**Python Introduction:**

* Python is a general-purpose , interpreted, interactive, object-oriented, high-level programming language.
* It was created by Guido van Rossum during 1985- 1990.
* Python got its name from “Monty Python’s flying circus”.
* Python was released in the year 2000.

**Python Features:**

* **Python is interpreted**: Python is processed at runtime by the interpreter. You do not need to compile your program before executing it.
* **Python is Interactive**: You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Compiler | | | | Interpreter | | | | |
| Compiler Takes **Entire** program as input | | | | Interpreter | Takes **Single** instruction | | | as |
| input |  | | |  |
| Intermediate Object Code is **Generated** | | | | **No** Intermediate | | Object | Code | |
| is **Generated** | |  |  | |
| Conditional | Control | Statements | are | Conditional | Control | Statements | | are |
| Executes **faster** | |  |  | Executes **slower** | |  | |  |
| **Memory Requirement** is **More**(Since Object | | | | **Memory Requirement** is **Less** | | | | |
| C  ode is Generated) | |  |  |
| Program need not be **compiled** every time | | | | Every time | higher | level program | | is |
| converted into lower level program | | | |  |
| **Errors** are | displayed | after **entire** | | **Errors** are | displayed | | for **every** | |
| **program** is checked | |  | | **instruction** interpreted (if any) | | | |  |
| **Example** : C Compiler | | | | **Example** : PYTHON | | | | |

* **Python is Object-Oriented**: Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
* **Python is a Beginner's Language:** Python is a great language for the beginner- level programmers and supports the development of a wide range of applications.
* **Easy-to-learn:**Python is clearly defined and easily readable. The structure of the program is very simple. It uses few keywords.
* **Easy-to-maintain:** Python's source code is fairly easy-to-maintain.
* **Portable:** Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* **Interpreted:** Python is processed at runtime by the interpreter. So, there is no need to compile a program before executing it. You can simply run the program.
* **Extensible:** Programmers can embed python within their C,C++,Java script ,ActiveX, etc.
* **Free and Open Source:** Anyone can freely distribute it, read the source code, and edit it.
* **High Level Language:** When writing programs, programmers concentrate on solutions of the current problem, no need to worry about the low level details.
* **Scalable:** Python provides a better structure and support for large programs than shell scripting.

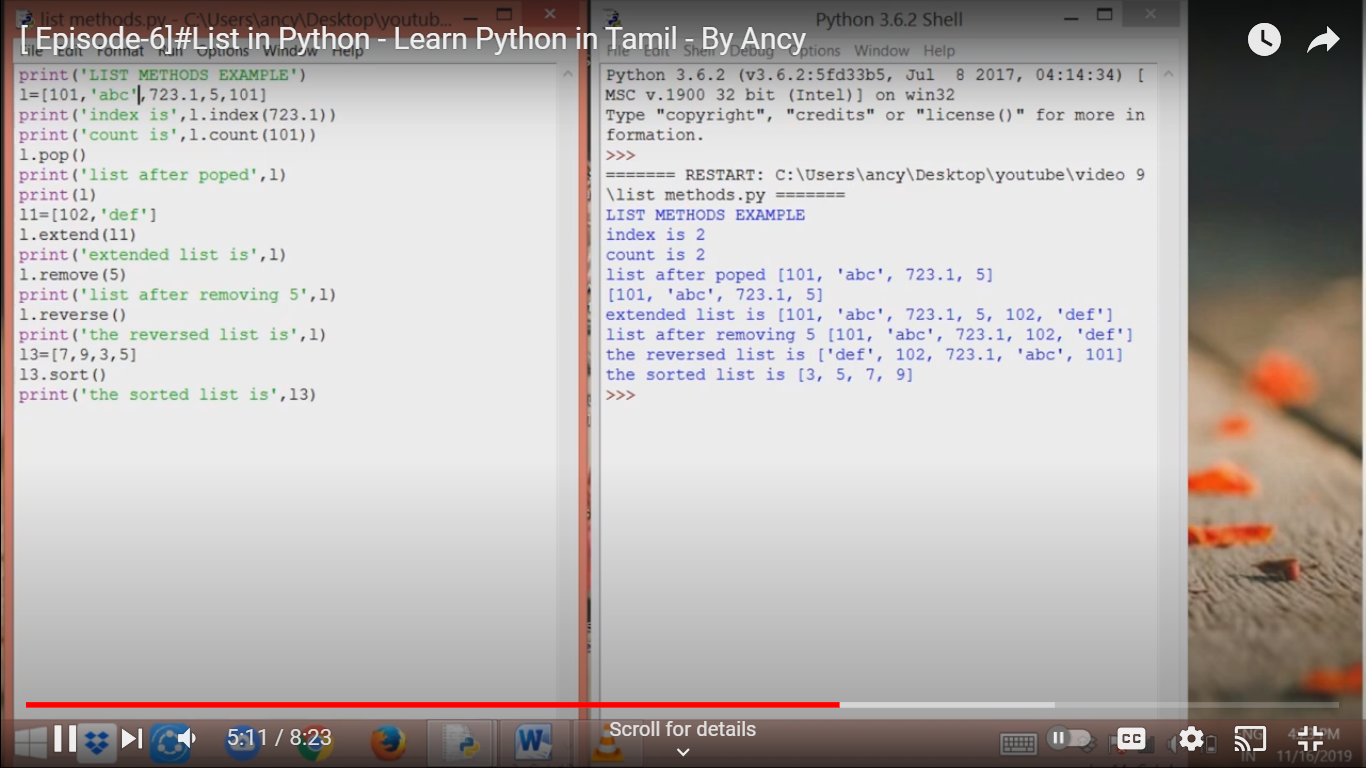
**Applications:**

|  |  |
| --- | --- |
| **Application** | **Features** |
| Console Based Application | Python can be used to develop console based  applications. For example: **IPython**. |
| Audio or Video based Applications | Python proves handy in multimedia section. Some of real applications are: Tim Player, cplay  etc. |
| 3D CAD Applications | Fandango is a real application which provides  full features of CAD. |
| Web Applications | Python can also be used to develop web based application. Some important developments are: Python Wiki Engines, Pocoo, Python Blog  Software etc. |
| Enterprise Applications | Python can be used to create applications which can be used within an Enterprise or an Organization. Some real time applications are:  OpenErp, Tryton, Picalo etc. |
| Applications for Images | Using Python several application can be developed for image. Applications developed  are: VPython, Gogh, imgSeek etc. |

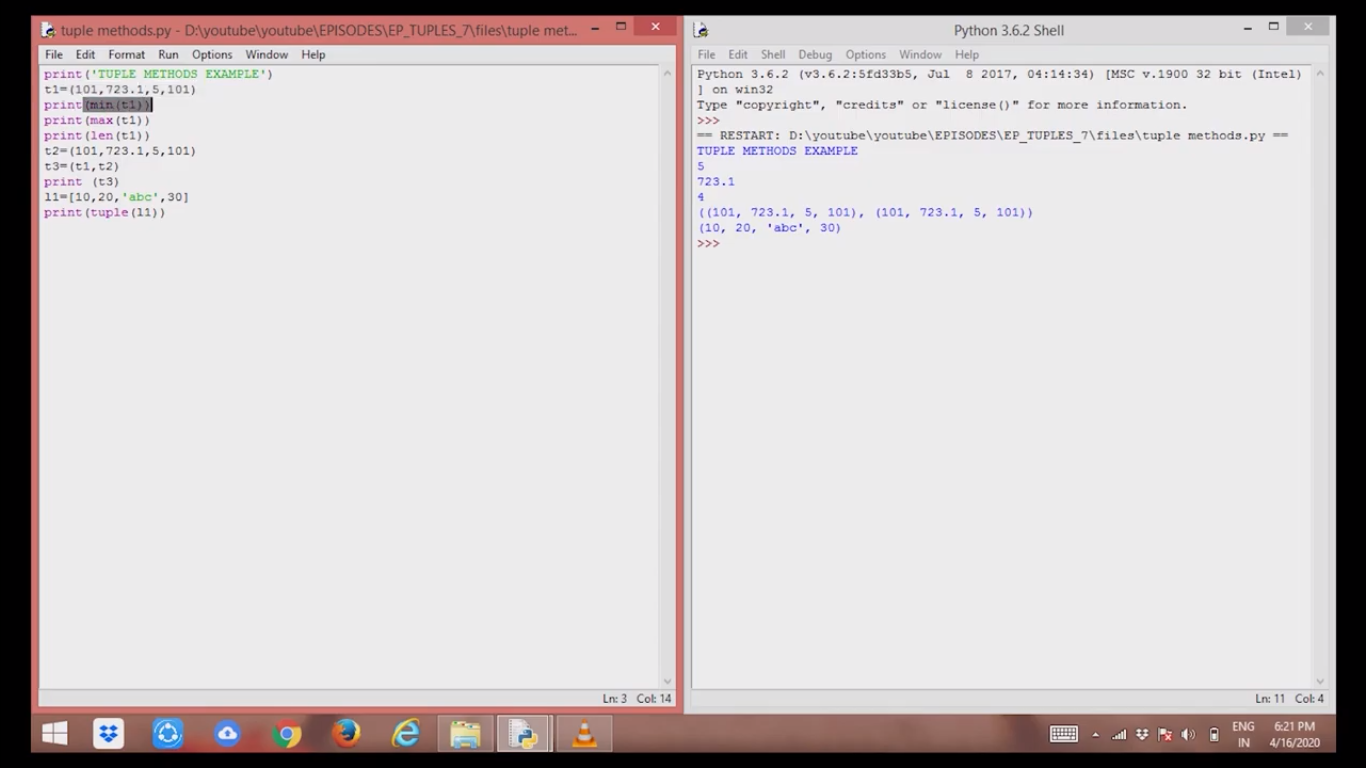
**Company uses python**

* Bit Torrent file sharing
* Google search engine
* Youtube
* Dropbox
* Intel
* Cisco,
* HP
* IBM
* i–Robot
* NASA

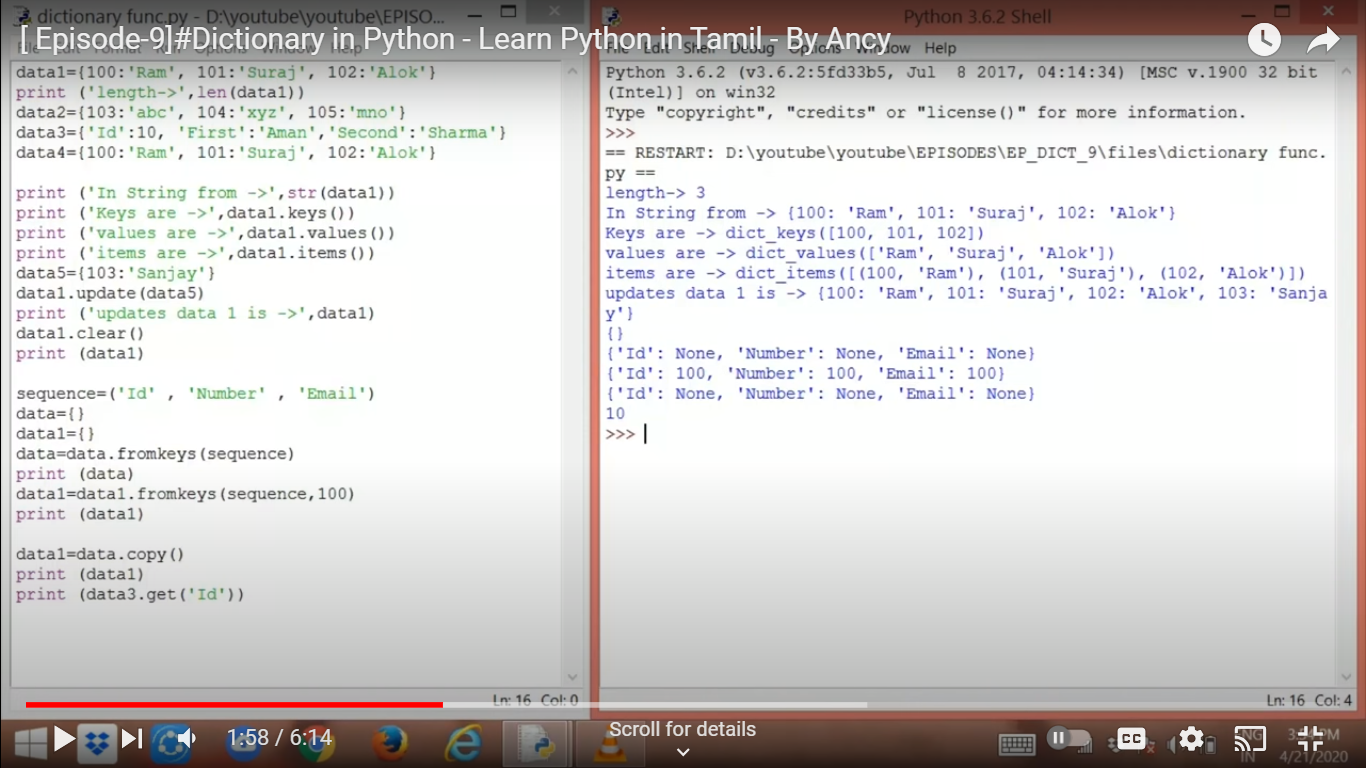
LIST AND ARRAY EXAMPLE



**TUPLE EXAMPLES**



**DICTIONARY:**



**DATA TYPES:**

a=5

print('the type of ',a, type(a))

a=4.6

print('the type of ',a, type(a))

a=1+2j

print('the type of ',a, type(a))

a='python programming'

print('the type of ',a, type(a))

a=[1,'a','python',4.6]

print('the type of ',a, type(a))

a=(1,'a','python',4.6)

print('the type of ',a, type(a))

a={'name':'aaa', 'dept':'bbb'}

print('the type of ',a, type(a))

a={1,2,3}

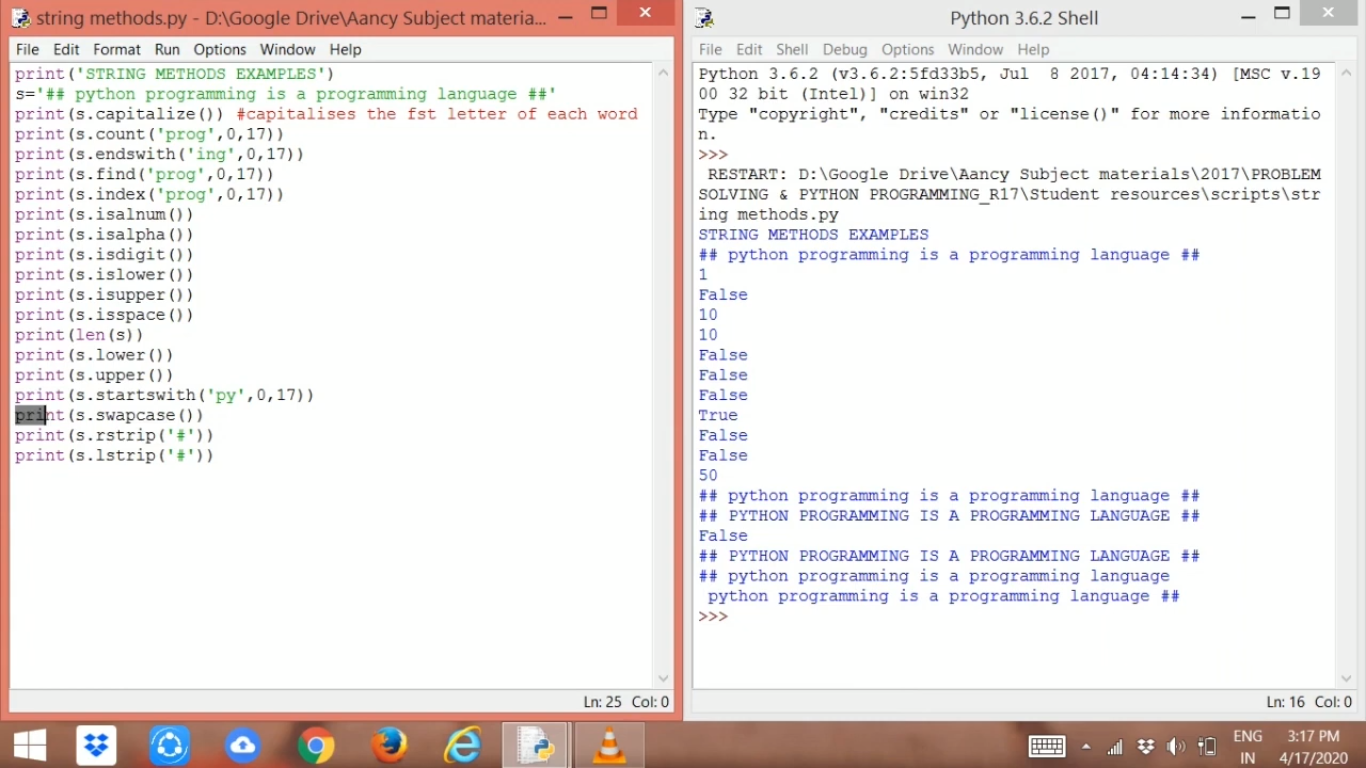
print('the type of ',a, type(a))

datatypes.py

**Sign in**

Displaying datatypes.py.

**Strings in python:**

===

Self-Introduction

1. Good Day Sir
2. Myself Saranya Karthik, completed my UG Bsc computer science in Nehru Memorial College, Trichy.
3. After completing my UG, I did my PG MSc Computer science in Govt Arts college, Salem.
4. After completing my graduation, I worked as admin in Cooperative Bank, Attur Taluk.
5. I have Experience in performing Finance maintenance

Data – Entry and Documentation Work

1. As I am interested in IT field, I am doing Python course with MySQL database at present.
2. Looking for an opportunity where I can utilize my skills and abilities while being innovative, resourceful and flexible. I am certain that my presence in your team will prove to be beneficial to your organization.
3. Thankyou Sir

**1.http using basic authentication in python flask:**

<https://roytuts.com/python-flask-http-basic-authentication/>(saranya)

<https://datagy.io/python-requests-authentication/>

<https://stackoverflow.com/questions/6999565/python-https-get-with-basic-authentication>

https://www.geeksforgeeks.org/authentication-using-python-requests/

**2.Asynchronous:**

<https://www.javatpoint.com/python-asynchronous-programming-asyncio-and-await>

<https://www.geeksforgeeks.org/asyncio-in-python/>

**3.Read jproperties (read)**

<https://www.digitalocean.com/community/tutorials/python-read-properties-file>

**4.Mysql stored procedures:**

<https://mysqlcode.com/call-mysql-stored-procedure-in-python/>

**5.Swagger-ui**

<https://code.likeagirl.io/swagger-and-postman-build-a-swagger-ui-for-your-python-flask-application-141bb4d0c203>

<https://progressstory.com/tech/python/swagger-api-doc-automation-with-flask-restful/>

<https://code.likeagirl.io/swagger-and-postman-build-a-swagger-ui-for-your-python-flask-application-141bb4d0c203>

**6.dijango:**

**https://www.w3schools.com/django/index.php**

**7.python event driven programming:**

**https://www.javatpoint.com/python-event-driven-programming**

**8.MYSQL:**

[**https://dev.mysql.com/downloads/file/?id=520407**](https://dev.mysql.com/downloads/file/?id=520407)

**9.REST API using FLASK:**

[**https://www.geeksforgeeks.org/python-build-a-rest-api-using-flask/**](https://www.geeksforgeeks.org/python-build-a-rest-api-using-flask/)

**10.Access modifier in python (public and protruded):**

**https://www.geeksforgeeks.org/python-build-a-rest-api-using-flask/**

**Dijango tutorial:**

**youtube`;payilagam**