

**FEDERAL INSTITUTE OF
SCIENCE AND TECHNOLOGY
(FISAT)TM**

HORMIS NAGAR, MOOKKANNOOR

ANGAMALY-683577



‘FOCUS ON EXCELLENCE’

20MCA131 PROGRAMMING LAB

.....
LABORATORY RECORD

Name: ANAMIKA C P

Branch: MASTER OF COMPUTER APPLICATIONS

Semester: 1 Batch: SEMESTER -1 A Roll No: 19

MARCH 2022

**FEDERAL INSTITUTE OF
SCIENCE AND TECHNOLOGY
(FISAT)TM**

HORMIS NAGAR, MOOKKANNOOR

ANGAMALY-683577



‘FOCUS ON EXCELLENCE’

Name : ANAMIKA C P

Branch : MASTER OF COMPUTER APPLICATIONS

Semester : 1 Roll No: 19

University Exam Reg. No: FIT21MCA-2019

CERTIFICATE

This is to certify that this is a Bonafide record of the Practical work done by **ANAMIKA C P** in the **20MCA131 PROGRAMMING** Laboratory towards the partial fulfillment for the award of the Master Of Computer Applications during the academic year 2021-2022.

Signature of Staff in Charge

Name:

Date:

Signature of H.O.D

Name:

Date of University practical examination

Signature of

Internal Examiner

Signature of

External Examiner

CONTENTS

SI No	Date of Experiment	Name of Experiment	Page No:	Signature of Staff –In – Charge:
1		Display future leap years from current leap year to a final year entered by user	7	
2		List comprehensions:(a) Generate positive list of numbers from a given list of integers(b) Square of N numbers(c) Form a list of vowels selected from a given word(d) List ordinal value of each element of a word	8-10	
3		Count the occurrences of each word in a line of text	11	
4		Prompt the user for a list of integers. For all values greater than 100, store 'over' instead	12	
5		Store a list of first names. Count the occurrences of 'a' within the list	13	
6		Enter 2 list of integers. Check (a) Whether list are of same length (b) whether list sums to same value (c) whether any value occur in both	14-15	
7		Get a string from an input string where all occurrences of first character replaced with '\$', except first character. [eg: onion->oni\$ni]	16	
8		Create a string from given string where first and last characters exchanged. [eg: python->nythop]	17	
9		Accept the radius from user and find area of circle	18	
10		Find biggest of 3 numbers entered	19	
11		Accept a file name from user and print extension of that	20	

12		Create a list of colors from comma-separated color names entered by user Display first and last colors	21	
13		Accept an integer n and compute $n+nn+nnn$	22	
14		Print out all colors from color-list1 not contained in color-list2	23	
15		Create a single string separated with space from two strings by swapping the character at position 1	24	
16		Sort dictionary in ascending and descending order	25	
17		Merge two dictionaries	26	
18		Find gcd of 2 numbers	27	
19		From a list of integers, create a list removing even numbers	28	
20		Program to find the factorial of a number	29	
21		Generate Fibonacci series of N terms	30	
22		Find the sum of all items in a list	31	
23		Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square	32-33	
24		Display the given pyramid with step number accepted from the user	34	
25		Count the number of characters (character frequency) in a string	35	
26		Add 'ing' at the end of a given string. If it already ends with 'ing', then add 'ly'	36	
27		Accept a list of words and return length of longest word	37	
28		Construct following pattern using nested loop	38-39	

29		Generate all factors of a number	40	
30		Create a package graphics with modules rectangle, circle and sub-package 3D-graphics with modules cuboid and sphere. Include methods to find area and perimeter of respective figures in each module. Write programs that finds area and perimeter of figures by different importing statements. (Include selective import of modules and import * statements)	41-43	
31		Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two Rectangle objects by their area	44-45	
32		Create a Bank account with members account number, name, type of account and balance. Write constructor and methods to deposit at the bank and withdraw an amount from the bank	46-47	
33		Create a class Rectangle with private attributes length and width. Overload '<' operator to compare the area of two rectangles	48-49	
34		Create a class Time with private attributes hour, minute and second. Overload '-' operator to find sum of two time	50	
35		Create a class Publisher(name). Derive a class Book from Publisher with attributes title and author. Derive class Python from Book with attributes price and number_of_pages. Write a program that displays information about a Python book. Use base class constructor invocation and method overriding	51-52	
36		Write a Python program to read a file line by line and store it into a list	53	
37		Write a Python program to read each row from a given csv file and print a list of strings	54	

CO1

1.

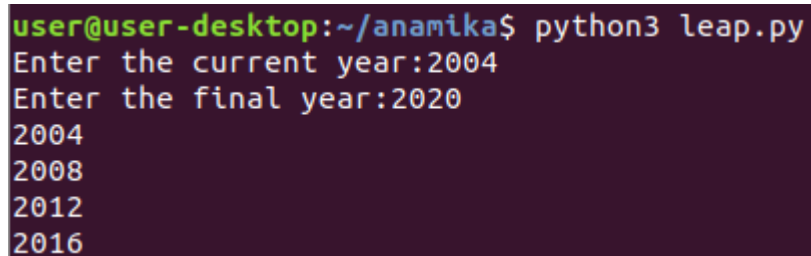
Aim:

Display future leap years from current leap year to a final year entered by user.

Input

```
current_year=int(input("Enter the current year:"))
final_year=int(input("Enter the final year:"))
for year in range(current_year,final_year):
    if(year%400==0)or(year%100!=0)and(year%4==0):
        print(year)
```

Output



```
user@user-desktop:~/anamika$ python3 leap.py
Enter the current year:2004
Enter the final year:2020
2004
2008
2012
2016
```

2. List comprehensions:

(a)

Aim:

Generate positive list of numbers from a given list of integers.

Input

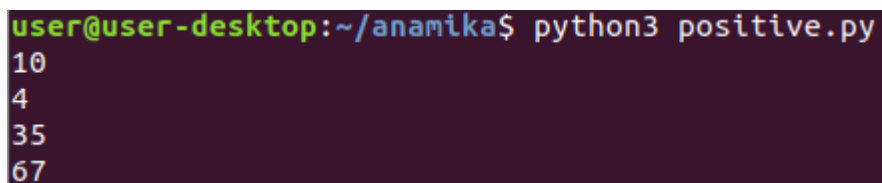
```
list=[10,-5,4,-8,35,67,-22]
```

```
for num in list:
```

```
if num>0:
```

```
print(num)
```

Output



```
user@user-desktop:~/anamika$ python3 positive.py
10
4
35
67
```

(b)

Aim:

Square of N numbers

Input

```
lst=[]
```

```
n=int(input("Enter a number:"))
```

```
for num in range(1,n+1):
```

```
num=num*num
```

```
lst.append(num)
```

```
print(lst)
```


Output

```
user@user-desktop:~/anamika$ python3 squarenumbers.py
Enter a number:5
[1, 4, 9, 16, 25]
```

(c)

Aim:

Form a list of vowels selected from a given word

Input

```
L=[]
s="India is my country"
for i in s:
    if i in ("aeiouAEIOU"):
        L.append(i)
print(L)
```

Output

```
user@user-desktop:~/anamika$ python3 vowels.py
['I', 'i', 'a', 'i', 'o', 'u']
```

(d)

Aim:

List ordinal value of each element of a word

Input

```
ordinal=input("Enter a word:")
print("The ASCII value of the letters in the word is")
for letter in ordinal:
    n=ord(letter)
    print(n)
```

Output

```
user@user-desktop:~/anamika$ python3 ordinal.py
Enter a word:anamika
The ASCII value of the letters in the word is
97
110
97
109
105
107
97
```

3.

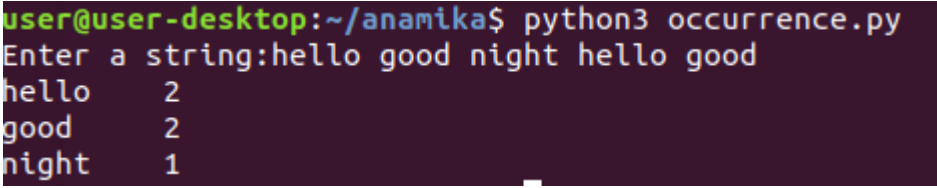
Aim:

Count the occurrences of each word in a line of text.

Input

```
list1=[]
list2=[]
x=input("Enter a string:")
for i in x.split(" "):
    list1.append(i)
    if i not in list2:
        list2.append(i)
for i in list2:
    print(i,"\t",list1.count(i))
```

Output



```
user@user-desktop:~/anamika$ python3 occurrence.py
Enter a string:hello good night hello good
hello      2
good       2
night      1
```

4.

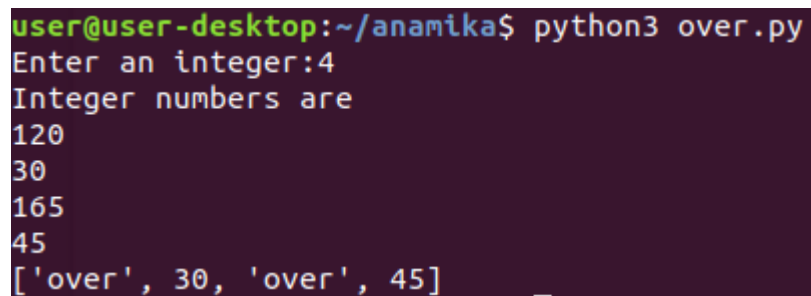
Aim:

Prompt the user for a list of integers. For all values greater than 100, store 'over' instead.

Input

```
lst=[]
n=int(input("Enter an integer:"))
print("Integer numbers are")
for i in range(0,n):
    j=int(input())
    if j>100:
        lst.append("over")
    else:
        lst.append(j)
print(lst)
```

Output



```
user@user-desktop:~/anamika$ python3 over.py
Enter an integer:4
Integer numbers are
120
30
165
45
['over', 30, 'over', 45]
```

5.

