Allah says

عَلَّمَهُ الْبَيَانَ o

He has taught him speech (and intelligence).

Al-Qur'an, 055.004 (Ar-Rahman)

Lecture-2.1
Python data

Fayyaz ul Amir Afsar Minhas, Ph.D.

fayyazafsar@gmail.com

Department of Computer and Information Sciences
Pakistan Institute of Engineering and Applied Sciences (PIEAS)
P.O. Nilore, Islamabad.

Lecture Plan

- Review of assignment
- Revision of concepts of objects
- Revision of concepts of Python variables
- Revision of python data structures
  - Mutable
  - Immutable
  - Strings
  - Tuples
  - List
- Revision of indexing: a[::-1], a[::-2]
- More on lists and sets
- What are Python variables exactly?
- What is Deep copy?
- What is ducktyping?
- What are dictionaries? Basic concepts of hashing
- Why do we have Mutable and Immutable types in Python?

Assignment

```
def cleanup(fname):
    with open(fname,'r') as ofile:
        text=ofile.read()
    return text,text.replace(' ','').lower()

def encrypt(ptext,shift):
    k=''
    for c in ptext:
        k+=chr((ord(c)-97+shift)%26+97)
    return k

def decrypt(ctext,shift):
    ptext=''
    for c in ctext:
        ptext+=chr((ord(c)-97-shift)%26+97)
    return ptext

def breakCipher(text):
    import numpy as np
    abc='abcdefghijklmnopqrstuvwxyz'
    psh=np.argmax([text.count(i) for i in abc])-5+1
    return psh
```

```
# Alternate Cleanup
```

```
def cleanup(file):
    st, cs = '', ''
    f = open(file)
    for line in f:
        st += line
    f.close()

    for i in st:
        if(ord(i)>=65 and ord(i) <=90):
            i = chr(ord(i)+32)
        if(ord(i)>=97 and ord(i) <=122):
            cs += i
    return st, cs
```

Assignment Alternate Solutions

```
# Opening the file differently
    ofile = open(fname,'r')
    text=ofile.read()
    ofile.close()
```

```
# Finding the most commonly occurring element
c  = ‘’
V = 0
For d in abc:
        v_ = mystring.count(d)
        if v_ > v:
                v = v_
                c = d
```

```
# Finding the most commonly occurring element
from collections import Counter
c = Counter(enc_text)
m_common=c.most_common(1)[0][0]
```

Poetry & Python Programming

```
def clean(ifile):
        with open(ifile, 'r') as f:
                istr = f.read().lower()
        return ''.join(c for c in istr if 122>=ord(c)>=97)


def encrypt (istr,shift): return ''.join(chr((ord(c)-97+shift)%26+97) for c in istr)

def decrypt (istr,shift): return encrypt (istr, -shift)

########################

# OR create a lambda function
# decrypt = lambda istr,shift : encrypt (istr,-shift)

# Testing
decrypt(encrypt(istr,shift), shift)==istr
```

Potential Issues

- Overchecking!
  - Use exception handling instead
- sys.exit(0)

Quick class assignment

- Generate a random string of 100 lower case ASCII characters
- Given any string, find all the unique characters in it

Objects

- Python uses objects for abstraction of data
- All data in Python is represented by objects or relations between objects
- Every object has, among others, a:
  - Identity: A unique number
  - Type: What is the class of this object
  - Value: What is the value
- An object also has behavior and properties

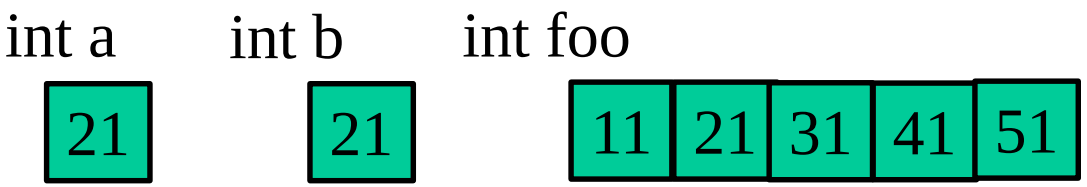
Python data model

- Given any object '`obj`' or data you can:
  - Use `dir(obj)` to get the list of all its properties and behaviours
  - Use `id(obj)` to get its Identity
  - Use `type(obj)` to get its type
  - And access its value by simply typing `obj`
  - Use `obj1 is obj2` to see if both `obj1` and `obj2` actually refer to the same object

Self Reading:
https://docs.python.org/2/reference/datamodel.html

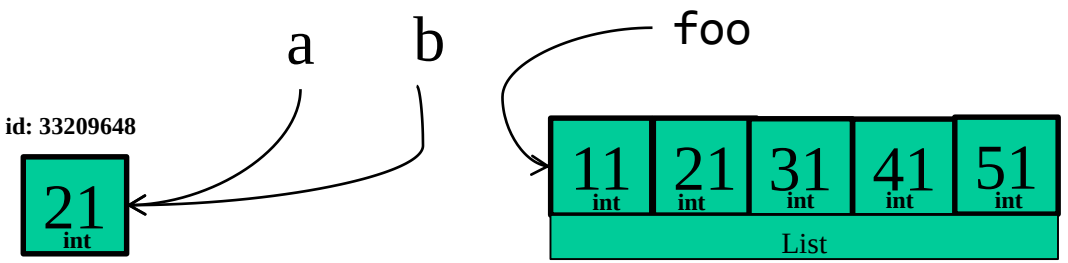
Python variables

- Major difference of Python with other languages is how its variables work
- In a language like C++, when you create a variable you essentially create a box in memory with the value of variable written in it and the variable identifying it.
  - The variable has type and object value




```
int a,b;
a=21;
b=a;
int foo [5] = {11,21,31,41,51};
```

- In Python, a variable is essentially a reference
  - The object has type, id and value whereas the variable is untyped




```
a=21
b=a
foo = [11,21,31,41,51]
```

Python Data types

- Sequences
  - Immutable
    - Strings: a='hello'
    - Unicode
    - Tuples: a=(1,2)
  - Mutable
    - Lists: a=[1,2]
    - Byte Array
- Set types
  - Set: a={1,2}
  - Frozen Set
- Mapping
  - Dictionary: a={1: 2, 2: 4}

```
str(arg)
list(arg)
tuple(arg)
len(arg)
Seq[i]
Seq[i:j]
Seq[i:]
Seq[:j]
Seq[i:j:step]
Seq[-i]
```

```
set(arg)
```

```
dict(arg)
```

Revision of indexing

- What will x[::-1] do?
- What about x[::-2]?

Lists and Sets

- What are Python Variables Exactly?
- What is meant by deep copying?
- Conceptual Challenge

```
a = 1
b = 1
print id(a)
print id(b)
print a is b

a = 1000
b = 1000
print id(a)
print id(b)
print a is b
```

OOP and Ducktyping

- What is duck typing?
  - Cover it in OOP
  - http://bitsquid.blogspot.com/2011/02/some-systems-need-to-manipulate-objects.html
  - https://github.com/pydanny/pydanny-event-notes/blob/master/Pycon2013/keynotes.rst

Mutability

- http://nedbatchelder.com/text/names.html
- http://stackoverflow.com/questions/9755990/why-can-tuples-contain-mutable-items

- Cover in mutability
  - Why do we have mutable and immutable types?
  - Why can tuples contain mutable items?

End of Lecture-3