1.1

Logistics, Python Review, Strings

[10] Introductions (icebreaker)

Lesson Plan

- [10] Introductions (ice breaker)
- [5] Who is this guy?
- [5] Course Components, Grading Criteria
- [20] Python Review
- [15] Strings Review
- [40] Warm Up Problems!

Who is this guy?

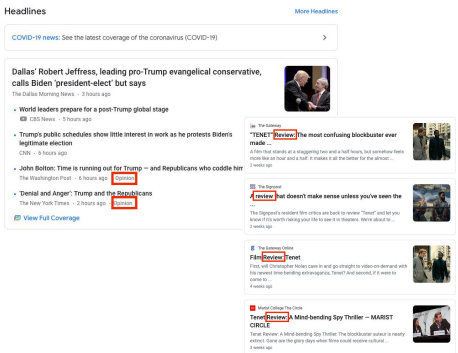
THETAKE AI

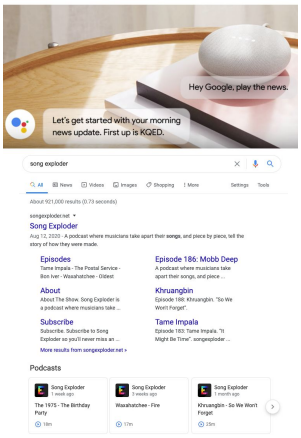
Google Podcasts

G Pay

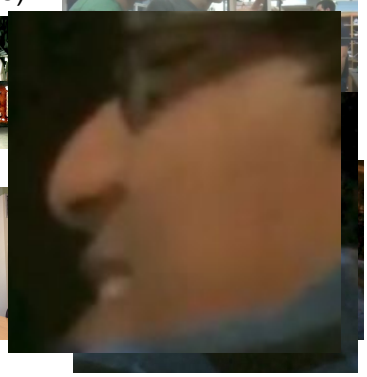
Google News

Who is this guy (at Google)?





Who was this guy (before Google)?



Who is this guy (outside of Google)?

My Goals

- I want you to love computer science
- I want you to succeed in ADS and beyond
- I want to remove whatever barriers I can between you and those first two things

How to Contact Me

shankerd@google.com

Discord

My Office Hours

Wednesday / Thursday 10:45-11:45 Eastern

(directly after class)

Or by appointment!

Course Components

- Class (x2) – 95 min
  - Lectures are interactive!
  - Live coding / problem solving
- In-class quizzes
  - Quick feedback for your instructors
  - Mandatory but not graded
- Homework
  - ~4-5 problems every week
  - You **are** allowed to collaborate with others beside your instructors and TAs
  - Please let us know who you collaborate with
- Tests
  - ~1 per unit
  - Timed (coding assessments and/or multiple-choice questions)
  - You **are not** allowed to collaborate with others

Grading Criteria

% of grade	Component
10%	Section Attendance
10%	In-class quizzes
40%	Homework
40%	Tests
TBD	Extra Credit

Course Units

#	Unit
1	Strings, Lists, and Tuples
2	Dictionaries and Sets
3	Recursion
4	Searching and Sorting
5	Linked Lists
6	Stacks and Queues
7	Trees
8	Graphs
9	Additional Problem Solving Strategies

Lectures are interactive!

Let's try these out:

- Chat waterfall
  - Everyone type your favorite animal - then, on the count of 3, we all hit Enter!
- Hand-Raise Button
- Thumbs Up / Down
- Google Meet Emoji
- Google Meet
  - Q & A
  - Poll

[20] Python Review

I/O

Input & Output (docs)

```
print('hello world')

name = input('What is your name? ')
print('Hello ' + name + '!')

# String formatting!
print('Hello %s!' % name)
print(f'Hello {name}!')
```


Code Style

Comments

```
# Single-line comment
x = 2 # another comment

'''
multi-line comment
in triple quotes
'''
```


Strings and Variables

```
rule = 'single quotes'
exception = "Unless it's needed"

# Variables and function names
# should be snake_case
big_scary_snake = 'python'
def cool_function():
 return 'cool'
```


Variable names & Function names

Should be short, but descriptive!
In general, avoid 1-letter names

[Chat Waterfall] On the count of 3, we all hit Enter!

