In [1]:

```
if True: # indentiation is always 4 spaces
    print('Data Science')
```

```
Data Science
```

In [2]:

```
if False:
    print('Data Science')
print('bye for now')
```

```
bye for now
```

In [3]:

```
if True: # indentiation is always 4 spaces
    print('Data Science')
print('bye for now')
```

```
Data Science
bye for now
```

In [4]:

```
#to print only even number

x = 14
r = x % 2

if r == 0:
    print('Even number')

if r == 1:
    print('Odd Number')
```

```
Even number
```

In [5]:

```
#to print only even number

x = 14
r = x % 2

if r == 0:
    print('Even number')
else:
    print('Odd Number')
```

```
Even number
```

In [6]:

```
#to print only even number
x = 11
r = x % 2

if r == 0:
    print('Even number')
```

In [7]:

```
x = 5
r = x % 2

if r == 0:
    print('Even number')
print('odd number')
```

```
odd number
```

In [8]:

```
x = 15
r = x % 2

if r == 0:
    print('Even number')
print('odd number')
```

```
odd number
```

In [9]:

```
x = 8
r = x % 2

if r == 0:
    print('Even number')
print('odd number')
```

```
Even number
odd number
```

In [10]:

```
x = 8
r = x % 2

if r == 0:
    print('Even number')
if r == 1:
    print('odd number')
```

```
Even number
```

In [11]:

```python
x = 7
r = x % 2

if r == 0:
    print('Even number')
if r == 1:
    print('odd number')
```

```
odd number
```

In [12]:

```python
x = 13
r = x % 2

if r == 0:
    print('Even number')

if r != 0:
    print('odd number')
```

```
odd number
```

if we observe the code its too many line cuz many of the coder always they wanted to reduce the code lenght which is very good practise. instead of 2 if we can use if-- else

In [13]:

```python
x = 2
r = x % 2

if r == 0:
    print( ' Even number')
else:
    print('Odd Number')
```

```
 Even number
```

In [14]:

```python
x = 3
r = x % 2

if r == 0:
    print('Even number')
    if x>5:
        print('greater number')
```

```
else:
    print('Odd Number')
```

```
Odd Number
```

In [15]:

```
x = 4
r = x % 2

if r == 0:
    print('Even number')
    if x>5:
        print('greater number')
else:
    print('Odd Number')
```

```
Even number
```

In [16]:

```
x = 2
r = x % 2

if r == 0:
    print('Even number')
    if x>5:
        print('greater number')
    else:
        print('not greater')
else:
    print('Odd Number')
```

```
Even number
not greater
```

We do have concept of (IF - ELIF- ELSE) e.g i want to print (1--> one , 2 --> two, 3--> three, 4--> four, 5- five)

In [17]:

```
#when you use if it will check all condition but if we mention as elif then it wont check all condition
x = 1

if(x == 1):
    print('one')
if(x == 2):
    print('Two')
if(x == 3):
    print('Three')
```

```
if(x == 4):
    print('four')
```

```
one
```

In [18]:

```
x = 3

if(x == 1):
    print('one')
elif(x == 2):
    print('Two')
elif(x == 3):
    print('Three')
elif(x == 4):
    print('four')
```

```
Three
```

In [19]:

```
x = 5

if(x == 1):
    print('one')
elif(x == 2):
    print('Two')
elif(x == 3):
    print('Three')
elif(x == 4):
    print('four')
```

In [20]:

```
x = 5

if(x == 1):
    print('one')
elif(x == 2):
    print('Two')
elif(x == 3):
    print('Three')
elif(x == 4):
    print('four')
else:
    print('wrong output')
```

```
wrong output
```

In [21]:

```
x = 15

if(x == 1):
    print('one')
elif(x == 2):
    print('Two')
elif(x == 3):
    print('Three')
elif(x == 4):
    print('four')

else:
    print('wrong output')
```

```
wrong output
```

In [22]:

```
print('data science')
```

```
data science
```

In [23]:

```
print('data science')
print('data science')
```

```
data science
data science
```

LOOPS -- in programing world some time we keep on repeating , may be you want to repeat 5 statement so one way is copy & paste multiple times or other way is. if you want to print the datascience 10 times then what you will you cant copy for 10 times , if you want to print 1000 times then you cant do manualy . that is the reason why we need to apply loop -> 2 type of loops -- While loop & For loop

In [24]:

```
i = 1          # initializing
while i<=5:    # condition
    print('data science')
    i = i + 1  # increment
```

```
data science
data science
data science
data science
data science
```

In [25]:

```python
i = 5          # initializing
while i>=1:    # condition
    print('data science')
    i = i - 1  # decrement
```

```
data science
data science
data science
data science
data science
```

In [26]:

```python
i = 1          # initializing
while i<=5:    # condition
    print('data science',':',i)
    i = i + 1  # increment
```

```
data science : 1
data science : 2
data science : 3
data science : 4
data science : 5
```

In [27]:

```python
i = 5          # initializing
while i>=1:    # condition
    print('data science',':',i)
    i = i - 1  # decrement
```

```
data science : 5
data science : 4
data science : 3
data science : 2
data science : 1
```

can we use multiple while loop || nested while loop to understand nested whild indepth understand you can use pycharm debug with f8 option

In [28]:

```
i = 1

while i<=5:
    print(' data science') # when we mention end then new line will not create
    j = 1
    while j<=4:
        print(' technology')
        j = j + 1

    i = i + 1
    print()
```

```
data science
technology
technology
technology
technology

data science
technology
technology
technology
technology

data science
technology
technology
technology
technology

data science
technology
technology
technology
technology

data science
technology
technology
technology
technology
```

In [29]:

```python
i = 1
while i<=5:
    print(' data science', end = "") # when we mention end then new line will not create
    j = 1
    while j<=4:
        print(' technology', end="")
        j = j + 1

    i = i + 1
    print()
```

```
data science technology technology technology technology
data science technology technology technology technology
data science technology technology technology technology
data science technology technology technology technology
data science technology technology technology technology
```

In [30]:

```python
i = 1
while i<=5:
    print(' data science', end = "  *") # when we mention end then new line will not create
    j = 1
    while j<=4:
        print(' technology', end="  *")
        j = j + 1

    i = i + 1
    print()
```

```
data science  * technology  * technology  * technology  * technology  *
data science  * technology  * technology  * technology  * technology  *
data science  * technology  * technology  * technology  * technology  *
data science  * technology  * technology  * technology  * technology  *
data science  * technology  * technology  * technology  * technology  *
```

In [31]:

```python
i = 1
while i <= 4 :
    j = 0
    while  j <= 3 :
        print(i*j, end=" ")
        j += 1
    print()
    i += 1
```

```
0 1 2 3
0 2 4 6
0 3 6 9
0 4 8 12
```

FOR LOOP - normally while loop it work with condition but for loop it will work with sequence (list, string,int)

In [32]:

```python
name = 'nit'
for i in name:
    print(i)
```

```
n
i
t
```

In [33]:

```
for i in [2, 3, 7.8, 'hi']:
    print(i)
```

```
2
3
7.8
hi
```

In [34]:

```
for i in range(5):
    print(i)
```

```
0
1
2
3
4
```

In [35]:

```
for i in range(1,5):
    print(i)
```

```
1
2
3
4
```

In [36]:

```
for i in range(1,10,3):
    print(i)
```

```
1
4
7
```

In [37]:

```
# print the numer which is not divisible by 5

for i in range(1,11):

    if i%5 != 0 :
      print(i)
```

```
1
2
3
4
6
7
8
9
```

In [38]:

```python
# can you write the python code for 5 multiplication table

for i in range(1,51):

    if i%5 == 0:
      print(i)
```

```
5
10
15
20
25
30
35
40
45
50
```

