Lecture 2: Python

August 28, 2024

References

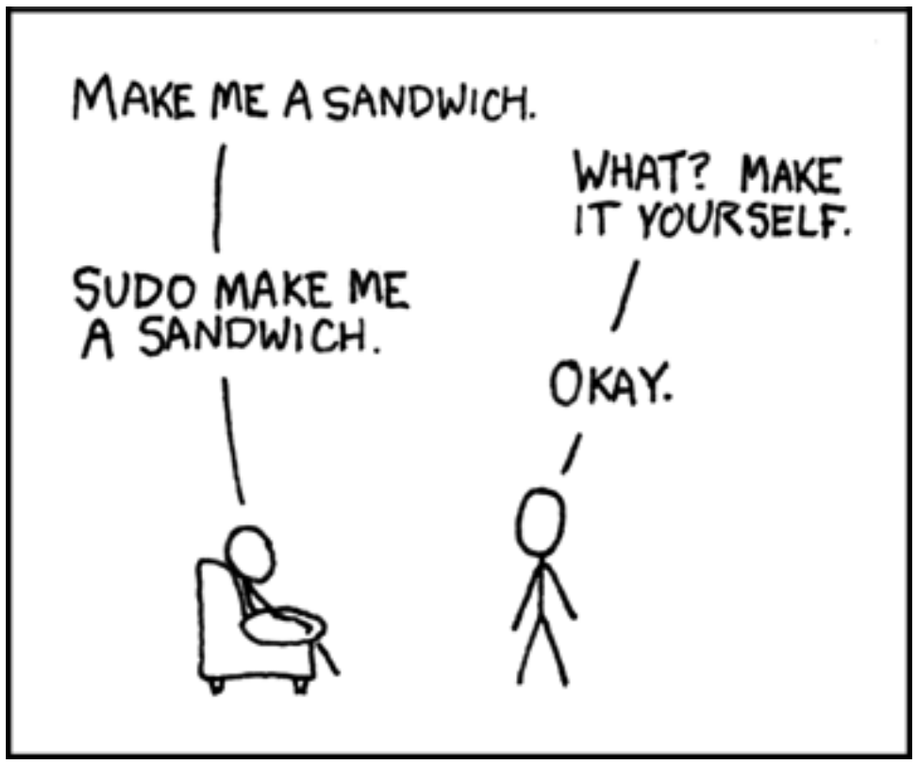
- https://www.w3schools.com/python/python_intro.asp

- https://wiki.python.org/moin/BeginnersGuide/Programmers

- https://docs.python.org/3/index.html

Hello World

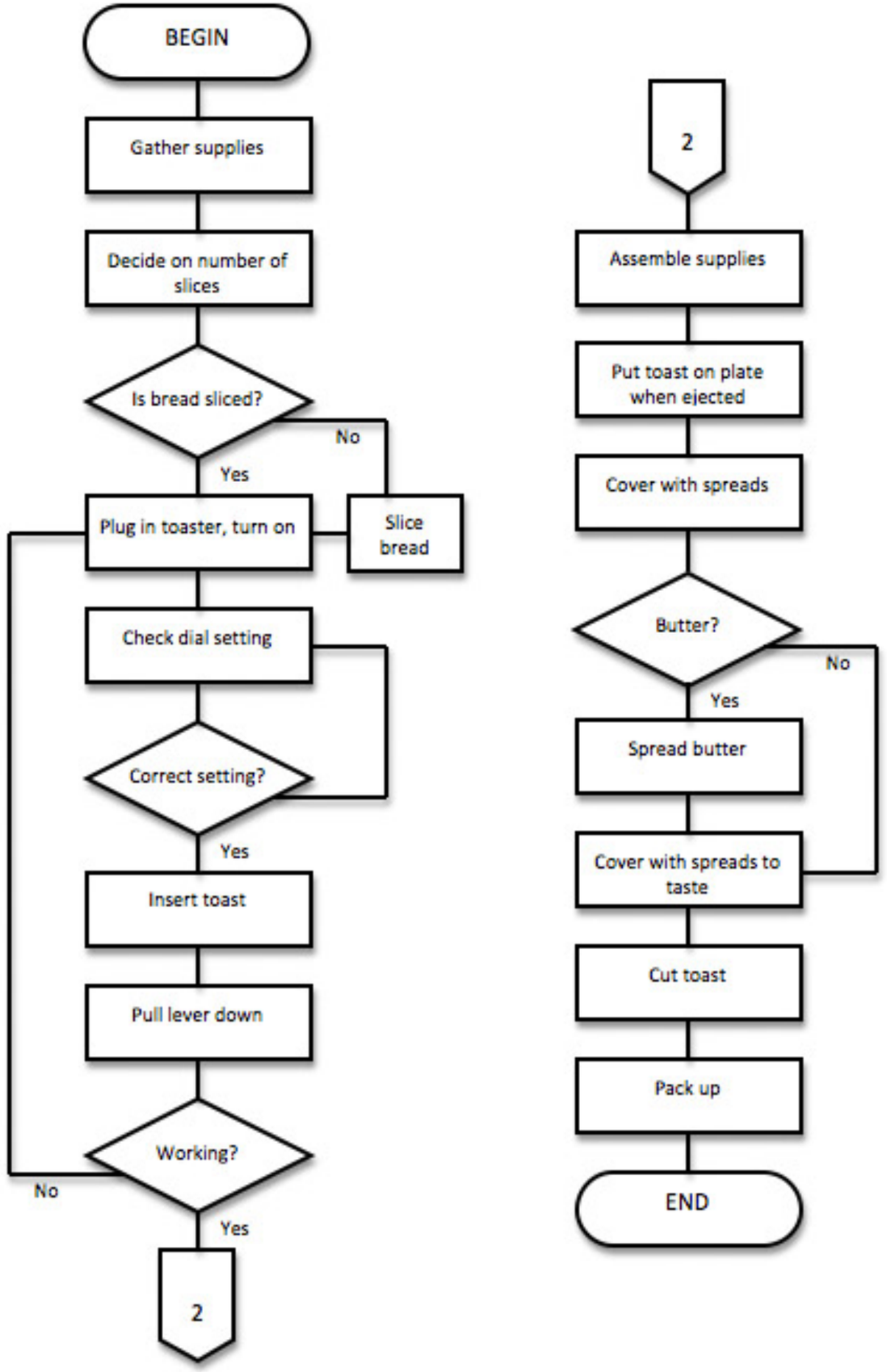
Programming is a skill



“Make me a sandwich”

- Type of sandwich (hot/cold/etc)
- Type of bread
- Ingredients to use/not use
- What equipment do you need?
- Time limit
- Recipes you can look up, or do you have to make your own

Create a flowchart to map out logic



Think, Code, Test, Debug, Repeat

- Think: make sure you understand what you're being asked to do.
  - Recipes = algorithms
- Code: get your hands dirty and experiment with combinations of ingredients, substitutions, repetitive parts.
  - Implementation of algorithm
- Test: determine if final product matches what task was expecting you to produce
  - Actual output matches expected output?
- Debug: tweak your recipe

Understanding the Task

- Never begin to write code right away
- Try to break problem into smaller problems with simpler and smaller steps
- Ask yourself:
  - What is problem supposed to accomplish
  - Are there interactions with the user?
  - What type of input is user giving you?
  - What does user want from program, and in what form?

Writing Pseudocode

- Come up with test cases, special behaviors, and visual representation of sequence of steps
- If you drew sequence of steps, now is the time to put them into words
- Pseudocode: mix of english and programming

Find the area of a circle

- Get radius from the user
- Apply a formula
- Show the result
- Repeat steps 1-3 until the user says to stop

Writing Readable Code

- Use descriptive and meaningful names

```
a = 3.1
b = 2.2
c = a*b*b
```

```
pi = 3.1
radius = 2.2
#use the formula to calculate the area of a circle
circle_area = pi*radius*radius
```

Commenting Your Code

```
pi = 3.1
radius = 2.2
#use the formula to calculate the area of a circle
circle_area = pi*radius*radius
```

- Comments should help others (and yourself) understand why you wrote code in that way.
- Explain why the code is correct to use, and not what the code is implementing
  - #multiply pi times the radius times the radius

Variables and Expressions

Giving Names to Things

- In programming, you reference things by using variables
- In math, lines of equations state an equivalence
  - x = 1
    - x is equivalent to 1
- In programming, lines of code with an equal sign stand for an assignment

Objects

- In Python, everything is an object. Every *thing* you can create in Python has the following:
  - A type
  - A set of operations
- Type: tells you data/values/attributes/properties associated with it
- Operations: commands that you can tell the object to do

Attributes and Operations

- Phone
- Dog
- Mirror
- Credit Card

Objects Have Names

- Every *thing* you create in a program can be given a name so you can refer to it later. These names are *variables*.

```
a = 1
```

```
greeting = "hello"
```

Rules for Object Names

- Must begin with a letter or an underscore
- Other characters in the name can be letters, numbers, or an underscore
- Names are case sensitive
- Names can be any length
- Programming languages have a few reserved words that you can't use as variable names
  - print
  - sum
  - if
  - for

Creating a Variable

- Before you can work with a variable, you have to set it to a value. You *initialize* the variable by assigning it to an object, using the equal sign.

```
int a = 1;
int b = 2;
int c = a + b;
```

```
a = 1
b = 2
c = a+b
```

Object Types

- Four people: Meg, Jo, Beth, Amy
- Three cats: Snap, Crackle, Pop
- Two dogs: Laurel, Hardy

Basic Object Types

- In most programming languages, a few types are the basic building blocks for each language
  - Primitives (or scalars)
- Every other type of object can be made up of combinations of these primitives
- Five basic Python types
  - integers
  - floating point
  - Booleans
  - strings
  - null [absence of a value]

Quick Check

- 2.7
- 27
- False
- “False”
- “0.0”
- -21
- 9999999
- “None”
- None

- Float
- Int
- Bool
- String
- Int
- Int
- String
- Null

Converting Between Types

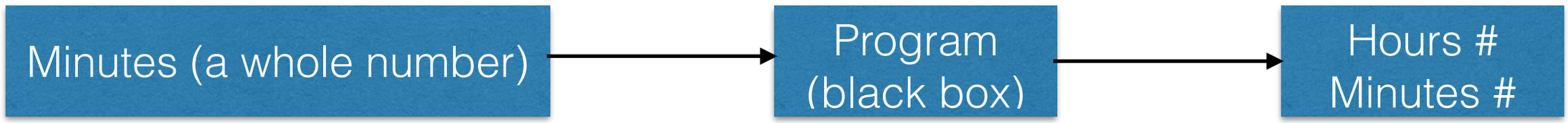
- If you're not sure of the type of an object, use *type()* command
- Surround the object with parentheses and name of the type you want to convert to
  - *float()*
  - *int()*
  - *str()*
  - *int()*

First Python Program

- Write a program in Python that converts minutes to hours. Start with a variable that contains the number of minutes. Your program will take that number, do some calculations, and print out the conversion to hours and minutes
- If the number of minutes is 121, then the program should print

Hours
2
Minutes
1

Think-code-test-debug-repeat



- 60 minutes is converted to 1 hour and 0 minutes
- 30 minutes is converted to 0 hours and 30 minutes
- 123 minutes is converted to 2 hours and 3 minutes

Divide Your Task

- Code to set up the input
  - Initialize a variable with a value
  - minutes_to_convert = 123
- Code to set up output
  - Format required is as follows:

    Hours

    <some number>

    Minutes

    <some number>

Implement Conversion Formula

- Divide the minutes by 60 and convert the result to an integer to give you the whole number of hours
- To find the minutes:
  - Use decimal portion and multiply by 60
  - Use the remainder operator %

Strings, Tuples, and Interacting with the User

Strings

- String is a sequence of characters, and are denoted inside quotation marks.
- Can use “” or ‘’ as long as it’s consistent for one string.
  - “Simple”
  - ‘also a string’
  - “a long string with Spaces and special sym&@L5_!”
  - “525600”
  - “”
  - ‘’

Basic Operations

- Creating a string object

```
num_one = "one"
num_two = "2"
```

- Understanding indexing into a string
- In CS, you start counting from 0

P	y	t	h	o	n		r	u	l	e	s	!
0	1	2	3	4	5	6	7	8	9	10	11	12
-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

- “Python rules!”[0] evaluates to ‘P’
- “Python rules!”[7] evaluates to r
- cheer = “Python rules!” and then cheer[2] gives ’t'

Basic Operations on a String

- Slicing *[start_index:stop_index:step]*
  - cheer = “Python rules!”
  - cheer[2:7:1] evaluates to ‘thon ‘
- String length *len()*
  - *len(*“Boston 4 ever”*)* evaluates to 13
- Converting between letter cases
  - cheer.lower()
  - cheer.upper()
  - cheer.swapcase()
  - cheer.capitalize()

Other Operations on Strings

- find()
- count()
- replace()
- concatenation and multiplication

```
"some_string".find('g')

fruit = "banana"
fruit.count('an')

"variables have no spaces".replace(" ", "_")

"one" + "two"

3 * "a"
```

Tuple Objects

- Strings store sequences of characters
- More convenient if there were a way to store individual objects in a sequence
- Tuple is a data type that represents sequences of objects.
- (1, “a”, 9.9)
- ()
- (1, 2, 3)
- (“a”, “b”, “cde”)
- (1, “2”, False)

Operations on Tuples

- len()
  - len((3, 5, “7”, ”9”))
- indexing into and slicing
  - (3, 5, “7”, “9”)[1] evaluates to 5
  - (3, (3,5), “7”, “9”)[1] evaluates to (3,5)
  - (3, (3, (“5”, 7), 9), “a”)[1][1][1] evaluates to?
- concatenation and multiplication

Interacting with the User

- Use the *print* command to show the user values on the console
  - print(“hello!”)
  - print(3*2)
  - print(“abc”+”def”)
- Use *input()* command to get input from the user
  - input(“What’s your name? “)

Storing Input into a Variable

- Anything a user types is converted into a string object

```
1  user_place = input("Where do you live? ")
2  text = user_place.capitalize() + "!"
3
4  print(text)
5  print("I hear it's nice there!")
```

Decisions

Conditionals

```
1  num = int(input("Enter a number: "))
2  if num > 0:
3      print("num is positive")
4  print("finished comparing num to 0")
```

- What happens if num has a value of 5?
- What happens is num has a value of 0?

Making Many Decisions

```
1  num = int(input("Pick a number: "))
2  if num > 0:
3      print("Your number is positive")
4  if num < 0:
5      print("Your number is negative")
6  if num == 0:
7      print("Your number is zero")
8
9  print("Finished")
```

Nested Conditionals

```
1  num_a = int(input("Pick a number: "))
2  num_b = int(input("Pick a number: "))
3  if num_a < 0:
4      print("num_a is negative")
5      if num_b < 0:
6          print("num_b is negative")
7
8  print("Finished")
```

```
1  num_a = int(input("Pick a number: "))
2  num_b = int(input("Pick a number: "))
3  if num_a < 0:
4      print("num_a is negative")
5  if num_b < 0:
6      print("num_b is negative")
7
8  print("Finished")
```

Do This or That

```
num_a = 5
num_b = 7
lucky_num = 7

if(type(num_a)) != int or type(num_b) != int:
    print("You did not enter integers")

else:
    if num_a > 0 and num_b > 0:
        print("Both numbers are positive")
    elif num_a < 0 and num_a < 0:
        print("both numbers are negative")
    else:
        print("Numbers have opposite sign")
if num_a == lucky_num or num_b == lucky_num:
    print("You also guessed my lucky numbers!")
else:
    print("I have a secret number in mind")
```

Loops

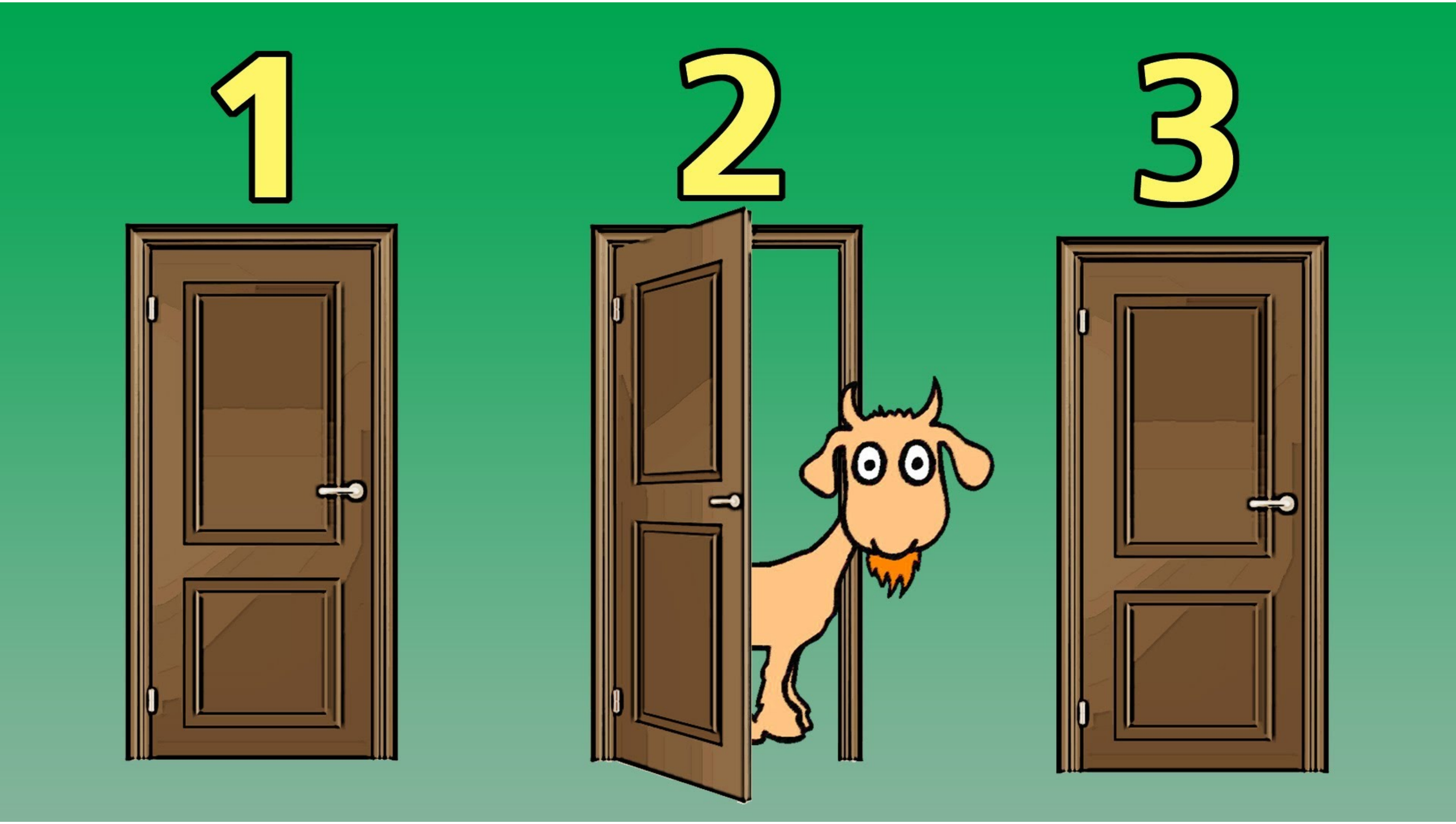
For Loops

for <loop variable> in <values>:
** <do something>**

range(start, end, step)

```
1  for i in range(4):
2      print("echo")
3
4
5  for ch in "Python is fun so far!":
6      print("The character is "+ch)
7
```

Monty Hall Problem



http://www.mathwarehouse.com/monty-hall-simulation-online/

Monty Hall Simulation Online

Play the Monty Hall game or run the simulation to better understand what might be one of the most famous math riddles ever.

Pick one of three doors

Change Choice	cars:0 50%	goats:0 50%
Keep Choice	cars:0 50%	goats:0 50%

Simulation

Run simulation 10 times And keep the choice Simulate

$$y = \frac{\sqrt{x^2 + 2}}{5 +}$$

Enter YOUR math problem...

Answer

Algebra
Geometry
Trigonometry
Calculus
Worksheets
Math Gifs
Teacher Tools

From Mathwarehouse

Make a Graph
Graphing Calculator

Partner Sites

$$y = \frac{\sqrt{x^2 + 2}}{5 +}$$

```python
import random

items = ['goat', 'goat', 'car']
num_trials = 10000
num_wins = 0

for trial in range(num_trials):
    random.shuffle(items)
    player = random.randrange(3)
    monty = next(i for i,v in enumerate(items)if i!= player and v!= 'car')
    player = next(x for x in range(3) if x not in (player,monty))
    if items[player] == 'car':
        num_wins+= 1

print('{}/{}={}'.format(num_wins, num_trials, num_wins/num_trials))
```

Functions

Building programs to last

```python
a = 1
b = 2

print(a+b)
print(a-b)
print(a*b)
print(a/b)

```

```
3
-1
2
0.5
```

```python
a = 1
b = 2
print(a+b)
print(a-b)
print(a*b)
print(a/b)

a = 3
b = 4
print(a+b)
print(a-b)
print(a*b)
print(a/b)


a = 5
b = 6
print(a+b)
print(a-b)
print(a*b)
print(a/b)

```

```
a = 1
b = 2
<wrapper for operations_with_a_and_b>
a = 3
b = 4
<wrapper for operations_with_a_and_b>
a = 5
b = 6
<wrapper for operations_with_a_and_b>
```

```python
def take_attendance(classroom, who_is_here):
    """
    classroom: tuple
    who_is_here: tuple

    Checks if every item in classroom is in who_is_here
    and prints their name if so
    Returns "finished taking attendance"

    """
    for kid in classroom:
        if kid in who_is_here:
            print(kid)


classroom = ['Alice', 'Bob', 'Carol', 'Dave']
here = ['Bob', 'Carol']

take_attendance(classroom, here)

```

```
1  def get_word_length(word1, word2):
2      word = word1+word2
3      return len(word)
4      print("This never gets printed")
5
```

Calling a Function

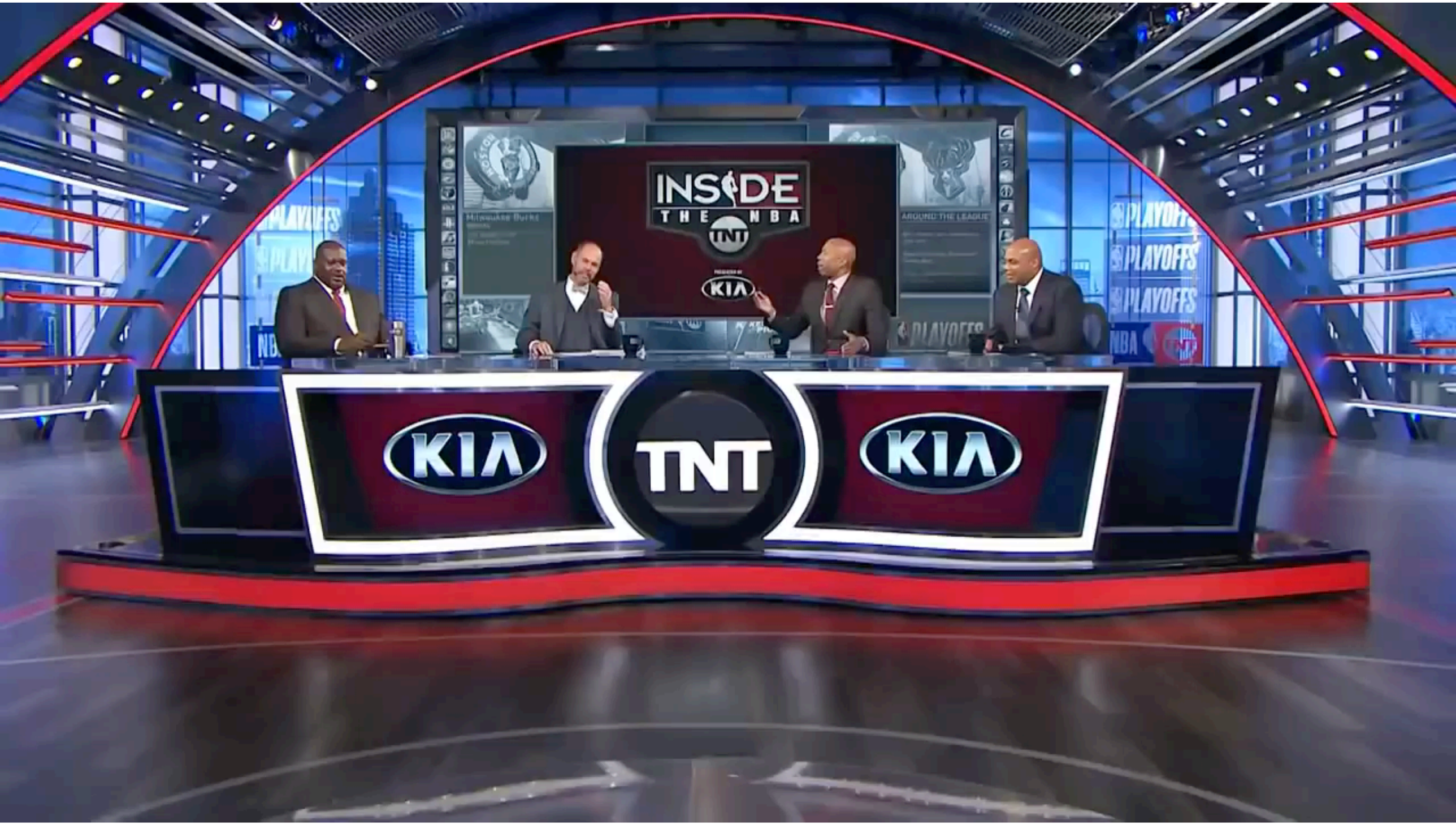
```
1  def get_word_length(word1, word2):
2      word = word1+word2
3      return len(word)
4
5  length1 = get_word_length("Grace","Hopper")
6  length2 = get_word_length("Katherine", "Johnson")
7  length3 = get_word_length("Mae", "Jemison")
8
9  print("One name is "+str(length1)+" letters long.")
10 print("One name is "+str(length2)+" letters long.")
11 print("One name is "+str(length3)+" letters long.")
12
```

Returning more than one value

```
1  def add_sub(n1,n2):
2      add = n1+n2
3      sub = n1-n2
4      return (add,sub)
5
6  (a,b) = add_sub(3,4)
7  print(a)
8  print(b)
9
```

Shaq's Price of Gas

https://www.youtube.com/watch?v=EuH91bQXDuE



```python
def gas_price(mtg, mpg, gpt, dpt):
    price = mtg/mpg/gpt*dpt
    return price

def shaq_price():
    return "???"

def main():
    miles_to_go = 800
    miles_per_gallon = 20
    gallons_per_tank = 10
    dollars_per_tank = 80

    price = gas_price(miles_to_go, miles_per_gallon, gallons_per_tank, dollars_per_tank)
    shaq = shaq_price()

    print("Price of gas is: ${}".format(price))
    print("Shaq's price of gas is: ${}".format(shaq))

main()
```

Lists

Collection of any object type.

```
grocery = ["milk", "eggs", "bread"]

len(grocery)

print(grocery[0])
print(grocery[1])
print(grocery[2])

```

```
years = [1984, 1986, 1988, 1988]

len(years)

years.count(1988)
years.count(2017)
years.index(1986)
years.index(1988)

```

```
first3letters = []
first3letters.append('a')
first3letters.append('c')
first3letters.insert(1,'b')
print(first3letters)


last3letters = ['x','y','z']
first3letters.extend(last3letters)
print(first3letters)

last3letters.extend(first3letters)
print(last3letters)

```

```
1  polite = ['please', 'and', 'thank', 'you']
2
3  polite.pop()
4
5  polite
6
7  polite.pop(1)
8
9  polite
```

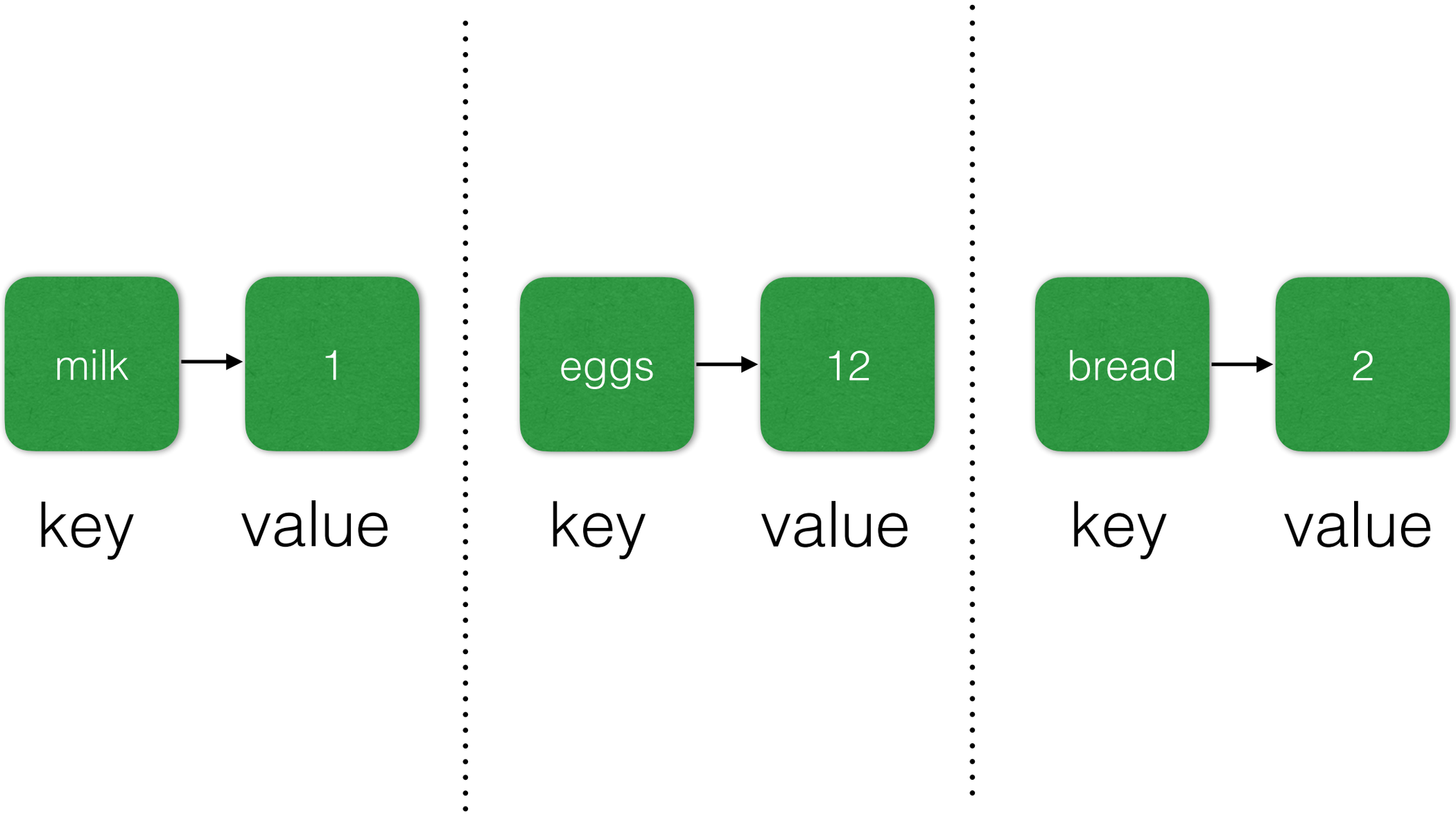
```
1  colors = ['red', 'blue', 'yellow']
2
3  colors[0] = 'orange'
4
5  colors[1] = 'green'
6
7  colors[2] = 'purple'
```

Dictionaries

Pairs of words/Maps between objects

```
grocery = {"milk" : 1, "eggs" : 12, "bread" : 2}

```



```
legs = {}
legs['human]'] = 2
legs['cat'] = 4
legs['snake'] = 0

len(legs)

legs['cat']=3
len(legs)
legs
```

```
household = {'person':4, 'cat':2, 'dog':1, 'fish':2}

removed = household.pop('fish')

print(removed)

```

```python
grades = {}

grades['Alice'] = [100,70]
grades['Bob'] = [90, 100]
grades['Carol'] = [80, 40]
grades['Dave'] = [70,70]

for students in grades.keys():
    print(students)

for quizzes in grades.values():
    print(str(sum(quizzes)/2))

for student in grades.keys():
    scores = grades[student]
    grades[student].append(sum(scores)/2)

print(grades)
```

Building a frequency dictionary

```
lyrics = "Happy birthday to you Happy birthday to you Happy birthday
    dear Happy birthday to you"

counts = {}

words = lyrics.split(' ')
for w in words:
    w = w.lower()
    if w not in counts:
        counts[w] = 1
    else:
        counts[w] += 1
print(counts)

```

Building unconventional dictionaries

```
def square(x):
    return x*x

def circle(r):
    return 3.14*r*r

def equilateraltriangle(s):
    return (s*s)*(3**0.5)/4

areas = {'sq':square, 'ci':circle, 'eqtri':equilateraltriangle}

n = 2

print(str(areas['sq'](n)))

print(str(areas['ci'](n)))

print(str(areas['eqtri'](n)))

```

Shaq vs The Moon

https://www.youtube.com/watch?v=spcJW2MQHnE