What are good names for:

- A variable representing how many login attempts the user has left
- A function that returns whether or not a user was found in the database

Operators

Math Operators (docs)

5 - 2
5 + 2
5 * 2
5 / 2

5 // 2 # Floor division (divide, then round down)
5 % 2 # Modulus (remainder)
5 ** 2 # Exponent

These are different!

Operator Precedence (docs)

P	()
E	**
MD	*, /, //, %
AS	+, -

Types

Types (docs)

String
- 'Hello', "It's raining", 'It\'s still raining'

Integer
- 123, -34

Float
- 3.14, -4.5

Python floats have the precision of a "double" in C++ or Java

int or float? (the answer may surprise you)

int = thumbs up, float = thumbs down
- 2.5 * 2
- 4 * 2
- 4 / 2

Beware of using floats when you don't need to!
You could lose precision!

Type Conversion (float, int, str)

```
price = float(input('Price per item: '))
amount = int(input('# of items: '))

print('Total cost: ' + str(price * amount))
```


Booleans, Conditionals, While Loops

Booleans, Logical & Comparison Operators

Booleans
- True
- False

Uppercase!

Logical Operators
- **not** True
- True **and** False
- True **or** False

In other languages: !, &&, ||

Comparison Operators
- ==, !=, <=, >=, <, >

Logical Operator Precedence (docs)

Highest	**not**
	and
Lowest	**or**

Conditionals

```
grade = 85
if grade >= 90:
  print('A')
elif grade >= 80:
  print('B')
elif grade >= 70:
  print('C')
else:
  print('oh no')
```

No curly braces or parentheses!

Use tab for indentation! Whitespace matters!

Don't forget the colon!

While Loops

```
answer = ''
while answer != 'y':
  answer = input('quit? (y/n): ')
print('goodbye!')
```

```
quit? (y/n): n
quit? (y/n): n
quit? (y/n): y
goodbye!
```

Use a while loop if you need to do something until a condition is met

Functions

Defining a function (docs)

```
import math

def area(radius):
  return math.pi * radius ** 2
```

No curly braces!

Use tabs for indentation

Calling a function

```
medium_pizza_area = area(8)
personal_pizza_area = area(5)
```

Don't forget the colon!

pass

```
def tricky_function(x, y, z):
  # TODO(me): write this later
  pass
```

Can be used as a placeholder in functions, loops, and conditionals!

How is Python different from other languages?

[Chat Waterfall] On the count of 3, we all hit Enter!
What differences have you noticed so far?

Examples:

- No types required when defining variables
- Code doesn't have to be inside a class
- Style differences (single quotes, snake_case)
- Indentation instead of curly braces
- No semicolons!
- Comments (# and triple quotes)
- Single / doesn't do floor division

Modules & Python documentation

Importing modules - random

```
import random

flip = random.randint(0, 1)
if flip == 0:
  print('Heads')
else:
  print('Tails')
```

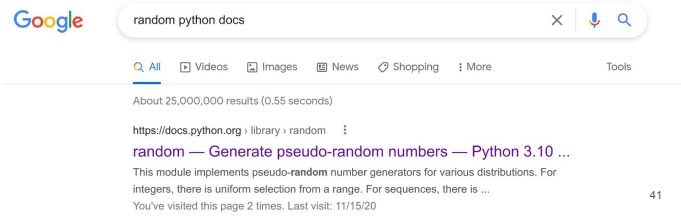

Importing modules - math

```
import math

a = int(input('a: '))
b = int(input('b: '))

c = math.sqrt(a**2 + b**2)

print(f'c: {c}')
```

```
a: 3
b: 4
c: 5.0
```


Python documentation

https://docs.python.org/3/ - **Tutorial** & **Library Reference** are most helpful
Include "**python docs**" in your Google search query



Python documentation

`random.`**`randint`**(*a*, *b*)

Return a random integer *N* such that `a <= N <= b`. Alias for `randrange(a, b+1)`.

