SRM PYTHON PROGRAMS

Saturday, 23 March 2019

Google Local Connect

The world is suffering from plethora of conundrums. You just name it and the list goes rampant. Climate Change, Terrorism, Poverty, Hatred, Jealousy et cetera. The recent Pulwama terror attack in India, Chrishtchurch mosque mass shooting in New Zealand, Idai cyclone havoc in Mozambique, Sweeping away of entire villages upon breaking of dam in Brazil, Chemical plant explosion in China and millions of apocalypses are yet to shadow over the earth.

But a lotus grows in the mud. Isn't Google Maps one of the Lotus? It motivates people to tour around the whole world by placing the same in his/her hand. As the saying goes "Information is knowledge" and "Knowledge is wisdom." Isn't Google making you the wisest of all the sagacious? One can know a place better than a local guide. One can be a connoisieur of places.

Posted by Unknown at 11:48 11 comments:

Wednesday, 7 March 2018

You are kindly requested to view this blog only when you have tried at least 10 times to solve any program.

SESSION: Input and Output

Q.1811110024: Day Old Bread

QUESTION DESCRIPTION

A bakery sells loaves of bread for 185 rupees each. Day old bread is discounted by 60 percent. Write a program that begins by reading the number of loaves of day old bread being purchased from the user.

Then your program should display the regular price for the bread, the discount because it is a day old, and the total price.

All of the values should be displayed using two decimal places, and the decimal points in all of the numbers should be aligned when reasonable values are entered by the user.

```
a = int(input())
print("Loaves Discount")
r=185*a
d=0.6*185*a
a=0.4*185*a
print("Regular Price " + str(r))
print("Total Discount "+ str(d))
print("Total Amount to be paid " + str(a))
```

SESSION: Input and Output

Q.1811110004: Display It

QUESTION DESCRIPTION

Create a program that displays your name and complete mailing address formatted in the manner that you would usually see it on the outside of an envelope.

About Me

Unknown

View my complete profile

Blog Archive

▼ 2019 (1)
▼ March (1)
Google Local Connect

► 2018 (1)

Your program needs to read any input from the user

```
a=str(input())
b=str(input())
c=str(input())
d=int(input())
print (a)
print (b)
print (c)
print (d)
```

SESSION: Input and Output

Q.1811110040: Sum of N series

QUESTION DESCRIPTION
Python Program to Read a number n and Compute n+nn+nnn

```
n=int(input())
temp=str(n)
t1=temp+temp
t2=temp+temp+temp
comp=n+int(t1)+int(t2)
print(comp)
```

SESSION: Input and Output

Q.1811110018: Units of Time

QUESTION DESCRIPTION
Create a program that reads a duration from the user as a number of days, hours, minutes, and seconds. Compute and display the total number of seconds represented by this duration.

```
a=int(input())
b=int(input())
c=int(input())
d=int(input())
e=d+c*60+b*3600+a*24*3600
print("Days= " + str(a))
print("Hours= " + str(b))
print("Minutes= " + str(c))
print("Seconds= " + str(d))
print("Total= " + str(e))
```

Q.1811110007: Area of a Field

QUESTION DESCRIPTION
Create a program that reads the length and width of a farmers field from the user in feet. Display the area of the field in acres.

Hint:
There are 43,560 square feet in an acre

SQ=43560; (Squarefeet)

Formula = (length*width)/squarefeet

```
a=float(input())
b=float(input())
c=a*b/43560
d=round(c,2)
print("The area of the field is " + str(d) + " acres")
```

SESSION: Input and Output

Q.1811110027: Conversion

QUESTION DESCRIPTION

Chris was given a task by his friend rajesh. When rajesh tells a number then chris needs to convey the value in binary, octal, hexadecimal for the decimal equivalent value told by his friend rajesh.

Help them to achieve this task

```
a=int(input())
print("The Binary Value " + str(bin(a)))
print("The octal Value " + str(oct(a)))
print("The hexadecimal Value " + str(hex(a)))
```

SESSION: Input and Output

Q.1811110036: Sum of the Series

QUESTION DESCRIPTION

Write a program to find sum of series

Hint:1+2+3+4+n

```
a=int(input())
s=0
for i in range(1,a+1):
 s=s+i
print(s)
```

SESSION: Input and Output

Q.1811110006: Area of a Room

QUESTION DESCRIPTION

Write a program that asks the user to enter the width and length of a room.

