0. Introduction

Topics

- Installing
    - -- versions
    - -- virtualenv
- Sublime
    - -- Package Manager (Ctrl+Shift+P), plugins, console, settings
    - -- PyCharm
- Start
    - -- hello world
    - -- expressions
    - -- vars
    - -- idea of types as a set of associated operations:
        - -- Strings (‘, “, “”“): multiplication, concatenation, slicing, indexing.
        - -- Numbers, integers, real: power, different types of division //, /, %.
        - -- Arrays (string as an array of chars): length, map, filter, reverse.
        - -- Boolean: only 2 values with negation relation.

Class work

- install python, sublime, build
- run “Hello, <Name>” from console and from module
- arithmetic expressions
- var of different types: sum of 2 numbers (strings, but not string and number)

Home work

- install python, sublime, virtualenv
- try different versions
- create a simple program that prints “Hello, <Name>”

1. Strings and lists

Class work

- string manipulation
- list manipulation
- list as a collection of elements
- string as a list of chars

Home work

- Coding bat (http://codingbat.com/python/String-1)
- Create char pattern using string multiplication (i.e. “/\/\/\/\”)
- Print several pattern to create dotted square
- return the last/first/3rd/Nth element of a string
- swap the 1st and last chars in a string
- uppercase, lowercase all string/1st/last/3rd

- *above but for even/odd/every 3rd elements
- find the index of the 1st ‘a’

2. Conditions

Topics

- Conditions
  - -- simple, chaining, example of difference between elif and 2 ifs
  - -- check range inclusion (Python way!)
- Negation
  - -- boolean type
  - -- (not) empty list, >0

Class work

- isEven
- Find largest from 2
- Coding bat (http://codingbat.com/python/Logic-1, 2)

Home work

- Coding bat (http://codingbat.com/python/Logic-1, 2)

3. Functions & modules

Topics

- Modules
  - -- modules as a way to handle complexity
  - -- py files
  - -- importing system modules
  - -- writing user modules, importing them (with vars example)
- Functions
  - -- syntax, positioned parameters, named parameters
  - -- name scope, collisions
  - -- builtin functions
  - -- functions composition
  - – lambdas

Class work

- isEven (as function)
- Find largest from 2 (as function)
- Power of number
- Square of number
- Fibbonachi
- Greatest Common Denominator
- *zip/map/filter

Home work

- Create different modules (~libs) for different topics
- Use reverse() function to reverse list
- Create function that sums 2 numbers
- Create function that adds tags to the passed string

4. Loops

*Topic*s:

- Loops and iterations
    - -- for/while loop
    - -- continue/break
    - -- enumeration for lists

Class work:

- Create string pattern
- Create Pythagoras table
- Printing pyramid
- Find largest, smallest
- Reverse function

Home work:

- Sum all elems in the list
- Sum all elems that are >N
- Sum all elems that are >N and <M
- Calc diff of all positive and negative
- Calc diff of all >N and <=N (when learn lists show how it is also possible with filters)
- Given a list of strings, print all elements which length is >5,

5. Further types exploration

*Topic*s:

- dictionaries
- tuples (swap, return 2+ values)
- arrays

Class work:

- create simple phone book

Home work:

- extend phone book

6. Classes

Topics:

- motivation: complexity is handled with abstraction (main idea w/o details. example: car)
- syntax
- difference between class and instance: blueprint or template vs real equipment or produced item
- private/public
- dynamic class extension
- name resolving
- OOP

Class work

- Phone book entry example
- Student, Professor example.

Howe work

- Class work extention

6. Files

Topics

- binary vs text files
- open modes (r, w, r+, w+)
- read, seek
- write

Class work

- Phone book save/restore example

Home work

- Write in the file squares and cubes of natural numbers from 1 to 10. Each 3 numbers in one line.

  Numbers are delimited by tabular symbol.

  Example:

  1 1 1
  2 4 8
  ...
- Add to the existing file from p.1 squares and cubes for numbers from 11 to 20.
- Read file content and print them on screen.
- Print out dir content (recursively)

7. Advanced topics

- lambdas, map/filter/zip
- metaprogramming

- CLI (word count tool)
- python idioms

H/W

- *Strings, logic, loops:*
  http://codingbat.com/python

- *Dict*
  -- symbol frequency distribution
  -- words frequency
  -- Given the following dictionary:

```
inventory = {
  'gold' : 500,
  'pouch' : ['flint', 'twine', 'gemstone'],
  'backpack' : ['xylophone','dagger', 'bedroll','bread loaf']
}
```

Try to do the followings:

Add a key to inventory called 'pocket'.
Set the value of 'pocket' to be a list consisting of the strings 'seashell', 'strange berry', and 'lint'.
.sort()the items in the list stored under the 'backpack' key.
Then .remove('dagger') from the list of items stored under the 'backpack' key.
Add 50 to the number stored under the 'gold' key.

-- Create a new dictionary called prices.

Put these values in your prices dictionary:

```
"banana": 4,
"apple": 2,
"orange": 1.5,
"pear": 3
```

Create a new dictionary called stocks.

Put these values in your stocks dictionary:

```
"banana": 6,
"apple": 0,
"orange": 32,
"pear": 15
```

Loop through each key in prices. For each key, print out the key along with its price and stock information. Print the answer in the following format:

```
apple
price: 2
stock: 0
banana
```

price: 4
stock: 6
...

Calculate total sum for the given food
Calculate total sum for all foods

-- Given a file with text:
--- calculate word frequency
--- calculate symbols frequency (horizontal graph)
--- split lines into tokens and calculate num of tokens

-- Given a file with president and number of votes in a state:
--- calculate num of total votes (for a list of votes from different states)
--- calculate distribution of votes between states

-- Read from file cities.csv with the following columns: country;city;population
--- Sort cities by name
--- Sort cities by population
--- Sort countries by population

-- Read from file salary.csv with the following columns: name;month;salary
--- прочитать из файла salary.csv все фамилии в список, затем подсчитать кол-во повторений каждой фамили используя словарь (dic_lib)
--- аналогично проделать для месяцев

-- создать список с названиями стран, затем создать словарь соответствия стран и континентов, и затем подсчитать кол-во стран на каждом из континентов

-- создать список актеров, затем создать словарь в котором бы хранилось соответствие актеров и фильмов, и затем для каждого фильма подсчитать кол-во актеров которые в нем снялись

-- создать словарь в котором есть соответствие актера и его гонорара, затем используя словарь соответствия актера и фильма из предыдущей задачи, необходимо просуммировать общий бюджет фильма исходя из гонорара всех снявшихся в нем актеров

-- сделать функцию для фильма и списка фильмов. Потом отрефакторить, чтобы одна вызывалась из другой

- *Files*
- Create a file and fill it with 20 symbols 'a', so that the result is like:
aaaaaaaaaaaaaaaaaaaa
- Open file from p.1 and modify it by inseritng symbols 'b', so that the result is like:
abababababab...ab

- Open file from p.1 and modify it by inseritng symbols 'b', so that the result is like:
  abbabbabbabbabbabbabb...abb
- Open file from p.1 and modify it by inseritng symbols 'b', so that the result is like:
  ababbabbbabbbbabbbbbabbbbbbab...abbbbbbbbbbbbbbbbbbbb
- Create a file and fill it with 20 lines in the following way (each next line is longer thatn the previous one):

  a
  aa
  aaa
  aaaa
  ...
  aaaaaaaaaaaaaaaaaaaa
- Create a file and fill it with 20 lines in the following way (each next line has one addditional number than the previous one):

  1
  1 2
  1 2 3
  1 2 3 4
  ...
  1 2 3 4 5 ... 19 20
- Create a file and fill it with 24 lines in the following way (each next pyramid is taller than the previous one):

  1
  1 2
  1
  1
  1 2
  1 2 3
  1 2
  1
  1
  1 2
  1 2 3
  1 2 3 4
  1 2 3
  1 2
  1
  1
  1 2
  1 2 3
  1 2 3 4
  1 2 3 4 5
  1 2 3 4
  1 2 3
  1 2
  1

- Modify the function so that it takes an extra argument to be used as a symbol to draw the piramyd.

Different topics

- FSM (draw diagram)
- github, plugins for sublime
- hex, regexp, json, csv

Materials:

- https://vk.com/wall-54530371_1075
- http://www.shashkovs.ru/_prog/Lutc_M._-_Izuchaem_Python_(4-e_izdanie)-_2011.pdf
- http://pythontutor.com/visualize.html
- https://www.checkio.org/

Hacks:

- When introduce new syntax constructions (if, loop, func) do it with formal BNF rules!
- There are tips to write program: explain in natural language => pseudocode => code
- from small to big
- debugging: start with working core and iteratively add functionality until it breaks
- seek is like cursor when edit document
- when you give theory, they should practice immediately. This will recall gained knowledge (considered the best learning hack ever).
- create modules that collect typical operations under lists and dics and use then use them to demostrate how we can export functions