

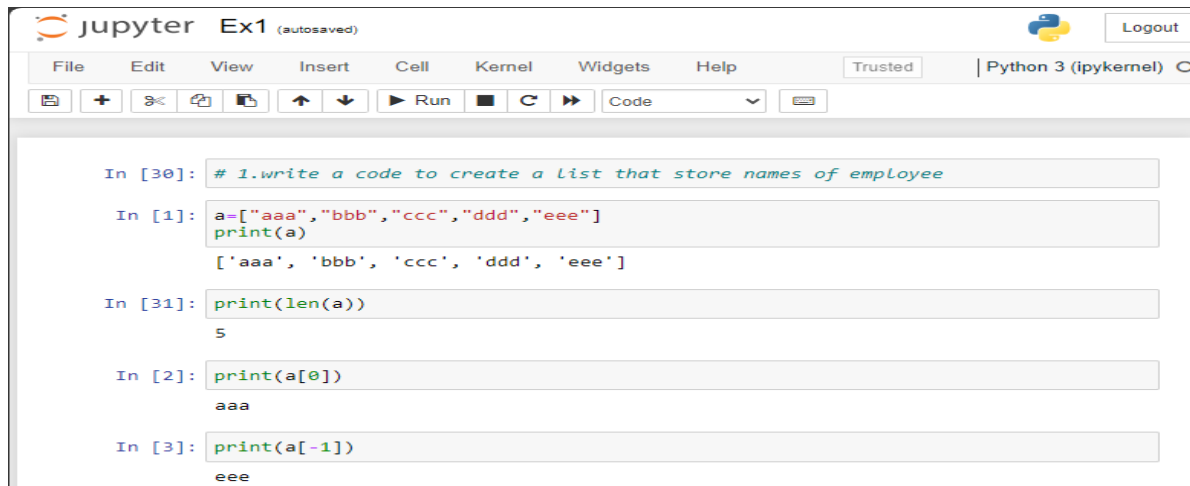
Machine Learning

Ex: 01 Basics of Python

Name: Athithraja R

Reg.no: 2022503702

1)



A screenshot of a Jupyter Notebook interface titled 'Ex1 (autosaved)'. The interface includes a top menu bar with 'File', 'Edit', 'View', 'Insert', 'Cell', 'Kernel', 'Widgets', and 'Help'. Below the menu is a toolbar with icons for file operations and code execution. The notebook content shows a series of code cells. The first cell is a comment: '# 1.write a code to create a list that store names of employee'. The second cell creates a list 'a' with five string elements and prints it. The third cell prints the length of the list. The fourth cell prints the first element. The fifth cell prints the last element.

```
In [30]: # 1.write a code to create a list that store names of employee

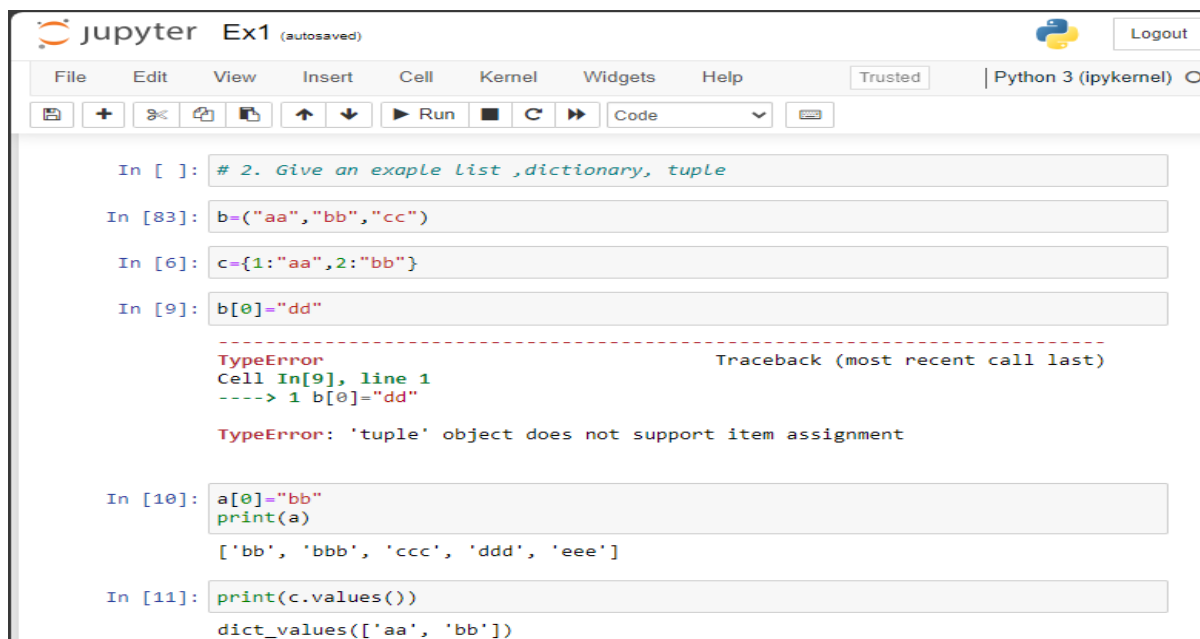
In [1]: a=["aaa","bbb","ccc","ddd","eee"]
        print(a)
        ['aaa', 'bbb', 'ccc', 'ddd', 'eee']

In [31]: print(len(a))
        5

In [2]: print(a[0])
        aaa

In [3]: print(a[-1])
        eee
```

2)



A screenshot of a Jupyter Notebook interface titled 'Ex1 (autosaved)'. The interface is similar to the one in the previous block. The notebook content shows code cells for creating a tuple, a dictionary, and attempting to modify them. The first cell is a comment: '# 2. Give an exaple list ,dictionary, tuple'. The second cell creates a tuple 'b' and prints it. The third cell creates a dictionary 'c' and prints its values. The fourth cell attempts to modify the first element of the tuple 'b', which results in a 'TypeError: 'tuple' object does not support item assignment'. The fifth cell prints the first element of a list 'a' (which is not shown in the previous block). The sixth cell prints the values of the dictionary 'c'.

```
In [ ]: # 2. Give an exaple list ,dictionary, tuple

In [83]: b=("aa","bb","cc")

In [6]: c={1:"aa",2:"bb"}

In [9]: b[0]="dd"

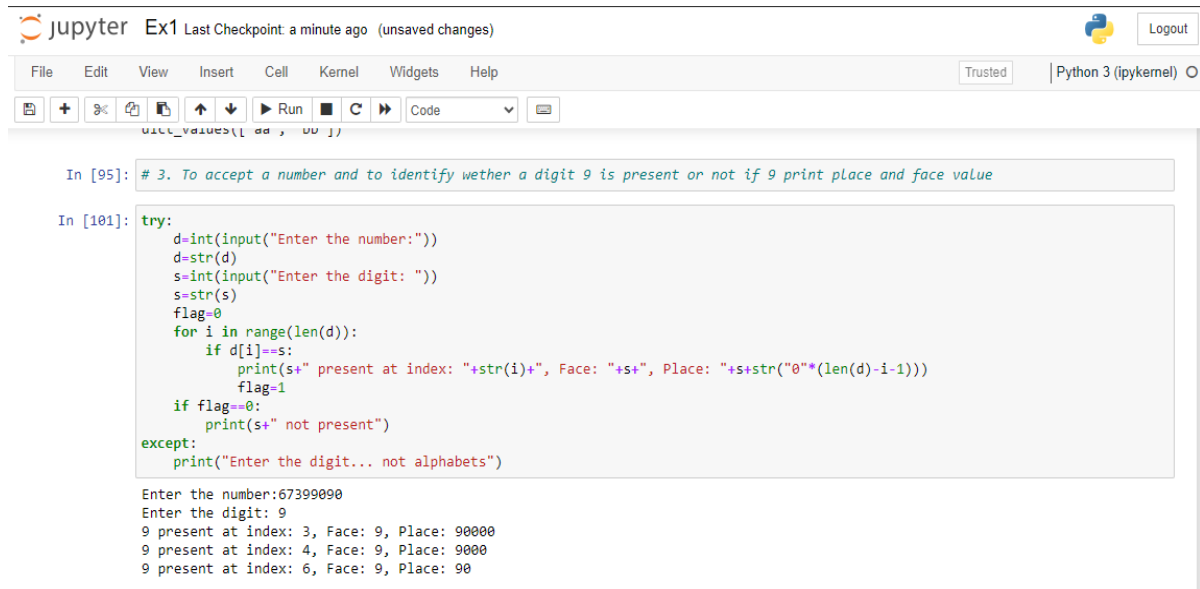
-----
TypeError                                 Traceback (most recent call last)
----> 1 b[0]="dd"

TypeError: 'tuple' object does not support item assignment

In [10]: a[0]="bb"
        print(a)
        ['bb', 'bbb', 'ccc', 'ddd', 'eee']

In [11]: print(c.values())
        dict_values(['aa', 'bb'])
```

3)



Jupyter Ex1 Last Checkpoint: a minute ago (unsaved changes)

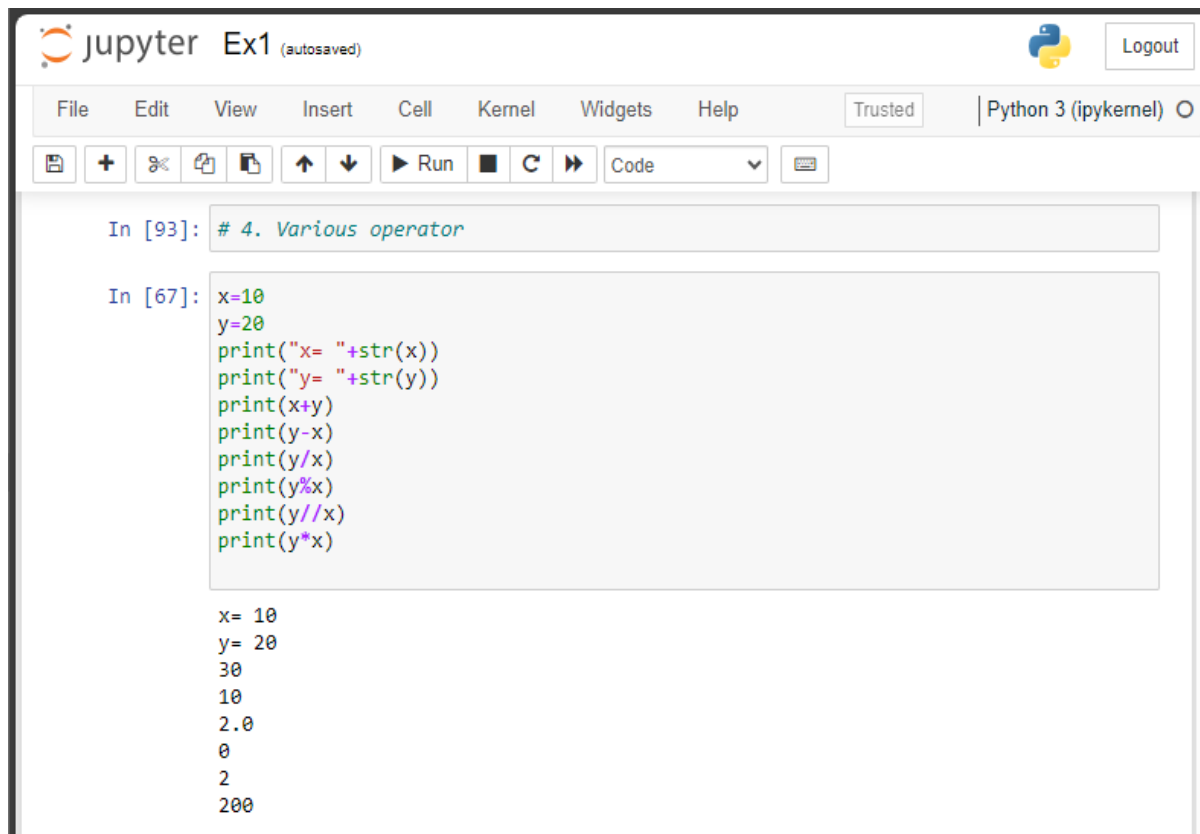
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

```
In [95]: # 3. To accept a number and to identify whether a digit 9 is present or not if 9 print place and face value
```

```
In [101]: try:
d=int(input("Enter the number:"))
d=str(d)
s=int(input("Enter the digit: "))
s=str(s)
flag=0
for i in range(len(d)):
    if d[i]==s:
        print(s+" present at index: "+str(i)+", Face: "+s+", Place: "+s+str("0"*(len(d)-i-1)))
        flag=1
    if flag==0:
        print(s+" not present")
except:
    print("Enter the digit... not alphabets")
```

Enter the number:67399090
Enter the digit: 9
9 present at index: 3, Face: 9, Place: 90000
9 present at index: 4, Face: 9, Place: 9000
9 present at index: 6, Face: 9, Place: 90

4)



Jupyter Ex1 (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

```
In [93]: # 4. Various operator
```

```
In [67]: x=10
y=20
print("x= "+str(x))
print("y= "+str(y))
print(x+y)
print(y-x)
print(y/x)
print(y%x)
print(y//x)
print(y*x)
```

x= 10
y= 20
30
10
2.0
0
2
200

5)

```
# 5. python program that prints the no.of ways a robot can take steps to climb the staires case(fibonacci series based) use function in program

def count_ways(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    elif n == 2:
        return 2
    a, b = 1, 2
    for _ in range(3, n + 1):
        a, b = b, a + b
    return b

stairs = int(input("Enter the number of stairs: "))
print("The robot can climb " + str(stairs) + " stairs in " + str(count_ways(stairs)) + " different ways.")

... The robot can climb 5 stairs in 8 different ways.
```

6)

Jupyter Ex1 (unsaved changes) Python 3 (ipykernel)

```
In [69]: # 6. Create multidimensional list and display it

In [70]: li=[[2,3,4],[3,6,7],[3,7,8]]
          for i in range(len(li)):
              for j in range(len(li[0])):
                  print(li[i][j])

          2
          3
          4
          3
          6
          7
          3
          7
          8
```

7)

[] #7. Patter generation

```
num = int(input("Enter the number of rows for the first pattern: "))
for i in range(num, 0, -1):
    for j in range(num - i):
        print(" ", end="")
    if i == 1:
        print("*")
        break
    for k in range(2 * i - 2):
        print("*", end="")
    print()

num = int(input("Enter the number of rows for the second pattern: "))
for i in range(1, num + 1):
    if i % 2 != 0:
        print(" ** ")
    else:
        print("*****")
```

```
Enter the number of rows for the first pattern: 4
*****
****
***
**
*

Enter the number of rows for the second pattern: 6
**
*****
**
*****
**
*****
```