**SESSION - 2**

**PYTHON OBJECT TYPES**

In Python, data takes the form of objects:

* Built-in objects that Python provides, or
* Objects we create using Python classes or
* External language tools such as C extension libraries



object has set of values and associated operations. We can say, in python 'Object' is fundamental notion.

**PYTHON BUILD IN OBJECT TYPES**

**WHY TO USE BUILD-IN TYPES**

* Because we get powerful tools such as collections (lists) and search tables (dictionaries) for free, you can use them immediately.
* Built-in objects are often more efficient (already optimized) than custom data structures.
* Built-in objects are a standard part of the language.

**PYTHON’S CORE DATA TYPES**

**Object type Example literals/creation**

Numbers 1234 , 3.1415 , 3+4j , 0b111 , Decimal() , Fraction()

Strings 'spam' , "Bob's" , b'a\x01c' , u'sp\xc4m'

Lists [1, [2, 'three'], 4.5] , list(range(10))

Dictionaries {'food': 'spam', 'taste': 'yum'} , dict(hours=10)

Tuples (1, 'spam', 4, 'U') , tuple('spam') , namedtuple

Files open('eggs.txt') , open(r'C:\ham.bin', 'wb')

Sets set('abc') , {'a', 'b', 'c'}

Other core types Booleans, types, None

Program unit types Functions, modules, classes

In formal terms, Python is dynamically typed, a model that keeps track of types for you automatically instead of requiring declaration code, but it is also strongly typed, a constraint that means you can perform on an object only operations that are valid for its type.

**PYTHON PROGRAM**

Python program can be decomposed in to

* modules,
* statements,
* expressions, and
* objects

The relationship defined as:

* Programs are composed of modules.
* Modules contain statements.
* Statements contain expressions.
* Expressions create and process objects.