Problem Solving and programming

Data 12 june 2019

###Day Objectives

```
- String Slicing
- Functions in Python
- Basic Problems related to conditional statements using functions
- Iteration in Python
- Problem set for practice
```

```
In [ ]:
```

string silicing

In [58]:
```
s1 = "python"

s1[0]      # Accessing the first character in a string

s1[1]       # Accessing the second character in a string

s1[-1]       # Accessing the last character in string

s1[len(s1)-1] # Accessing the last character in the another way

s1[-2]         # Accessing the penultimate character of the string

s1[-2:len(s1)]  # Accessing the  two or more character of the string

#Accessing all character first and last charater
s1[1:-1]
# Accessing the middle charater in a string
s1[len(s1)//2]

# Reverse of the string
s1[ -1: :-1 ]

#Access last two charater in reverse of string

s1[-1:-4:-1]

# How do you reverse the middle two characters in an even length string

s1[len(s1)//2:len(s1)//2-2:-1]

#Accessing alternate charaters in a string
# "python" -> "pto"
s1[::2]


# Accessing alternate charaters of a string in reverse of a string
#"python" ->  "nhy"
s1[::-2]
```

Out[58]:
```
'nhy'
```

Functions

In [60]:
```
# Function to reverse a string
def reverseString(s):
    return s[::-1]


reverseString("Python")
```

Out[60]:
```
'nohtyP'
```

In [67]:
```
#Function to check if a string is a palindrome
def palindrome(s):
    if s == s[::-1]:
        return True
    else:
        return False
palindrome("123321")
```

Out[67]:
```
True
```

In [92]:
```
# Function to check if a year is leap year or not
def leapyear(year):
    if year%400 == 0 or (year%4 == 0 and year%100!= 0):
        return True
    else:
        return False
leapyear(2019)
```

Out[92]:
```
False
```

In [84]:
```
# Function to count the number of digits in a given number
def countdigits(s):
    return len(str(s))
countdigits(123123)
```

Out[84]:
```
6
```

In [88]:
```
# Function to identify the greatest of 4 numbers
def greatest(s,r,t,u):
    if s > r and s > t and s > u:
        return s
    elif r > t and r > u :
        return r
    elif t > u :
         return t
    else:
        return u
greatest(123,243,345,346)
```

Out[88]:
```
346
```

In [ ]:
```
```

Iteration

- for
- while

###For Loop In Python

[101,210]

for number in range(101,211) print number o/p:-

101 102 . . . 210

In [91]:
```
# Functions to print n numbers
def printNaturalNumbers(n):
    for counter in range(1, n+1):
        print (counter, end=" ")
    return
printNaturalNumbers(50)
```

```
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 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
```

In [15]:
```
# Functions to print odd numbers
def printoddnumbers(n):
    for i  in range(1, n+1):
        if i%2==0:
             print ("even number", i)
        else:
            print ("odd number", i)
    return


printoddnumbers(30)
```

```
odd number 1
even number 2
odd number 3
even number 4
odd number 5
even number 6
odd number 7
even number 8
odd number 9
even number 10
odd number 11
even number 12
odd number 13
even number 14
odd number 15
even number 16
odd number 17
even number 18
odd number 19
even number 20
odd number 21
even number 22
odd number 23
even number 24
odd number 25
even number 26
odd number 27
even number 28
odd number 29
even number 30
```

In [1]:
```
# Function to print N naturals using a while loop

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


nNaturalnumbers(30)
```

```
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
```

In [11]:
```
#Function to print all numbers divisible by 6 and not a fator of 100 in given ran
def Numbersdivisibleby6(lb,ub):

    for i in range(lb,ub):
        if i%6 == 0 and i%100!=0:
            print (i, end =" ")
    return

Numbersdivisibleby6(10,1000)
```

```
12 18 24 30 36 42 48 54 60 66 72 78 84 90 96 102 108 114 120 126 132 138 144 15
0 156 162 168 174 180 186 192 198 204 210 216 222 228 234 240 246 252 258 264 2
70 276 282 288 294 306 312 318 324 330 336 342 348 354 360 366 372 378 384 390
396 402 408 414 420 426 432 438 444 450 456 462 468 474 480 486 492 498 504 510
516 522 528 534 540 546 552 558 564 570 576 582 588 594 606 612 618 624 630 636
642 648 654 660 666 672 678 684 690 696 702 708 714 720 726 732 738 744 750 756
762 768 774 780 786 792 798 804 810 816 822 828 834 840 846 852 858 864 870 876
882 888 894 906 912 918 924 930 936 942 948 954 960 966 972 978 984 990 996
```

In [10]:
```
# Function to find the average of  cubes of all even numbers in a given range(lb,
def Average(lb,ub):
    count=0
    tem=0
    for i in range(lb,ub):
        if i%2==0:
            tem = tem + i ** 3
            count = count+1
    print(tem/count, end=" ")
    return
Average(20,30)
```

```
14400.0
```

In [14]:
```
#Function to calculate the factorial of a given number
def factorial(n):
    fact=1
    for i in range(1, n+1):
        fact=fact*i
    print (fact, end=" ")
    return
factorial(12)
```

```
479001600
```

In [6]:
```
#Function to check given number is prime
def prime(n):
    for i in range(2,n):
         if n%i==0:
            return False
    return True

prime(5)
```

Out[6]:
```
True
```

In [47]:
```
# Function calculate the average of first N prime numbers

def avgNprime(i):
    count = 0
    sum = 0
    avg = 0
    for n in range(1,i+1):
        if prime(n):
            sum = sum + n
            count = count+1
            avg = sum /count
    return avg

avgNprime(10)
```

Out[47]:
```
3.6
```

In [5]:
```
# Function generate all perfect numbers in a given range
def perfectnum(n):
    sum = 0
    for i in range(1,n):
        if n%i==0:
            sum = sum + i
    if sum==n:
        return True
    return False



def generateperfect(lb,ub):
    for n in range(lb,ub+1):
        if perfectnum(n):
            print (n,end=" ")
    return

generateperfect(2,50)
```

```
6 28
```

In [78]:
```
# Function generate factors of given numbers
#12- 2,4,6,12
def factors(n):
    for i in range(1,n+1):
        if n%i == 0:
            print(i, end =" ")
    return

factors(120)
```

```
1 2 3 4 5 6 8 10 12 15 20 24 30 40 60 120
```

In [4]:
```
# Function to print the reverse range in the same line
# [500,550]   -> 500 502 504.........550
#(500,550)    -> 501 503 505.........549
#range(500,550) ->500 502 504.........548
def alternativenumbers(lb,ub):
    for i in range(lb,ub+1, 2):
        print (i, end=" ")
    return

alternativenumbers(500,534)
```

```
500 502 504 506 508 510 512 514 516 518 520 522 524 526 528 530 532 534
```

In [6]:
```
#Function to print reverse of given range in the same line
def reverse(lb,ub):
    for i in range(ub,lb-1,-1):
        print (i,end=" ")
    return

reverse(500,525)
```

```
525 524 523 522 521 520 519 518 517 516 515 514 513 512 511 510 509 508 507 506
505 504 503 502 501 500
```

In [7]:
```
# function to print odd numbers in reverse order
def reverseoddnumbers(lb,ub):
    for i in range (ub,lb-1,-1):
        if i%2!=0:
            print (i, end=" ")
    return
reverseoddnumbers(500,525)
```

```
525 523 521 519 517 515 513 511 509 507 505 503 501
```

In [54]:
```
#Function to calculte the sum of numbers in a range
def sumofthenumbers(lb,ub):
        sum1 = 0
        for i in range(lb,ub+1):
            sum1 = sum1 + i
        return sum1




sumofthenumbers(100,200)
```

Out[54]:
```
15150
```

In [60]:
```
#Function  to calculate average of a given range
def Averageofnumbers(lb,ub):
    sum1 = 0
    count = 0
    avg = 0
    for i in range(lb,ub+1):
        sum1 = sum1+i
        count = count+1
        avg = sum1/count
    return avg

Averageofnumbers(12,45)
```

Out[60]:
```
28.5
```

In [10]:
```
# FUnction to generate the leap years in given time period
def Leapyear(year):
        if year%400 == 0 or (year%4 == 0 and year%100!=0):
            return True
        else:
            return  False




def genreateleapyear(ly,uy):
    for year in range(ly,uy+1):
        if Leapyear(year):
            print (year,end=" ")
    return

genreateleapyear(2000,2019)
```

```
2000 2004 2008 2012 2016
```

In [6]:
```
def generateleapyear(ly,uy):
    for i in range(ly,uy+1):
        if i%400 == 0 or(i%4==0 and i%100!=0):
            print (i,end=" ")
    return
generateleapyear(200,2020)
```

```
204 208 212 216 220 224 228 232 236 240 244 248 252 256 260 264 268 272 276 280
284 288 292 296 304 308 312 316 320 324 328 332 336 340 344 348 352 356 360 364
368 372 376 380 384 388 392 396 400 404 408 412 416 420 424 428 432 436 440 444
448 452 456 460 464 468 472 476 480 484 488 492 496 504 508 512 516 520 524 528
532 536 540 544 548 552 556 560 564 568 572 576 580 584 588 592 596 604 608 612
616 620 624 628 632 636 640 644 648 652 656 660 664 668 672 676 680 684 688 692
696 704 708 712 716 720 724 728 732 736 740 744 748 752 756 760 764 768 772 776
780 784 788 792 796 800 804 808 812 816 820 824 828 832 836 840 844 848 852 856
860 864 868 872 876 880 884 888 892 896 904 908 912 916 920 924 928 932 936 940
944 948 952 956 960 964 968 972 976 980 984 988 992 996 1004 1008 1012 1016 102
0 1024 1028 1032 1036 1040 1044 1048 1052 1056 1060 1064 1068 1072 1076 1080 10
84 1088 1092 1096 1104 1108 1112 1116 1120 1124 1128 1132 1136 1140 1144 1148 1
152 1156 1160 1164 1168 1172 1176 1180 1184 1188 1192 1196 1200 1204 1208 1212
1216 1220 1224 1228 1232 1236 1240 1244 1248 1252 1256 1260 1264 1268 1272 1276
1280 1284 1288 1292 1296 1304 1308 1312 1316 1320 1324 1328 1332 1336 1340 1344
1348 1352 1356 1360 1364 1368 1372 1376 1380 1384 1388 1392 1396 1404 1408 1412
1416 1420 1424 1428 1432 1436 1440 1444 1448 1452 1456 1460 1464 1468 1472 1476
1480 1484 1488 1492 1496 1504 1508 1512 1516 1520 1524 1528 1532 1536 1540 1544
1548 1552 1556 1560 1564 1568 1572 1576 1580 1584 1588 1592 1596 1600 1604 1608
1612 1616 1620 1624 1628 1632 1636 1640 1644 1648 1652 1656 1660 1664 1668 1672
1676 1680 1684 1688 1692 1696 1704 1708 1712 1716 1720 1724 1728 1732 1736 1740
1744 1748 1752 1756 1760 1764 1768 1772 1776 1780 1784 1788 1792 1796 1804 1808
1812 1816 1820 1824 1828 1832 1836 1840 1844 1848 1852 1856 1860 1864 1868 1872
1876 1880 1884 1888 1892 1896 1904 1908 1912 1916 1920 1924 1928 1932 1936 1940
1944 1948 1952 1956 1960 1964 1968 1972 1976 1980 1984 1988 1992 1996 2000 2004
2008 2012 2016 2020
```

In [38]:
```
#Calculate number of days in a given time days
def numberofdays(ly,uy):
    days=0
    for year in range(ly,uy+1):
        if  Leapyear(year):
             days= days+366
        else:
            days= days+365
    return days

numberofdays(2000,2019)
```

Out[38]:
```
7305
```

In [1]:
```
# Function to calculate number of hours for a given period
#numberof Hours(11,1975,3,1999) ->204504 or 205248
#numberof hours(5,2019,6,2019)  ->1464
#calculate days from jan to end month year
#Excluding Feb
# First seven  months   -1,3,4,5,6,7
                           # All odd num have 31 days
                            # All even num have 30 days
#next months - 8,9,10,11,12
                          # All even num have 31 days
                           # All odd num have 30 days
# 31 days -(month<=7 and month !=2)
#         return 31
#     else
#        return 30

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

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

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

def numberofhours(startmonth,startyear,endmonth,endyear):
    days=0
    days += daysInstartyear(startmonth,startyear)
    days += daysInendyear(endmonth,endyear)
    if endyear-startyear == 1:
        days+=numberofdays(startyear+1,startyear+2)
    elif endyear-startyear > 2:
        days+=numberofdays(startyear+1,endyear-1)
    return 24*days

numberofhours(11,1975,3,1999)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-1-4274da7923db> in <module>
     47     return 24*days
     48
---> 49 numberofhours(11,1975,3,1999)
```

```
<ipython-input-1-4274da7923db> in numberofhours(startmonth, startyear, endmont
h, endyear)
     39 def numberofhours(startmonth,startyear,endmonth,endyear):
     40     days=0
---> 41     days += daysInstartyear(startmonth,startyear)
     42     days += daysInendyear(endmonth,endyear)
     43     if endyear-startyear == 1:

<ipython-input-1-4274da7923db> in daysInstartyear(startmonth, startyear)
     28     days=0
     29     for month in range(startmonth,13):
---> 30         days += numberdaysinmonth(month,startyear)
     31     return days
     32 

TypeError: numberdaysinmonth() takes 1 positional argument but 2 were given
```

In [105]:

```python
#Function to generate N odd armstrong numbers
def armstrong(n):
    for i in range(1,n+1):
```

```
  File "<ipython-input-105-06d585252549>", line 2
    def
        ^
SyntaxError: invalid syntax
```

In [ ]: