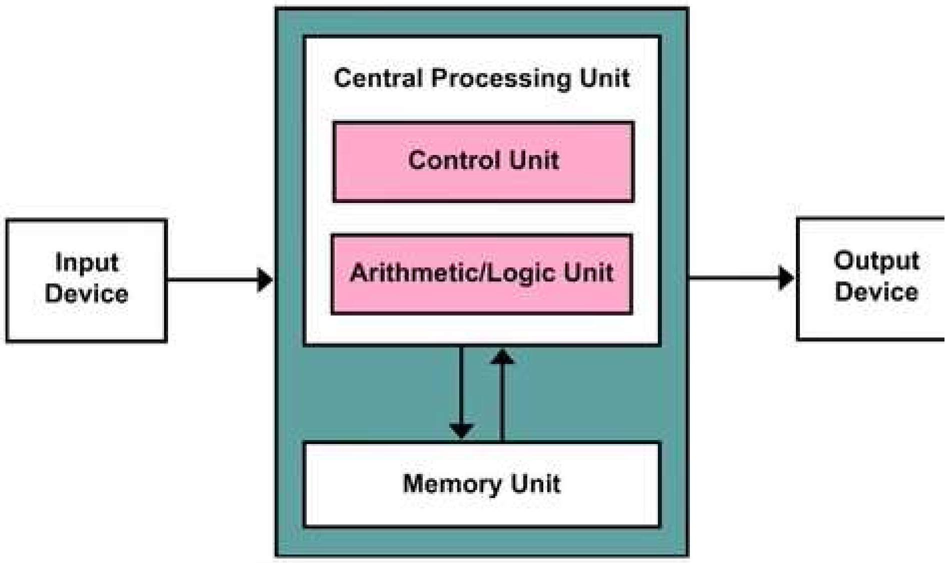


cpu

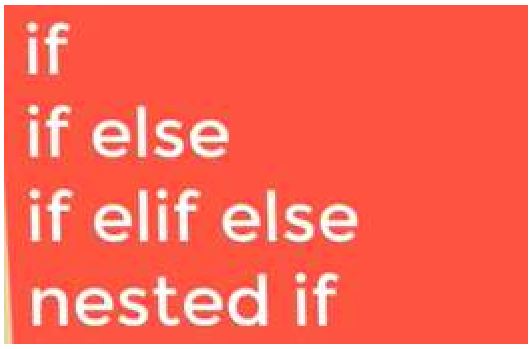
- cpu has 3 part - CU (Contol Unit) | MU(Memory Unit) | ALU (Arithmetic | Logic unit)
- MEMORY UNIT -->

so far we work in memory unit-that means we save variable in memory & call the varible from memory & trying to save the data with the help of variable

- ALU (ARITHMETIC UNIT & LOGIC UNIT) -->

ARITHMETIC UNIT --> when it comes to Arithmetic unit it performs calculation & we done so far addition, substraction, multiplication , division. LOGICAL UNIT --> when i talk about logical unit which makes your computer to think means user programed so that and let your computer think. in real life also we apply many condition to reach the goal. if this not possible then we can do that for example. e.g after complet +2 which field you can go for if not this field else what is other files. while we talk amoung each other many situation unknowily we are using if else condition.

LETS UNDERSTAND THE CONDITION TO COMPUTER TO THINK FOR THAT WE HAVE SOME

SPECIAL KEYWORD - 
if
if else
if elif else
nested if

In [1]:
```
if True: # indentiation is always 4 spaces
    print('Data Science')
```

```
Data Science
```

In [2]:
```python
if False:
    print('Data Science')
print('bye for now')
```

```
bye for now
```

In [3]:
```python
if True: # indentiation is always 4 spaces
    print('Data Science')
print('bye for now')
```

```
Data Science
bye for now
```

Lets do one program as if number is divide by 2 then reminder is 0 then it is even number if reminder is not 0 then it is odd number

In [4]:
```python
#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]:
```python
#to print only even number

x = 14
r = x % 2

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

```
Even number
```

In [6]:
```python
#to print only even number
x = 11
r = x % 2

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

In [7]:
```python
x = 5
r = x % 2

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

```
odd number
```

In [8]:
```python
x = 8
r = x % 2

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

```
Even number
odd number
```

In [9]:
```python
x = 8
r = x % 2

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

```
Even number
```

In [10]:
```python
x = 7
r = x % 2

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

```
odd number
```

In [11]:
```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 [12]:
```python
x = 2
r = x % 2

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

```
 Even number
```

In [13]:
```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 [14]:
```python
x = 4
r = x % 2

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

```
Even number
```

NESTED IF (if we have 2 condition so we need to implment with nested if)

In [15]:
```python
x = 6
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
greater number
```

In [16]:
```python
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
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 = 2

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

```
Two
```

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]:
```
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]:
```
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]:
```
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 c
    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 no
    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]:
```
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]:
```
name = 'nit'
for i in name:
    print(i)
```

```
n
i
t
```

In [33]:
```
name1 = [1,3.5,'hallo']

for i in name1:
    print(i)
```

```
1
3.5
hallo
```

In [34]:
```
for i in [2, 3, 7.8, 'hi']:
    print(i)
```

```
2
3
7.8
hi
```

In [35]:
```
for i in range(5):
    print(i)
```

```
0
1
2
3
4
```

In [36]:
```
for i in range(1,5):
    print(i)
```
```
1
2
3
4
```

In [37]:
```
for i in range(1,10,3):
    print(i)
```
```
1
4
7
```

In [38]:
```
# 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 [39]:
```
# 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
```

LETS DISCUSS ABOUT 3 KEYWORDS -- BREAK || CONTINUE || PASS BREAK STATEMNT - if you apply break statment in a loop then it will end the loop # Pass = skips block of code(function, class etc) # Continue= skips 1 step/iteration during loop # Break= jumps out of the function/loop

In [40]:
```
for i in range(1,11):
    print(i)
```

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

In [41]:
```
for i in range(1,11):
    if i == 5:
        break #==> WHILE YOU WORK WITH COMPUTER VISION PROJECT
```

In [42]:
```
for i in range(1,11):
    if i == 5:
        break #==> WHILE YOU WORK WITH COMPUTER VISION PROJECT
    print(i)
```

```
1
2
3
4
```

In [43]:
```
for i in range(1,11):
    if i == 5:
        break #==> WHILE YOU WORK WITH COMPUTER VISION PROJECT
print(i)
```

```
5
```

in continue , loop wont be terminate

In [44]:
```
for i in range(1,11):
    if i == 5:
        continue
    print(i)
```

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

In [45]:
```
for i in range(1,11):
    if i == 5:
        continue
    print('hello ',i)
```

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

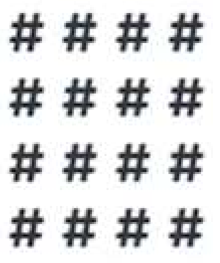
#PASS Statement - pass the code & it wont go

In [46]:
```
for i in range(1,11):
```

```
  Cell In[46], line 1
    for i in range(1,11):
                         ^
SyntaxError: incomplete input
```

In [47]:
```
for i in range(1,11):
    pass
```

PRINTING PATTERN IN PYTHON



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

In [48]:
```
print('# # # #')
print('# # # #')
print('# # # #')
print('# # # #')
```

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

In [49]:
```
for j in range(4):
    print('#', end="  ")
```

```
#  #  #  #
```

In [50]:
```
for j in range(4):
    print('#', end="  ")

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

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

In [51]:
```
for j in range(4):
    print('#', end="  ")
print()

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

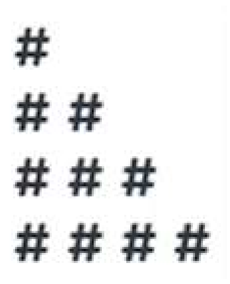
```
#  #  #  #
#  #  #  #
```

In [52]:
```
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 [53]:
```
for i in range(4):
    for j in range(4):
        print('#', end="  ")
    print()
    # pease use debug mode
```

