problem Solving and Programming

Date 12 June 2019

Day Objectives

Day Objective:

String Slicing

Function in Python

Basic Problems related to conditional statements using functions

Python data Structures - Lists , Tuples and Dictionaries

Basics operations on data structures

Apply Data structures to solve problems

In [ ]:

String Slicing

In [13]:
```
s1="python"
s1[-1] or s1[len(s1)-1]
s1[-3]
```

Out[13]: 'h'

In [ ]:
```
s1[1]
len(s1)
s1[-1]#prints last one
s[-1:-10:-1]#it prints from last one to -10 with decrementing order
```

In [15]:
```
s2=len(s1)
s1[1:s2-1]
s1[1:-1]
```

Out[15]: 'ytho'

In [33]:
```
# Accessing the middle character in an odd length
s="python1"
s3="python"
s[len(s1)//2]#middle number for odd
a=s3[int(len(s3)/2)]
b=s3[(int(len(s3)/2)-1)]
print(b,a,end="")#middle number for even
```

```
t h
```

In [37]:
```
#reverse the string
s="python"
s1=s[-1::-1]
print(s1)
s[-1:-3:-1]
```

```
nohtyp
```

Out[37]:
```
'no'
```

In [47]:
```
#print middle number for a even string in reverse order
s3="python"
a=s3[int(len(s3)/2)]
b=s3[(int(len(s3)/2)-1)]
print(a,b,end="")
```

```
h t
```

In [40]:
```
#accessing alternative characters of a string in reverse order
s3[-1::-2]
```

Out[40]:
```
'nhy'
```

In [41]:
```
#accessing alternative characters of a string in reverse order
s3[0::2]
```

Out[41]:
```
'pto'
```

In [ ]:
```

```

Functions

In [49]:
```
#Function to reverse a string
def reversestring(s1):
    s2=s1[-1::-1]
    print(s2)
reversestring("alekhya")
```

```
ayhkela
```

In [51]:
```
def reversestring(s1):
    return s1[-1::-1]
reversestring("alekhya")
```

Out[51]: 'ayhkela'

In [55]:
```
#Function Palindrome
def palindrome(s1):
    if s1==s1[-1::-1]:
        return True
    else:
        return("false")
palindrome("aa")
```

Out[55]: True

In [64]:
```
#check the function is a leap year or not
def leapyear(year):
    if year%400==0 or (year%100!=0 and year%4==0):
        return True
    return False
leapyear(2019)
```

Out[64]: False

In [71]:
```
#Functions to count the number of digits in given number

def count(a):
    c=0
    while a>0:
        r=a%10
        c=c+1
        a=int(a/10)

    return(c)
count(123)
```

Out[71]: 3

In [74]:
```
def count(a):
    return len(str(a))
count(1234)
```

Out[74]: 4

In [80]:
```
#Function to identify the gretest of 4 numbers
def greatestnumber(a,b,c,d):
    if a>b and a>c and a>d:
        return("a")
    elif b>c  and b>d:
        return("b")
    elif c>d:
        return("c")
    else:
        return("d")
greatestnumber(7,4,4,7)
```

Out[80]: 'd'

Iterations

- for
- while

In [86]:
```
def naturalnumber(n):
    for i in range(1,n+1):
        print(i,end=" ")
naturalnumber(90)
```

```
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
30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55
56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81
82 83 84 85 86 87 88 89 90
```

In [100]:
```
#Function to print all numbers divisible by 6 and not factor of 100 in a given
range(lb,ub) iclusive
def factors(lb,ub):
    for i in range(lb,ub+1):
        print("i value",i)
        if 6%i==0 and 100%i!=0:
            return(i)

factors(1,100)
```

```
i value 1
i value 2
i value 3
```

Out[100]: 3

In [119]:
```
# Function to find the average of cubes of all even numbers ina given range(l
b.ub) iclusive
def even(a,b):
    c=0
    for i in range(a,b+1):
        if i%2==0:


            c=i*i*i+c
    print(c)
even(1,100)
```

```
13005000
```

In [140]:
```
#functions to find list of factors for a given number
def sumoffactors(a):
    sum=0
    for i in range(1,a+1):
        if a%i==0:
            sum += i
            print(sum)
factors(12)
```

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

In [141]:
```
#functions to find factorial of a number
def factorial(a):
    fact=1
    for i in range(1,a+1):
        fact=fact*i
    print(fact)
factorial(5)
```

```
120
```

In [13]:
```
#functions to calculate the average of first N prime numbers

def averagefirstprime(n):
    count1=0
    count2=0
    average=0
    for i in range(1,100):
        for a in range(1,i+1):
            if i%a==0:
                count1=count1+1
        if count1==2:
            count2=count2+1
            add=add+i
            average=add/count2
    if count2==4:
        a=average
    print(a)
averagefirstprime(4)
```

```
99
```

In [ ]:
```
#Function to generate all perfect numbers in a given range
```

In [4]:
```
#print N natural Numbers in Alternative
def alternative(n):
    for i in range(1,n+1,2):
        print(i,end=" ")
    return
alternative(100)
```

```
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 5
5 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99
```

In [5]:
```
#function to print reverse of given range in the same line
def reversenumberrange(high,low):
    for i in range(high,low-1,-1):
        print(i,end=" ")
reversenumberrange(20,10)
```

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

In [12]:
```
#function to print odd numbers in   given range in the same line
def reversenumberrange(high,low):
    for i in range(high,low-1,-1):
        if (i%2)!=0:
            print(i,end=" ")
reversenumberrange(10,1)
```

```
9 7 5 3 1
```

In [16]:
```
#Function to calculate the sum of numbers in a range
def sumrange(start,end):
    sum=0
    for i in  range(start,end+1):
        sum=sum+i
    return sum
sumrange(1,19)
```

Out[16]: False

In [28]:
```
#Function to calculate average in a given range

def average(start,end):
    sum=0
    avg=0
    #count=0
    for i in range(start,end+1):
        sum=sum+i
        #count=count+1
        avg=sum/(end+1-start)
    print(avg)
average(100,200)
```

```
150.0
```

In [14]:
```
#functions to check given number is a prime or not
def isprime(a):
    seqcount=0
    for i in range(1,a+1):
        if a%i==0:
            seqcount=seqcount+1
    if seqcount==2:
        return True
    return False


isprime(5)
```

Out[14]: True

In [34]:
```
def avgNprimes(lb,ub):
    n=int(input("enter how many prime numbers do you want"))
    primecount=0
    sum=0
    for i in range(lb,ub+1):
        if isprime(i):
            sum=sum+i
            print(i,"i")
            print(sum)
            primecount=primecount+1
    print(primecount,"every loop")
    if primecount==n:
        print(sum/primecount)




avgNprimes(1,10)
```

```
enter how many prime numbers do you want4
2 i
2
3 i
5
5 i
10
7 i
17
4 every loop
4.25
```

In [ ]:

In [ ]:

In [ ]:
```
#function to generate Multiple table in a given range
def multiplication(n):
    r=int(input("enter the range"))
    u=int(input("enter upto"))
    for i in range(r,u):
        print(n,'X',r,'=',u)
multiplication(10)
```

In [33]:
```
# Function to generate all leap years in a given time period
#2000-2020
def leapyear(lb,ub):
    for year in range(lb,ub+1):
        if year%400==0 or (year%100!=0 and year%4==0):
            print(year,end=" ")


leapyear(2000,2020)
```

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

In [10]:
```
def leapyear(year):
    if year%400==0 or (year%100!=0 and year%4==0):
        return True
    return False
```

In [52]:
```
def rangeyears(lb,ub):
    for year in range(lb,ub+1):
        if leapyear(year):
            print(year,end=" ")

rangeyears(2000,2018)
```

```
2000 2004 2008 2012 2016
```

In [6]:
```
#calculate the number of days in agiven time period using functions
def numberofdays(lb,ub):
    lp=366
    nlp=365
    add=0
    add1=0
    for year in range(lb,ub+1):
        if leapyear(year):
            add=add+lp
        else:
            add1=add1+nlp
    sum=add+add1
    return sum
```

In [16]:
```
#Function to calculate number of hours for a given time period
#(11,1975,3,1999)
def numberofhours(month1,year1,month2,year2):
    lp=366
    nlp=365
    add=0
    add1=0
    for year in range(year1+1,year2):
        if leapyear(year):
            add=add+lp
        else:
            add1=add1+nlp
    sum1=add+add1
    if leapyear(year1):
        s=0
        p=0
        q=0
        for month in range(month1,12+1):
            if month==1 or month==3 or month==5 or month==7 or month==9 or mon
th==11 or month==8:
                s=s+31
            elif month==4 or month==6 or month==10 or month==12:
                p=p+30
            elif month==2:
                q=q+29
        loop1=s+p+q
    else:
        s=0
        p=0
        q=0
        for month in range(month1,12+1):
            if month==1 or month==3 or month==5 or month==7 or month==9 or mon
th==11 or month==8:
                s=s+31
            elif month==4 or month==6 or month==10 or month==12:
                p=p+30
            elif month==2:
                q=q+28
        loop1=s+p+q

    if leapyear(year2):
        s1=0
        p1=0
        q1=0
        for month in range(1,month2+1):
            if month==1 or month==3 or month==5 or month==7 or month==9 or mon
th==11 or month==8:
                s1=s1+31
            elif month==4 or month==6 or month==10 or month==12:
                p1=p1+30
            elif month==2:
                q1=q1+29
        loop2=s1+p1+q1
    else:
        s1=0
        p1=0
```

```
        q1=0
        for month in range(1,month2+1):
            if month==1 or month==3 or month==5 or month==7 or month==9 or mon
th==11 or month==8:
                s1=s1+31
            elif month==4 or month==6 or month==10 or month==12:
                p1=p1+30
            elif month==2:
                q1=q1+29
        loop2=s1+p1+q1
    total_days=loop1+loop2+sum1
    total_hours=total_days*24
    print("Number of hours",total_hours)


numberofhours(2,2019,3,2020)
```

```
Number of hours 10224
```

In [31]:
```
#functions to find list of factors for a given number
def factors(a):

    for i in range(1,a+1):
        if a%i==0:

            print(i)
factors(12)
```

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

In [111]:
```
#functions to generate all perfect numbers in a given range
def perfectrange(lb,ub):
    sum=0
    for i in range(lb,ub+1):
        if prime(i):
            print(i)



perfectrange(1,10)
```

```
6
```

In [110]:
```
def prime(n):
    sum=0
    for j in range(1,n):
            if n%j==0:
                sum=sum+j


    if sum==n:

        #print("final Sum",sum)
        return True
```