Aim:

Store a list of first names. Count the occurrences of 'a' within the list.

Input

```
list1=["anamika","anagha","athira"]
```

```
count=0
```

```
for word in list1:
```

```
for letter in word:
```

```
if letter=="a":
```

```
count=count+1
```

```
print("The occurrences of 'a' within the list is "+str(count))
```

Output

```
user@user-desktop:~/anamika$ python3 firstnames.py
The occurrences of 'a' within the list is 8
```

6.

Aim:

Enter 2 list of integers. Check (a) Whether list are of same length (b) whether list sums to same value (c) whether any value occur in both.

Input

```
l1=[2,4,6,8,10]
```

```
l2=[3,5,7,9,10]
```

```
print(l1)
```

```
print(l2)
```

```
if len(l1)==len(l2):
```

```
print("Lists are of same length")
```

```
else:
```

```
print("Lists are of different length")
```

```
s1=0
```

```
s2=0
```

```
for i in range(len(l1)):
```

```
s1=s1+l1[i]
```

```
print("Sum of first list is",s1)
```

```
for j in range(len(l2)):
```

```
s2=s2+l2[j]
```

```
print("Sum of second list is",s2)
```

```
if (s1==s2):
```

```
print("Sum of lists is same")
```

```
else:
```

```
print("Sum of lists are different")
```

```
for i in l1:
```

```
if i in l2:
```

```
print(i,"occurs in both list")
```

Output

```
user@user-desktop:~/anamika$ python3 same.py
[2, 4, 6, 8, 10]
[3, 5, 7, 9, 10]
Lists are of same length
Sum of first list is 30
Sum of second list is 34
Sum of lists are different
10 occurs in both list
```

7.

Aim:

Get a string from an input string where all occurrences of first character replaced with '\$', except first character. [eg: onion->oni\$n]

Input

```
str1=input("Enter a string:")
print("Original string:",str1)
char=str1[0]
str1=str1.replace(char,'$')
str1=char+str1[1:]
print("String:",str1)
```

Output

```
user@user-desktop:~/anamika$ python3 string.py
Enter a string:onion
Original string: onion
String: oni$n
```


8.

Aim:

Create a string from given string where first and last characters exchanged. [eg: python->nythop]

Input

```
s="python"
t=s[0]
t1=s[-1]
n=len(s)
rs=t1+s[1:n-1]+t
print(rs)
```

Output

```
user@user-desktop:~/anamika$ python3 reverseletter.py
nythop
```

9.

Aim:

Accept the radius from user and find area of circle.

Input

```
r=int(input("Enter the radius:"))
```

```
a=3.14*r*r
```

```
print("Area of circle is",a)
```

Output

```
user@user-desktop:~/anamika$ python3 areaofcircle.py
Enter the radius:4
Area of circle is 50.24
```

10.

Aim:

Find biggest of 3 numbers entered.

Input

```
a=int(input("Enter the first number:"))
b=int(input("Enter the second number:"))
c=int(input("Enter the third number:"))
if a>b:
if a>c:
print(a)
else:
print(c)
else:
if b>c:
print(b)
else:
print(c)
```

Output

```
user@user-desktop:~/anamika$ python3 biggest.py
Enter the first number:9
Enter the second number:34
Enter the third number:12
34
```

11.

Aim:

Accept a file name from user and print extension of that.

Input

```
import os
a=input("Enter the file name:")
print("The extension of file",a,"is",os.path.splitext(a))
```

Output

```
user@user-desktop:~/anamika$ python3 extension.py
Enter the file name:swap.py
The extension of file swap.py is ('swap', '.py')
```

12.

Aim:

Create a list of colors from comma-separated color names entered by user. Display first and last colors.

Input

```
list1=[]  
string=input("Enter colors separated by comma:\n")  
for i in string.split(","):  
    list1.append(i)  
print("First and last colors in the list are",list1[0],"and",list1[-1])
```

Output

```
user@user-desktop:~/anamika$ python3 13.py  
Enter colors separated by comma:  
Red,Yellow,Green,Blue  
First and last colors in the list are Red and Blue
```

13.

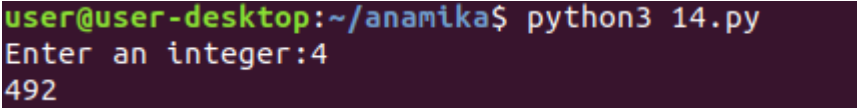
Aim:

Accept an integer n and compute $n+nn+nnn$.

Input

```
x=int(input("Enter an integer:"))
n1=str(x)
n2=n1+n1
n3=n2+n1
result=int(n1)+int(n2)+int(n3)
print(result)
```

Output



```
user@user-desktop:~/anamika$ python3 14.py
Enter an integer:4
492
```

14.

Aim:

Print out all colors from color-list1 not contained in color-list2.

Input

```
l1=["red","green","blue","maroon","peach","orange"]
```

```
l2=["white","green","maroon","black"]
```

```
print(l1)
```

```
print(l2)
```

```
for i in l1:
```

```
if i not in l2:
```

```
print(i)
```

Output

```
user@user-desktop:~/anamika$ python3 colourlist.py
['red', 'green', 'blue', 'maroon', 'peach', 'orange']
['white', 'green', 'maroon', 'black']
red
blue
peach
orange
```

15.

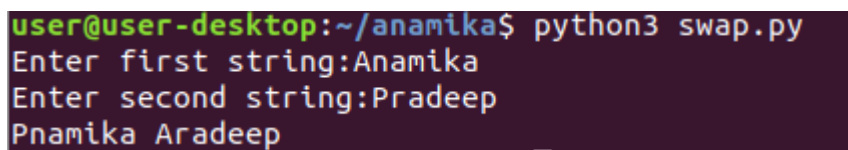
Aim:

Create a single string separated with space from two strings by swapping the character at position 1.

Input

```
str1=input("Enter first string:")  
str2=input("Enter second string:")  
str3=str2[0]+str1[1:]+""+str1[0]+str2[1:]  
print(str3)
```

Output



```
user@user-desktop:~/anamika$ python3 swap.py  
Enter first string:Anamika  
Enter second string:Pradeep  
Pnamika Aradeep
```


16.

Aim:

Sort dictionary in ascending and descending order.

Input

```
dict1={"a":1,"c":3,"d":2,"b":4}
```

```
l=list(dict1.items())
```

```
print(l)
```

```
l.sort()
```

```
print("Ascending order is\n",l)
```

```
l=list(dict1.items())
```

```
l.sort(reverse=True)
```

```
print("Descending order is\n",l)
```

Output

```
user@user-desktop:~/anamika$ python3 sortdic.py
[('a', 1), ('c', 3), ('d', 2), ('b', 4)]
Ascending order is
[('a', 1), ('b', 4), ('c', 3), ('d', 2)]
Descending order is
[('d', 2), ('c', 3), ('b', 4), ('a', 1)]
```

17.

Aim:

Merge two dictionaries.

Input

```
dict1={"Name":"Athidhi","Age":25}  
dict2={"Gender":"F","Qualification":"PG"}  
dict1.update(dict2)  
print(dict1)
```

Output

```
user@user-desktop:~/anamika$ python3 mergedic.py  
{'Name': 'Athidhi', 'Age': 25, 'Gender': 'F', 'Qualification': 'PG'}
```

18.

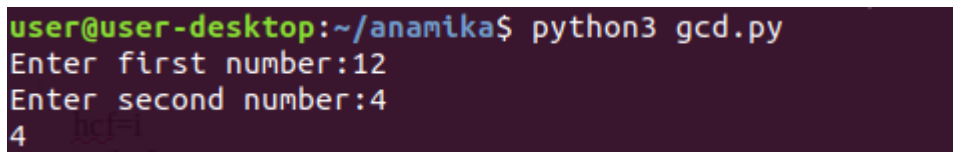
Aim:

Find gcd of 2 numbers.

Input

```
x=int(input("Enter first number:"))
y=int(input("Enter second number:"))
if x>y:
    s=y
else:
    s=x
for i in range(1,s+1):
    if(x%i==0)and(y%i==0):
        hcf=i
print(hcf)
```

Output



```
user@user-desktop:~/anamika$ python3 gcd.py
Enter first number:12
Enter second number:4
4
```

19.

Aim:

From a list of integers, create a list removing even numbers.

Input

```
l1=[1,2,3,4,5,6]
```

```
l2=[]
```

```
for i in l1:
```

```
    if i%2!=0:
```

```
        l2.append(i)
```

```
print(l2)
```

Output

```
user@user-desktop:~/anamika$ python3 evenremove.py
[1, 3, 5]
```

CO2

1.

Aim:

Program to find the factorial of a number.

Input

```
n=int(input("Enter a number:"))  
fact=1  
for i in range(1,n+1):  
    fact=fact*i  
print(fact)
```

Output

```
user@user-desktop:~/anamika$ python3 factorial.py  
Enter a number:5  
120
```

2.

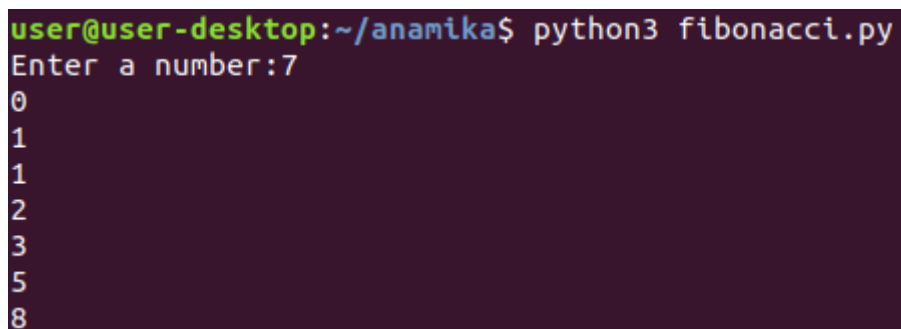
Aim:

Generate Fibonacci series of N terms.

Input

```
n=int(input("Enter a number:"))  
f1=0  
f2=1  
print(f1)  
print(f2)  
for i in range(0,n-2):  
    f3=f1+f2  
    print(f3)  
    f1=f2  
    f2=f3
```

Output



```
user@user-desktop:~/anamika$ python3 fibonacci.py  
Enter a number:7  
0  
1  
1  
2  
3  
5  
8
```

3.

Aim:

Find the sum of all items in a list.

Input

```
list1=[1,2,3,4,5]
```

```
sum=0
```

```
for i in list1:
```

```
sum=sum+i
```

```
print(sum)
```

Output

```
user@user-desktop:~/anamika$ python3 listsum.py
15
```

4.

Aim:

Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square.

Input

```
limit1=1000
```

```
limit2=9999
```

```
list1=[]
```

```
for i in range(limit1,limit2):
```

```
    j=i
```

```
    digit=[]
```

```
    while(i!=0):
```

```
        digit.append(i%10)
```

```
        i=int(i/10)
```

```
    count=0
```

```
    for n in digit:
```

```
        if n%2==0:
```

```
            count=count+1
```

```
    if count==4:
```

```
        for k in range(31,100):
```

```
            if((k**2)==j):
```

```
                list1.append(j)
```

```
print(k)
```

```
print(list1)
```


Output

```
user@user-desktop:~/anamika$ python3 perfectsq.py  
68  
78  
80  
92  
[4624, 6084, 6400, 8464]
```

5.

Aim:

Display the given pyramid with step number accepted from the user.

Eg: N=4

```
1
2 4
3 6 9
4 8 12 16
```

Input

```
n=int(input("Enter a number:"))
for i in range(1,n+1):
    for j in range(i,(i*i)+1,i):
        print(j,"\\t",end="")
    print("\\n")
```

Output

```
user@user-desktop:~/anamika$ python3 stepnumber.py
Enter a number:5
1
2      4
3      6      9
4      8      12      16
5      10      15      20      25
```

6.

Aim:

Count the number of characters (character frequency) in a string.

Input

```
string=input("Enter a string:")
```

```
list1=[]
```

```
for i in string:
```

```
if i not in list1:
```

```
list1.append(i)
```

```
for i in list1:
```

```
count=0
```

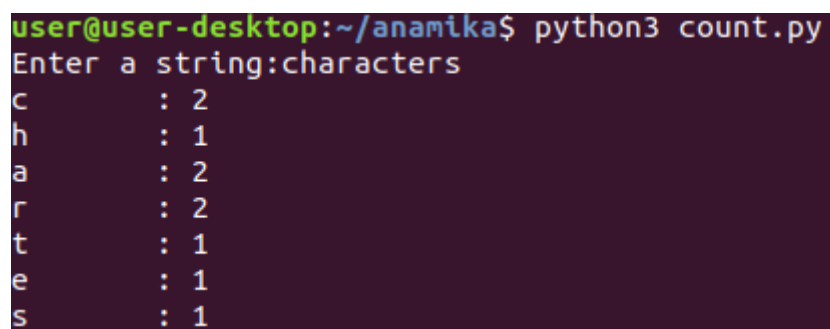
```
for j in string:
```

```
if(i==j):
```

```
count=count+1
```

```
print(i,"\t:",count)
```

Output



```
user@user-desktop:~/anamika$ python3 count.py
Enter a string:characters
c      : 2
h      : 1
a      : 2
r      : 2
t      : 1
e      : 1
s      : 1
```

7.

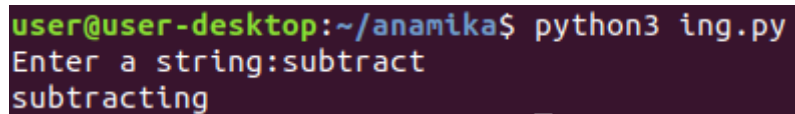
Aim:

Add 'ing' at the end of a given string. If it already ends with 'ing', then add 'ly'.

Input

```
string=input("Enter a string:")  
if(string[-3:]== "ing"):  
    string+="ly"  
else:  
    string+="ing"  
print(string)
```

Output



```
user@user-desktop:~/anamika$ python3 ing.py  
Enter a string:subtract  
subtracting
```

8.

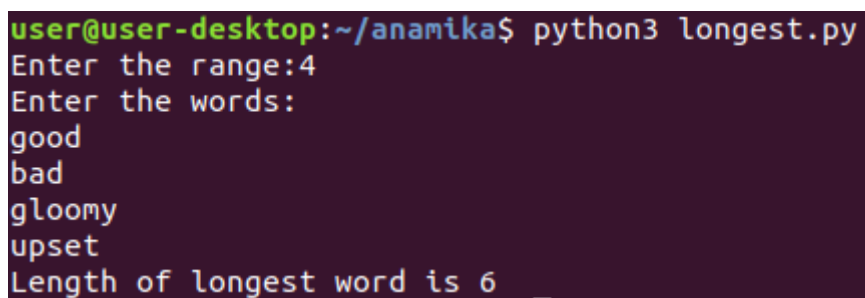
Aim:

Accept a list of words and return length of longest word.

Input

```
lis=[]
n=int(input("Enter the range:"))
print("Enter the words:")
for i in range(0,n):
    lis.append(input(""))
longest=lis[0]
for i in range(1,n):
    if(len(lis[i])>len(longest)):
        longest=lis[i]
print("Length of longest word is",len(longest))
```

Output



```
user@user-desktop:~/anamika$ python3 longest.py
Enter the range:4
Enter the words:
good
bad
gloomy
upset
Length of longest word is 6
```

9.

Aim:

Construct following pattern using nested loop.

```
*
* *
* * *
* * * *
* * * * *
* * * * *
* * * *
* * *
* *
*
```

Input

```
k='*'
for i in range(1,6):
    for j in range(1,i+1):
        print(k,end="")
    print("\n")
    for i in range(4,0,-1):
        for j in range(1,i+1):
            print(k,end="")
        print("\n")
```

Output

```
user@user-desktop:~/anamika$ python3 starpyramid.py
*
* *
* * *
* * * *
* * * * *
* * * * *
* * * *
* * *
* *
*
```

10.

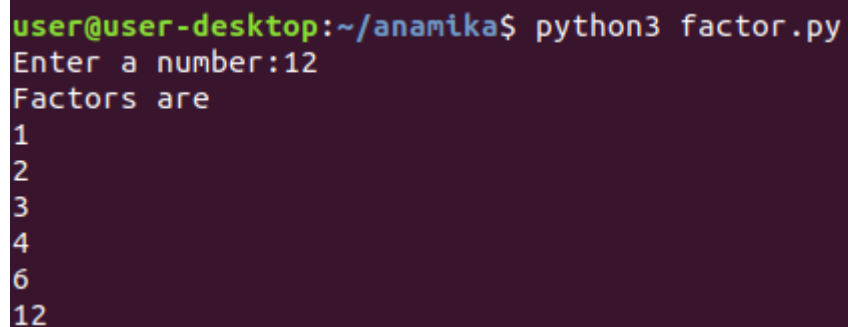
Aim:

Generate all factors of a number.

Input

```
n=int(input("Enter a number:"))
print("Factors are")
for i in range(1,n+1):
    if(n%i==0):
        print(i)
```

Output

A terminal window with a dark purple background. The prompt is 'user@user-desktop:~/anamika\$'. The user has entered 'python3 factor.py'. The program prompts 'Enter a number:12' and then 'Factors are'. It then lists the factors: 1, 2, 3, 4, 6, and 12, each on a new line.

```
user@user-desktop:~/anamika$ python3 factor.py
Enter a number:12
Factors are
1
2
3
4
6
12
```


CO3

1.

Aim:

Create a package graphics with modules rectangle, circle and sub-package 3D-graphics with modules cuboid and sphere. Include methods to find area and perimeter of respective figures in each module. Write programs that finds area and perimeter of figures by different importing statements. (Include selective import of modules and import * statements)

Input

circle.py

```
from math import pi

def area_circle(radius):
    return pi*radius*radius

def perimeter_circle(radius):
    return 2*pi*radius
```

rectangle.py

```
def area_rec(length,width):
    return length*width

def perimeter_rec(length,width):
    return 2*(length+width)
```

cuboid.py

```
def area_cuboid(l,b,h):
    return 2*(l*h + b*h + l*b)

def volume_cuboid(l,b,h):
    return l*b*h
```

sphere.py

```
from math import pi
def area_sphere(radius):
    return 4*(pi*radius*radius)
def perimeter_sphere(radius):
    return 2*pi*radius
```

graphics.py

```
import Graphics
from Graphics import circle,rectangle
from Graphics.threedgraphics import cuboid,sphere
from Graphics.circle import *
print("Area of the Circle : ",circle.area_circle(12))
print("Perimeter of the Circle : ",circle.perimeter_circle(12))
print("\n")

print("Area of the Rectangle : ",rectangle.area_rec(14,8))
print("Perimeter of the Rectangle : ",rectangle.perimeter_rec(14,8))
print("\n")

print("Area of the Cuboid : ",cuboid.area_cuboid(8,8,8))
print("Volume of the Cuboid : ",cuboid.volume_cuboid(8,8,8))
print("\n")

print("Area of the Sphere : ",sphere.area_sphere(6))
print("Perimeter of the Sphere : ",sphere.perimeter_sphere(6))
```

```

C:\Users\useresq\Desktop\Anamika>cd Python
C:\Users\useresq\Desktop\Anamika\Python>md Graphics
C:\Users\useresq\Desktop\Anamika\Python>cd Graphics
C:\Users\useresq\Desktop\Anamika\Python\Graphics>notepad __init__.py
C:\Users\useresq\Desktop\Anamika\Python\Graphics>notepad circle.py
C:\Users\useresq\Desktop\Anamika\Python\Graphics>notepad rectangle.py
C:\Users\useresq\Desktop\Anamika\Python\Graphics>md threedgraphics
C:\Users\useresq\Desktop\Anamika\Python\Graphics>cd threedgraphics
C:\Users\useresq\Desktop\Anamika\Python\Graphics\threedgraphics>notepad __init__.py
C:\Users\useresq\Desktop\Anamika\Python\Graphics\threedgraphics>notepad cuboid.py
C:\Users\useresq\Desktop\Anamika\Python\Graphics\threedgraphics>notepad sphere.py
C:\Users\useresq\Desktop\Anamika\Python\Graphics\threedgraphics>cd ..
C:\Users\useresq\Desktop\Anamika\Python\Graphics>cd ..

```

Output

```

C:\Users\useresq\Desktop\Anamika\Python>python graphics.py
Area of the Circle : 452.3893421169302
Perimeter of the Circle : 75.39822368615503

Area of the Rectangle : 112
Perimeter of the Rectangle : 44

Area of the Cuboid : 384
Volume of the Cuboid : 512

Area of the Sphere : 452.3893421169302
Perimeter of the Sphere : 37.69911184307752

```

CO4

1.

Aim:

Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two Rectangle objects by their area.

Input

```
class Rectangle:
```

```
    def __init__(self,length,breadth):
        self.length=length
        self.breadth=breadth

    def area(self):
        return self.length*self.breadth

    def perimeter(self):
        return 2*(self.length+self.breadth)
```

```
r1=Rectangle(6,4)
```

```
r2=Rectangle(10,7)
```

```
x=r1.area()
```

```
y=r2.area()
```

```
z=r1.perimeter()
```

```
w=r2.perimeter()
```

```
print("Area of rectangle1 is",x)
```

```
print("Area of rectangle2 is",y)
```

```
print("Perimeter of rectangle1 is",z)
```

```
print("Perimeter of rectangle2 is",w)
```

```
if(x>y):
```

```
    print("Area of rectangle 1 is larger")
```

```
else:
```

```
    print("Area of rectangle 2 is larger")
```

Output

```
user@user-desktop:~/anamika$ python3 rectangle.py
Area of rectangle1 is 24
Area of rectangle2 is 70
Perimeter of rectangle1 is 20
Perimeter of rectangle2 is 34
Rectangle 2 is larger
```

2.

Aim:

Create a Bank account with members account number, name, type of account and balance. Write constructor and methods to deposit at the bank and withdraw an amount from the bank.

Input

class bank:

```
    def __init__(self,acc_no,name,acc_type,balance):
        self.acc_no=acc_no
        self.name=name
        self.type=acc_type
        self.balance=balance

    def withdrawal(self,x):
        self.balance=self.balance-x
        print("Balance amount after withdrawal:",self.balance)

    def deposit(self,y):
        self.balance=self.balance+y
        print("Balance amount after deposit:",self.balance)

    def display(self):
        print("Account Number:",self.acc_no)
        print("Account Name:",self.name)
        print("Account Type:",self.type)
        print("Account Balance:",self.balance)
```

```
account1=bank(1234,"Anu","Savings",25000)
```

```
account2=bank(3456,"Ammu","Savings",5000)
```

```
account3=bank(7890,"Anju","Savings",15000)
```

```
account4=bank(4587,"Athira","Savings",4000)
```

```
account1.deposit(10000)
```

```
account1.withdrawal(2000)
```

```
account2.deposit(5000)
```

```
account2.withdrawal(1000)
```

```
account3.deposit(15000)
```

```
account3.withdrawal(3000)
```

```
account4.deposit(20000)
```

```
account4.withdrawal(10000)
```

Output

```
user@user-desktop:~/anamika$ python3 bank.py
Balance amount after deposit: 35000
Balance amount after withdrawal: 33000
Balance amount after deposit: 10000
Balance amount after withdrawal: 9000
Balance amount after deposit: 30000
Balance amount after withdrawal: 27000
Balance amount after deposit: 24000
Balance amount after withdrawal: 14000
```

3. Aim:

Create a class Rectangle with private attributes length and width. Overload '<' operator to compare the area of two rectangles.

Input

```
class Rectangle:
```

```
    def __init__(self,length,breadth):
```

```
        self.length=length
```

```
        self.breadth=breadth
```

```
    def area(self):
```

```
        return self.length*self.breadth
```

```
    def perimeter(self):
```

```
        return 2*(self.length+self.breadth)
```

```
    def __lt__(self,r2):
```

```
        if(self.length*self.breadth<r2.length*r2.breadth):
```

```
            return True
```

```
        else:
```

```
            return False
```

```
r1=Rectangle(10,6)
```

```
r2=Rectangle(6,4)
```

```
x=r1.area()
```

```
y=r2.area()
```

```
z=r1.perimeter()
```

```
w=r2.perimeter()
```

```
print("Area of rectangle 1 is",x)
```

```
print("Area of rectangle 2 is",y)
```

```
print("Perimeter of rectangle 1 is",z)
```

```
print("Perimeter of rectangle 2 is",w)
```

```
if(r1<r2):
```

```
    print("Rectangle 1 is smaller")
```

```
else:
```

```
    print("Rectangle 2 is smaller")
```


Output

```
user@user-desktop:~/anamika$ python3 rectangle1.py
Area of rectangle 1 is 60
Area of rectangle 2 is 24
Perimeter of rectangle 1 is 32
Perimeter of rectangle 2 is 20
Area of Rectangle 2 is smaller
```

4.

Aim:

Create a class Time with private attributes hour, minute and second. Overload '-' operator to find sum of two time.

Input

class Time:

```
    def __init__(self, hour, minute, second):
        self.__hour = hour
        self.__minute = minute
        self.__second = second

    def __add__(self, t2):
        a = self.__hour + t2.__hour
        b = self.__minute + t2.__minute
        c = self.__second + t2.__second
        print("The Sum of Two Times is", a, b, c)
```

```
t1 = Time(2, 40, 15)
```

```
t2 = Time(5, 12, 20)
```

```
t3 = t1 + t2
```

Output

```
user@user-desktop:~/anamika$ python3 time.py
The Sum of Two Times is 7 52 35
```

5.

Aim:

Create a class Publisher(name). Derive a class Book from Publisher with attributes title and author. Derive class Python from Book with attributes price and number_of_pages. Write a program that displays information about a Python book. Use base class constructor invocation and method overriding.

Input

```
class Publisher:
```

```
    def __init__(self,name):
        self.name=name
```

```
class Book(Publisher):
```

```
    def __init__(self,name,title,author):
        super().__init__(name)
        self.title=title
        self.author=author

    def display(self):
        print("Name:",self.name)
        print("Title:",self.title)
        print("Author:",self.author)
```

```
class Python(Book):
```

```
    def __init__(self,name,title,author,price,no_of_pages):
        super().__init__(name,title,author)
        self.price=price
        self.no_of_pages=no_of_pages

    def display(self):
        print("Name:",self.name)
```

```
print("Title:",self.title)

print("Author:",self.author)

print("Price:",self.price)


print("Number of Pages:",self.no_of_pages)


p1=Python("Khanna Publishing House","Taming Python","Jeeva Jose",200,500)
p1.display()
p2=Book("ABC Publications","Python For Everybody","Charles Severence")
p2.display()
```

Output

```
user@user-desktop:~/anamika$ python3 book.py
Name: Khanna Publishing House
Title: Taming Python
Author: Jeeva Jose
Price: 200
Number of Pages: 500
Name: ABC Publications
Title: Python For Everybody
Author: Charles Severence
```

CO5

1.

Aim:

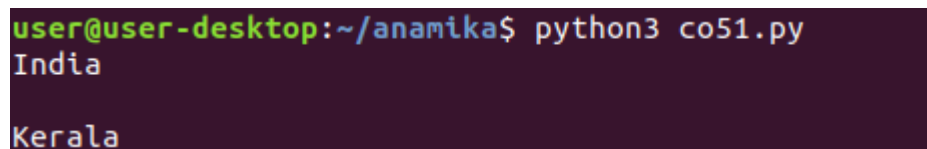
Write a Python program to read a file line by line and store it into a list.

Input

```
f=open("data_file.txt","w")  
f.write("India")  
f.write("\n")  
f.write("Kerala")  
f.close()
```

```
f=open("data_file.txt","r")  
for x in f.readlines():  
    print(x)
```

Output



```
user@user-desktop:~/anamika$ python3 co51.py  
India  
Kerala
```

2.

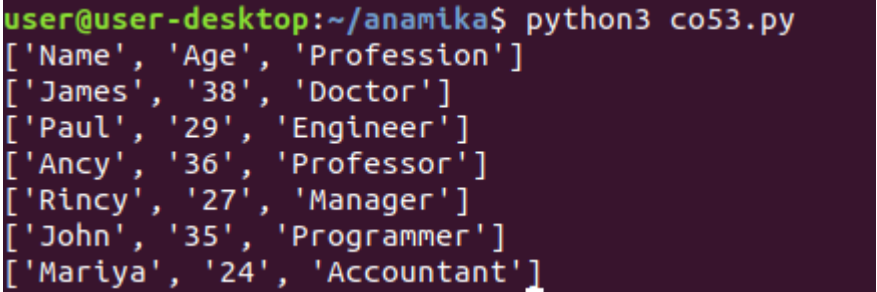
Aim:

Write a Python program to read each row from a given csv file and print a list of strings.

Input

```
import csv  
with open("excel.csv","r")as file:  
    reader=csv.reader(file)  
    for row in reader:  
        print(row)
```

Output



```
user@user-desktop:~/anamika$ python3 co53.py  
['Name', 'Age', 'Profession']  
['James', '38', 'Doctor']  
['Paul', '29', 'Engineer']  
['Ancy', '36', 'Professor']  
['Rincy', '27', 'Manager']  
['John', '35', 'Programmer']  
['Mariya', '24', 'Accountant']
```