CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

< ロ ト < 団 ト < 三 ト < 三 ト</p>

CSci 127 (Hunter)

Lecture 3

11 February 2020 1 / 46

From lecture slips & emails.

999

<ロト < 部 ト < 注 ト < 注 ト - 注</p>

From lecture slips & emails.

• What does it mean that this course is hybrid?

990

From lecture slips & emails.

• What does it mean that this course is hybrid?

Instead of having a second weekly lecture, you work independently through the weekly **Online Lab**: reinforces material introduced in lecture, introduces new material, and provide knowledge necessary to work on programming assignments.

From lecture slips & emails.

- What does it mean that this course is hybrid? Instead of having a second weekly lecture, you work independently through the weekly Online Lab: reinforces material introduced in lecture, introduces new material, and provide knowledge necessary to work on programming assignments.
- Can I work ahead on programs?

From lecture slips & emails.

 What does it mean that this course is hybrid? Instead of having a second weekly lecture, you work independently through the weekly Online Lab: reinforces material introduced in lecture, introduces new material, and provide knowledge necessary to work on programming assignments.

• Can I work ahead on programs?

Absolutely!!! You should be 5 or so programs ahead. If you are working on today's program you are already falling behind!!!

From lecture slips & emails.

- What does it mean that this course is hybrid? Instead of having a second weekly lecture, you work independently through the weekly Online Lab: reinforces material introduced in lecture, introduces new material, and provide knowledge necessary to work on programming assignments.
- Can I work ahead on programs?

Absolutely!!! You should be 5 or so programs ahead. If you are working on today's program you are already falling behind!!!

• What do you mean by Input and Output?

From lecture slips & emails.

- What does it mean that this course is hybrid? Instead of having a second weekly lecture, you work independently through the weekly Online Lab: reinforces material introduced in lecture, introduces new material, and provide knowledge necessary to work on programming assignments.
- Can I work ahead on programs?

Absolutely!!! You should be 5 or so programs ahead. If you are working on today's program you are already falling behind!!!

• What do you mean by Input and Output?

Input is data provided to a program each time it runs (e.g. input() in Lab2); Output is data produced by a program each time it runs (e.g. display text or graphics on screen). Not all programs have Input or Output.

From lecture slips & emails.

- What does it mean that this course is hybrid? Instead of having a second weekly lecture, you work independently through the weekly Online Lab: reinforces material introduced in lecture, introduces new material, and provide knowledge necessary to work on programming assignments.
- Can I work ahead on programs? Absolutely!!! You should be 5 or so programs ahead. If you are working on today's program you are already falling behind!!!
- What do you mean by Input and Output? Input is data provided to a program each time it runs (e.g. input() in Lab2); Output is data produced by a program each time it runs (e.g. display text or graphics on screen). Not all programs have Input or Output.
- I don't understand the ASCII Table?

From lecture slips & emails.

- What does it mean that this course is hybrid? Instead of having a second weekly lecture, you work independently through the weekly Online Lab: reinforces material introduced in lecture, introduces new material, and provide knowledge necessary to work on programming assignments.
- Can I work ahead on programs? Absolutely!!! You should be 5 or so programs ahead. If you are working on today's program you are already falling behind!!!
- What do you mean by Input and Output? Input is data provided to a program each time it runs (e.g. input() in Lab2); Output is data produced by a program each time it runs (e.g. display text or graphics on screen). Not all programs have Input or Output.
- I don't understand the ASCII Table?

Intro/Survey course: introduce high-level concepts before low-level. Cannot store characters on a computer chip, only numbers. ASCII is simply an agreement on how to map characters to numbers so they can be stored on computer chips.

From lecture slips & emails.

- What does it mean that this course is hybrid? Instead of having a second weekly lecture, you work independently through the weekly Online Lab: reinforces material introduced in lecture, introduces new material, and provide knowledge necessary to work on programming assignments.
- Can I work ahead on programs? Absolutely!!! You should be 5 or so programs ahead. If you are working on today's program you are already falling behind!!!
- What do you mean by Input and Output? Input is data provided to a program each time it runs (e.g. input() in Lab2); Output is data produced by a program each time it runs (e.g. display text or graphics on screen). Not all programs have Input or Output.
- I don't understand the ASCII Table? Intro/Survey course: introduce high-level concepts before low-level. Cannot store characters on a computer chip, only numbers. ASCII is simply an agreement on how to map characters to numbers so they can be stored on computer chips.
- Why are we learning about the command line?

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

From lecture slips & emails.

- What does it mean that this course is hybrid? Instead of having a second weekly lecture, you work independently through the weekly Online Lab: reinforces material introduced in lecture, introduces new material, and provide knowledge necessary to work on programming assignments.
- Can I work ahead on programs? Absolutely!!! You should be 5 or so programs ahead. If you are working on today's program you are already falling behind!!!
- What do you mean by Input and Output? Input is data provided to a program each time it runs (e.g. input() in Lab2); Output is data produced by a program each time it runs (e.g. display text or graphics on screen). Not all programs have Input or Output.

• I don't understand the ASCII Table?

Intro/Survey course: introduce high-level concepts before low-level. Cannot store characters on a computer chip, only numbers. ASCII is simply an agreement on how to map characters to numbers so they can be stored on computer chips.

 Why are we learning about the command line? Starting with Lab2, bottom section will introduce shell commands. Command line is widely used among Computer Scientists and in Industry; very useful for automating tasks and working remotely. Do not overlook!!! Will be tested on both Quizzes and Final Exam. CSci 127 (Hunter) Lecture 3 11 February 2020 2/46

From lecture slips & emails.

- What does it mean that this course is hybrid? Instead of having a second weekly lecture, you work independently through the weekly Online Lab: reinforces material introduced in lecture, introduces new material, and provide knowledge necessary to work on programming assignments.
- Can I work ahead on programs? Absolutely!!! You should be 5 or so programs ahead. If you are working on today's program you are already falling behind!!!
- What do you mean by Input and Output? Input is data provided to a program each time it runs (e.g. input() in Lab2); Output is data produced by a program each time it runs (e.g. display text or graphics on screen). Not all programs have Input or Output.

• I don't understand the ASCII Table?

Intro/Survey course: introduce high-level concepts before low-level. Cannot store characters on a computer chip, only numbers. ASCII is simply an agreement on how to map characters to numbers so they can be stored on computer chips.

 Why are we learning about the command line? Starting with Lab2, bottom section will introduce shell commands. Command line is widely used among Computer Scientists and in Industry; very useful for automating tasks and working remotely. Do not overlook!!! Will be tested on both Quizzes and Final Exam. CSci 127 (Hunter) Lecture 3 11 February 2020 2/46

Today's Topics



- Research Survey
- More on Strings
- Arithmetic
- Indexing and Slicing Lists
- Colors & Hexadecimal Notation

Э

990

Today's Topics



• Research Survey

- More on Strings
- Arithmetic
- Indexing and Slicing Lists
- Colors & Hexadecimal Notation

3

Why All the Handouts Today?



Lecture Slip



Overview

		NAME FORM
I you have appear measure that you COM Research De pot can write be	ann allest year right- readd bla in dinaan e splano Administrator	or a revents periopor, or you have constrain or th neurons rober than the researchers, above of the d 100-101-072.8 or used SXPPA and als. Marnadoly
COVERENCE Alter Stern Research Con- 201 Sunt 424 Room Rear Tools, 201 200	Feetbacoller for bear planet inhibitionate 1	
SAME AND A	Aprel .	
f you agos to get	terpere in this research	study piece sign and date below. You will be grant a
Peters Name of Pe	r isipati	
Spoon-throa	рак	
Spatare at Sales	idead MetaloingCourse	
Pitch-IPitane-Ch	Polat Milling Soc	
lipster d'habi	ing Broking Corners	- be

· · · ·				100.00	
			Annual Art		
for man	i be at descet 14 property	the period	ipata indiki-siada		
			Manager Manager		
		P.013		Terreri.	
			-A-03287 N-18		
				olation,	
hibrar	make the LIT during	. I general	CONTRACT.		
					18.64D
	. BIRD				
	DENIETIC		Ohili 722		
	NAME AND				NOTES
	049983		66		
	101400				
	DOBRATE .				
	. NTENES				
When an	moniting the spanninger	intere, inte		end beliefs along	computer mileters.
1.1	what retret the products	at a part	. 1	Annihister	Under Sprong and
	one suspervise	~			
0.	distant.				
_ ri 0	and the second se			I down also down and	
	The second se				
				1000	
- Яt					
-88	derived.				
-83	stands and stands		E	Arr	
	al cala la ca			Apre Strongly agree	

Consent Form

Survey

990

This study investigates students' emotions, cognitions, motivation, and learning related to computer science.



Part 1: Consists of two brief surveys completed in class.

Prof. John Ranellucci

Educational Psychology

This study investigates students' emotions, cognitions, motivation, and learning related to computer science.



Part 1: Consists of two brief surveys completed in class.

Part 2: I'm asking you to answer three extra questions at the end of your "lecture slips".

Prof. John Ranellucci

Educational Psychology

This study investigates students' emotions, cognitions, motivation, and learning related to computer science.



Prof. John Ranellucci

Educational Psychology

- Part 1: Consists of two brief surveys completed in class.
- Part 2: I'm asking you to answer three extra questions at the end of your "lecture slips".
- Part 3: Consists of six questions per week for 10 weeks (three before class and three after class) via text message.

This study investigates students' emotions, cognitions, motivation, and learning related to computer science.



Prof. John Ranellucci

Educational Psychology

Part 1: Consists of two brief surveys completed in class.

- Part 2: I'm asking you to answer three extra questions at the end of your "lecture slips".
- Part 3: Consists of six questions per week for 10 weeks (three before class and three after class) via text message.

(Participants will be compensated with a \$20 Amazon gift certificate for completing the text-message portion of the survey - \$1 for 3-question sets)

This study investigates students' emotions, cognitions, motivation, and learning related to computer science.



Prof. John Ranellucci

Educational Psychology

- Part 1: Consists of two brief surveys completed in class.
- Part 2: I'm asking you to answer three extra questions at the end of your "lecture slips".
- Part 3: Consists of six questions per week for 10 weeks (three before class and three after class) via text message.

(Participants will be compensated with a \$20 Amazon gift certificate for completing the text-message portion of the survey - \$1 for 3-question sets)

This study is not part of the class, and no individual analyses will be shared with your instructor.

Today's Topics



- Research Survey
- More on Strings
- Arithmetic
- Indexing and Slicing Lists
- Colors & Hexadecimal Notation

3

From Final Exam, Fall 2017, Version 1, #1:

Name:

EmpID:

CSci 127 Final, V1, F17

1. (a) What will the following Python code print:

```
s = "FridaysSaturdaysSundays"
num = s.count("s")
days = s[:-1].split("s")
print("There are", num, "fun days in a week")
mess = days[0]
print("Two of them are", mess, days[-1])
result = ""
for i in range(len(mess)):
    if i > 2:
        result = result + mess[i]
print("My favorite", result, "is Saturday.")
```

Output:

CSci 127 (Hunter)

11 February 2020 8 / 46

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○



```
s = "FridaysSaturdaysSundays"
num = s.count("s")
days = s[:-1].split("s")
print("There are", num, "fun days in a week")
mess = days[0]
result = ""
for i in range(len(mess)):
    if i > 2:
        result = result + mess[i]
    print("Wy favorite", result, "is Saturday.")
```

• Some we have seen before, some we haven't.

") "

3



- Some we have seen before, some we haven't.
- Don't leave it blank- write what you know & puzzle out as much as possible.



- Some we have seen before, some we haven't.
- Don't leave it blank- write what you know & puzzle out as much as possible.
- First, go through and write down what we know:



- Some we have seen before, some we haven't.
- Don't leave it blank- write what you know & puzzle out as much as possible.
- First, go through and write down what we know:
 - ► There are 3 print().

Name:	EmpID:		CSci 127 Final, V1, F1
1. (a)	What will the following Python code print:		
	<pre>s = "FridaysSaturdaysSundays" num = s.count("s") days = s[:-1].split("s") print("There are", num, "fun days in a week") mess = days[0] result = "" for i in range(len(mess)): if i > 2: result = result + mess[i] print("Wy forvrite", result, "is Saturday.")</pre>	Output:	

- Some we have seen before, some we haven't.
- Don't leave it blank- write what you know & puzzle out as much as possible.
- First, go through and write down what we know:
 - There are 3 print().
 - Output will have at least:

< ロ ト < 団 ト < 三 ト < 三 ト</p>



- Some we have seen before, some we haven't.
- Don't leave it blank- write what you know & puzzle out as much as possible.
- First, go through and write down what we know:
 - ▶ There are 3 print().
 - Output will have at least: There are ??? fun days in a week

Name:	EmpID:		CSci 127 Final, V1, F1
1. (a)	What will the following Python code print:		
	<pre>s = "FridaysSaturdaysSundays" num = s.count("s") days = s[:-1].split("s") print("There are", num, "fun days in a week") mess = days[0] result = "" for i in range(len(mess)): if i > 2: result = result + mess[1] print("Wy orrite", result, "is Saturday.")</pre>	Output:	

- Some we have seen before, some we haven't.
- Don't leave it blank- write what you know & puzzle out as much as possible.
- First, go through and write down what we know:
 - ▶ There are 3 print().
 - Output will have at least: There are ??? fun days in a week Two of them are ???

CSci 127 (Hunter)

3

Sac

Name:	EmpID:	CSci 127 Final, V1, F1
1. (a)	What will the following Python code print:	
	<pre>s = "FridaysSaturdaysSundays" num = s.count('s") days = s[:-1].split('s") print("There are", num, "fun days in a week") mess = days[0] result = "" for i in range(len(mess)): if i > 2: result = result + mess[i] print("My favorite", result, "is Saturday.")</pre>	Output:

- Some we have seen before, some we haven't.
- Don't leave it blank- write what you know & puzzle out as much as possible.
- First, go through and write down what we know:
 - ▶ There are 3 print().

```
    Output will have at least:
There are ??? fun days in a week
Two of them are ???
My favorite ??? is Saturday.
```

CSci 127 (Hunter)

11 February 2020 9 / 46

Sac



- Some we have seen before, some we haven't.
- Don't leave it blank- write what you know & puzzle out as much as possible.
- First, go through and write down what we know:
 - ▶ There are 3 print().

```
    Output will have at least:
There are ??? fun days in a week
Two of them are ???
My favorite ??? is Saturday.
```

• Will get 1/3 to 1/2 points for writing down the basic structure.

CSci 127 (Hunter)

```
s = "FridaysSaturdaysSundays"
num = s.count("s")
```

• The first line creates a variable, called s, that stores the string: "FridaysSaturdaysSundays"

イロト (四) (三) (三) (二) (0)

```
s = "FridaysSaturdaysSundays"
num = s.count("s")
```

- The first line creates a variable, called s, that stores the string: "FridaysSaturdaysSundays"
- There are many useful functions for strings (more in Lab 2).

イロト (四) (三) (三) (三) (0)

```
s = "FridaysSaturdaysSundays"
num = s.count("s")
```

- The first line creates a variable, called s, that stores the string: "FridaysSaturdaysSundays"
- There are many useful functions for strings (more in Lab 2).
- s.count(x) will count the number of times the pattern, x, appears in s.

イロト (四) (三) (三) (三) (0)

```
s = "FridaysSaturdaysSundays"
num = s.count("s")
```

- The first line creates a variable, called s, that stores the string: "FridaysSaturdaysSundays"
- There are many useful functions for strings (more in Lab 2).
- s.count(x) will count the number of times the pattern, x, appears in s.
 - ▶ s.count("s") counts the number of lower case s that occurs.

◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 - つへへ
More on Strings: String Methods

```
s = "FridaysSaturdaysSundays"
num = s.count("s")
```

- The first line creates a variable, called s, that stores the string: "FridaysSaturdaysSundays"
- There are many useful functions for strings (more in Lab 2).
- s.count(x) will count the number of times the pattern, x, appears in s.
 - ▶ s.count("s") counts the number of lower case s that occurs.
 - num = s.count("s") stores the result in the variable num, for later.

◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 ◇◇◇

11 February 2020

10/46

CSci 127 (Hunter)

More on Strings: String Methods

```
s = "FridaysSaturdaysSundays"
num = s.count("s")
```

- The first line creates a variable, called s, that stores the string: "FridaysSaturdaysSundays"
- There are many useful functions for strings (more in Lab 2).
- s.count(x) will count the number of times the pattern, x, appears in s.
 - ▶ s.count("s") counts the number of lower case s that occurs.
 - num = s.count("s") stores the result in the variable num, for later.
 - What would print(s.count("sS")) output?

◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 ◇◇◇

More on Strings: String Methods

```
s = "FridaysSaturdaysSundays"
num = s.count("s")
```

- The first line creates a variable, called s, that stores the string: "FridaysSaturdaysSundays"
- There are many useful functions for strings (more in Lab 2).
- s.count(x) will count the number of times the pattern, x, appears in s.
 - ▶ s.count("s") counts the number of lower case s that occurs.
 - num = s.count("s") stores the result in the variable num, for later.
 - What would print(s.count("sS")) output?
 - What about:

```
mess = "10 20 21 9 101 35"
mults = mess.count("0 ")
print(mults)
```

CSci 127 (Hunter)

◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 ◇◇◇

11 February 2020

10 / 46

More on Strings...



• Don't leave it blank- write what you know & puzzle out as much as possible:

3

イロト イロト イヨト イヨト

More on Strings...



• Don't leave it blank- write what you know & puzzle out as much as possible:

```
There are 3 fun days in a week
Two of them are ???
My favorite ??? is Saturday.
```

for i in range(len(mess)):
 if i > 2:

result = result + mess[i]
print("My favorite", result, "is Saturday.")

CSci 127 (Hunter)

11 February 2020 11 / 46

= nar

イロト イボト イヨト イヨト

```
s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")
```

• Strings are made up of individual characters (letters, numbers, etc.)

```
s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

0	1	2	3	4	5	6	7	8	 16	17	18	19	20	21	22
F	r	i	d	а	у	S	S	а	 S	u	n	d	а	у	S

< ロ > < 同 > < 三 > < 三 > < 三 > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ >

s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

0	1	2	3	4	5	6	7	8	 16	17	18	19	20	21	22
F	r	i	d	а	У	s	S	а	 S	u	n	d	а	у	S
												-4	-3	-2	-1

```
s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

0	1	2	3	4	5	6	7	8	 16	17	18	19	20	21	22
F	r	i	d	а	У	s	S	а	 S	u	n	d	а	у	S
												-4	-3	-2	-1

● s[0] is

```
s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

0	1	2	3	4	5	6	7	8	 16	17	18	19	20	21	22
F	r	i	d	а	У	s	S	а	 S	u	n	d	а	у	S
												-4	-3	-2	-1

● s[0] is 'F'.

```
s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

0	1	2	3	4	5	6	7	8	 16	17	18	19	20	21	22
F	r	i	d	а	У	s	S	а	 S	u	n	d	а	у	S
												-4	-3	-2	-1

● s[1] is

```
s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

0	1	2	3	4	5	6	7	8	 16	17	18	19	20	21	22
F	r	i	d	а	У	s	S	а	 S	u	n	d	а	у	S
												-4	-3	-2	-1

• s[1] is 'r'.

```
s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

0	1	2	3	4	5	6	7	8	 16	17	18	19	20	21	22
F	r	i	d	а	у	s	S	а	 S	u	n	d	а	у	S
												-4	-3	-2	-1

• s[-1] is

```
s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

0	1	2	3	4	5	6	7	8	 16	17	18	19	20	21	22
F	r	i	d	а	У	s	S	а	 S	u	n	d	а	у	S
												-4	-3	-2	-1

● s[-1] is 's'.

```
s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

0	1	2	3	4	5	6	7	8	 16	17	18	19	20	21	22
F	r	i	d	а	у	s	S	а	 S	u	n	d	а	У	S
												-4	-3	-2	-1

• s[3:6] is

```
s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

0	1	2	3	4	5	6	7	8	 16	17	18	19	20	21	22
F	r	i	d	а	у	s	S	а	 S	u	n	d	а	у	S
												-4	-3	-2	-1

• s[3:6] is 'day'.

```
s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

0	1	2	3	4	5	6	7	8	 16	17	18	19	20	21	22
F	r	i	d	а	У	s	S	а	 S	u	n	d	а	у	S
												-4	-3	-2	-1

• s[:3] is

```
s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

0	1	2	3	4	5	6	7	8	 16	17	18	19	20	21	22
F	r	i	d	а	У	s	S	а	 S	u	n	d	а	у	S
												-4	-3	-2	-1

• s[:3] is 'Fri'.

s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

0	1	2	3	4	5	6	7	8	 16	17	18	19	20	21	22
F	r	i	d	а	У	S	S	а	 S	u	n	d	а	У	S
												-4	-3	-2	-1

• s[:-1] is

s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a "substring" of the string.

0	1	2	3	4	5	6	7	8	 16	17	18	19	20	21	22
F	r	i	d	а	У	S	S	а	 S	u	n	d	а	у	s
												-4	-3	-2	-1

 s[:-1] is 'FridaysSaturdaysSunday'. (no trailing 's' at the end)

CSci 127 (Hunter)

```
s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")
```

• split() divides a string into a list.

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

```
s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")
```

• split() divides a string into a list.

• Cross out the delimiter, and the remaining items are the list.

s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")

- split() divides a string into a list.
- Cross out the delimiter, and the remaining items are the list.

"Friday XSaturday XSunday"

◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 - つへへ

s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")

split() divides a string into a list.

• Cross out the delimiter, and the remaining items are the list.

"FridayXSaturdayXSunday"
days = ['Friday', 'Saturday', 'Sunday']

s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")

- split() divides a string into a list.
- Cross out the delimiter, and the remaining items are the list.

"FridayXSaturdayXSunday"
days = ['Friday', 'Saturday', 'Sunday']

• Different delimiters give different lists:

s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")

- split() divides a string into a list.
- Cross out the delimiter, and the remaining items are the list.

"FridayXSaturdayXSunday"
days = ['Friday', 'Saturday', 'Sunday']

Different delimiters give different lists:
 days = s[:-1].split("day")

< □ > < □ > < 豆 > < 豆 > < 豆 > < 豆 > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □

s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")

- split() divides a string into a list.
- Cross out the delimiter, and the remaining items are the list.

"FridayXSaturdayXSunday"
days = ['Friday', 'Saturday', 'Sunday']

 Different delimiters give different lists: days = s[:-1].split("day") "FrixxsSaturxxsSunxxx"

◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 - つへへ

s = "FridaysSaturdaysSundays"
days = s[:-1].split("s")

- split() divides a string into a list.
- Cross out the delimiter, and the remaining items are the list.

"FridayXSaturdayXSunday"
days = ['Friday', 'Saturday', 'Sunday']

• Different delimiters give different lists: days = s[:-1].split("day") "Frix*sSatur**sSun**" days = ['Fri', 'sSatur', 'sSun']

CSci 127 (Hunter)

< □ > < □ > < 豆 > < 豆 > < 豆 > < 豆 > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □

More on Strings...



• Don't leave it blank- write what you know & puzzle out as much as possible:

3

イロト イロト イヨト イヨト

More on Strings...



• Don't leave it blank- write what you know & puzzle out as much as possible:

```
There are 3 fun days in a week
Two of them are Friday Sunday
My favorite ??? is Saturday.
```

for i in range(len(mess)):
 if i > 2:

result = result + mess[i]
print("My favorite", result, "is Saturday.")

CSci 127 (Hunter)

イロト イロト イヨト イヨト

11 February 2020

20 / 46

Today's Topics



- Research Survey
- More on Strings
- Arithmetic
- Indexing and Slicing Lists
- Colors & Hexadecimal Notation

3

200

Some arithmetic operators in Python:

Addition:



1

996

Some arithmetic operators in Python:

• Addition: sum = sum + 3



999

Some arithmetic operators in Python:

- Addition: sum = sum + 3
- Subtraction:



996

Some arithmetic operators in Python:

- Addition: sum = sum + 3
- Subtraction: deb = deb item


Some arithmetic operators in Python:

- Addition: sum = sum + 3
- Subtraction: deb = deb item
- Multiplication:



999

< □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > <

Some arithmetic operators in Python:

- Addition: sum = sum + 3
- Subtraction: deb = deb item
- Multiplication: area = h * w



< □ > < 同 > < 臣 > < 臣 > 三 = - の < ⊙

Some arithmetic operators in Python:

- Addition: sum = sum + 3
- Subtraction: deb = deb item
- Multiplication: area = h * w
- Division:



< □ > < 同 > < 臣 > < 臣 > 三 = - の < ⊙

Some arithmetic operators in Python:

- Addition: sum = sum + 3
- Subtraction: deb = deb item
- Multiplication: area = h * w
- Division: ave = total / n



<ロト < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □

Some arithmetic operators in Python:

- Addition: sum = sum + 3
- Subtraction: deb = deb item
- Multiplication: area = h * w
- Division: ave = total / n
- Floor or Integer Division:



500

イロト イボト イヨト イヨト 二日

Some arithmetic operators in Python:

- Addition: sum = sum + 3
- Subtraction: deb = deb item
- Multiplication: area = h * w
- Division: ave = total / n
- Floor or Integer Division: weeks = totalDays // 7 15 // 7 = 2



200

イロト 不得 トイラト イラト・ラー

Some arithmetic operators in Python:

- Addition: sum = sum + 3
- Subtraction: deb = deb item
- Multiplication: area = h * w
- Division: ave = total / n
- Floor or Integer Division: weeks = totalDays // 7 15 // 7 = 2
- Remainder or Modulus:



200

イロト イボト イヨト イヨト 二日



Some arithmetic operators in Python:

- Addition: sum = sum + 3
- Subtraction: deb = deb item
- Multiplication: area = h * w
- Division: ave = total / n
- Floor or Integer Division: weeks = totalDays // 7
 15 // 7 = 2
- Remainder or Modulus: days = totalDays % 7
 15 % 7 = 1

11 February 2020 22 / 46

Sac

イロト イボト イヨト イヨト 二日



Some arithmetic operators in Python:

- Addition: sum = sum + 3
 Subtraction: deb = deb item
 Multiplication: area = h * w
 Division: ave = total / n
 Floor or Integer Division: weeks = totalDays // 7 15 // 7 = 2
 Remainder or Modulus: days = totalDays % 7 15 % 7 = 1
- Exponentiaion:

11 February 2020 22 / 46



Some arithmetic operators in Python:

Addition: sum = sum + 3 Subtraction: deb = deb - item Multiplication: area = h * w Division: ave = total / n Floor or Integer Division: weeks = totalDays // 7 15 // 7 = 2 Remainder or Modulus: days = totalDays % 7 15 % 7 = 1• Exponentiaion:

pop = 2**time

11 February 2020 22 / 46

What does this code do?

```
#Mystery code for lecture 3
```

```
startTime = int(input('Enter starting time: '))
duration = int(input('Enter how long: '))
```

print('Your event starts at', startTime, "o'clock.")

endTime = (startTime+duration)%12
print('Your event ends at', endTime, "o'clock.")

What does this code do?

```
#Mystery code for lecture 3
startTime = int(input('Enter starting time: '))
duration = int(input('Enter how long: '))
print('Your event starts at', startTime, "o'clock.")
endTime = (startTime+duration)%12
print('Your event ends at', endTime, "o'clock.")
```

In particular, what is printed...

• If the user enters, 9 and 2.

What does this code do?

```
#Mystery code for lecture 3
startTime = int(input('Enter starting time: '))
duration = int(input('Enter how long: '))
print('Your event starts at', startTime, "o'clock.")
endTime = (startTime+duration)%12
print('Your event ends at', endTime, "o'clock.")
```

In particular, what is printed...

- If the user enters, 9 and 2.
- If the user enters, 12 and 4.

What does this code do?

```
#Mystery code for lecture 3
startTime = int(input('Enter starting time: '))
duration = int(input('Enter how long: '))
print('Your event starts at', startTime, "o'clock.")
endTime = (startTime+duration)%12
print('Your event ends at', endTime, "o'clock.")
```

In particular, what is printed...

- If the user enters, 9 and 2.
- If the user enters, 12 and 4.
- If the user enters, 8 and 20.

What does this code do?

```
#Mystery code for lecture 3
startTime = int(input('Enter starting time: '))
duration = int(input('Enter how long: '))
print('Your event starts at', startTime, "o'clock.")
endTime = (startTime+duration)%12
print('Your event ends at', endTime, "o'clock.")
```

In particular, what is printed...

- If the user enters, 9 and 2.
- If the user enters, 12 and 4.
- If the user enters, 8 and 20.
- If the user enters, 11 and 1.

What does this code do?

```
#Mystery code for lecture 3
startTime = int(input('Enter starting time: '))
duration = int(input('Enter how long: '))
print('Your event starts at', startTime, "o'clock.")
endTime = (startTime+duration)%12
print('Your event ends at', endTime, "o'clock.")
```

In particular, what is printed...

• If the user enters, 9 and 2.

What does this code do?

```
#Mystery code for lecture 3
startTime = int(input('Enter starting time: '))
duration = int(input('Enter how long: '))
print('Your event starts at', startTime, "o'clock.")
endTime = (startTime+duration)%12
print('Your event ends at', endTime, "o'clock.")
```

In particular, what is printed...

```
    If the user enters, 9 and 2.
    Enter starting time: 9
    Enter how long: 2
    Your event starts at 9 o'clock.
    Your event ends at 11 o'clock.
```

CSci 127 (Hunter)

What does this code do?

```
#Mystery code for lecture 3
startTime = int(input('Enter starting time: '))
duration = int(input('Enter how long: '))
print('Your event starts at', startTime, "o'clock.")
endTime = (startTime+duration)%12
print('Your event ends at', endTime, "o'clock.")
```

In particular, what is printed...

• If the user enters, 12 and 4.

What does this code do?

```
#Mystery code for lecture 3
startTime = int(input('Enter starting time: '))
duration = int(input('Enter how long: '))
print('Your event starts at', startTime, "o'clock.")
endTime = (startTime+duration)%12
print('Your event ends at', endTime, "o'clock.")
```

In particular, what is printed...

```
    If the user enters, 12 and 4.
    Enter starting time: 12
    Enter how long: 4
    Your event starts at 12 o'clock.
    Your event ends at 4 o'clock.
```

CSci 127 (Hunter)

What does this code do?

```
#Mystery code for lecture 3
startTime = int(input('Enter starting time: '))
duration = int(input('Enter how long: '))
print('Your event starts at', startTime, "o'clock.")
endTime = (startTime+duration)%12
print('Your event ends at', endTime, "o'clock.")
```

In particular, what is printed...

• If the user enters, 8 and 20.

What does this code do?

```
#Mystery code for lecture 3
startTime = int(input('Enter starting time: '))
duration = int(input('Enter how long: '))
print('Your event starts at', startTime, "o'clock.")
endTime = (startTime+duration)%12
print('Your event ends at', endTime, "o'clock.")
```

In particular, what is printed...

```
    If the user enters, 8 and 20.
Enter starting time: 8
Enter how long: 20
Your event starts at 8 o'clock.
Your event ends at 4 o'clock.
```

CSci 127 (Hunter)

What does this code do?

```
#Mystery code for lecture 3
startTime = int(input('Enter starting time: '))
duration = int(input('Enter how long: '))
print('Your event starts at', startTime, "o'clock.")
endTime = (startTime+duration)%12
print('Your event ends at', endTime, "o'clock.")
```

In particular, what is printed...

• If the user enters, 11 and 1.

What does this code do?

```
#Mystery code for lecture 3
startTime = int(input('Enter starting time: '))
duration = int(input('Enter how long: '))
print('Your event starts at', startTime, "o'clock.")
endTime = (startTime+duration)%12
print('Your event ends at', endTime, "o'clock.")
```

In particular, what is printed...

```
    If the user enters, 11 and 1.
    Enter starting time: 11
    Enter how long: 1
    Your event starts at 11 o'clock.
    Your event ends at 0 o'clock.
```

CSci 127 (Hunter)

Today's Topics



- Research Survey
- More on Strings
- Arithmetic
- Indexing and Slicing Lists
- Colors & Hexadecimal Notation

Э

Sar

Mostly review:

```
1 for d in range(10, 0, -1):
 2
        print(d)
 3
   print("Blast off!")
 4
 5
   for num in range(5,8):
 6
       print(num, 2*num)
 7
 8
   s = "City University of New York"
 9
   print(s[3], s[0:3], s[:3])
10 print(s[5:8], s[-1])
11
12
   names = ["Eleanor", "Anna", "Alice", "Edith"]
13 for n in names:
14
        print(n)
```

CSci 127 (Hunter)

イロト (四) (三) (三) (二) (0)

Python Tutor

(Demo with pythonTutor)

= nar



The three versions:

1

996

< ロ ト < 団 ト < 三 ト < 三 ト</p>



The three versions:
 range(stop)

3

900

< □ > < □ > < □ > < □ > < □ >



The three versions:

- range(stop)
- range(start, stop)

3

590

イロト イロト イヨト イヨト



The three versions:

- range(stop)
- o range(start, stop)
- range(start, stop, step)

3

Sac

イロト イロト イヨト イヨト

 Similar to range(), you can take portions or slices of lists and strings:

```
1 for d in range(10, 0, -1):
    print(d)
3 print("Blast off!")
4 
5 for num in range(5,8):
    print(num, 2*num)
7 
8 s = "(ity University of New York"
9 print(s[3], s[0:3], s[:3])
10 print(s[3:6], s[-1])
11 nomes = ["Eleanor", "Anna", "Alice", "Edith"]
13 for n in nomes:
4 print(n)
```

3

Sac

イロト イボト イヨト イヨト

Similar to range(), you can take portions or slices of lists and strings:

s[5:8]

1 for d in range(10, 0, -1): print(d) 3 print("Blast off!") 4 5 for num in range(5,8): print(num, 2*num) 7 8 s = "(ity University of New York" 9 print(s[3], s[0:3], s[:3]) 10 print(s[3:6], s[-1]) 11 nomes = ["Eleanor", "Anna", "Alice", "Edith"] 13 for n in nomes: 4 print(n)

gives: "Uni"

= nar

イロト イポト イヨト イヨト

Similar to range(), you can take portions or slices of lists and strings:

s[5:8]

gives: "Uni"

Also works for lists:

= nar

イロト イロト イヨト イヨト

Similar to range(), you can take portions or slices of lists and strings:

s[5:8]

gives: "Uni"

Also works for lists:

names[1:3]

= nar

イロト イボト イヨト イヨト

Similar to range(), you can take portions or slices of lists and strings:

s[5:8]

gives: "Uni"

Also works for lists:

names[1:3]

gives: ["Anna", "Alice"]

3

Sac

イロト イボト イヨト イヨト

Similar to range(), you can take portions or slices of lists and strings:

s[5:8]

gives: "Uni"

Also works for lists:

names[1:3]

gives: ["Anna", "Alice"]

 Python also lets you "count backwards": last element has index: -1.

CSci 127 (Hunter)

Sac

イロト イボト イヨト イヨト 二日
Today's Topics



- Arithmetic
- Indexing and Slicing Lists
- Design Challenge: Planes
- Colors & Hexadecimal Notation

3

Sac

Color Name	HEX	Color
Black	<u>#000000</u>	
Navy	#000080	
DarkBlue	<u>#00008B</u>	
MediumBlue	#0000CD	
Blue	#0000FF	

• Can specify by name.

1

900

Color Name	HEX	Color
Black	<u>#000000</u>	
Navy	#000080	
DarkBlue	<u>#00008B</u>	
MediumBlue	#0000CD	
Blue	#0000FF	

- Can specify by name.
- Can specify by numbers:

E

590

< ロ ト < 団 ト < 三 ト < 三 ト</p>

Color Name	HEX	Color
Black	<u>#000000</u>	
Navy	#000080	
DarkBlue	<u>#00008B</u>	
MediumBlue	#0000CD	
Blue	#0000FF	

- Can specify by name.
- Can specify by numbers:
 - ► Amount of Red, Green, and Blue (RGB).

3

Sac

Color Name	HEX	Color
Black	<u>#000000</u>	
Navy	#000080	
DarkBlue	<u>#00008B</u>	
MediumBlue	#0000CD	
Blue	#0000FF	

- Can specify by name.
- Can specify by numbers:
 - ► Amount of Red, Green, and Blue (RGB).
 - Adding light, not paint:

CSci 127 (Hunter)

3

Sac

Color Name	HEX	Color
Black	<u>#000000</u>	
Navy	#000080	
DarkBlue	<u>#00008B</u>	
MediumBlue	#0000CD	
Blue	#0000FF	

- Can specify by name.
- Can specify by numbers:
 - ► Amount of Red, Green, and Blue (RGB).
 - Adding light, not paint:
 - ★ Black: 0% red, 0% green, 0% blue

CSci 127 (Hunter)

3

Sac

Color Name	HEX	Color
Black	<u>#000000</u>	
Navy	#000080	
DarkBlue	<u>#00008B</u>	
MediumBlue	#0000CD	
Blue	#0000FF	

- Can specify by name.
- Can specify by numbers:
 - ► Amount of Red, Green, and Blue (RGB).
 - Adding light, not paint:
 - ★ Black: 0% red, 0% green, 0% blue
 - ★ White: 100% red, 100% green, 100% blue

CSci 127 (Hunter)

= nar

Color Name	HEX	Color
Black	<u>#000000</u>	
Navy	#000080	
DarkBlue	<u>#00008B</u>	
MediumBlue	#0000CD	
Blue	#0000FF	

• Can specify by numbers (RGB):

<ロト < 部 > < 目 > < 目 > < 目 > < 目 > < 目 > < 0 < 0</p>

Color Name	HEX	Color
Black	<u>#000000</u>	
Navy	<u>#000080</u>	
DarkBlue	<u>#00008B</u>	
MediumBlue	#0000CD	
Blue	<u>#0000FF</u>	

- Can specify by numbers (RGB):
 - ► Fractions of each:

996

Color Name	HEX	Color
Black	<u>#000000</u>	
Navy	<u>#000080</u>	
DarkBlue	<u>#00008B</u>	
MediumBlue	#0000CD	
Blue	<u>#0000FF</u>	

- Can specify by numbers (RGB):
 - Fractions of each:

e.g. (1.0, 0, 0) is 100% red, no green, and no blue.

▲□▶ ▲□▶ ▲三▶ ▲三▶ 三三 - ∽0,00

Color Name	HEX	Color
Black	<u>#000000</u>	
Navy	<u>#000080</u>	
DarkBlue	<u>#00008B</u>	
MediumBlue	#0000CD	
Blue	<u>#0000FF</u>	

- Can specify by numbers (RGB):
 - Fractions of each:
 - e.g. (1.0, 0, 0) is 100% red, no green, and no blue.
 - ▶ 8-bit colors: numbers from 0 to 255:

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

Color Name	HEX	Color
Black	<u>#000000</u>	
Navy	<u>#000080</u>	
DarkBlue	<u>#00008B</u>	
MediumBlue	#0000CD	
Blue	<u>#0000FF</u>	

• Can specify by numbers (RGB):

- ► Fractions of each:
 - e.g. (1.0, 0, 0) is 100% red, no green, and no blue.
- ▶ 8-bit colors: numbers from 0 to 255:
 e.g. (0, 255, 0) is no red, 100% green, and no blue.

CSci 127 (Hunter)

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

Color Name	HEX	Color
Black	<u>#000000</u>	
Navy	<u>#000080</u>	
DarkBlue	<u>#00008B</u>	
MediumBlue	#0000CD	
Blue	<u>#0000FF</u>	

• Can specify by numbers (RGB):

- ► Fractions of each:
 - e.g. (1.0, 0, 0) is 100% red, no green, and no blue.
- ▶ 8-bit colors: numbers from 0 to 255: e.g. (0, 255, 0) is no red, 100% green, and no blue.
- Hexcodes (base-16 numbers)...

CSci 127 (Hunter)

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

Decimal & Hexadecimal Numbers

Counting with 10 digits:



(from i-programmer.info)

3

Sac



(from i-programmer.info)

900

◆□▶ ◆圖▶ ◆臣▶ ◆臣▶ 三臣.

00 01 02 03 04 05 06 07 08 09



(from i-programmer.info)

3

999

00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19



(from i-programmer.info)

999

イロト イロト イモト イモト 二日

- 00 01 02 03 04 05 06 07 08 09
- 10 11 12 13 14 15 16 17 18 19
- 20 21 22 23 24 25 26 27 28 29



(from i-programmer.info)

3

990



(from i-programmer.info)

- 00 01 02 03 04 05 06 07 08 09
- 10 11 12 13 14 15 16 17 18 19
- 20 21 22 23 24 25 26 27 28 29
- 30 31 32 33 34 35 36 37 38 39

3

Sar



(from i-programmer.info)

 00
 01
 02
 03
 04
 05
 06
 07
 08
 09

 10
 11
 12
 13
 14
 15
 16
 17
 18
 19

 20
 21
 22
 23
 24
 25
 26
 27
 28
 29

 30
 31
 32
 33
 34
 35
 36
 37
 38
 39

 40
 41
 42
 43
 44
 45
 46
 47
 48
 49

3

< □ > < □ > < □ > < □ > < □ >



(from i-programmer.info)

 00
 01
 02
 03
 04
 05
 06
 07
 08
 09

 10
 11
 12
 13
 14
 15
 16
 17
 18
 19

 20
 21
 22
 23
 24
 25
 26
 27
 28
 29

 30
 31
 32
 33
 34
 35
 36
 37
 38
 39

 40
 41
 42
 43
 44
 45
 46
 47
 48
 49

 50
 51
 52
 53
 54
 55
 56
 57
 58
 59

3

< □ > < □ > < □ > < □ > < □ >



00	01	02	03	04	05	06	07	80	09
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69

(from i-programmer.info)

900

◆□▶ ◆圖▶ ◆臣▶ ◆臣▶ 三臣.



00	01	02	03	04	05	06	07	80	09
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79

(from i-programmer.info)

900

◆□▶ ◆圖▶ ◆臣▶ ◆臣▶ 三臣.



	C		· ·	١
(trom	I-programmer.	unto)
- 1		· · · · · · · · · · · · · · · · · · ·		

00	01	02	03	04	05	06	07	08	09
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89

900

《口》《圖》《臣》《臣》 三臣



(from i-programmer.info)

00	01	02	03	04	05	06	07	08	09
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89
90	91	92	93	94	95	96	97	98	99

900

◆□▶ ◆圖▶ ◆臣▶ ◆臣▶ 三臣.



(from i-programmer.info)

00	01	02	03	04	05	06	07	80	09
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89
90	91	92	93	94	95	96	97	98	99

 $10^1 + 10^0$

イロト イロト イモト イモト 二日

Max Number = 99

11 February 2020 37 / 46



(from i-programmer.info)

00	01	02	03	04	05	06	07	80	09
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89
90	91	92	93	94	95	96	97	98	99

 $10^1 + 10^0$

Max Number = 99

 $90 = (9 * 10^1) + (0 * 10^0)$

CSci 127 (Hunter)

Lecture 3

< □ ▶ < □ ▶ < 三 ▶ < 三 ▶ < 三 ▶ 三 の Q (~ 11 February 2020 37 / 46



(from i-programmer.info)

00	01	02	03	04	05	06	07	08	09
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89
90	91	92	93	94	95	96	97	98	99

 $10^1 + 10^0$

 $99 = (9 * 10^{1}) + (9 * 10^{0}) \rightarrow \text{ and } \text{ for a first set of a state o$

Max Number = 99

$$90 = (9 * 10^1) + (0 * 10^0)$$

Lecture 3

11 February 2020 37 / 46

Decimal & Hexadecimal Numbers

Counting with 16 digits:



(from i-programmer.info)

Э

200

00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F



(from i-programmer.info)

CSci 127 (Hunter)

Lecture 3

< □ ト < □ ト < 三 ト < 三 ト < 三 ト 三 の へ (~ 11 February 2020 39 / 46

00	01	02	03	04	05	06	07	08	09	ΟA	0B	0C	0D	0E	0F
10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F



(from i-programmer.info)

CSci 127 (Hunter)

Lecture 3

900 11 February 2020 39 / 46

イロト イロト イモト イモト 二日

00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F



(from i-programmer.info)

CSci 127 (Hunter)

Lecture 3

990 11 February 2020 39 / 46

00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F



(from i-programmer.info)

CSci 127 (Hunter)

Lecture 3

< ロ ト < 団 ト < 三 ト < 三 ト</p> 590 11 February 2020 39 / 46

 00
 01
 02
 03
 04
 05
 06
 07
 08
 09
 0A
 0B
 0C
 0D
 0E
 OF

 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C
 1D
 1E
 1F

 20
 21
 22
 24
 25
 26
 27
 28
 29
 2A
 2D
 2D
 2E
 2F

 30
 31
 32
 33
 34
 35
 36
 37
 38
 39
 3C
 D)
 3E
 3F
 3F

 40
 41
 42
 43
 44
 45
 46
 47
 48
 49
 4A
 4B
 4C
 4D
 4E
 4F
 4F



(from i-programmer.info)

CSci 127 (Hunter)

Lecture 3

11 February 2020 39 / 46

3

990

< ロ ト < 団 ト < 三 ト < 三 ト</p>

 00
 01
 02
 03
 04
 05
 06
 07
 08
 09
 0A
 0B
 0C
 0D
 0E
 0F

 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C
 1D
 1E
 1F

 20
 21
 22
 23
 24
 25
 26
 27
 28
 29
 A
 28
 20
 21
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 23
 33
 33
 33
 33
 33
 33
 33
 33
 33
 33
 33
 33
 33
 33
 34
 44
 44
 44
 44</td



(from i-programmer.info)

CSci 127 (Hunter)

Lecture 3

11 February 2020 39 / 46

3

590

< ロ ト < 回 ト < 三 ト < 三 ト</p>

 00
 01
 02
 03
 04
 05
 06
 07
 08
 09
 0A
 0B
 0C
 0D
 0E
 0F

 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C
 1D
 1E
 1F

 20
 21
 22
 23
 24
 25
 26
 27
 28
 29
 2A
 2B
 2D
 2E
 2D
 2E
 2F
 27
 28
 29
 AB
 2C
 D
 2E
 2F
 2F
 26
 27
 28
 29
 AB
 2C
 D
 2E
 2F
 3D
 3E
 3F
 3G
 3D
 3E
 3F
 3G



(from i-programmer.info)

CSci 127 (Hunter)

Lecture 3

11 February 2020 39 / 46

3

590

< □ > < □ > < □ > < □ > < □ >


(from i-programmer.info)

CSci 127 (Hunter)

Lecture 3

11 February 2020 39 / 46

3

990



(from i-programmer.info)

3

990



(from i-programmer.info)

Э



(from i-programmer.info)

 00
 01
 02
 03
 04
 05
 06
 07
 08
 09
 0A
 0B
 0C
 0D
 0E
 0F
 <td

Э

< □ > < □ > < □ > < □ > < □ >



(from i-programmer.info)

 00
 01
 02
 03
 04
 05
 06
 07
 08
 09
 0A
 0B
 0C
 0D
 0E
 0F

 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 1A
 1B
 1C
 1D
 1E
 1F

 20
 21
 22
 23
 24
 25
 26
 27
 28
 29
 2A
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 22
 25
 55
 55
 55
 55
 55
 55
 55
 55
 55
 55
 55
 55
 55
 55
 55
 55
 55
 55
 55
 56
 76
 70
 70</t

Э

< □ > < □ > < □ > < □ > < □ >



(from i-programmer.info)

3

< □ > < □ > < □ > < □ > < □ >



(from i-programmer.info)

3



(from i-programmer.info)

CSci 127 (Hunter)

Lecture 3

11 February 2020 39 / 46



00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F 60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F 80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F AO A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF CO C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF DO D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF EO E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF

CSci 127 (Hunter)

11 February 2020 39 / 46

(from i-programmer.info)

00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F 60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F 80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F AO A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF CO C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF DO D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF EO E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF

 $16^1 + 16^0$

イロト イボト イヨト イヨト

CSci 127 (Hunter)

Lecture 3

11 February 2020 39 / 46

Э



00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F 60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F 80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F AO A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF CO C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF DO D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF EO E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF

 $16^1 + 16^0$

イロト イボト イヨト イヨト

Max Number = 255



00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F 60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F 80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F AO A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF DO D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF EO E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF

 $16^1 + 16^0$

イロト イボト イヨト イヨト

Max Number = 255

 $F0 = (F * 16^{1}) + (0 * 16^{0})$

F0 = (240) + (0) = 240



00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F 60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 7E 7F 80 81 82 83 84 85 86 87 88 89 8A 8B 8C 8D 8E 8F 90 91 92 93 94 95 96 97 98 99 9A 9B 9C 9D 9E 9F AO A1 A2 A3 A4 A5 A6 A7 A8 A9 AA AB AC AD AE AF B0 B1 B2 B3 B4 B5 B6 B7 B8 B9 BA BB BC BD BE BF C0 C1 C2 C3 C4 C5 C6 C7 C8 C9 CA CB CC CD CE CF DO D1 D2 D3 D4 D5 D6 D7 D8 D9 DA DB DC DD DE DF EO E1 E2 E3 E4 E5 E6 E7 E8 E9 EA EB EC ED EE EF F0 F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FB FC FD FE FF

 $16^1 + 16^0$

Max Number = 255

 $F0 = (F * 16^{1}) + (0 * 16^{0})$ F0 = (240) + (0) = 240 $FF = (F * 16^{1}) + (F * 16^{0})$ FF = (240) + (15) = 255

CSci 127 (Hunter)

Lecture 3

11 February 2020 39 / 46

3

Colors

Color Name	HEX	Color
Black	#000000	
Navy	#000080	
DarkBlue	<u>#00008B</u>	
MediumBlue	#0000CD	
Blue	#0000FF	

- Can specify by numbers (RGB):
 - ► Fractions of each:

e.g. (1.0, 0, 0) is 100% red, no green, and no blue.

- ▶ 8-bit colors: numbers from 0 to 255: e.g. (0, 255, 0) is no red, 100% green, and no blue.
- Hexcodes (base-16 numbers):

CSci 127 (Hunter)

◆□▶ ◆□▶ ◆三▶ ◆三▶ ○○○

Colors

Color Name	HEX	Color
Black	<u>#000000</u>	
Navy	<u>#000080</u>	
DarkBlue	<u>#00008B</u>	
MediumBlue	#0000CD	
Blue	#0000FF	

- Can specify by numbers (RGB):
 - ► Fractions of each:

e.g. (1.0, 0, 0) is 100% red, no green, and no blue.

- 8-bit colors: numbers from 0 to 255:
 e.g. (0, 255, 0) is no red, 100% green, and no blue.
- Hexcodes (base-16 numbers):
 e.g. #0000FF is no red, no green, and 100% blue.

CSci 127 (Hunter)

◆□▶ ◆□▶ ◆三▶ ◆三▶ 三三 - つへへ

In Pairs or Triples...

Some review and some novel challenges:

```
import turtle
1
 2
    teddy = turtle.Turtle()
 3
4
    names = ["violet", "purple", "indigo", "lavender"]
 5 -
    for c in names:
6
      teddy.color(c)
 7
      teddy.left(60)
8
      teddy.forward(40)
9
      teddy.dot(10)
10
11
    teddy.penup()
12
    teddy.forward(100)
13
    teddy.pendown()
14
15
    hexNames = ["#FF00FF", "#990099", "#550055", "#111111"]
16 -
    for c in hexNames:
17
      teddy.color(c)
18
      teddy.left(60)
      teddy.forward(40)
19
20
      teddy.dot(10)
 CSci 127 (Hunter)
                              Lecture 3
```

Sar

Trinkets

```
1 import turtle
 2 teddy = turtle.Turtle()
4 names = ["violet", "purple", "indigo", "lavender"]
 5 - for c in names:
 6
     teddy.color(c)
 7
     teddy.left(60)
8
     teddy.forward(40)
9
     teddy.dot(10)
10
11 teddy.penup()
12 teddy.forward(100)
13 teddy.pendown()
14
15 hexNames = ["#FF00FF", "#990099", "#550055", "#111111"]
16 - for c in hexNames:
17
     teddy.color(c)
18
     teddy.left(60)
19
     teddy.forward(40)
20
    teddy.dot(10)
```

(Demo with trinkets)

Sac

イロト イポト イヨト イヨト 二日

Lecture Slip

LECTURE 3, CSCI 127 SPRING 2020
EmpID:

1. Write down the names of your team members:



2. What is printed? Write your answer for each in the output box.



CSci 127 (Hunter)

Lecture 3

< □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □



• On lecture slip, write down a topic you wish we had spent more time (and why).

Э

590

< ロ ト < 回 ト < 三 ト < 三 ト</p>



 On lecture slip, write down a topic you wish we had spent more time (and why).

• In Python, we introduced:

Э

Sac



- On lecture slip, write down a topic you wish we had spent more time (and why).
- In Python, we introduced:
 - ► Indexing and Slicing Lists

3



- On lecture slip, write down a topic you wish we had spent more time (and why).
- In Python, we introduced:
 - Indexing and Slicing Lists
 - Colors

3



- On lecture slip, write down a topic you wish we had spent more time (and why).
- In Python, we introduced:
 - Indexing and Slicing Lists
 - Colors
 - Hexadecimal Notation

Э

< ロ ト < 団 ト < 三 ト < 三 ト</p>



- On lecture slip, write down a topic you wish we had spent more time (and why).
- In Python, we introduced:
 - Indexing and Slicing Lists
 - Colors
 - Hexadecimal Notation
- Pass your lecture slips to the end of the rows for the UTA's to collect.

3



• Since you must pass the final exam to pass the course, we end every lecture with final exam review.

Image: A matrix and a matrix

-



- 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).



- 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:

b 4 = b

Image: A matrix and a matrix



- 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;



- 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



- 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.

∃ ► < ∃ ►</p>

Image: A matrix and a matrix



- 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).

CSci 127 (Hunter)



- 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 Fall 2017, Version 2.

CSci 127 (Hunter)

Writing Boards



• Return writing boards as you leave...

CSci 127 (Hunter)

Lecture 3

< □ > < □ > < □ > < □ > < □ > 990 11 February 2020 46 / 46