In [39]:

```python
for i in range(1,11):
    print(i)
```

```
1
2
3
4
5
6
7
8
9
10
```

In [43]:

```python
for i in range(1,11):
    if i == 5:
        break #==> WHILE YOU WORK WITH COMPUTER VISION PROJECT
    print(i)
```

```
1
2
3
4
```

In [44]:

```
for i in range(1,11):
    if i == 5:
        break #==> WHILE YOU WORK WITH COMPUTER VISION PROJECT
    print(i)
```

```
1
2
3
4
```

In [45]:

```
for i in range(1,11):
    if i == 5:
        break #==> WHILE YOU WORK WITH COMPUTER VISION PROJECT
print(i)
```

```
5
```

In [46]:

```
# in continue ,  loop wont be terminate
```

In [47]:

```
for i in range(1,11):
    if i == 5:
        continue
    print(i)
```

```
1
2
3
4
6
7
8
9
10
```

In [48]:
```
for i in range(1,11):
    if i == 5:
        continue
    print(i)
```

```
1
2
3
4
6
7
8
9
10
```

In [49]:
```
#PASS Statement - pass the code & it wont go
```

In [51]:
```
for i in range(1,11):
    pass
```

PRINTING PATTERN IN PYTHON

In [52]:
```
print('# # # #')
print('# # # #')
print('# # # #')
print('# # # #')
```

```
# # # #
# # # #
# # # #
# # # #
```

In [53]:
```
for j in range(4):
    print('#', end="  ")
```

```
#  #  #  #
```

In [54]:
```
for j in range(4):
    print('#', end="  ")

for j in range(4):
    print('#', end="  ")
```

```
#  #  #  #  #  #  #  #  
```

In [55]:

```python
for j in range(4):
    print('#', end="  ")

print()

for j in range(4):
    print('#', end="  ")
```

```
#  #  #  #  
#  #  #  #  
```

In [56]:

```python
for j in range(4):
    print('#', end="  ")

print()

for j in range(4):
    print('#', end="  ")

print()

for j in range(4):
    print('#', end="  ")

print()

for j in range(4):
    print('#', end="  ")
```

```
#  #  #  #  
#  #  #  #  
#  #  #  #  
#  #  #  #  
```

In [57]:

```python
for i in range(4):
    for j in range(4):
        print('#', end="  ")
    print()
    # pease use debug mode
#  #  #  #
#  #  #  #
#  #  #  #
#  #  #  #
```

In [58]:

```
for i in range(5):
    for j in range(i):
        print('#', end="  ")
    print()
```

```
#
#  #
#  #  #
#  #  #  #
```

In [59]:

```
for i in range(4):
    for j in range(i+1):
        print('#', end="  ")
    print()
```

```
#
#  #
#  #  #
#  #  #  #
```

In [60]:

```
for i in range(4):
    for j in range(4-i):
        print('*', end="  ")
    print()
```

```
*  *  *  *
*  *  *
*  *
*
```

In [61]:

```
for i in range(4):
    for j in range(4-i):
        print('#', end="  ")
    print()
```

```
#  #  #  #
#  #  #
#  #
#
```

For|Else in python

in other language for else not supportable but in python it is supportable eg- lets print the number from 1- 20 & we dont want print number which is divisible by 5

In [62]:

```
nums = [12,15,18,21,26]

for num in nums:
    if num % 5 == 0:
        print(num)
```

```
15
```

In [63]:

```
nums = [12,14,18,21,25]
for num in nums:
    if num % 5 == 0:
        print(num)
```

```
25
```

In [64]:

```
nums = [12,14,18,21,25,20]
for num in nums:
    if num % 5 == 0:
        print(num)
```

```
25
20
```

In [65]:

```
nums = [12,14,18,21,25,20]
for num in nums:
    if num % 5 == 0:
        print(num)
        break
```

```
25
```

In [66]:

```
nums = [10,14,18,21,20,25]
for num in nums:
    if num % 5 == 0:
        print(num)
        break #it will print only 1 number then it break
```

```
10
```

In [67]:

```
nums = [7,14,18,21,23,27] #hear there is no number which is divisible by 5 we got output as blank
for num in nums:
    if num % 5 == 0:
        print(num)
       # break
```

In [68]:

```
nums = [7,14,18,21,23,27,22] #hear there is no number which is divisible by 5 we got output as blank
for num in nums:
    if num % 5 == 0:
        print(num)
        break
    else:
        print('Number Not Found') #every iteration it cheking condition

```

```
Number Not Found
Number Not Found
Number Not Found
Number Not Found
Number Not Found
Number Not Found
Number Not Found
```

In [69]:

```
nums = [7,14,18,21,23,27] #hear there is no number which is divisible by 5 we got output as blank
for num in nums:
    if num % 5 == 0:
        print(num)
        #break
else:
        print('Not Found') # hear else we dont write in if block but we can write in for block only
```

```
Not Found
```

In [70]:

```
nums = [10,14,18,21,20,27] #hear there is no number which is divisible by 5 we got output as blank
for num in nums:
    if num % 5 == 0:
        print(num)
        #break
else:
        print('Not Found')
```

```
10
20
Not Found
```

In [71]:

```
nums = [10,14,18,21,20,27] #hear there is no number which is divisible by 5 we got output as blank
for num in nums:
    if num % 5 == 0:
        print(num)
        break
else:
        print('Not Found')
```

```
10
```

prime number - how to check given number is prime number or not

In [72]:

```
num = 12

for i in range(2,num):
    if num % i == 0:
        print('Not prime Number')
        break
else:
    print('Prime Number')
```

```
Not prime Number
```

In [73]:

```
num = 13

for i in range(2,num):
    if num % i == 0:
        print('Not prime Number')
        break
else:
    print('Prime Number')
```

```
Prime Number
```

In [76]:

```
from array import *

arr = array('i',[])

n = int(input('Enter the length of the array'))

for i in range(5):
    x = int(input('Enter the next value'))
    arr.append(x)
    print(arr)
```

```
array('i', [3])
array('i', [3, 4])
array('i', [3, 4, 6])
array('i', [3, 4, 6, 8])
array('i', [3, 4, 6, 8, 9])
```

Way of creating array using numpy

In [77]:

```
from numpy import *
arr = array([1,2,3,4,5])
print(arr)
type(arr)
```

```
[1 2 3 4 5]
```

Out[77]:

```
numpy.ndarray
```

In [78]:

```
print(arr.dtype)
```

```
int32
```

In [79]:

```
arr = array([1,2,3,4,5.9])
print(arr)
```

```
[1.  2.  3.  4.  5.9]
```

In [80]:

```
print(arr.dtype)
```

```
float64
```

In [81]:

```
arr2 = array([1,2,3,4,5.9],float)
arr2
```

```
Out[81]: array([1. , 2. , 3. , 4. , 5.9])
```

```
In [82]: arr3 = array([1,2,3,4,5.6],int)
         arr3
```

```
Out[82]: array([1, 2, 3, 4, 5])
```

```
In [83]: import numpy as np
```

```
In [84]: arr4 = np.linspace(0, 16, 10) # break the code between 10 spaces between 0 to 16 but why decimal becuase we break int
         arr4
```

```
Out[84]: array([ 0.        ,  1.77777778,  3.55555556,  5.33333333,  7.11111111,
                 8.88888889, 10.66666667, 12.44444444, 14.22222222, 16.        ])
```

```
In [85]: arr5 = np.arange(0,10,2) # arange - as range
         arr5
```

```
Out[85]: array([0, 2, 4, 6, 8])
```

```
In [86]: arr6 = np.zeros(5)
         arr6
```

```
Out[86]: array([0., 0., 0., 0., 0.])
```

```
In [87]: arr7 = np.ones(5)
         arr7
```

```
Out[87]: array([1., 1., 1., 1., 1.])
```

```
In [ ]:
```