In [1]:
```
#question 1
string="learning"

temp="machine"
string=temp+ string;
```

In [2]:
```
print(string)
```

```
machinelearning
```

In [3]:
```
#1.b.
len(string)
```

Out[3]:
```
15
```

In [4]:
```
string.find("learning")
```

Out[4]:
```
7
```

In [5]:
```
strencode=string.encode()
```

In [6]:
```
strdcd=strencode.decode()
```

In [7]:
```
strencode
```

Out[7]:
```
b'machinelearning'
```

In [8]:
```
strdcd
```

Out[8]:
```
'machinelearning'
```

In [11]:
```
o=string.isalpha()
```

In [12]:
```
o
```

Out[12]:
```
True
```

In [15]:
```
p=[2,3,4,5,2,2,8]
```

In [17]:
```
c=p.count("2")
```

In [18]:
```
c
```

Out[18]:
```
0
```

In [19]:
```
c=p.count(2)
```

In [20]:
```
c
```

Out[20]:
```
3
```

In [21]:
```
j=string.isdigit()
```

In [22]:
```
j
```

Out[22]:
```
False
```

In [23]:
```
#question 2
p[3]=66;
```

In [24]:
```
p
```

Out[24]:
```
[2, 3, 4, 66, 2, 2, 8]
```

In [26]:
```
print(p[1:3])
```

```
[3, 4]
```

In [27]:
```
y=p.index(2)
```

In [28]:
```
y
```

Out[28]:
```
0
```

In [29]:
```
p.append(99)
```

In [30]:
```
p
```

Out[30]:
```
[2, 3, 4, 66, 2, 2, 8, 99]
```

In [31]:
```
q=[999,555]
```

In [35]:
```
yy=p.extend(q)
```

In [39]:
```
p
```

Out[39]:
```
[2, 3, 4, 66, 2, 2, 8, 99, 999, 555, 999, 555]
```

In [40]:
```
p.insert(1,88)
```

In [41]:
```
p
```

Out[41]:
```
[2, 88, 3, 4, 66, 2, 2, 8, 99, 999, 555, 999, 555]
```

In [44]:
```
p.pop(8)
```
Out[44]: 99

In [45]:
```
p.remove(66)
```

In [46]:
```
p
```
Out[46]: [2, 88, 3, 4, 2, 2, 8, 999, 555, 999, 555]

In [47]:
```
p.reverse()
```

In [48]:
```
p
```
Out[48]: [555, 999, 555, 999, 8, 2, 2, 4, 3, 88, 2]

In [49]:
```
p.sort()
```

In [50]:
```
p
```
Out[50]: [2, 2, 2, 3, 4, 8, 88, 555, 555, 999, 999]

In [66]:
```
t=("virat","rohit","dhoni","shewag","dhawan")#q3
```

In [67]:
```
c=("rahul")
```

In [68]:
```
t=c
```

In [69]:
```
t
```
Out[69]: 'rahul'

In [70]:
```
tpl=(1,22,333,4,6,7)
```

In [71]:
```
tpl.index(22)
```
Out[71]: 1

In [74]:
```
print(tpl[1:3])
```
(22, 333)

In [75]:
```
max(tpl)
```
Out[75]: 333

In [76]:
```
min(tpl)
```
Out[76]: 1

In [77]:
```
len(tpl)
```

Out[77]:
```
6
```

In [85]:
```
del tpl
```

In [86]:
```
#question 4
```

In [87]:
```
dic={"1999: vandana","2006: ambika"}
```

In [88]:
```
dic2={"2001:tulika"}
```

In [90]:
```
dic2=dic
```

In [91]:
```
dic
```

Out[91]:
```
{'1999: vandana', '2006: ambika'}
```

In [92]:
```
dic2
```

Out[92]:
```
{'1999: vandana', '2006: ambika'}
```

In [93]:
```
del(dic2)
```

In [94]:
```
dic.update({"1:ff"})
```

In [95]:
```
dic
```

Out[95]:
```
{'1999: vandana', '1:ff', '2006: ambika'}
```

In [99]:
```
dic.values()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[99], line 1
----> 1 dic.values()

AttributeError: 'set' object has no attribute 'values'
```

In [138]:
```
dicn={1:"a",2:"b"}
```

In [139]:
```
dicn.values()
```

Out[139]:
```
dict_values(['a', 'b'])
```

In [140]:
```
dicn.get(1)
```

Out[140]:
```
'a'
```

In [141]:
```
dic.clear()
```

In [142]:
```
dic
```

Out[142]:
```
set()
```

In [143]:
```
k=dicn.copy()
```

In [144]:
```
k
```

Out[144]:
```
{1: 'a', 2: 'b'}
```

In [145]:
```
p=len(dicn)
```

In [146]:
```
p
```

Out[146]:
```
2
```

In [152]:
```
dict_1="Mohite"
type(dict_1)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[152], line 2
      1 dict_1="Mohite"
----> 2 type(dict_1)

TypeError: 'dict' object is not callable
```

In [148]:
```
q
```

Out[148]:
```
{1: 'a', 2: 'b'}
```

In [137]:
```
type(dicn)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[137], line 1
----> 1 type(dicn)

TypeError: 'dict' object is not callable
```

In [136]:
```
#quest5
```

In [153]:
```
t20={"rohit","kohli","dhoni"}
```

In [154]:
```
odi={"kohli","dhoni","rahul"}
```

In [155]:
```
test={"rishabh","kohli","dhawan","pujara"}
```

In [156]:
```
g=t20.union(odi)
```

In [157]:
```
g
```

Out[157]:
```
{'dhoni', 'kohli', 'rahul', 'rohit'}
```

In [158]:
```
f=t20.intersection(test)
f
```

Out[158]:
```
{'kohli'}
```

In [159]:
```
r=odi.difference(test)
```

In [160]:
```
r
```

Out[160]:
```
{'dhoni', 'rahul'}
```

In [161]:
```
#question 6
```

In [163]:
```
s1 = "amazon"
s2 = "netflix"
s3 = "google"
s4 = "oracledatabase"

if len(s1) > len(s2) and len(s1) > len(s3) and len(s1) > len(s4):
    lrg = s1
elif len(s2) > len(s1) and len(s2) > len(s3) and len(s2) > len(s4):
    lrg = s2
elif len(s3) > len(s1) and len(s3) > len(s2) and len(s3) > len(s4):
    lrg = s3
else:
    lrg = s4

print(lrg)
```

```
oracledatabase
```

In [164]:
```
#question 7
```

In [165]:

```
def generate_even_numbers():
    even_numbers = []
    for num in range(1, 31):
        if num % 2 == 0:
            even_numbers.append(num)
    return even_numbers

even_numbers = generate_even_numbers()
squared_numbers = [num ** 2 for num in even_numbers]

print("Squared Numbers List:", squared_numbers)

filtered_even_numbers = list(filter(lambda x: x % 2 == 0, squared_numbers))

print("Filtered Even Numbers List:", filtered_even_numbers)
```

```
Squared Numbers List: [4, 16, 36, 64, 100, 144, 196, 256, 324, 400, 484, 576,
676, 784, 900]
Filtered Even Numbers List: [4, 16, 36, 64, 100, 144, 196, 256, 324, 400, 48
4, 576, 676, 784, 900]
```

In [ ]: