Markdown Basics

Markdown Basics

Markdown Basics

- **point1** (Bold)
- *point2* (Italic)
- ***point3*** (Bold and Italic)
- Normal Text
    - Sublist 1 *Sublist2

> 1. Point1
> 2. Point2

- Google site -- 1 (http://www.google.com): http://www.google.com (http://www.google.com)
- msn site -- 2 (http://www.msn.com): http://www.msn.com (http://www.msn.com)
- []option 1
- Google site -- [Google].1 (http://www.google.com)

- []option 2
- msn site -- [msn].2 (http://www.msn.com)

Python Basics

- Python version 3.7
- Functional Programming
- Object oriented Programming
- Sripting Programming

In [3]:

```
print("Hello Gitam")
```

```
Hello Gitam
```

In [4]:

```
print("Hyderabad")
```

```
Hyderabad
```

In [13]:

```
print("Hello Gitam","Hyderabad ",end ="")
print("welcome ",end ="")
print("ECE")
```

```
Hello Gitam Hyderabad welcome ECE
```

In [44]:

```
n1=100
a=b=c=20
a1,b1,c1=111,222,333
print(n1)
print(a,b,c)
print(a1,b1,c1)
```

```
100
20 20 20
111 222 333
```

In [43]:

```
n1=1
print(n1)
print(n1, n1)
print(n1, n1, n1)
```

```
1
1 1
1 1 1
```

In [45]:

```
a=100;
s1="python"
s2='p'
s3=19.1
print(a,s1,s2,s3)
print(type(a),type(s1),type(s2),type(s3))
```

```
100 python p 19.1
<class 'int'> <class 'str'> <class 'str'> <class 'float'>
```

In [1]:

```
n1=5
n2=6
a=b=c=d=300
e=0
print(n1)
```

```
5
```

In [2]:

```
i=100
print(type(i))
s1=str(i)
print(type(s1))
f1=float(i)
print(type(f1))
```

```
<class 'int'>
<class 'str'>
<class 'float'>
```

In [5]:

```
s1="100"
print(type(s1))
a=int(s1)
print(type(a))
f=1.5
a1=int(f)
print(type(a1))
print(a1)
```

```
<class 'str'>
<class 'int'>
<class 'int'>
1
```

In [7]:

```
# length of a string
a1=1234
print(len(str(a1)))
```

```
4
```

In [13]:

```
#Reading a value-input function
s1=input("enter your name")
print(s1)
print(type(s1))
```

```
enter your namevinay
vinay
<class 'str'>
```

In [16]:

```
# want a number as input
n1= int(input("enter a number"))
print(n1,type(n1))
```

```
enter a number19
19 <class 'int'>
```

Operators

- operator is a symbols used to perform specific operations

In [18]:

```
n1=1234
print(n1+10)
print(n1-10)
print(n1*10)
print(n1/10)
print(n1%10)
print(n1//10)
print(n1**10)
```

```
1244
1224
12340
123.4
4
123
8187505353567209228244052427776
```

Precedence of the Arth operators .parenthesis .power .division .multiplication .addition .sub

In [19]:

```
x=1+2**3/4+5
print(x)
```

```
8.0
```

Relation Operatos

.== .!= .Greater(>) .less than(<) .less than and equals to(<=) .greater than and equals to(=>)

Logical Operators

used to combine more than single comdition

.and .or .not

In [20]:

```
i=100
a1= (i>15) and (i<75)
a2= (i>15) and (i<150)
```

In [26]:

```
#To check a given number is even or odd
n= int(input("enter a number"))
if n%2 ==0:
    print("even")
else:
    print("odd")
```

```
enter a number6
even
```

In [32]:

```
#To check a given number is perfectly multiple of 3 and 5
n=int(input("enter a number"))
if n%3==0 and n%5==0:
    print("yes")
else:
    print("no")
```

```
enter a number6
no
```

In [29]:

```
# Test given number is positive,negative or zero
n=int(input("enter a number"))
if n==0:
    print("zero")
elif n>0:
    print("positive Number")
elif n<0:
    print("negative Number")
```

```
enter a number5
positive Number
```

In [2]:

```
# Find the large number from the given numbers
a1=int(input("enter number 1 "))
a2=int(input("enter number 2 "))
a3=int(input("enter number 3 "))

if a1>a2 and a1>a3:
    print(a1,"is the largest number")
elif a2>a1 and a2>a3:
    print(a2,"is the largest number")
elif a3>a1 and a3>a2:
    print(a3,"is the largest number")
```

```
enter number 1 5
enter number 2 6
enter number 3 4
6 is the largest number
```

In [6]:

```
# check 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 leap year")
```

```
enter a year2012
leap year
```

Iterative Statement

In [7]:

```
# Need print "Gitam" for 5 times
print("Gitam")
print("Gitam")
print("Gitam")
print("Gitam")
print("Gitam")
```

```
Gitam
Gitam
Gitam
Gitam
Gitam
```

In [13]:

```
x=0
while x<5:
    print("Gitam")
    x=x+1
```

```
Gitam
Gitam
Gitam
Gitam
Gitam
```

In [4]:

```
# Print N Natural numbers using while loop
n=int(input("enter n "))
x=1
while x<=n:
    print(x,end =" ")
    x=x+1
```

```
enter n 15
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
```

In [2]:

```
# Sum of Even number Series
n=int(input("enter n "))
i=1
sum=0
while i<=n:
    if i%2==0:
        sum=sum+i
    i=i+1
print(sum)
```

```
enter n 10
30
```

In [7]:

```
#reverse of number
n=int(input("Enter a number"))
while n !=0:
    r=n%10
    print(r,end =" ")
    n=n //10
```

```
Enter a number123
3 2 1 
```

Functional programming

. Simple . Easy read . Lengthy program divides into sub programs

In [15]:

```
# def nameofthefunction(<parameter>)
      statement
      return
```

```
  File "<ipython-input-15-58136d847ed9>", line 2
    statement
    ^
IndentationError: unexpected indent
```

In [4]:

```
def addEvenDigits(n):
    sum=0
    while n!= 0:
        r=n%10
        if r%2 ==0:
            sum=sum+r
        n=n//10
    print(sum)
    return
addEvenDigits(1234)
```

```
6
```

In [7]:

```
#Read a number--19528
#Print the largest digit

def printLarge(n):
    large=0
    while n!=0:
        r=n%10
        if large <r:
            large=r
        n=n//10
    return large
printLarge(19528)
```

Out[7]:

```
9
```

In [1]:

```
x=10
print(x)
```

```
10
```

In [1]:

```
# Reverse of Number
#Output is Reverse of given number

def reverseNumber(n):
    rev=0
    while n !=0:
        r=n%10
        rev=rev * 10 + r
        n=n//10
    return rev
reverseNumber(123)
```

Out[1]:

```
321
```

In [8]:

```
# Palindrome or not

def isPalindrome(n):

    rev=0
    buffer =n
    while n!=0:
        r=n%10
        rev=rev*10+r
        n=n//10
    if buffer == rev:
        return "Yes"
    else:
        return "No"

print(isPalindrome(252))
print(isPalindrome(123))
```

```
Yes
No
```

In [9]:

```
#Function to print N Natural numbers with for loop
def printNaturalNumbers(n):
    for x in range(1,n+1):
        print(x,end=" ")
    return
printNaturalNumbers(10)
```

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

In [13]:

```
def printnumbers(p,q):
    for x in range(p,q+1):
        print(x,end=" ")
    return
printnumbers(10,20)
```

```
10 11 12 13 14 15 16 17 18 19 20
```

In [14]:

```
# Printing Alternate numbers
#[500,520] -- 500 502 504 506 ----

def printaltnumbers(p,q):
    for x in range(p,q+1,4):
        print(x,end=" ")
    return
printaltnumbers(500,520)
```

```
500 504 508 512 516 520
```

In [16]:

```
# Printing reverse series
# i/p-1,10
# o/p-10 9 8 -----1

def printrevseries(start,end):
    for x in range(end,start-1,-1):
        print(x,end=" ")
    return
printrevseries(1,10)
```

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

In [1]:

```
#Problem name
#Algorithm
#Sample i/p and o/p
# Flowchart
# Python code
```

In [3]:

```
#Given number is prime or not

def isprime(n):
    flag= True
    for i in range(2,n//2+1):
        if n%i == 0:
            flag=false
            return flag
    return flag
isprime(11)
```

Out[3]:

```
True
```

In [8]:

```
# Function to find Prime numbers count from 1 to N
# 10 -- 4(2,3,5,7)

def primeCount(n):
    cnt=0
    for a in range(2,n+1):
        k=0
        for i in range(2,a//2+1):
            if a%i ==0:
                k=k+1
        if(k<=0):
            cnt=cnt+1
    return cnt
primeCount(10)
```

Out[8]:

```
4
```

In [10]:

```
# Factors of a given number

def factorslist(n):
    for i in range(1,n+1):
        if n % i ==0:
            print(i,end=" ")
    return
factorslist(12)
```

```
1 2 3 4 6 12
```

In [11]:

```
#Individual digit fatorial sum is equal to the original number or not
# ex: 145 -- Yes(5! +4! + 1!=145)
# 123 -- No(3! + 2!+ 1! =9)

def factorial(n):
    fact =1
    for i in range(2,n+1):
        fact*=i
    return fact
def digitFactSum(n):
    sum=0
    buffer=n
    while n!=0:
        r=n%10
        sum+=factorial(r)
        n=n//10
    if sum==buffer:
        return "Yes"
    else:
        return "No"
    return
print(digitFactSum(145))
```

```
Yes
```

In [13]:

```
# Function to return the count of palindrome numbers
#i/p: 1 10
#o/p: 9 (1,2,3,4,5,6,7,8,9)

def isPalindrome(n):
    rev=0
    buffer= n
    while n!=0:
        r=n% 10
        rev=rev* 10+r
        n=n//10
    if rev ==buffer:
        return True
    else:
        return False
    return

def countPalindrome(lb,ub):
    cnt=0
    while lb != ub:
        #Implement
        if isPalindrome(lb) == True:
            cnt=cnt+1
        lb=lb+1
    return cnt
countPalindrome(1,10)
```

Out[13]:

```
9
```

In [16]:

```python
#Function to generate the Perfect numbers in a given range
# Perfect Number :sum of all its factors same as original number
#Ex: 6 --1 2 3 (1+2+3)
# i/p : 1  10
#o/p : 6

def factorsList(n):
    sum= 0
    for i in range(1,n//2+1):
        if n%i==0:
            sum=sum+i

    return sum
def isPerfect(n):
    if factorsList(n) ==n:
        return True
    return False
def generatePerfect(lb,ub):
    for x in range(lb,ub+1):
        if isPerfect(x):
            print(x,end =" ")
    print()
    return
generatePerfect(1,10)
generatePerfect(1,10000)
```

```
6
6 28 496 8128
```

In [19]:

```python
a,b,c = range(1,12,4)
print(a,b,c)
```

```
1 5 9
```

In [34]:

```python
# Strings
s1='Python'
print(s1[0])
print(s1[1])
print(s1[2])
print(s1[3])
print(s1[-2])
print(s1[-1])
print(s1[len(s1)-1])
```

```
P
y
t
h
o
n
n
```

In [33]:

```
s1='Python'
print(s1[-2])
print(s1[0:2]) #access the first characters
print(s1[:2]) #access the first characters
print(s1[-3:]) #access the last three characters
print(s1[2:]) #from second character to last character
```

```
o
Py
Py
hon
thon
```

In [40]:

```
s1='Python'
print(s1[1:-1])  #printing all characters except first and last
print(s1[len(s1)//2]) #middle of the string
print(s1[-1::-1]) #Reverse of the string
print(s1[-1:-3:-1]) #access the last two characters in reverse order
#Access the alternate characters
print(s1[::2]) #two characters
print(s1[::3]) #three characters
```

```
ytho
h
nohtyP
no
Pto
Ph
```

In [38]:

```
def reverseString(str):
    return str[-1::-1]
reverseString("Programming")
```

Out[38]:

```
'gnimmargorP'
```

In [41]:

```
def isPalindrome(str):
    if str ==str[::-1]:
        return True
    else:
        return False
    return
print(isPalindrome("Python"))
print(isPalindrome("ganag"))
```

```
False
True
```

In [2]:

```
# Function to print the Upper case and Lower case alphabets
#ASCII
#A-Z : 65-90
#a-z : 97-122
#0-9 : 48 -57
#space : 32

def printUpper(x):
    for i in range(len(x)):
        if ord(x[i]) >=65 and ord(x[i])<= 90:
            print(x[i],end=" ")

    return
printUpper("PyThon")
```

```
P T
```

In [3]:

```
def printLower(x):
    for i in range(len(x)):
        if ord(x[i]) >=97 and ord(x[i])<= 122:
            print(x[i],end=" ")

    return
printLower("PyThon")
```

```
y h o n
```

In [5]:

```
ord('A') # It give the equivalent ASCII number for i/p character
```

Out[5]:

```
65
```

In [9]:

```
#Function to print "Python" if the count of
#Upper and Lower case is same
#Print "Programming" if not same
#ex: PyThOn ---3 P T O (upper case)
#              3 y h n (lower case)
#PytHon --P H (2)
#       --y t o n(4) -Programming


def findCount(str):
    cntUpper =0
    cntLower =0
    for x in range(len(str)):
        if ord(str[x]) >=65 and ord(str[x])<=90:
            cntUpper=cntUpper+1
        elif ord(str[x]) >=97 and ord(str[x])<=122:
            cntLower=cntLower+1
    if cntUpper == cntLower:
        return "SameCount"
    else:
        return "Programming"
    return
print(findCount('PyThOn'))  #SameCount
print(findCount('PYTHon'))  #Programming
```

```
SameCount
Programming
```

In [8]:

```
x=10
s=str(x)
type(s)
```

Out[8]:

```
str
```

In [11]:

```
#example:
#i/p: Appli1cat8ion89
#o/p: 1 8 8 9
def extractDigits(str):
    return
extractDigits("Appli1cat8ion89")
```

In [13]:

```
#Function to return the sum of digits in a given string
#ex:
#i/p: Appli1cat8ion89
#o/p: 26(1+8+8+9)
#0 1 2 3 4 5----9
#48 49 50 51 ----57
def sumofDigits(str):
    sum=0
    for x in range(len(str)):
        if ord(str[x]) >=48 and ord(str[x])<=57:
            sum=sum+ (ord(str[x])-48)
    return sum
sumofDigits('Appli1cat8ion89')
```

Out[13]:

```
26
```

In [16]:

```
def sumofEvenDigits(str):
    sum=0
    for x in range(len(str)):
        if ord(str[x]) >=48 and ord(str[x])<=57:
            if (ord(str[x])-48) % 2==0:
                  sum= sum+ (ord(str[x])-48)
    return sum
sumofEvenDigits('Appli1cat8ion89')
```

Out[16]:

```
16
```

In [ ]: