Python Language Fundamentals-2

- Working with path:
  - Normal path in computer system writes like: C:\ESRI\Python
  - In Python, a backslash is treated as an escape character. So you should avoid backslash in paths.
  - In Python, paths are stored as strings
  - Three correct ways to write path in Python:
    1. Use forward slash (/): “C:/ESRI/Python”
    2. Use two backslashes(\\): “C:\\ESRI\\Python”
    3. Use a string literal by placing the letter r before a string: r“C:\ESRI\Python”. Letter r stands for “raw string” meaning that a backslash will not be treated as an escape character.

- Working with modules:
  - Module are extensions that can be imported into Python; a module consists of a number of specialized functions; modules are imported with a statement import
  - <module>.<function>
  - import math;math.cos(2)
  - __doc__ statement to get the description.

    Print math.cos.__doc__
  - Get the list of all the functions in the a module: dir(module)

- Working with modules:
  - If no more than one function with the same name will be used in your codes, you can use

    from module import function

    then in the codes you can use the function without its module prefix

- Working with modules:
  - time module:

    import time

    print dir(time)

    time.time() (determines the current time as number of seconds since the “epoch” or reference date(0 hours January 1 1970, also the python default
  - time.asctime(): asctime function convert time to a string
  - Python keywords cannot be used as variables names. To see a list of keywords, use the keyword module

    import keyword

    print keyword.kwlist

- Controlling workflow using conditional statements
  - Branching: making a decision to take one path or another; Branching typically uses if structure and its variants
  - if structure have a condition that is either true or false: True; False
    - Comparison operators to create condition:
      - == equal to
      - != not equal to
      - > greater than
      - < less than
      - >= greater than or equal to
      - <= less than or equal to

- Controlling workflow using conditional statements
  - if statement is followed by a colon (:);
  - if statement can be used on its own, without being followed by an else or endif, as is often required in other languages.
  - The line following if statement is indented; Indenting a line, the code becomes a block; Python knows where is the end of the if structure by detecting you stop indenting the code.(so we don’t need to use endif)

- Controlling workflow using conditional statements
  - elif statement : get executed only if the condition in the if statement is False; elif statements can be repeated as many times as necessary
  - else statement(if used) comes after all the elif statements and dose not include a condition; else statement is executed only if all the previous conditions are False and can be used only once in a single if structure.

```
import random
x=random.randint(0,9)
print x
if x==8:
    print " You Win!"
elif x==3:
    print "Try Again"
else:
    print "You Loose"
```

- Getting user input
  - Use input function :only in PythonWin

```
x=input(“”)
```

- Commenting scripts
  - Comments are preceded by the number sign # ; when the script is run, any line that starts with the number sign is not executed
  - Comments can also be placed on the same line after pieces of codes.
  - ## can also be used to start the comments.