In [ ]:
```
# print the output with the string
```

In [1]:
```
num1=2
num2=4
add=num1+num2
print("the addition of the ",num1,"and",num2,"is=",add)
print("the addition of the {} and {} is {}".format(num1,num2,add))
print(f"addition of the{num1} and{num2}is{add}")
```

```
the addition of the  2 and 4 is= 6
the addition of the 2 and 4 is 6
addition of the2 and4is6
```

In [3]:
```
num1=3
num2=4
num3=6
average=round(num1+num2+num3/3,2)
print("the average of the",num1,num2,"and",num3,"is",average)
```

```
the average of the 3 4 and 6 is 9.0
```

end statement

In [4]:
```
print("hello",end=" ")
print("how are you")
```

```
hello how are you
```

seprator

In [6]:
```
print("hii","hello","how r u",sep="@")
```

```
hii@hello@how r u
```

In [7]:
```
print("hii","hello","how r u",sep="$")
```

```
hii$hello$how r u
```

In [8]:
```
print("hii","hello","how r u",sep="&")
```

```
hii&hello&how r u
```

keywords in python

In [9]:
```
import keyword
print(keyword.kwlist)
```

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'clas
s', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from',
'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass',
'raise', 'return', 'try', 'while', 'with', 'yield']
```

In [10]:
```
a="ruchtha"
a
```

```
Out[10]:  'ruchtha'
```

address of the vriable or memory location

```
In [11]:  id(a)
```

```
Out[11]:  2382568599072
```

```
In [12]:  a
```

```
Out[12]:  'ruchtha'
```

```
In [13]:  a=18
          b=18
          c=a
```

```
In [14]:  print(id(a))
          print(id(b))
          print(id(c))
```

```
140720543902680
140720543902680
140720543902680
```

```
In [15]:  a=20
          b=30
```

```
In [16]:  print(id(a))
          print(id(b))
```

```
140720543902744
140720543903064
```

strings

indexing and slicing

```
In [20]:  s="ruchitha"
```

```
In [21]:  s[0]
```

```
Out[21]:  'r'
```

```
In [22]:  s[::]
```

```
Out[22]:  'ruchitha'
```

```
In [23]:  s[3:]
```

```
Out[23]:  'hitha'
```

```
In [24]: s[:3]

Out[24]: 'ruc'

In [25]: s[0:4:8]

Out[25]: 'r'
```

string functions

```
In [37]: l=["ruchi","hi","hello"]
         l.append("ruchitha")
         l

Out[37]: ['ruchi', 'hi', 'hello', 'ruchitha']

In [42]: l.count(1)

Out[42]: 0

In [45]: l.index("hi")

Out[45]: 1

In [46]: l.insert(1,3)
         l

Out[46]: ['ruchi', 3, 'hi', 'hello', 'ruchitha']

In [47]: l

Out[47]: ['ruchi', 3, 'hi', 'hello', 'ruchitha']

In [56]: l2=[]

In [57]: l2

Out[57]: []

In [59]: list2=[2]

In [60]: list2.extend(l)

In [61]: list2

Out[61]: [2, 'ruchi', 3, 'hi', 'hello', 'ruchitha']

In [62]: l.extend(l)

In [63]: l2

Out[63]: []
```

In [65]:
```
l3=[4]
l3
```

Out[65]:
```
[4]
```

In [70]:
```
l3.extend(l2)
```

In [71]:
```
l3
```

Out[71]:
```
[4]
```

In [72]:
```
l2.clear()
l2
```

Out[72]:
```
[]
```

In [74]:
```
l3.pop(0)
l3
```

Out[74]:
```
[]
```

In [76]:
```
l3.reverse
```

Out[76]:
```
<function list.reverse()>
```

In [77]:
```
l3
```

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

In [79]:
```
l4=[3,4,7,9,2,6]
l4
```

Out[79]:
```
[3, 4, 7, 9, 2, 6]
```

In [80]:
```
l4.sort()
l4
```

Out[80]:
```
[2, 3, 4, 6, 7, 9]
```

In [ ]:
```
l4.sort(reverse=True)
```

In [82]:
```
l4
```

Out[82]:
```
[9, 7, 6, 4, 3, 2]
```

tuple

In [84]:
```
t=()
t
```

Out[84]:
```
()
```

In [85]:
```
t=(2,True,"hello",1+2j,3.4,2,2)
```

```
t
```

```
Out[85]: (2, True, 'hello', (1+2j), 3.4, 2, 2)
```

```
In [86]: t.count(2)
```

```
Out[86]: 3
```

```
In [87]: t.index("hello")
```

```
Out[87]: 2
```

```
In [88]: t1=[1,3,6,8,9]
         t
```

```
Out[88]: (2, True, 'hello', (1+2j), 3.4, 2, 2)
```

tuple slicing

```
In [89]: t1[::]
```

```
Out[89]: [1, 3, 6, 8, 9]
```

```
In [90]: t1[:3]
```

```
Out[90]: [1, 3, 6]
```

```
In [91]: t1[3:]
```

```
Out[91]: [8, 9]
```

```
In [92]: t1[0:5:2]
```

```
Out[92]: [1, 6, 9]
```

tuple indexing

```
In [94]: t2=(3,2,9,5,0)
```

```
In [95]: t2[2]
```

```
Out[95]: 9
```

```
In [96]: len(t2)
```

```
Out[96]: 5
```

loop in the tuple

```
In [98]: for i in t2:
    print(i)
```

```
3
2
9
5
0
```

In [ ]:

```
as tuple is immutable we cant make any changes
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: