# ASSIGNMENT-1(WEEK 0-PYTHON)
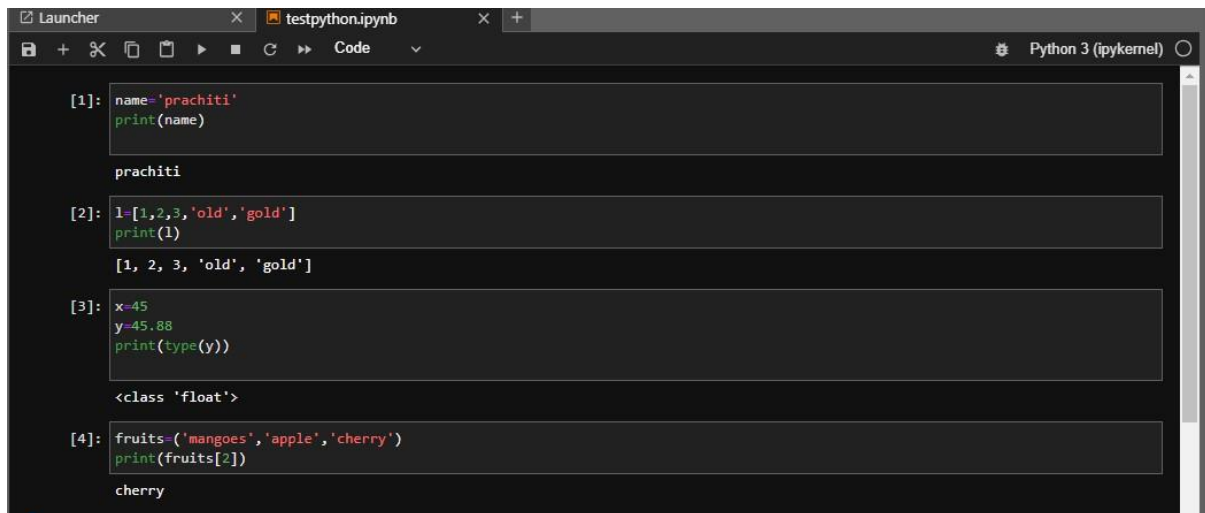
**Q1. Create one variable containing following type of data:**

(i) string

(ii) list

(iii) float

(iv) tuple

**Ans:**



**Q2. Given are some following variables containing data:**

(i) var1 = ' '

(ii) var2 = '[ DS , ML , Python]'

(iii) var3 = [ 'DS' , 'ML' , 'Python' ]

(iv) var4 = 1.

What will be the data type of the above given variable.

**Ans:**

(1) Integer

(2) String

(3) List

(4) Float

**Q3. Explain the use of the following operators using an example:**

(i) /

(ii) %

(iii) //

(iv) **

**Ans:**

(1) /

It is used to divide the no.

(2)%

It is used to find the modulus

(3) //

It is used as a floor division

(4) **

It is used as a exponentiation

```
[1]: 4/2

[1]: 2.0

[2]: 6%8

[2]: 6

[3]: 6//9

[3]: 0

[4]: 3**2

[4]: 9
```

**Q4. Create a list of length 10 of your choice containing multiple types of data. Using for loop print the element and its data type.**

```
[10]: l=[1,2,3,4,'prachiti','adam',5.77]
      for i in l:
          print(type(i))
          if i==0:
              continue
          print(i)


      <class 'int'>
      1
      <class 'int'>
      2
      <class 'int'>
      3
      <class 'int'>
      4
      <class 'str'>
      prachiti
      <class 'str'>
      adam
      <class 'float'>
      5.77
```

## Q6. Create a list containing 25 int type data. Using for loop and if-else condition print if the element is divisible by 3 or not.

```
a= range(25)
for i in a:
    if i%3==0:
        print(i)
else:
    print("not divided")
```

```
0
3
6
9
12
15
18
21
24
not divided
```

## Q7. What do you understand about mutable and immutable data types? Give examples for both showing this property.

### Ans:

Immutable object can't change the object's state after you've created it. In contrast, a mutable object allows you to modify its internal state after creation.

Likewise List is mutable it can change no. words etc

```
[1]: l=[1,3,4,5,'hello','fruits',4.66]

[2]: l
[2]: [1, 3, 4, 5, 'hello', 'fruits', 4.66]

[3]: l[2]
[3]: 4

[4]: l[4]=300

[5]: l
[5]: [1, 3, 4, 5, 300, 'fruits', 4.66]
```

Likewise string is immutable it can't change no. words etc.

```
[6]: s='fruits'

[7]: s
[7]: 'fruits'

[8]: s[2]
[8]: 'u'

[10]: s[2]='a'
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[10], line 1
----> 1 s[2]='a'

TypeError: 'str' object does not support item assignment
```