

CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

Frequently Asked Questions

From previous semesters.

Frequently Asked Questions

From previous semesters.

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

Frequently Asked Questions

From previous semesters.

- **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 previous semesters.

- **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 previous semesters.

- **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 previous semesters.

- **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 previous semesters.

- **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.
- **What are types of variables?**

Frequently Asked Questions

From previous semesters.

- **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.
- **What are types of variables?**
Di erent kinds of information takes di erent amounts of space.
Types we have seen so far: int, float, str and objects (e.g. turtles).

Frequently Asked Questions

From previous semesters.

- **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.
- **What are types of variables?**
Di erent kinds of information takes di erent amounts of space.
Types we have seen so far: int, float, str and objects (e.g. turtles).
- **How can I tell strings from variables?**

Frequently Asked Questions

From previous semesters.

- **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.
- **What are types of variables?**
Di erent kinds of information takes di erent amounts of space.
Types we have seen so far: int, float, str and objects (e.g. turtles).
- **How can I tell strings from variables?**
Strings are surrounded by quotes (either single or double).

Frequently Asked Questions

From previous semesters.

- **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.
- **What are types of variables?**
Di erent kinds of information takes di erent amounts of space.
Types we have seen so far: int, float, str and objects (e.g. turtles).
- **How can I tell strings from variables?**
Strings are surrounded by quotes (either single or double).
Variables names (identi ers) for memory locations are not.

Frequently Asked Questions

From previous semesters.

- **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.
- **What are types of variables?**
Di erent kinds of information takes di erent amounts of space.
Types we have seen so far: int, float, str and objects (e.g. turtles).
- **How can I tell strings from variables?**
Strings are surrounded by quotes (either single or double).
Variables names (identi ers) 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 (algorithms) in plain English without worrying about syntax (code).

One More FAQ: Why Paper Planes?

Why paper planes?

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

- | Practice writing solutions (algorithms) in plain English without worrying about syntax (code).
- | 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 (algorithms) in plain English without worrying about syntax (code).
- | Practice thinking (and writing) precisely.

Why collaboratively?

One More FAQ: Why Paper Planes?

Why paper planes?

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

- | Practice writing solutions (algorithms) in plain English without worrying about syntax (code).
- | Practice thinking (and writing) precisely.

Why collaboratively?

- | 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 (algorithms) in plain English without worrying about syntax (code).
- | Practice thinking (and writing) precisely.

Why collaboratively?

- | 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 (algorithms) in plain English without worrying about syntax (code).
- | Practice thinking (and writing) precisely.

Why collaboratively?

- | Improves mastery of material.
- | Our industry partners want strong communication skills:
 - F 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 (algorithms) in plain English without worrying about syntax (code).
- | Practice thinking (and writing) precisely.

Why collaboratively?

- | Improves mastery of material.
- | Our industry partners want strong communication skills:
 - F communicating technical ideas precisely, and
 - F communicating and working in teams.

Today's Topics

Recap: Decisions

Logical Expressions

Circuits

Binary Numbers

Tech Interview Classic

Today's Topics

Recap: Decisions

Logical Expressions

Circuits

Binary Numbers

Tech Interview Classic

Challenge Problem...

Some challenges with types & decisions:

Python Tutor

(Demo with pythonTutor)

Decisions

Side Note: Reading Flow Charts

(xkcd/518)

Today's Topics

Recap: Decisions

Logical Expressions

Circuits

Binary Numbers

Tech Interview Classic

Challenge Problem

Predict what the code will do:

Python Tutor

(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 Problem

Predict what the code will do:

Python Tutor

(Demo with pythonTutor)

Today's Topics

Recap: Decisions

Logical Expressions

Circuits

Binary Numbers

Tech Interview Classic

Circuit Demo

(Demo with neuroproductions)

Challenge Problem

Predict when these expressions are true:

`in1 or not in1` :

`not(in1 or in2)` :

`(in1 and in2) and in3` :

Circuit Demo

(Demo with neuroproductions)

Challenge Problem

Draw a circuit that corresponds to each logical expression:

$\text{in1} \text{ or } \text{in2}$

$(\text{in1} \text{ or } \text{in2}) \text{ and } (\text{in1} \text{ or } \text{in3})$

$(\text{not}(\text{in1} \text{ and } \text{not } \text{in2})) \text{ or } (\text{in1} \text{ and } (\text{in2} \text{ and } \text{in3}))$

Circuit Demo

(Demo with neuroproductions)

Today's Topics

Recap: Decisions

Logical Expressions

Circuits

Binary Numbers

Tech Interview Classic

Binary Numbers

Logic! Circuits! Numbers

Binary Numbers

Logic! Circuits! Numbers

Digital logic design allows for two states:

Binary Numbers

Logic! Circuits! Numbers

Digital logic design allows for two states:

- | True / False

Binary Numbers

Logic! Circuits! Numbers

Digital logic design allows for two states:

- | True / False
- | On / 0 (two voltage levels)

Binary Numbers

Logic! Circuits! Numbers

Digital logic design allows for two states:

- | True / False
- | On / O (two voltage levels)
- | 1 / 0

Binary Numbers

Logic! Circuits! Numbers

Digital logic design allows for two states:

- | True / False
- | On / O (two voltage levels)
- | 1 / 0

Computers store numbers using the Binary system (base 2)

Binary Numbers

Logic! Circuits! Numbers

Digital logic design allows for two states:

- | True / False
- | On / O (two voltage levels)
- | 1 / 0

Computers store numbers using the Binary system (base 2)

A bit (binary digit) being 1 (on) or 0 (o)

Binary Numbers

Two digits: 0 and 1

Binary Numbers

Two digits: 0 and 1

Each position is a power of two

Binary Numbers

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

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

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 v can obtain the decimal equivalent as follows:

Binary Numbers

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

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

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

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:

$$11001_{\text{base}2} = 16 + 8 + 1 = 25_{\text{base}10}$$

Today's Topics

Recap: Decisions

Logical Expressions

Circuits

Binary Numbers

Tech Interview Classic

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".

Write down the output to see the pattern:

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

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

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

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

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

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

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

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

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

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)
```

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

Recap

- In Python, we introduced:
 - | Decisions
 - | Logical Expressions
 - | Circuits
 - | Binary Numbers
- [Log in to Gradescope for Quiz 5.](#)