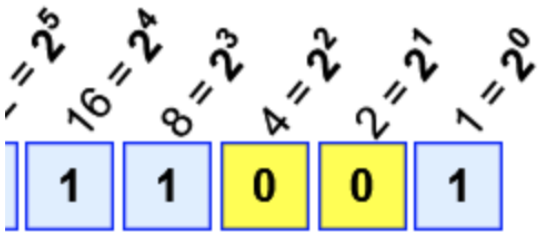
Binary Numbers



Example: 1×16 + 1×8 + 1×1 = 16+8+1 = 25

- Two digits: **0** and **1**
- Each position is a power of two
  - Decimal: the "ones", "tens", "hundreds" and so on (powers of 10)
  - Binary: the "ones", "twos", "fours", "sixteens" and so on (powers of 2)
- In each position the digit is either 0 or 1, so given a binary number we can obtain the decimal equivalent as follows:
  - In the "ones" position we either have a 1 or not

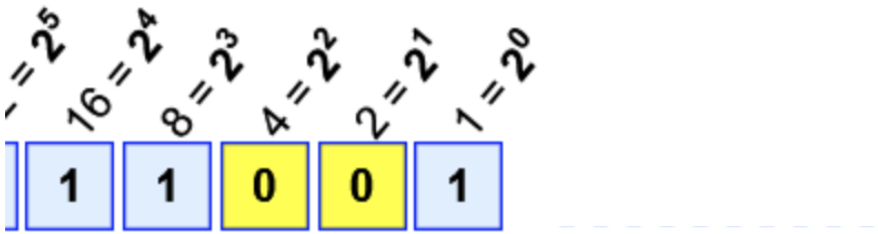
Binary Numbers



Example: 1×16 + 1×8 + 1×1 = 16+8+1 = 25

- Two digits: **0** and **1**
- Each position is a power of two
  - Decimal: the "ones", "tens", "hundreds" and so on (powers of 10)
  - Binary: the "ones", "twos", "fours", "sixteens" and so on (powers of 2)
- In each position the digit is either 0 or 1, so given a binary number we can obtain the decimal equivalent as follows:
  - In the "ones" position we either have a 1 or not
  - In the "twos" position we either have a 2 or not

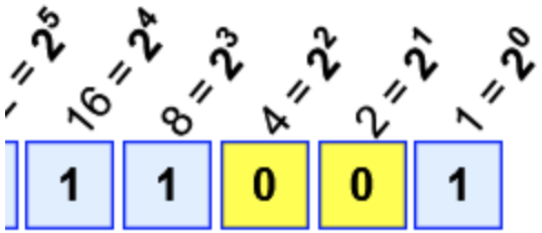
Binary Numbers



Example: 1×16 + 1×8 + 1×1 = 16+8+1 = 25

- Two digits: **0** and **1**
- Each position is a power of two
  - Decimal: the "ones", "tens", "hundreds" and so on (powers of 10)
  - Binary: the "ones", "twos", "fours", "sixteens" and so on (powers of 2)
- In each position the digit is either 0 or 1, so given a binary number we can obtain the decimal equivalent as follows:
  - In the "ones" position we either have a 1 or not
  - In the "twos" position we either have a 2 or not
  - In the "fours" position we either have a 4 or not ...

Binary Numbers



Example: 1×16 + 1×8 + 1×1 = 16+8+1 = 25

- Two digits: **0** and **1**
- Each position is a power of two
  - Decimal: the "ones", "tens", "hundreds" and so on (powers of 10)
  - Binary: the "ones", "twos", "fours", "sixteens" and so on (powers of 2)
- In each position the digit is either 0 or 1, so given a binary number we can obtain the decimal equivalent as follows:
  - In the "ones" position we either have a 1 or not
  - In the "twos" position we either have a 2 or not
  - In the "fours" position we either have a 4 or not ...
- **Example:**

$$11001_{base2} = 16 + 8 + 1 = 25_{base10}$$

Lecture Slip Challenge: Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.

Lecture Slip Challenge: Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- Write down the output to see the pattern:

Lecture Slip Challenge: Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- Write down the output to see the pattern:
  1

Lecture Slip Challenge: Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz".
- Write down the output to see the pattern:
  1
  2

Lecture Slip Challenge: Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- Write down the output to see the pattern:
  1
  2
  Fizz

Lecture Slip Challenge: Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- Write down the output to see the pattern:
  1
  2
  Fizz
  4

Lecture Slip Challenge: Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- Write down the output to see the pattern:
  1
  2
  Fizz
  4
  Buzz

Lecture Slip Challenge: Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- Write down the output to see the pattern:
  1
  2
  Fizz
  4
  Buzz
  Fizz

Lecture Slip Challenge: Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- Write down the output to see the pattern:
  1
  2
  Fizz
  4
  Buzz
  Fizz
  7

Lecture Slip Challenge: Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- Write down the output to see the pattern:
  1
  2
  Fizz
  4
  Buzz
  Fizz
  7
  ...
  14

Lecture Slip Challenge: Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- Write down the output to see the pattern:
  1
  2
  Fizz
  4
  Buzz
  Fizz
  7
  ...
  14
  FizzBuzz

Lecture Slip Challenge: Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz".
- Write down the output to see the pattern:
  1
  2
  Fizz
  4
  Buzz
  Fizz
  7
  ...
  14
  FizzBuzz
- Write the **algorithm** then, if time, write the code.

Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.

Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- To Do List:

Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- To Do List:
  - Create a loop that goes from 1 to 100.

Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- To Do List:
  - Create a loop that goes from 1 to 100.
  - If the number is divisible by 3, print “Fizz”.

Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- To Do List:
  - Create a loop that goes from 1 to 100.
  - If the number is divisible by 3, print “Fizz”.
  - If the number is divisible by 5, print “Buzz”.

Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- To Do List:
  - Create a loop that goes from 1 to 100.
  - If the number is divisible by 3, print “Fizz”.
  - If the number is divisible by 5, print “Buzz”.
  - **If divisible by both, print “FizzBuzz”.**

Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- To Do List:
  - Create a loop that goes from 1 to 100.
  - If the number is divisible by 3, print “Fizz”.
  - If the number is divisible by 5, print “Buzz”.
  - **If divisible by both, print “FizzBuzz”.**
  - Otherwise print the number.

Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz".
- To Do List:
  - Create a loop that goes from 1 to 100.
  - If the number is divisible by 3, print "Fizz".
  - If the number is divisible by 5, print "Buzz".
  - **If divisible by both, print "FizzBuzz".**
  - Otherwise print the number.
    Order matters!!! To print FizzBuzz when i is divisible by both it should be checked first, otherwise it will never get to this case!

Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print “Fizz” instead of the number and for the multiples of five print “Buzz”. For numbers which are multiples of both three and five print “FizzBuzz”.
- To Do List (**Reordered**):

Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz".
- To Do List (**Reordered**):
  - Create a loop that goes from 1 to 100.
  - **If divisible by both 3 and 5, print "FizzBuzz".**

Tech Interview Classic

- Write a program that prints the numbers from 1 to 100. But for multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz".
- To Do List (**Reordered**):
  - Create a loop that goes from 1 to 100.
  - **If divisible by both 3 and 5, print "FizzBuzz".**
  - If the number is divisible by 3, print "Fizz".
  - If the number is divisible by 5, print "Buzz".
  - Otherwise print the number.
  - Also should print a new line (so each entry is on its own line).

Tech Interview Classic

- To Do List:
  - Create a loop that goes from 1 to 100.
  - If divisible by both 3 and 5, print “FizzBuzz”.
  - If the number is divisible by 3, print “Fizz”.
  - If the number is divisible by 5, print “Buzz”.
  - Otherwise print the number.
  - Also should print a new line (so each entry is on its own line).

Tech Interview Classic

- To Do List:
  - Create a loop that goes from 1 to 100.
  - If divisible by both 3 and 5, print "FizzBuzz".
  - If the number is divisible by 3, print "Fizz".
  - If the number is divisible by 5, print "Buzz".
  - Otherwise print the number.
  - Also should print a new line (so each entry is on its own line).

```
for i in range(1,101):
```

Tech Interview Classic

- To Do List:
  - Create a loop that goes from 1 to 100.
  - If divisible by both 3 and 5, print "FizzBuzz".
  - If the number is divisible by 3, print "Fizz".
  - If the number is divisible by 5, print "Buzz".
  - Otherwise print the number.
  - Also should print a new line (so each entry is on its own line).

```
for i in range(1,101):
    if i%3 == 0 and i%5 == 0:
        print("FizzBuzz")
```

Tech Interview Classic

- To Do List:
  - Create a loop that goes from 1 to 100.
  - If divisible by both 3 and 5, print "FizzBuzz".
  - If the number is divisible by 3, print "Fizz".
  - If the number is divisible by 5, print "Buzz".
  - Otherwise print the number.
  - Also should print a new line (so each entry is on its own line).

```
for i in range(1,101):
    if i%3 == 0 and i%5 == 0:
        print("FizzBuzz")
    elif i%3 == 0:
        print("Fizz")
```

Tech Interview Classic

- To Do List:
  - Create a loop that goes from 1 to 100.
  - If divisible by both 3 and 5, print "FizzBuzz".
  - If the number is divisible by 3, print "Fizz".
  - If the number is divisible by 5, print "Buzz".
  - Otherwise print the number.
  - Also should print a new line (so each entry is on its own line).

```
for i in range(1,101):
    if i%3 == 0 and i%5 == 0:
        print("FizzBuzz")
    elif i%3 == 0:
        print("Fizz")
    elif i%5 == 0:
        print("Buzz")
```

Tech Interview Classic

- To Do List:
  - Create a loop that goes from 1 to 100.
  - If divisible by both 3 and 5, print "FizzBuzz".
  - If the number is divisible by 3, print "Fizz".
  - If the number is divisible by 5, print "Buzz".
  - Otherwise print the number.
  - Also should print a new line (so each entry is on its own line).

```
for i in range(1,101):
    if i%3 == 0 and i%5 == 0:
        print("FizzBuzz")
    elif i%3 == 0:
        print("Fizz")
    elif i%5 == 0:
        print("Buzz")
    else:
        print(i)
```

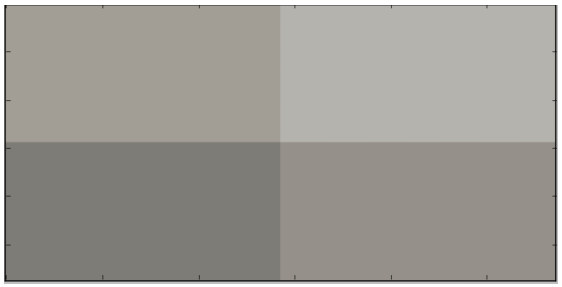
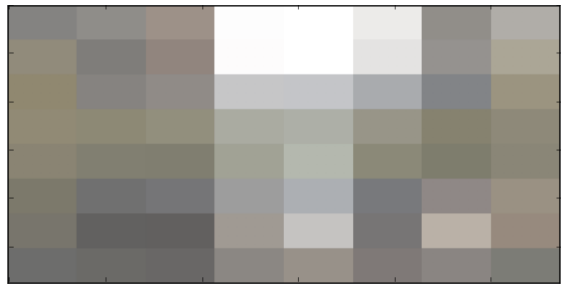
Recap

- In Python, we introduced:

Recap

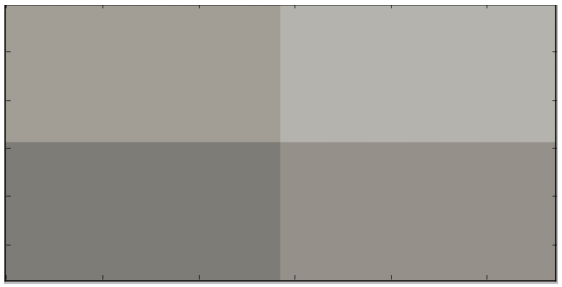
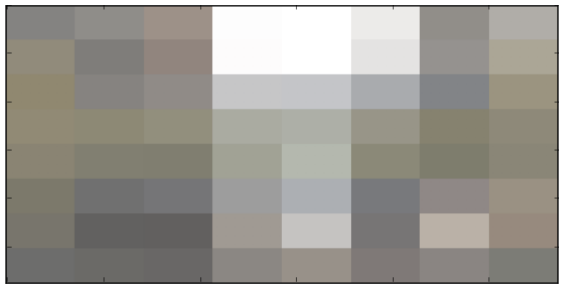
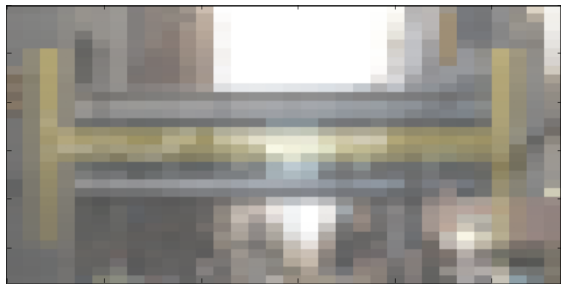
- In Python, we introduced:
  - Decisions
  - Logical Expressions
  - Circuits
  - Binary Numbers

Practice Quiz & Final Questions


- Since you must pass the final exam to pass the course, we end every lecture with final exam review.
- Pull out something to write on (not to be turned in).

Practice Quiz & Final Questions

- Since you must pass the final exam to pass the course, we end every lecture with final exam review.
- Pull out something to write on (not to be turned in).
- Lightning rounds:
  - write as much you can for 60 seconds;
  - followed by answer; and
  - repeat.
- Past exams are on the webpage (under Final Exam Information).
- We're starting with Spring 2018, Version 1.

Weekly Reminders!

Before next lecture, don't forget to:

- Work on this week's Online Lab

Weekly Reminders!

Before next lecture, don't forget to:

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

Weekly Reminders!

Before next lecture, don't forget to:

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

Weekly Reminders!

Before next lecture, don't forget to:

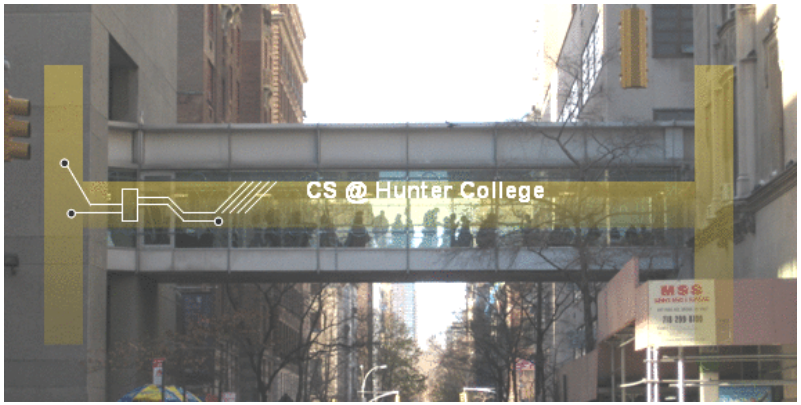
- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001E Hunter North
- If you haven't already, schedule an appointment to take the Code Review (**one every two weeks**) in lab 1001E Hunter North
- Submit this week's 5 programming assignments **(programs 21-25)**

Weekly Reminders!

Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 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 21-25)**
- If you need help, schedule an appointment for Tutoring in lab 1001E 11am-5pm

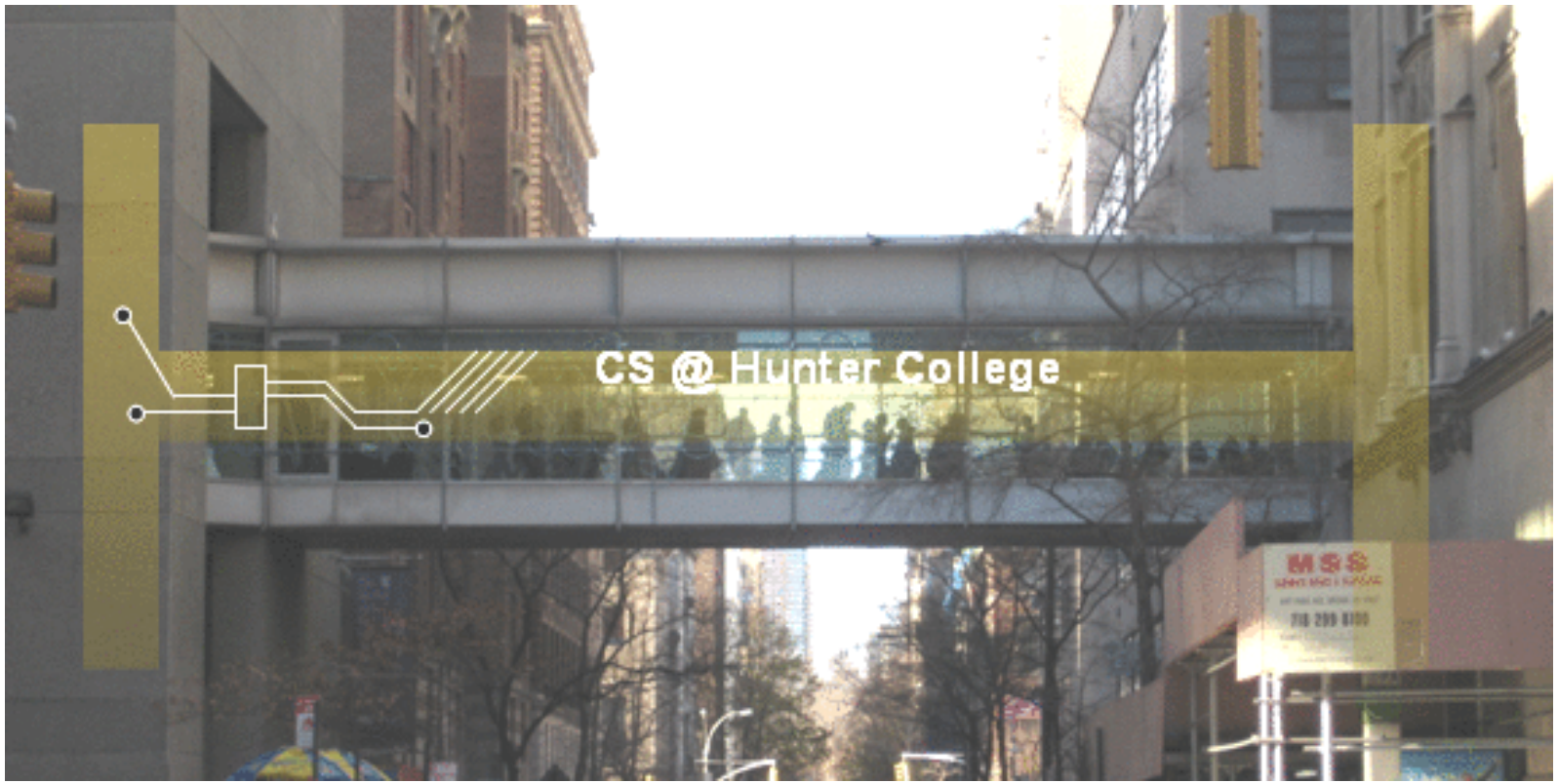
Weekly Reminders!



Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 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 21-25)**
- 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)

Lecture Slips & Writing Boards



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