CS50 Lecture 6: Python

Jordan Mandel

February 17, 2022

Intro

```
print("hello, world")
```

can get strings

```
from cs50 import get_string
answer = get_string("What's your name? ")
print("hello, " + answer)
```

F strings

```
print(f"hello, {anser})
```

if/else:

```
if x < y: #lala
    print("x is less than y")
elif x > y:
    print("x is greater than y")
else:
    print("x is equal to y")
```

boolean expression:

```
while True:
    print("hellow, world")
```

Loops

we can write a while loop:

```
i=0
while i < 3:
    print("hello, world")
    i += 1
```

can write for loop:

```
for i in [0, 1, 2]: // this is like a list
    print("cough")
```

`range(3)` gives `[0, 1, 2]` `range(0, 101, 2)` goes from `0` to `100` in increments of `2`.

Instead of do-while

```
def get_positive_int():
    while True:
        n = get_int("Positive Integer: ")
        if n > 0:
            break
    return n
```

we can print an integer with print `i`.

data types

- `bool`: `True`, `False`
- `float`
- `int`
- `str`

More complex types include

- `range`: sequence of numbers
- `list`: sequence of mutable values: can grow or shrink
- `tuple`: tuple: collection of ordered values
- `dict`: key value pairs
- `set`: unique values with no duplicates

CS50 Library

- `get_float`
- `get_int`
- `get_string`

Can import individual functions

```
from cs50 import get_flost
from cs50 import get_int
from cs50 import get_string
```

or

```
from cs50 import get_float, get_int, get_string
```

or

```
import cs50
```

Examples

Blur

- It's a higher level language

```
from PIL import Image, ImageFilter
before = Image.open("bridge.bmp")
after = before.filter(ImageFilter.BoxBlur(1))
after.save("out.bmp")
```

- Can change arguments to BoxBlur to see how many are included.

Dictionary

```
words = set() # just a collection of values

def check(word):
    if word.lower() in words:
        return True
    else:
        return False

def size():
    return len(words)

def unload()
    return true

def load(dictionary):
    file = open(dictionary, "r")
    for line in file:
        words.add(line.rstrip())
file.close()
return True
```

input

```
# can get a string
input("what's your name? ")

x = int(input("x: "))
# can cast as integer
y = int(input("y:"))
print(x + y)
```

- In this case `int` can return error if we mess it up by putting something in that's not an int.

Division

In this case 1/2 returns .5 whereas in c it would have returned zero

```
print(1/2)
```

Conditions

```
from cs50 import get_int
x = get_int("x: ")
y = get+int("y: ")

if x < y:
    print("x less than y")
elif x > y:
    print("x is greater than y")
else:
    print("x is equal to y")
```

aggree.py

```
from cs50 import get_string
s = get_string("Do you agree?")

if s == "Y" or s == "y":
    print("Agreed.")
elif s == "N" or s == "n":
    print("Not agreed")
```

- can use single or double quotes or single quotes in python.
- everything that is a character is a string in python

can do `if s.lower() in ["y", "yes"]`

meow.py

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

or

```
def main()
    for i in range(3):
        meow()

def meow():
    print("meow")

#if __name__ == "__main__":
main()
```

- can change it to meow(3) by moving it into the loop.

positive.py

```
from cs50 import get_int
    i = get_positive_int()
    print(i)

def get_positive_int():
    while True:
        n = get_int("Poasitive int: ")
        if n > 0:
            break
    return(n)
```

- Loops don't have their own scope

Mario.py

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

to get rid of newline

```
for i in range(4):
    print("?", end = "")
```

Or can go

```
print("?" * 4)
```

or

```
for i in range(3):
    for j in range(3):
        print("#", end = "")
    print()
```

int.py

```
i = 1
while True:
    print(i)
    i *=2
```

Ints are a finite size in C: we won't have integer overflow in python. We still have floating point imprecision: but there are libraries that allow us to be more precise.

scores.py

```
scores = [72, 73, 33]
print("Average: " +  str(sum(scores) /len(scores)))
```

Can use a format string.

```
scores = [72, 73, 33]
print(f"Average: {sum(scores) / len(scores)}" )
```

Could have used variable in f string.

```
from cs50 import get_int

scores = []
for i in range(3)
    scores.append(get_int("Score: "))

average = sum(scores) / len(scores)
print(f"Average: {average}")
```

uppercase

```
from cs50 import get_string

s = get_string("Before: ")
print("After: ", end="")
# print(s.upper())
for c in s:
    print(c.upper(), end = "")
print()
```

argv

```
from sys import argv
if len(argv) == 2:
    print(f"hello, {argv[1]}")
else:
    print("hello, world")
```

or could do

```
for arg in argv:
    print arg
```

exit.py

```
import sys # have to mention the package
if len(sys.arv) != 2:
    print("missing command-line argument")
    sys.exit(1)

print(f"helo, {sys.argv[1]}")
sys.exit(0)
```

numbers.py

```
import sys

numbers = [4, 5, 8, 2, 7, 4, 0]

if 0 i numbers:
    print("fount")
    sys.exit(0)
else:
    print("not found")
    sys.exit(1)
```

there are dictionaries

```
from cs50 import get_string

people = {
    "Brian": "+1-617-945-1000",
    "David": "+1-949-458-2740"
    }

name = get_string("Name: ")
if name in people:
    print(f"Number: {people[name]}")
```

swap.py

```
x = 1
y = 2
```

```
x, y = y, x
```

csv

```
import csv
from cs50 import get_string

file = open("phonebook.csv", "a")

name = get_string("Name: ")
number = get_string("Number: ")

writer = csv.writer(file)
writer.writerow([name, number])

file.close()
```

