CSCI 120
Introduction to CompSci and Programming I
Lec 0: Introduction

Jetic Gū

Overview

- Focus: Python Programming
- Architecture: von Neumann
- Core Ideas:
  1. Administration
  2. Review some Data Structures
  3. It will be quick today

Administration

P0
Admin

Course Arrangement

- Week 8-13: Jetic
  - Instructor @Columbia College, PhD Candidate in Computational Neuroscience
- Zoom Link
  - Well, you are already here
  - Lectures: the sooner we finish, the sooner we finish
- Course Website
  - https://jetic.org/kurs/csci-120/

Concept

P0
Admin

Grading Scheme

- 5 Labs, released on Wednesday
- 1 Assignment
- Small Test
- Final Exam
    - About tests and exams, I **DO NOT** require you to turn on your **camera**. Most likely I will be working on **stuff** when you take the test, but I will check your code to see if it is copied.

Concept

Questions?

Data Structures

Lists, Strings, Tuple, Dictionary, etc.

Data Structures

- Data structures are particular ways of storing data to make some operation easier or more efficient. That is, they are tuned for certain tasks

- Data structures are suited to solving certain problems, and they are often associated with algorithms.

Kinds of data structures

Roughly two kinds of data structures:

- built-in data structures, data structures that are so common as to be provided by default
- user-defined data structures (classes in object oriented programming) that are designed for a particular task

Python built in data structures

- Python comes with a general set of built in data structures:
  - lists
  - tuples
  - string
  - dictionaries
  - sets
  - others...

Lists

- An ordered group of items
- Does not need to be the same type
- List notation

```
A = [1, ”This is a list”, c]
```

Methods of Lists

- `alist.append(x)`
  - adds an item to the end of the list
- `alist.extend(L)`
  - Extend the list by appending all in the given list L
- `alist.insert(I,x)`
  - Inserts an item at index I
- `alist.remove(x)`
  - Removes the first item from the list whose value is x

Concept

Examples of other methods

- Define and print List, then count values

```
a = [66.25, 333, 333, 1, 1234.5]
print (a.count(333), a.count(66.25), a.count('x')) //calls method
2 1 0      //output
```

- Return the first index where the given value appears

```
a.index(333)
1    //output
```

- Reversing a list **(in place)**

```
a.reverse() //Reverses order of list
[333, 1234.5, 1, 333, -1, 66.25]
```

- Sorting **(in place)**

```
a.sort()
[-1, 1, 66.25, 333, 333, 1234.5]
```

Concept

The del statement

- Deleting at index: `del a[0]`
- Deleting at index range: `del a[2:5]`

Concept

Tuples

- Tuple
  - A number of values separated by commas
  - Immutable
    - Cannot assign values to individual items of a tuple
    - However tuples can contain mutable objects such as lists
  - Single items must be defined using a comma
    - `Singleton = 'hello',`

Concept

Sets

- An unordered collection with no duplicate elements

```
Basket = ['apple', 'orange', 'apple', 'pear']

Fruit = set(basket)

Fruit // Set(['orange', 'apple', 'pear'])
```

Dictionaries

- Indexed by **keys**
    - This can be any immutable type (strings, numbers…)
    - Tuples can be used if they contain only immutable objects

Looping Techniques

- `adict.items()`
  - for retrieving key and values through a dictionary

```
stuff = {
    'name': 'Jetic',
    'age': '44',
    'favourite food': 'cheese'}
for k, v in stuff.items():
    print("His " + k + " is " + v)
```

Concept

Looping Techniques

- `enumerate(alist)`
  - for the position index and values in a sequence

```
for i, v in enumerate(['a', 'b', 'c']):
    print(i, v)

// Output
0 a
1 b
2 c
```

Looping Techniques

- `zip()`
  - for looping over two or more sequences

```
key = ['name', 'age', 'favourite food']
val = ['Jetic', '44', 'cheese']
for k, v in zip(key, val):
    print("His " + k + " is " + v)

// Result
His name is Jetic
His age is 44
His favourite food is cheese
```

Comparisons

- Operators “in” and “not in” can be used to see if an item exists in a sequence
- Comparisons can be chained
    - `a < b == c`
      This tests whether a is less than b and that b equals c

Environmental Setup

Linux, Command Line, Vim

Summary

Introduction to Linux

- It’s just another OS
- It’s open source
    - You can see the kernel code here: https://github.com/torvalds/linux
- It’s most useful in a wide-range of fields
    - ALL major Online Servers, even Microsoft’s own Webservers / cloud system
    - ALL scientific computers, including smallest cluster to SFU’s super computers
    - Used in ALL software development except for Windows apps

For This Course

- You will need access to Linux or Unix-based OS (macOS is Unix)
  - Use Linux/Unix as your primary desktop OS
  - Dual OS desktop
  - Install a Linux distribution in Virtual Machine
  - Install Windows 10's Linux subsystem (SLOWWWWWW)
- You need to be comfortable with command line

Common Questions

- Do we have to?
  - Yes. In your later life, you can use whatever GUI based code editor. But you must know how to work using command line only, so for this course you must complete all coding assignments using command line.
- Why is macOS ok but not Windows?
  - macOS is a UNIX operating system which shares many common features with Linux, such as intuitive command line. In fact most commands and operations are identical, including the way we compile and debug everything.
  - Windows on the other hand is more cumbersome. Even Microsoft employees use linux/mac for webdev and building cloud services.

P3
Linux

List of Linux commands you will need

- Open Terminal, a command line prompt will show up
  - `ls`: list present directory
  - `cd`: change directory
  - `cat`: view file as text
  - `man`: see manual
  - `vim`: text editor

We will have a short tutorial tmr.

Technical