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



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

In [54]:
```
for i in range(5):
    for j in range(i):
        print('#', end="  ")
    print()
```

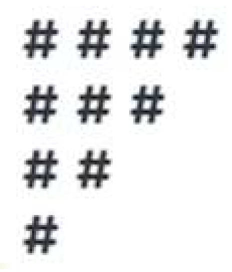
```
#
#  #
#  #  #
#  #  #  #
```

In [55]:
```python
for i in range(4):
    for j in range(i+1):
        print('#', end="  ")
    print()
```

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

In [56]:
```python
for i in range(4):
    for j in range(4-i):
        print('*', end="  ")
    print()
```

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


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

In [57]:
```python
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 [58]:
```python
nums = [12,15,18,21,26]

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

```
15
```

In [59]:
```
nums = [12,14,18,21,25]
for num in nums:
    if num % 5 == 0:
        print(num)
```

```
25
```

In [60]:
```
nums = [12,14,18,21,25,20]
for num in nums:
    if num % 5 == 0:
        print(num)
```

```
25
20
```

In [61]:
```
nums = [12,14,18,21,25,20]
for num in nums:
    if num % 5 == 0:
        print(num)
        break
```

```
25
```

In [62]:
```
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 [63]:
```
nums = [7,14,18,21,23,27] #hear there is no number which is divisible by 5 we got o
for num in nums:
    if num % 5 == 0:
        print(num)
       # break
```

In [64]:
```
nums = [7,14,18,21,23,27,22] #hear there is no number which is divisible by 5 we go
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 [65]:
```
nums = [7,14,18,21,23,27] #hear there is no number which is divisible by 5 we got o
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 i
```

```
Not Found
```

In [66]:
```
nums = [10,14,18,21,20,27] #hear there is no number which is divisible by 5 we got
for num in nums:
    if num % 5 == 0:
        print(num)
        #break
else:
        print('Not Found')
```

```
10
20
Not Found
```

In [67]:
```
nums = [10,14,18,21,20,27] #hear there is no number which is divisible by 5 we got
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 [68]:
```
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 [69]:
```
num = 13

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

```
Prime Number
```

In [70]:
```
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)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[70], line 5
      1 from array import *
      3 arr = array('i',[])
----> 5 n = int(input('Enter the length of the array'))
      7 for i in range(5):
      8     x = int(input('Enter the next value'))

ValueError: invalid literal for int() with base 10: 'q'
```

Way of creating array using numpy

In [71]:
```
from numpy import *
arr = array([1,2,3,4,5])
print(arr)
type(arr)
```

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

Out[71]:
```
numpy.ndarray
```

In [72]:
```
print(arr.dtype)
```

```
int32
```

In [73]:
```
arr = array([1,2,3,4,5.9])
print(arr)
```

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

In [74]:
```
print(arr.dtype)
```

```
float64
```

In [75]:
```
arr2 = array([1,2,3,4,5.9],float)
arr2
```

Out[75]:
```
array([1. , 2. , 3. , 4. , 5.9])
```

In [76]:
```
arr3 = array([1,2,3,4,5.6],int)
arr3
```

Out[76]:
```
array([1, 2, 3, 4, 5])
```

In [77]:
```
import numpy as np
```

In [78]:
```
arr4 = np.linspace(0, 16, 10) # break the code between 10 spaces between 0 to 16 bu
arr4
```

```
Out[78]: array([ 0.        ,  1.77777778,  3.55555556,  5.33333333,  7.11111111,
                 8.88888889, 10.66666667, 12.44444444, 14.22222222, 16.        ])
```

```
In [79]: arr5 = np.arange(0,10,2) # arange - as range
         arr5
```

```
Out[79]: array([0, 2, 4, 6, 8])
```

```
In [80]: arr6 = np.zeros(5)
         arr6
```

```
Out[80]: array([0., 0., 0., 0., 0.])
```

```
In [81]: arr7 = np.ones(5)
         arr7
```

```
Out[81]: array([1., 1., 1., 1., 1.])
```

```
In [ ]:
```

```
In [ ]:
```