Running Python

- Working in Terminal mode
- Working using IDLE
- Python character set is UTF (Unicode Transformation Format) 8
- Token in python are:
  - Line termination
  - Indent – leading spaces / tab
  - Identifiers
  - Keywords
  - Literals
  - Operators
  - Delimiters (line termination, indent, **( )**, **[ ]**, **{ }**, **,**, **:** , **;**)

List of keywords

False	class	finally	
is	return	None	
continue	for	lambda	
try	True	def	
from	nonlocal	while	
and	del	global	
not	with	as	
elif	if	or	
yield	assert	else	
import	pass	break	
except	in	raise	

Data Types

- Integer (decimal, binary, octal, hexadecimal)
  - Maximum and minimum possible value is not restricted.
  - Integer value without prefix is decimal value.

- Float
  - Maximum and minimum value not restricted
  - As per iEEE standard - it's 64bit double precision value, in the form $x*10^n$
  - where value of n can be in -324 to 307
  - Python will store a value larger than 1.8e308 by indicating it as string – **inf** *(stands for infinity)*

```
>>>a = 1.79e308
>>>a
float
>>>a = 1.8e309
>>>a
inf
```

- String
  - Collection of characters, enclosed in '' or "".
  - Using "" in string literal allows us to use ' as part of string

```
"he said 'hello' to her"
```

  - \ is escape character
    - Used to represent whitespace char
    - Prefixing \ to special char converts it to normal char and vice versa.
    - Prefixing a string literal with r / R changes interpretation of \
  - ord() and chr() functions can be used to get UTF and character equivalent.
  -

Function	Purpose	Example
ord(character)	Displays the ASCII equivalent of the character	>>>ord('A') 65 >>>ord('a') 97
chr(integer)	Displays the character corresponding to the UNICODE integer	>>>chr(70) F >>>chr(99+7) j

- List
- Tuple
- Dictionary

Converting between types

Functions in Python :

int() anything to integer, provided result is valid integer value

float() anything to float, provided result is valid floating point value

str() will convert anything to string

Type method

- Python has an inbuilt method or function called **type** which can be used to find out the type of variable at run-time. The syntax is :

```
type(constant/ variable name)
```

```
>>>a=90
>>>type(a)
int64
```

comments

- Used for annotation of code / script

- # is python comment character

```
>>>print ("Hello")     #This is a simple print statement
```

- # character and everything beyond it in the line is ignored.
- So a comment can be part of line or may be an entire line.
- A comment running across multiple line can be written either using # infront of each line or by enclosing it in triple quotes (''' ……''' OR """ …….. """)

Continuation statement

- Explicit joining
  - **** is used to join a python statement running across more than one line on screen
  - It can't be used to continue comment or token (except for string literal)
- Implicit joining
  - Expressions contained in (), [] or {} can be written across lines, without using \.

Variables / identifiers

attaching name to value

- Python is case sensitive
- Rules for creating an identifier are :
  - should start with a letter or underscore
  - Can have any number of letter, digit or underscore
  - Should not be keyword
- Whenever a user defined identifier is encountered in the code, python substitute it's attached value at that place.
- No need to declare a variable, they get created with first definition.

- Variable in Python, denotes memory space, which holds reference of the actual value.
- A value can have more than one referencing variable.

Invalid identifier	***Reason***
Admit card	space
45lane	starting with digit
*All*gotit*	special character
d+f	+ not allowed
while	keyword