[20] Strings

Indexing & Slicing

Indexing

```
name = 'Mikayla'

len(name) # 7

name[0] # 'M'

name[len(name) - 1] # ?
```


Slicing

```
name = 'Mikayla'

# A slice makes a copy of the
# substring from [begin, end)
# begin is inclusive
# end is exclusive
name[0:len(name)] # 'Mikayla'

name[2:5] # ?

name[2:] # ?

name[:5] # ?
```


You can index and slice lists as well!

```
foods = ['apple', 'banana', 'carrot', 'donut', 'egg', 'falafel']

foods[0] # 'apple'
foods[1:3] # ['banana', 'carrot']
foods[-1] # ?
```

Slicing makes a copy of that segment of the list

Splitting & Joining

Splitting (str → lst)

```
sentence = 'she sells sea shells'
words = sentence.split()
print(words)
```

```
['she', 'sells', 'sea', 'shells']
```

You can pass a string to split() function to split on something other than space!

Joining (lst → str)

```
words = ['she', 'sells', 'sea', 'shells']
sentence = ' '.join(words)
print(sentence)
```

```
she sells sea shells
```

You can use something other than space as glue!

Joining letters to form words

```
letters = ['Y', 'M', 'C', 'A']
gym = ''.join(letters)
print(gym)
```

```
YMCA
```

This is probably the most common use of join!

For Loops

For Loops (range)

```
for i in range(3):
  print(i)

for i in range(0, 3):
  print(i)

for i in range(0, 3, 1):
  print(i)
```

These all print out 0, 1, 2.
range(begin, end, step) goes…

- From **begin** (inclusive)
- To **end** (exclusive)
- By intervals of size **step**

Countdown

I want my program to count down 5, 4, 3, 2, 1.
What should be the value of:

- begin?
- end?
- step?

```
for i in range(begin, end, step):
  print(i)
```

[Chat Waterfall] On the count of 3, we all hit Enter!

For loops (for-each)

```
word = 'hello'
for letter in word:
  print(letter)

lst = ['hey', 'sup', 'hi']
for word in lst:
  print(word)

sentence = 'Welcome to tech exchange!'
for word in sentence.split():
  print(word)
```

Use a for-each loop if you only care about the **value**, not the index

For loops work the exact same way for lists!

For loops (enumerate)

```
word = 'hello'
for i, letter in enumerate(word):
  print(f'The {i}th letter is {letter}')
```

```
The 0th letter is h
The 1th letter is e
The 2th letter is l
The 3th letter is l
The 4th letter is o
```

Use an enumerate loop if you care about the value and its index

For loops (range again)

```
word = 'hello'
for i in range(len(word)):
  print(f'The {i}th letter is {word[i]}')
```

```
The 0th letter is h
The 1th letter is e
The 2th letter is l
The 3th letter is l
The 4th letter is o
```

It's often easier to use enumerate instead!

Same syntax for lists!

```
foods = ['apple', 'banana', 'carrot', 'donut', 'egg', 'falafel']

for i, food in enumerate(foods):
  print(f'The {i}th food is {food}')

for i in range(len(foods)):
  print(f'The {i}th food is {foods[i]}')
```


Practice Problems

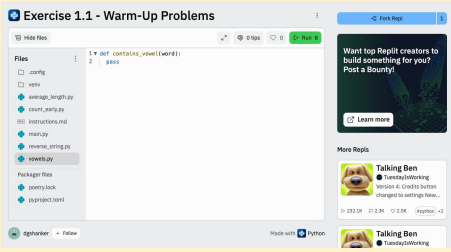
You will be working in teams of 2 or 3. The goal is to collaboratively find a solution and be able to explain it to the class. Use the table below to figure out what your role is.

Role	Responsibilities	Assignment Criteria
Driver	Copy and share the repl.it, write the code, make sure you're listening to ideas from your teammates	Person with the most letters in their middle name
Tester	Play devil's advocate, thinks of edge cases, write unit tests for the driver's code	Person with the fewest letters in their middle name
Presenter	Document the code, be prepared to present the team's design decisions, and share one thing the team learned from the problem	Person with the middlest number of letters in their middle name

If there are only 2 members in your team, the tester will also take on the presenter role.

Practice Problems

1. Select a team captain
2. Team captain:
   a. go to the link provided in class
   b. click "Fork repl"
   c. Click "Invite" Invite
   d. Generate a join link Generate a join link
   e. Copy/paste the join link into your breakout room chat!
3. Everyone else:
   a. Click the join link! You should now be able to edit your group's code



HW 0

Feedback / Attendance

https://docs.google.com/forms/d/e/1FAIpQLSecL_cEcCqWAcLbWq3FEp3FuMM72ZI_fhLzl0uR8TNajoZ6_A/alreadyresponded