Day objectives

- String Slicing
- Function In Python
- Basic Problems related to conditional statements using functions
- Iteration In Python
- Python Data structures

In [ ]:
```
1
```

String Slicing

In [18]:
```
1  s1="Python"
2
3  s1[0] #Accessing the first character in a string
4  s1[1] #Accessing the second character in a string
5  s1[len(s1)-1] #Accessing the last character in a string
6
7  s1[-1] #Another way of accessing the last character in a string
8
9  s1[-2] #Another way of accesing the penultimate character of the string
10
11 s1[0:2] #Accessing the first two characters in a string
12
13 s1[-2:] #Accessing the last two characters in a string
14
15 s1[4:] #only it is apllicable when string length is 6 #Accessing all charact
16
17 s1[1:-1] #Accessing all characters except first and last character
18
19 #Accessing the middle character
20 s1[len(s1)//2]
21
22 s1[-1::-1] # Reverse of a string
23
24 s1[-1:-3:-1] #Accessing the last two characters in reverse order
25
26 s1[len(s1)//2:(len(s1)//2)-2:-1]#Reverse the middle two characters in an eve
27
28 #Accessing alternate characters in a string
29 #"Python" -> "pto"
30
31 s1[::2]
32
33 #Accessing alternate characters of a string in reverse order
34 #"Python" -> "nhy"
35 s1[::-2]
```

Out[18]:
```
'nhy'
```

In [ ]:
```
1
```

Functions

In [19]:
```
1 # Function to reverse a string
2 def reverseString(s):
3     return s[::-1]
4
5 reverseString("Python")
6
```

Out[19]: 'nohtyP'

In [31]:
```
1 # Function check if a string is a palindrome
2
3 def palindrome(s):
4     if s==s[::-1]:
5         return True
6     else:
7         return False
8 palindrome("abc")
```

Out[31]: False

In [34]:
```
1 # Function check if a given year is a leap year
2 year=int(input("Enter a year"))
3 def leapyear(year):
4     if year%400==0 or (year%100!=0 and year%4==0):
5         return True
6     else:
7         return False
8
9 leapyear(year)
```

```
Enter a year2016
```

Out[34]: True

In [40]:
```
1 # Function to count number of digits in a given number
2 n=(input("Enter a number"))
3 def noofdigits(n):
4     return len(n)
5 noofdigits(n)
6
7
```

```
Enter a number132454657658
```

Out[40]: 12

In [43]:
```
# Function to identify the gretest of 4 numbers
def greatest(n1,n2,n3,n4):
    if n1>n2 and n1>n3 and n1>n4:
        return n1
    elif n2>n3 and n3>n4:
        return n2
    elif n3>n4:
        return n3
    else:
        return n4

greatest(15,25,53,65)
```

Out[43]: 65

Type *Markdown* and LaTeX: α^2

Iteration

- for
- while

For Loop in Python

[101,210+1]

for number in range(101,210+1)

In [52]:
```
# Function to print n natural numbers using for loop
def printNnaturalNumbers(n):
    for counter in range(1,n+1):
        print(counter,end=" ")
    print()
    return
printNnaturalNumbers(30)
printNnaturalNumbers(20)
```

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 3
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
```

In [55]:
```
# Function to print n natural numbers using a while loop

def nNaturalNumbers(n):
    counter=1
    while counter <=n:
        print(counter,end=" ")
        counter=counter+1
    return

nNaturalNumbers(9)


```

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

In [19]:
```
# Function to print the alternate values in a given number

#[500,550] -> 500 502 504 ......550
#(500,550) -> 501 503 505 507 .....549
#range(500,550) -> 500 501 502 .....549
#For all set based functions in python have start value as inclusive and end

def alternate(lb,ub):
    for i in range(lb,ub+1,2):
        print(i,end=" ")
    return
alternate(500,525)
```

```
500 502 504 506 508 510 512 514 516 518 520 522 524
```

In [29]:
```
# Function to print reverse of a given range in the

def reverse(lb,ub):
    for i in range(ub,lb-1,-1):
        print(i,end=" ")
    return
reverse(1,10)
```

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

In [30]:
```
# Function to print the odd numbers in reverse in a range

def oddNumbers(lb,ub):
    for i in range(ub,lb-1,-1):
        if i%2==1:
            print(i,end=" ")
    return
oddNumbers(1,10)
```

```
9 7 5 3 1
```

In [34]:
```
# Function to calculate the sum of numbers in a range

def sum(lb,ub):
    s=0
    for i in range(lb,ub+1):
        s=s+i
    return s
sum(1,10)

```

Out[34]: 55

In [4]:
```
#Function to print all numbers divisible by 6
#and not a factor of 100 in a given (lb,ub) inclusive

def divisible(lb,ub):
    for i in range(lb,ub+1):
        if i%6==0 and 100%i!=0:
            print(i,end=" ")
divisible(1,20)
```

```
6 12 18
```

In [7]:
```
## Function to generate the list of factors for a given numbers
#12-> 1 2 3 4 6 12

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

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

In [9]:
```
#Function to calculate the averages of cubes of all even numbers in a given

def average(lb,ub):
    s=0
    count=0
    for i in range(lb,ub+1):
        if i%2==0:
            s=s+i**3
            count=count+1
    return s//count
average(1,5)
```

Out[9]: 36

In [21]:
```
#Function to calculate the factorial of a given number

def factorial(n):
    m=1
    for i in range(1,n+1):
        m=m*i
    return m
factorial(5)
```

Out[21]: 120

In [14]:
```
# Check if given number is a prime or not

def isprime(n):
    count=0
    for i in range(1,n+1):
        if n%i==0:
            count=count+1
    if count==2:
        return True
    else:
        return False
isprime(8)
```

Out[14]: False

In [17]:
```
#Function to calculate the average of first N Prime numbers

def avgofprime(n):
    sum=0
    count=0
    for i in range(1,n+1):
        x=isprime(i)
        if x==1:
            sum+=i
            count+=1
    avg=sum/count
    return avg
avgofprime(10)
```

Out[17]: 4.25

In [5]:
```
#Function to generate all perfect numbers in a given range
def isperfect(i):
    s=0
    for j in range(1,i):
        if i%j==0:
            s=s+j
    if s==i:
        return True
    else:
        return False

def allperfect(start,end):
    for i in range(start,end+1):
        if isperfect(i):
            print(i,end=" ")
    return

allperfect(1,30)
```

```
6 28
```

Advanced probelm set

In [25]:
```
#Function to calculate average of all factorials in a given range

def avgoffactorials(n):
    sum=0
    count=0
    m=1
    for i in range(1,n+1):
        m=m*i
        print(m,end=" ")
        sum+=m
        count+=1
    return sum/count
avgoffactorials(5)
```

```
1 2 6 24 120
```

Out[25]:
```
30.6
```

In [4]:
```
#Function to generate N odd armstrong numbers
def isarmstrong(n):
    n1=n
    sum=0
    l=len(str(n))
    while(n>0):
        r=n%10
        sum=sum+r**l
        n=n//10
    if sum==n1:
        return True
    else:
        return False




def Noddarmstrong(n):
    for i in range(1,n+1):
        if isarmstrong(i):
            print(i,end=" ")
    return
Noddarmstrong(200)

```

```
1 2 3 4 5 6 7 8 9 153
```

In [5]:
```
### Function to generate Multiplication table for a number in a given range
### 10 in the range(100, 102) inclusive
### 10 x 100 = 1000
### 10 x 101 = 1010
### 10 x 102 = 1020

def multilplication(n,lb,ub):
    for i in range(lb,ub+1):
        print(n, "x", i, "=",n*i)
    return

multilplication(10,100,110)


```

```
10 x 100 = 1000
10 x 101 = 1010
10 x 102 = 1020
10 x 103 = 1030
10 x 104 = 1040
10 x 105 = 1050
10 x 106 = 1060
10 x 107 = 1070
10 x 108 = 1080
10 x 109 = 1090
10 x 110 = 1100
```

In [43]:
```
# Function to print average of all numbers in a given range
#(1,5)-> 3

def average(lb,ub):
    sum=0
    for i in range(lb,ub+1):
        sum=sum+i
        #count=count+1
    return sum/(ub-lb+1)
average(1,5)
```

Out[43]: 3.0

In [50]:
```
# Function to generate all leap years in a given time period
#2000-2020 -> 2000 2004 2008 2012 2016

def isLeapYear(i):
    if (i%400==0 or i%100!=0 and i%4==0):
        return True
    else:
        return False

def leapyear(start,end):
    for i in range(start,end+1):
        if isLeapYear(i):
            print(i,end=" ")
    return
leapyear(2000,2020)

```

```
2000 2004 2008 2012 2016 2020
```

In [4]:
```
# Calculate number of days in a given timeperiod using leapyear
#for every year in the given time period, if the year is leap year

def isLeapYear(i):
    if (i%400==0 or i%100!=0 and i%4==0):
        return True
    else:
        return False

def numberofDays(start,end):
    sum=0
    for i in range(start,end+1):
        if isLeapYear(i):
            sum=sum+366
        else:
            sum+=365
    return sum
numberofDays(2000,2020)

```

Out[4]: 7671

In [42]:

```
## Function to calculate the number of hours for a given period
## numberOfHours (11,1975, 3,1999)
## numberOfHours(5,2019, 6,2019)

def isLeapYear(i):
    if (i%400==0 or i%100!=0 and i%4==0):
        return True
    else:
        return False

def numberofDays(start,end):
    sum=0
    for i in range(start,end+1):
        if isLeapYear(i):
            sum=sum+366
        else:
            sum+=365
    return sum



def numberofDaysMonth(month,year):
    if month==2:
        if isLeapYear(year):
            return 29
        else:
            return 28
    elif(month <=7 and month % 2 != 0) or (month >= 8 and month % 2 == 0):
        return 31
    else:
        return 30

# numberofDaysMonth(2,2019)

def daysInStartYear(startmonth,startyear):
    days=0
    for month in range(startmonth,13):
        days+=numberofDaysMonth(month,startyear)
    return days

# daysInStartYear(2,2019)

def daysInEndYear(endmonth,endyear):
    days=0
    for month in range(1,endmonth+1):
        days+=numberofDaysMonth(month,endyear)
    return days

# daysInEndYear(6,2019)


# daysInMiddleYear(2000,2020)


def numberOfHours(startmonth,startyear,endmonth,endyear):
    days=0
```

```
    if startyear != endyear:
        days+=daysInStartYear(startmonth,startyear)
        days+=daysInEndYear(endmonth,endyear)
    if endyear-startyear==2:
        days+=numberofDays(startyear+1,startyear+1)
    elif endyear-startyear>2:
        days+= numberofDays(startyear+1,endyear-1)
    else:
        for month in range(startmonth,endmonth+1):
            days+= numberofDaysMonth(month,startyear)
    return 24*days

numberOfHours(5,2019,6,2019)

```

Out[42]: 1464

In [ ]:

```

```