- using `with` keyword to avoid having to write file.close

```
import csv
from cs50 import get_string

number = get_string("Number: ")
name = get_string("Name: ")

with open("phonebook.csv", "a") as file:
    writer = csv.writer(file)
    writer.writerow([name, number])
```

Hogwarts

```
import csv

houses = { #this is a dictionary
 "Gryffindor": 0,
 "Hufflepuff": 0,
 "Ravenclaw": 0,
 "Slytherin": 0
 }

with open("sorting Hat - Form Responses 1.csv", "r") as file:
    reader = csv.reader(file)
    next(reader) #ignores first row
    for row in reader:
        house = row[1] # gives the second column of the first row
        houses[house] += 1

for house in houses:
    print(f"{house}: {houses[house]}")
```

speech

```
import pyttsx3

engine = pyttsx3init()
name = input("whats your  name")
engine.say(f"hello {name}")
engine.runAndWait()
```

facial detection

```
from PIL import Image
import face_recognition

# Load the  jpg file into a numpy array
image = face_recognition.load_image_file("offiece.jpg")

face_locations = face_recognitoin.face_locations(image)

for face_location in face_locations:
    #print fae locations
    top, right, bottom, left = face_location

    face_image = image[top:bottom, left:right]
    pil_image = Image.fromarray(face_image)
    pil_image.show()
```

qr

```
import os
import qrcoede
img = qrcode.make("nick.com")
image.save("qr.png", "PNG")
os.system("open qr.png")
```

recognition

```
recognizer = speech_recognition.Recognizer()
with speechrecognition.Microphone() as source:
    print("say something: ")
    audio = recognizer.listen(source)

words = recognier.recognize_google(audio)


words = input("Saying something: ").lower()

if "hello" in words:
    print("Hello to you too!")
elif "how are you" in words:
    print("ladeedaadeedaaa")
else:
```

```
    print("Huh?")
```

python short

Recall in c we had

```
bool alphabetic = isalpha(var) ? true : false
```

```
letters_only = True if input().isalpha() else False
```

list comprehension (need to study more)

```
nums = [x for x in range(500)]
```

other ways to do this

```
nums = list()
nums = [1, 2,3,4]
nums.append(5)
nums.insert(4,5)
nums[len(nums):] = [5] #tacking on something
```

tuples

an ordered immutable set of data

```
# a list with some tuples included
presidents = [
("George washington", 1789), ("marvin meely", 1453)
]

for prez, year in presidents: # might have use enumerate
    print("In {1}, {0} took office".format(prez, year))
```

Dictionaries

```
pizzas = {
    "shese": 9,
    "pyanapl": 3,
    "srhinspms": 3}
    }
```

can go:

```
for pie in pizzas: # might be difficult to determine the order of things.

pizzas["cheese"] = 8 # can be an update or an append

for pie, price in pizzas.items(): # transforms into list
    print(price)
    print("a whole {} pizza costs ${}".format(pie, price))
    print("a whole " + pie + "pizza costs %" + str(price))  # can also append with plus signs
    print(" a whole %s pizza costs $%2d" % (pie, price)) # deprecated
```

fuctions

```
if __name__ == "__main__":
    main() # here we go with starting the main progrm in some cases.
```

- Defining a function

```
def square(x):
    return x ** 2
```

Could also do `x * x` or

```
def square(x)
    result = 0
    for i in range(0, x):
        result += x
    return result
```

objects

In `C` for example: struct car { int year; char* model; }

The properties are tied to the struct. You can't initialize the struct and then use a property inside the struct. In C the objects also have. We are constructing an object.

```
class Student(): # have to include the  self keyword

    def __init__(self, name, id): # this is the constructor.
        self.name = name
        self.id = id

    def changeID(self, id):
        self.id = id

    def print(self):
        print("{} - {}".format(self.name, self.id))
```

We can go `import cs50` and then be like `cs50.get_int()`

- can include shebang `#!/usr/bin/env python3` to just invoke the file without prepending `python3`.

Additional Notes

Subsitute for %in% operator

```
x = [2, 3, 5]
y = [1, 2, 3]

# for loop
for i in x: [].append(i in y)

Out: [True, True, False]


# list comprehension
[i in y for i in x]

Out: [True, True, False]
```