Markdown notes

- **Bold**
- *Italic*
- ***BI***
- Normal text
  - sublist 1
  - sublist 2

-[]Option 1
-[]Option 2
-[x]Option 3

markdown@google.com (mailto:markdown@google.com)

Python Basics

Python version 3.7

- Scripting
- Object Oriented
- Functional

In [1]:
```
#Python comments
```

In [8]:
```
print("Good Afternoon","!!",end="||")   #Basic output
print("Hello Python")
```

```
Good Afternoon !!||Hello Python
```

In [ ]:
```
1
```

Assignment

In [17]:
```
1 n1 = 1234567  #Single variable assignment
2
3 n2 = n3 = n4 = n1  #Multi Variable Assignment of the same value
4
5 a,b,c = 123, 234, 345 # Multi Variable Assignment with different values
6
7 print(a,b,c)
```

```
123 234 345
```

In [ ]:
```
1
```

DataTypes & Type Conversions

- Integer
- Float
- string

In [50]:
```
1  type(a)
2
3  s1 = "Python"
4  type(s1)
5
6  f1 = 12.345
7  type(f1)
8
9  int(f1)
10 float(str(int(f1)))
11
```

Out[50]:
```
12.0
```

Arithmetic Operations

- +
- -
- *
- /
- %
- **
- //

In [53]:
```
n1 % 11

n3 = n2 ** 12

type(n3)
len(str(n3))
n3

atoms = 10 ** 82
len(str(atoms))     # to calculate the length of a variable
type(str(atoms))     # to check the type of a variable



12.2321 ** 9
122321 ** 999
```

Out[53]:
```
259132476157026937968416257509659203288789300719948715022377926182638343382063340476537881009641582024759701922404623947103387289390214760783589387449759174707315949596073331411187958452280409617063236485572286559439561820132552677034424962529034805226004277368945602164539729751406921314013982546149585198149462314375125407390122967018281912287349906431754798504774043565305933009368989878102290789997155975752493945596951610210695112453950590773856270966194951627192626571073773111525263114591574533908553451216653293183776307115682191498490336690483061073991286314984738009786895013276325830254388801260477974633067186661556953047232011095168965359538696163972812837729009830384753390382830671666190738881052455350529342368709780761977889769364777845635317092902647041169203677691285195813362268966141781382594190610556743672532536382860710970541498997600807062553690899135938515943493404361493152771345649983714334313847685957387472481493301848824428193681311150061812526236625133301371829042187735336909742507016991871787281064281549939221292576007328966677879000161277713566839324146285832238104951110769289118264446628268728220483204343862297248841187204067231446291052183893652770157254983913418416276211728160228635855795305332951987962201887571568936425740898830928931245933552989489815598656307708207217001788335556888853524722661667236258950623794089147241288697765231035476734879578293100684073926938925940434185103079728819877117249055546699194313716688301425980521653552154941663274343693311160827635215883875026544237531444874270726492536774090464068123646000703361729793335105701061323472465772860356898621411020885301712354793726421180096515690430589882048737390565746007536142031608152593873018725781619420501904732562504717067973263014065196391468016595015537629036260167936548753020538722419755608869584901755204735933410907643389798319212466059893811095345550844258131434610752266468114628781573932211783730857187906496243892967647791534678105522394123603286064195026832028683689732997645709315044882570275869056756624316512616460301583678187804181905382105459898366063520799641613440327969299095999384733583398177694066204838931537597687753847450257414656816331027493418856989601649696770737037702589826112382884076051779335158903323192582103606213009339589182478800384800018047885431550711255007414114014528459087233554851649969941860539808816859284335990704703558355765375397550191516917919784345186034105762836880209934813671411977028168127747379017295408523794930216996937218773933159357114867942306526581438593532586289716648487233395344972095195385952698248142864975847289576932962964399841225211802850411811099896095899389036799546189940910033807824920690523650911252141849906824238312397218889155405721819078958778806409087176731572110401572187027261007581757762829819804917404953428708584773055208432065034314188147903425750242160564977288603995420138718236205368536729464547435898868372536316822459876764250224653625570782803936927223873164497782375358102697497906533626757423287232999145445210703334964190168379343337715279711607371213190850861889476253118000760355329016132282681987254489192490248096320464105032237632411
```

```
30099228583102487391133231754525347696207382315354239948644891340699871499376938007752481311200296218921756066489895771406154470650879919900271178321693031116134005788188846281788435575432267160822490136896442984470419681207316626379667116125870561430317530125663361094214558765148587918265594089628803363241948429401525442600011077769841611107737820440204344251170923197185062360025696693274488606043670279223322758428247986843564783701463567676751782923910890425004995250908537714422175569668826566396156052890084203263708020767238469070246255523794045770257721922791531133217034607644030478349993892232125669512402295002304953534989957539673261436676609574291835569193178656556761285881218232845384530640044759616593534245705805383109676784416096816918301007322001791683139647218207398540308345644426180170304366132312291467261374602403366809403819209050804218181506399988534881808558130654807273587826687081352862987012348010691943463704802120320047461209950891004266637120053770861334869790379641540471616748662117273572715883252472234392854702465817961847980452722218365035879568562223918850735113396176560253450047798314334311867264173612797600670163557555594441584390789231758194002838386928254240705543100560544065212106076919235733081192028407598378589440704452788822883918047833209916279757165910235444903303800350613463080640026109276443527537933506301204231347142606483283960817509168226008049013700386812922253962348738748735092715075447534828791657997545226513033327273465961845000992326630082945485987331492824782191255433450794693399597477633802016781822097239352691257329553353839604359873473460335496663120873474796750889413311800468040007936251562555319185692727708128834028040846666357594082374947714672286574638550277900056303149730736905434691177900116132887770864680935969928510869141810522310194020378048115973257290562276020566191466971630238405382997672855972840613399646416475628938072047817208 1
```

In [ ]:
```
1
```

Conditionals

In [57]:
```
1 if atoms < 10 ** 9:
2     print("TRUE")
3 else:
4     print("FALSE")
```

```
FALSE
```

In [ ]:
```
1
```

In [58]:
```
1 # Check if a number is even or odd
2
3 n = 123
4
5 if n% 2 == 0:
6     print("Even")
7 else:
8     print("Odd")
```

```
Odd
```

In [2]:
```
# Find the greatest of three numbers

n1 = int(input("Enter the first number "))
n2 = int(input("Enter the second number "))
n3 = int(input("Enter the third number "))

if n1 > n2 and n1 > n3:
    print(n1, "is the greatest")
elif n2 > n3:
    print(n2,"is the greatest")
else:
    print(n3,"is the greatest")

```

```
Enter the first number 3
Enter the second number 5
Enter the third number 10
10 is the greatest
```

In [24]:
```
# Check if a year is Leap year or not
year = int(input("Enter a year: "))

if year % 400 == 0 or (year % 100 != 0 and year % 4 ==0):
    print("Leap Year")
else:
    print("Not a Leap Year")

```

```
Enter a year: 2011
Not a Leap Year
```

In [22]:
```
# check if a number exists in given range ( both bounds are inclusive)

n1 = int(input("Enter a number: "))
lb = int(input("Enter lower bound: "))
ub = int(input("Enter upper bound: "))

if n1>=lb and n1<=ub:
    print("Number exists in given range")
else:
    print("Not in given range")


```

```
Enter a number: 10
Enter lower bound: 5
Enter upper bound: 15
Number exists in given range
```

In [9]:
```
# Calculate the number of digits in a number

Number = int(input("Enter a number"))
Count = 0
while(Number > 0):
    Number = Number // 10
    Count = Count + 1

print("\n Number of Digits in a Given Number = %d" %Count)
```

```
Enter a number123456789

 Number of Digits in a Given Number = 9
```

In [12]:
```
Number = int(input("Enter a number "))
print(len(str(Number)))
```

```
Enter a number10
2
```

In [21]:
```
# check if the number is multiple of 10
n=int(input("Enter a number"))
if n% 10 ==0:
    print(n,"is a factor")
else:
    print(n,"not a factor")

```

```
Enter a number25
25 not a factor
```

In [31]:
```
# check if the number is factor of 1000
n1=int(input("Enter a number"))
if 1000 % n1 == 0:
    print("factor")
else:
    print("not a factor")


```

```
Enter a number50
factor
```

In [19]:
```
# Check if the given string is equal to a number
s1="123456"
n1=123456
if str(n1) == s1:
    print(n1,"is equal to",s1)
else:
    print(n1,"is not equal to",s1)
```

```
123456 is equal to 123456
```

In [16]:
```
# Calculate the square root of a number without functions

n1 =123
n1 ** 0.5
```

Out[16]:
```
11.090536506409418
```

In [32]:
```
# Calculate the number of nano seconds in a given year (considering leap yea

year = 2016


if year % 400 == 0 or (year % 100 != 0 and year % 4 ==0):
    print(366 * 24 * 60 * 60 * (10**9))
else:
    print(365 * 24 * 60 * 60 * (10**9))
```

```
31622400000000000
```

In [ ]:
```

```