Once the values have been read, your program should compute and display the area of the room.

The length and the width will be entered as floating point numbers. Include units in your prompt and output message; either feet or meters, depending on which unit you are more comfortable working with.

Formula = length*width

```
a=float(input())
b=float(input())
c=a*b
d=round(c,2)
print("The area of the room is " + str(d) + " square feet")
```

SESSION: Control Stmt - if-elif-el

Q.1812110006: Name of the Shape

QUESTION DESCRIPTION

Write a program that determines the name of a shape from its number of sides.

Read the number of sides from the user and then report the appropriate name as part of a meaningful message.

Your program should support shapes with anywhere from 3 up to (and including) 6sides.

If a number of sides outside of this range is entered then your program should display an appropriate error message.

1. If the input is 5 then display as Pentagon

2. if the input is 6 display as Hexagon

3.if the input is any another number then display the message "Input should be from 3 to 6"

Note:
Use only if and elif

```
a=int(input())
if (a == 3):
   print("Triangle")
elif (a == 4):
   print("Quadrilateral")
elif (a == 5):
   print("Pentagon")
elif( a == 6):
   print("Hexagon")
else:
        print ("Input should be from 3 to 6")
```

SESSION: Control Stmt - if-elif-el

Q.1812110028: Richter Scale

QUESTION DESCRIPTION
The following table contains earthquake magnitude ranges on the Richter scale and their descriptors:

Magnitude Descriptor

Less than 2.0 Micro
2.0 to less than 3.0 Very minor
3.0 to less than 4.0 Minor
4.0 to less than 5.0 Light
5.0 to less than 6.0 Moderate
6.0 to less than 7.0 Strong
7.0 to less than 8.0 Major
8.0 to less than 10.0 Great
10.0 or more Meteoric

Write a program that reads amagnitude from the user and displays the appropriate descriptor as part of a meaningful message. For example, if the user enters 5.5 then your program should indicate that a magnitude 5.5 earthquake is considered to be a moderate earthquake

Mandatory:

Use if and elif

```
a= float(input())
if (a<2.0):
   print("Micro")
elif (a>=2.0 and a<3.0):
   print("Very Minor")
elif (a>=3.0 and a<4.0):
   print("Minor")
elif (a>=4.0 and a<5.0):
   print("Light")
elif (a>=5.0 and a<6.0):
   print("Moderate")
elif (a>=6.0 and a<7.0):
   print("Strong")
elif (a>=7.0 and a<8.0):
   print("Major")
```

```
elif (a>=8.0 and a<9.0):
    print("Great")
else:
    print("Meteoric")
```

SESSION: Control Stmt - if-elif-el

Q.1812110012: Mirror Image

QUESTION DESCRIPTION

Puck, the trickster, has again started troubling people in your city.

The people have turned on to you for getting rid of Puck. Puck presents to you a number consisting of numbers from 0 to 9 characters.

He wants you to reverse it from the final answer such that the number becomes Mirror number.

A Mirror is a number which equals its reverse. The hope of people are on you so you have to solve the riddle.

You have to tell if some number exists which you would reverse to convert the number into mirror

```
a=int(input())
p=0
b=a
while(b>0):
    r=b % 10
    p=p*10+r
    b=b//10
if (p==a):
    print ("Mirror")
else:
    print("No Mirror")
```

SESSION: Control Stmt - if-elif-el

Q.1812110009: May to August

QUESTION DESCRIPTION

The length of a month varies from 30 and 31 days.

In this exercise you will create a program that reads the name of a month from the user as a string.

The program should get input from May to August.

If the input is from may to other months then display error messages as "Invalid"

Note:
Use only if and elif

```
a=str(input())
if a=="May":
    print(31)
elif a=="Jun":
    print(30)
elif a=="Jul":
    print(31)
elif a=="Aug":
    print(30)
else:
    print("Invalid")
```

SESSION: Control Stmt - if-elif-el

Q.1812110027: Find Year

QUESTION DESCRIPTION

Most years have 365 days. However, the time required for the Earth to orbit the Sun is actually slightly more than that. As a result, an extra day, February 29, is included in some years to correct for this difference. Such years are referred to as leap years.

The rules for determining whether or not a year is a leap year follow:

Any year that is divisible by 400 is a leap year.
Of the remaining years, any year that is divisible by 100 is not a leap year.
Of the remaining years, any year that is divisible by 4 is a leap year.
All other years are not leap years.

Write a program that reads a year from the user and displays a message indicating
whether or not it is a leap year.

```
a=int(input())
if a%400==0:
    print(str(a) + " is a leap year")
elif a%100==0:
    print(str(a) + " is not a leap year")
elif a%4==0:
    print(str(a) + " is a leap year")
else:
    print(str(a) + " is not a leap year")
```

SESSION: Control Stmt - if-elif-el

Q.1812110018: Note To Frequency - G NOTE

QUESTION DESCRIPTION

The following table lists an octave of music notes, beginning with middle G, along with their frequencies.

Note Frequency (Hz)
G4 392.00

Begin by writing a program that reads the name of a note from the user and displays the notes frequency. Your program should support all of the notes listed previously.

Once you have your program working correctly for the notes listed previously you should add support for all of the notes from G0 to G8.

While this could be done by adding many additional cases to your if statement, such a solution is cumbersome,
inelegant and unacceptable for the purposes of this exercise. Instead, you should exploit the relationship between notes in adjacent octaves. In particular, the frequency of any note in octave n is half the frequency of the corresponding note in octave n+1.

By using this relationship, you should be able to add support for the additional notes without adding additional cases to your if statement.

Hint: To complete this exercise you will need to extract individual characters from the two-character note name so that you can work with the letter and the octave number separately.

Once you have separated the parts, compute the frequency of the note in the fourth octave using the data in the table above.

Then divide the frequency by 2 power(4-x), where x is the octave number entered by the user. This will halve or double the frequency the correct number of times.

Then divide the frequency by 2 power(4-x), where x is the octave number entered by the user. (freq=freq/2**(4-octave))

```
C4_FREQ = 261.63
D4_FREQ = 293.66
E4_FREQ = 329.63
F4_FREQ = 349.23
G4_FREQ = 392.00
A4_FREQ = 440.00
B4_FREQ = 493.88
# Read the note name from the user
name = input()
# Store the note and its octave in separate variables
note = name [0]
octave = int(name[1])
# Get the frequency of the note, assuming it is in the fourth octave
if note == "C":
    freq = C4_FREQ
elif note == "D":
    freq = D4_FREQ
elif note == "E":
    freq = E4_FREQ
elif note == "F":
    freq = F4_FREQ
elif note == "G":
    freq = G4_FREQ
elif note == "A":
    freq = A4_FREQ
elif note == "B":
    freq = B4_FREQ
# Now adjust the frequency to bring it into the correct octave
freq = freq / 2 ** (4 - octave)
# Display the result
print (freq)
```

SESSION: Control Stmt - if-elif-el

Q.1812110041: Combinations

QUESTION DESCRIPTION
Python Program to Accept Three Digits and Print all Possible Combinations from the Digits.

```
a=int(input())
b=int(input())
c=int(input())
d=[]
d.append(a)
d.append(b)
d.append(c)
for i in range(0,3):
    for j in range(0,3):
        for k in range(0,3):
            if(i!=j&j!=k&k!=i):
                print(d[i],d[j],d[k])
```

SESSION: Control Stmt - if-elif-el

Q.1812110010: September to December

QUESTION DESCRIPTION

The length of a month varies from 30 and 31 days.

In this exercise you will create a program that reads the name of a month from the user as a string.

The program should get input from September to December.I

If the input is from may to other months then display error messages as "Invalid"

Note:
Use only if and elif

```
a=input()
if a == "Sep":
    print(30)
elif a == "Oct":
 print(31)
elif a == "Nov":
 print(30)
elif a == "Dec":
 print(31)
else:
 print("Invalid")
```

SESSION: Control Stmt - if-elif-el

Q.1812110040: Sieve of Eratosthenes

QUESTION DESCRIPTION
Python Program to Read & Print Prime Numbers in a Range using Sieve of Eratosthenes.

To find all the prime numbers less than or equal to a given integer n by Eratosthenes' method:

Create a list of consecutive integers from 2 through n: (2, 3, 4, ..., n).

Initially, let p equal 2, the smallest prime number.

Enumerate the multiples of p by counting to n from 2p in increments of p, and mark them in the list (these will be 2p, 3p, 4p, ...; the p itself should not be marked).

Find the first number greater than p in the list that is not marked. If there was no such number, stop. Otherwise, let p now equal this new number (which is the next prime), and repeat from step 3.

When the algorithm terminates, the numbers remaining not marked in the list are all the primes below n.

```
n=int(input())
sieve=set(range(2,n+1))
while sieve:
    prime=min(sieve)
    print(prime,end="\n")
    sieve-=set(range(prime,n+1,prime))

print()
```

SESSION: Control Stmt - if-elif-el

Q.1812110013: Name that Triangle

QUESTION DESCRIPTION
A triangle can be classified based on the lengths of its sides as equilateral, isosceles or scalene.

All 3 sides of an equilateral triangle have the same length.

An isosceles triangle has two sides that are the same length, and a third side that is a different length.

If all of the sides have different lengths then the triangle is scalene.

Write a program that reads the lengths of 3 sides of a triangle from the user. Display a message indicating the type of the triangle.

Mandatory:

Use if and elif

```
a=int(input())
b=int(input())
c=int(input())
if (a==b and b==c and c==a):
 print("equilateral")
elif (a==b or b==c or c==a):
 print("isosceles")
else:
 print("scalene")
```

SESSION: Control Stmt - for-while

Q.1813110029: Sum of Even and Odd

QUESTION DESCRIPTION
Write a program to calculate the sum of even and odd numbers including its count.

Input:
1.Total number of Inputs
2. Input elements

Output:
1. Odd numbers count
2. Even numbers count
3. Sum of even numbers
4. Sum of odd numbers

```
lst = []
lst1=[]
e=0
se=0
o=0
so=0
num = int(input())
str=input()
lst1=str.split(' ')
for n in range(num):
    lst.append(int(lst1[n]))
for n in range(num):
```

```
    if lst[n]%2==0:
        e=e+1
        se=se+lst[n]
    else:
        o=o+1
        so=so+lst[n]
print (o)
print (e)
print (se)
print (so)
```

SESSION: Control Stmt - for-while

Q.1813110010: Exponentation- Part B

QUESTION DESCRIPTION

Write a program to find the exponentation of given number

Input 1: Base
Input 2: Number of inputs

Output:
List of exponentation terms for the given input

```
a=int(input())
b=int(input())
print("The total terms is:",b)
for i in range(b+1):
    print(a**i)
```

SESSION: Control Stmt - for-while

Q.1813110027: Star Pattern 1

QUESTION DESCRIPTION

Write a program to print the star pattern

Input:Number of rows

Output:Star pattern

Refer sample input and output for formatting specification.

```
a=int(input())
for i in range(a,0,-1):
    for j in range(i,0,-1):
        print ("*",end="")
    print()
```

SESSION: Control Stmt - for-while

Q.1813110002: Prime Generator-2

QUESTION DESCRIPTION

Write a program to find prime numbers between two range excluding the starting and ending number of given range

Input 1: Lower Range

Input 2: Upper Range

Output:
List of prime numbers

Refer sample input and output for formatting specification.

```
a=int(input())
b=int(input())
c=0
for i in range (a,b+1):
    for j in range (1,i+1):
        r=i%j
        if r==0:
            c=c+1
    if c==2:
        print(i)
    c=0
```

SESSION: Control Stmt - for-while

Q.1813110026: Multiple Layer a and b

QUESTION DESCRIPTION
Write a program to find the numbers divisible by a input numbers within the given range

Input 1:Lower range
Input 2:Upper range
Input 3:Positive number
Input 4:Positive number

Output:Numbers divisible by given input numbers within the specified range

Refer sample input and output for formatting specification.

```
a=int(input())
b=int(input())
c=int(input())
d=int(input())
for i in range (a,b+1):
    if (i%c==0 and i%d==0):
        print(i)
```

SESSION: Control Stmt - for-while

Q.1813110034: Multiplication Table

QUESTION DESCRIPTION
Write a program to print multiplication table as shown in the test cases

TEST CASE 1
INPUT

```
2

5
```

OUTPUT

```
1 * 2 = 2

2 * 2 = 4
```

```
3 * 2 = 6

4 * 2 = 8

5 * 2 = 10
```

```
a=int(input())
b=int(input())
for i in range(1,b+1):
    print(str(i) + " * " + str(a) + " = " + str(i*a) )
```

SESSION: Control Stmt - for-while

Q.1813110025: Strong Number

QUESTION DESCRIPTION

Write a program to check whether the given number is Strong number or not

Input:Positive number

Output:Display the appropriate message

Refer sample input and output for formatting specification.

TEST CASE 1

INPUT

```
145
```

OUTPUT

```
The number is a strong number
```

```
sum1=0
num=int(input())
temp=num
while(num):
    i=1
    f=1
    r=num%10
    while(i<=r):
        f=f*i
        i=i+1
    sum1=sum1+f
    num=num//10
if(sum1==temp):
    print("The number is a strong number")
else:
    print("The number is not a strong number")
```

SESSION: Control Stmt - for-while

Q.1813110017: Sum of Positive Negative

QUESTION DESCRIPTION

Write a program to find the sum of odd numbers, even numbers and negative numbers

Input:Positive and negative numbers

Output:Display Sum of ODD,EVEN and NEGATIVE numbers respectively.

Refer sample input and output for formatting specification

TEST CASE 1
INPUT

```
10

2

3

4

5

6

7

8

9

10

11
```

OUTPUT

```
Sum of positive even numbers: 30

Sum of positive odd numbers: 35

Sum of negative numbers: 0
```

TEST CASE 2
INPUT

```
5

10

-10

20

-20

45
```

OUTPUT

```
Sum of positive even numbers: 30

Sum of positive odd numbers: 45

Sum of negative numbers: -30
```

```
a=int(input())
```

```
l=[]
se=0
so=0
sn=0
for i in range(a):
    b=int(input())
    l.append(b)
for i in range(a):
    if (l[i]>0):
     if (l[i]%2==0):
         se=se+l[i]
     else:
         so=so+l[i]
    else:
        sn=sn+l[i]
print ("Sum of positive even numbers:",se)
print ("Sum of positive odd numbers:",so)
print ("Sum of negative numbers:",sn)
```

SESSION: Control Stmt - for-while

Q.1813110003: Factorial

QUESTION DESCRIPTION

Write a program to find the factorial for the given number

Input 1:Negative number

Output: Display the message "Sorry,factorial does not exist for negative numbers"

Input 2: Positive number

Output:
Display the factorial of the given number

Refer sample input and output for formatting specification.

```
a=int(input())
f=1
if a<0:
   print("Sorry,factorial does not exist for negative numbers")
else:
   for i in range(1,a+1):
      f=f*i
   print(f)
```

SESSION: Control Stmt - for-while

Q.1813110008: Sum on n number

QUESTION DESCRIPTION

Write a program to find the sum of numbers
Input 1:Negative number

Output:
Display the appropriate error message

Input 2: Positive number

Output:
Display the sum

Refer sample input and output for formatting specification.

```
n=int(input())
if n<0:
   print("Enter a positive number")
else:
   print("The sum is",int((n*(n+1))/2))
```

SESSION: Strings

Q.1814110033: Number of Occurences of a Character

QUESTION DESCRIPTION

Write a program find the number of occurrences of a character in a given string.
Input:
First line is the string, s
Second line is the character, c
Output:
An integer, i.e. the number of occurrences of c in s.

TEST CASE 1
INPUT

```
welcome

w
```

OUTPUT

```
1
```

```
a=input()
b=input()
c=0
for i in range(len(a)):
    if a[i]==b:
        c=c+1
print (c)
```

SESSION: Strings

Q.1814110003: Anagrams

QUESTION DESCRIPTION

Write a function to check whether two given strings are anagram of each other or not. An anagram of a string is another string that contains same characters, only the order of characters can be different. For example, abcd and dabc are anagram of each other.

Problem Solution

1. Take two strings from the user and store them in separate variables.
2. Then use sorted() to sort both the strings into lists.
3. Compare the sorted lists and check if they are equal.
4. Print the final result.
5. Exit

TEST CASE 1
INPUT

```
eLab is an Auto Evaluation Tool

Tool Evaluation Auto an is eLab
```

OUTPUT

```
The strings are anagrams
```

```
s1=input()
s2=input()
if(sorted(s1)==sorted(s2)):
      print("The strings are anagrams")
else:
      print("The strings are not anagrams")
```

SESSION: Strings

Q.1814110006: Vowels

QUESTION DESCRIPTION
Implement the following algorithm

1. Take a string from the user and store it in a variable.
2. Initialize a count variable to 0.
3. Use a for loop to traverse through the characters in the string.
4. Use an if statement to check if the character is a vowel or not and increment the count variable if it is a vowel.
5. Print the total number of vowels in the string.
6. Exit

TEST CASE 1
INPUT

```
eLab tool has been used by 40000 users in Tamilnadu alone
```

```
a=input()
c=0
for i in range(len(a)):
    if (a[i]=='a' or a[i]=='e' or a[i]=='i' or a[i]=='o' or a[i]
        c=c+1
print("Number of vowels are")
print (c)
```

SESSION: Strings

Q.1814110018: Panagram

QUESTION DESCRIPTION

Write a Python program to check whether the given string is Pangram or not

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output

TEST CASE 1

INPUT

```
The quick brown fox jumps over the lazy dog
```

OUTPUT

```
The string is a pangram
```

```
from string import ascii_lowercase as asc_lower
def check(s):
    return set(asc_lower) - set(s.lower()) == set([])
strng=input()
if(check(strng)==True):
      print("The string is a pangram")
else:
      print("The string is not a pangram")
```

SESSION: Strings

Q.1814110036: Remove Vowels

QUESTION DESCRIPTION

Write a program that accepts a string and redisplays the same string after removing vowels from it.

TEST CASE 1

INPUT

```
welcome
```

OUTPUT

```
wlcm
```

```
a=input()
b=[]
c=0
for i in range(len(a)):
    if (a[i]=="a" or a[i]=="e" or a[i]=="i" or a[i]=="o" or a[i]
        c=c+1
    else:
        b.append(a[i])
print (*b,sep='')
```

SESSION: Strings

Q.1814110019: Numbers, Alphabets and Spaces

QUESTION DESCRIPTION
Write a Python program to count the numbers, alphabets and spaces in the entered string.

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output

Output:
1. First Line: Numbers count
2. Second Line: Alphabets Count
3. Spaces Count

TEST CASE 1
INPUT

```
eLab Tool version 1 released in 2017
```

OUTPUT

```
5
```

```
x=input()
letters=0
digit=0
space=0
for i in x:
    if i.isalpha():
        letters+=1
    elif i.isnumeric():
        digit+=1
    else :
        space+=1
print(digit)
print(letters)
print(space)
```

SESSION: Strings

Q.1814110029: CamelCase

QUESTION DESCRIPTION
Alice wrote a sequence of words in CamelCase as a string of letters, S, having the following properties:

1. It is a concatenation of one or more words consisting of English letters.
2. All letters in the first word are lowercase.
3. For each of the subsequent words, the first letter is uppercase and rest of the letters is lowercase.

Given S, print the number of words in S on a new line.
Input:
A single line containing string S.
Output:
Print the number of words in string S.

TEST CASE 1
INPUT

```
thisIsIndiaWelcome
```

OUTPUT

```
4
```

```
x=input()
c=0
for i in x:
    if (ord(i)>64 and ord(i)<91):
        c+=1
print(c+1)
```

SESSION: Strings

Q.1814110004: Replace it

QUESTION DESCRIPTION
Problem Solution

1. Take a string from the user and store it in a variable.
2. Pass the string as an argument to a function.
3. In the function, split the string.
4. Then add the last character to the middle part of the string which is in turn added to the first character.
5. Print the modified string.
6. Exit.

Algorithm:

1. User must enter a string and store it in a variable.
2. This string is passed as an argument to a function.
3. In the function, using string slicing, the string is first split into three parts which is the last character, middle part and the first character of the string.
4. These three parts are then concatenated using the + operator.
5. The modified string is then printed.

TEST CASE 1
INPUT

```
eLab is an Auto Evaluation Tool
```

OUTPUT

```
lLab is an Auto Evaluation Tooe
```

```
x=input()
a=list(x)
b=a[0]
a[0]=a[-1]
a[-1]=b
s=''.join(a)
print(s)
```

SESSION: Strings

Q.1814110034: Replacing Whitespaces

QUESTION DESCRIPTION
Write a program to replace whitespaces in a string by *.

TEST CASE 1
INPUT

```
Welcome to India
```

OUTPUT

```
Welcome*to*India
```

```
a=input()
l=list(a)
for i in range(len(l)):
    if l[i]==' ':
        l[i]='*'
s=''.join(l)
print(s)
```

SESSION: Functions

Q.1815110001: Median of three Values

QUESTION DESCRIPTION
Write a function that takes three numbers as parameters, and returns the median value of those parameters as its result. Include a main program that reads three values from the user and displays their median.

Hint: The median value is the middle of the three values when they are sorted into ascending order. It can be found using if statements, or with a little bit of mathematical creativity

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output

TEST CASE 1
INPUT

```
9

12

22
```

OUTPUT

```
12
```

```
a=int(input())
b=int(input())
c=int(input())
def median(x,y,z):
    m1=min(x,y,z)
    m2=max(x,y,z)
```

```
    print ((x+y+z)-m1-m2)
median(a,b,c)
```

SESSION: Functions

Q.1815110023: Sine series

QUESTION DESCRIPTION
Python Program to Find the Sum of Sine Series

Input

First Line : The value of x in degrees
Second Line: The number of terms

Output
Print the Sum of Sine series

TEST CASE 1
INPUT

```
30

10
```

OUTPUT

```
0.5
```

```
import math
def sin(x,n):
    sine = 0
    for i in range(n):
        sign = (-1)**i
        pi=22/7
        y=x*(pi/180)
        sine = sine + ((y**(2.0*i+1))/math.factorial(2*i+1))*sig
    return sine
x=int(input())
n=int(input())
print(round(sin(x,n),2))
```

SESSION: Functions

Q.1815110014: Odd Print

QUESTION DESCRIPTION
Write a program to print all the odd numbers between two ranges(Include the range too)

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output

TEST CASE 1
INPUT

```
1
```

```
11
```

OUTPUT

```
1

3

5

7

9

11
```

```
a=int(input())
b=int(input())
def odd(x,y):
    for i in range(x,y+1):
        if (i%2==1):
            print (i)
odd(a,b)
```

SESSION: Functions

Q.1815110008: Is Valid Prime

QUESTION DESCRIPTION

A prime number is an integer greater than 1 that is only divisible by one and itself. Write a function that determines whether or not its parameter is prime, returning True if it is, and False otherwise.

Write a main program that reads an integer from the user and displays a message indicating whether or not it is prime.

Ensure that the main program will not run if the file containing your solution is imported into another program.

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output

TEST CASE 1
INPUT

```
7
```

OUTPUT

```
TRUE
```

```
a=int(input())
def prime(x):
    f=0
    for i in range(1,x+1):
        if (x%i==0):
            f=f+1
```

```
    if f==2:
        print ("TRUE")
    else:
        print("FALSE")
prime(a)
```

SESSION: Functions

Q.1815110007: Does a String Represent an Integer

QUESTION DESCRIPTION

In this exercise you will write a function named isInteger that determines whether or not the characters in a string represent a valid integer. When determining if a string represents an integer you should ignore any leading or trailing white space.

Once this white space is ignored, a string represents an integer if its length is at least 1 and it only contains digits, or if its first character is either + or - and the first character is followed by one or more characters, all of which are digits.

Write a main program that reads a string from the user and reports whether or not it represents an integer. Ensure that the main program will not run if the file containing your solution is imported into another program

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output

TEST CASE 1

INPUT

```
eLab
```

OUTPUT

```
Does not Contain
```

```
a=input()
def is_Integer(x):
    l=list(x)
    c=0
    for i in range(len(l)):
        ascii=ord(l[i])
        if (ascii>=48 and ascii<=57):
            c=c+1
    if c>0:
        print("Contains Integer")
    else:
        print("Does not Contain")
is_Integer(a)
```

SESSION: Functions

Q.1815110005: Shipping Calculator

QUESTION DESCRIPTION

An online retailer provides express shipping for many of its items at a rate of 750 for the first item, and 200 for each subsequent item. Write a function that takes the number of items in the order as its only parameter.

Return the shipping charge for the order as the functions result. Include a main program that reads the number of
items purchased from the user and displays the shipping charge.

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output

TEST CASE 1
INPUT

```
10
```

OUTPUT

```
2550
```

```
a=int(input())
def cost(x):
    if x==1:
        print(750)
    else:
        print(750+(x-1)*200)
cost(a)
```

SESSION: Functions

Q.1815110025: Natural Numbers

QUESTION DESCRIPTION
Python Program to Find the Sum of First N Natural Numbers

Input

Value of "N"

Output
Print the Sum of N natural Numbers

TEST CASE 1
INPUT

```
18
```

OUTPUT

```
171
```

```
a=int(input())
def sum(x):
    print(int(x*(x+1)/2))
sum(a)
```

SESSION: Functions

Q.1815110016: Sum Count

QUESTION DESCRIPTION
Write a program to find the sum of the entered numbers

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output

TEST CASE 1
INPUT

```
18
```

OUTPUT

```
The total sum of digits is: 9
```

TEST CASE 2
INPUT

```
25
```

```
a=int(input())
def sum(x):
    s=0
    while (x>0):
        r=x%10
        s=s+r
        x=x//10
    print("The total sum of digits is:",s)
sum(a)
```

SESSION: Functions

Q.1815110013: Grade System

QUESTION DESCRIPTION
Write a python function to calculate the GRADE systems of marks

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output

Input :
Input 5 numbers

Output:

1. If the average is above 90 then print Grade as "A" (Hint : 90 and above)
2. If the average is above 70 and less than 90 then print Grade as "B" (Hint: 71-89)
3. If the average is above 50 and less than 70 then print Grade as "C"(Hint: 51-70)
4. If the average is 50 or less then then print as "D"

TEST CASE 1
INPUT

```
99

98

87

99

90
```

OUTPUT

```
Grade:A
```

```
a=int(input())
b=int(input())
c=int(input())
d=int(input())
e=int(input())
def grade(v,w,x,y,z):
    avg=(v+w+x+y+z)/5
    if avg>=90:
        print("Grade:A")
    elif avg>70:
        print("Grade:B")
    elif avg>50:
        print("Grade:C")
    else:
        print("Grade:D")
grade(a,b,c,d,e)
```

SESSION: Functions

Q.1815110024: Cosine series

QUESTION DESCRIPTION

Python Program to Find the Sum of Cosine Series

Input:
First Line : Value of x in degrees
Second Line :Number of terms

Output:
Print the Sum of Cosine Series

TEST CASE 1
INPUT

```
65

10
```

OUTPUT

```
0.42
```

```
import math
def cosine(x,n):
    cosx = 1
    sign = -1
```

```
    for i in range(2, n, 2):
        pi=22/7
        y=x*(pi/180)
        cosx = cosx + (sign*(y**i))/math.factorial(i)
        sign = -sign
    return cosx
x=int(input())
n=int(input())
print(round(cosine(x,n),2))
```

SESSION: Lists and Tuples

Q.1816110003: Biggest Number

QUESTION DESCRIPTION

Write a program to find the biggest of "n" numbers using List

Input:

1. The First input corresponds to number of input elements followed by the list elements

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output.

TEST CASE 1
INPUT

```
4

99

45

33

45
```

OUTPUT

```
Largest element is

99
```

```
a=int(input())
l=[]
for i in range(a):
    b=int(input())
    l.append(b)
print ("Largest element is")
print(max(l))
```

SESSION: Lists and Tuples

Q.1816110002: Reverse Order

QUESTION DESCRIPTION

Write a program that reads integers from the user and stores them in a list. Use 0 as a sentinel value to mark the end of the input.

Once all of the values have been read your program should display them (except for the 0) in reverse order, with one value appearing on each line.

TEST CASE 1
INPUT

```
15

144

133

125

178

123

1345

376

0
```

OUTPUT

```
376

1345

123

178

125

133

144

15
```

```
l=[]
m=[]
for i in range(100):
    b=int(input())
    if b!=0:
        l.append(b)
    else:
        break
for i in range((len(l))):
    c=l[i]
    m.append(c)
m.reverse()
for i in range(len(m)):
    print (m[i])
```

SESSION: Lists and Tuples

Q.1816110023: Reverse and Remove

QUESTION DESCRIPTION

Write a program to add the list, display the list in the reverse order and delete the list element

Input:

1. The size of the first list
2. The size of the second list
3. The List elements of first and second list
4. The List element to be deleted

Output:

1. The appended or extended list
2. The reverse list
3. Th updated or final list after deleting

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output.

TEST CASE 1

INPUT

```
2

2

100

200

10

20

100
```

OUTPUT

```
The Extended List

[100, 200, 10, 20]

The Reverse List

[20, 10, 200, 100]

[20, 10, 200]
```

```
a=int(input())
b=int(input())
l1=[]
l2=[]
l3=[]
```

```
for i in range(a):
    c=int(input())
    l1.append(c)
for i in range(b):
    c=int(input())
    l1.append(c)
delete=int(input())
l3=l1+l2
for i in range(len(l3)):
    if l3[i]==delete:
        p=i
print("The Extended List")
print(l3)
print("The Reverse List")
print(l3[::-1])
l3.pop(p)
print(l3[::-1])
```

SESSION: Lists and Tuples

Q.1816110019: Index and Extend

QUESTION DESCRIPTION

Write a program to append two list and find the list index of the list element

Input:

1. The Size of First List
2. The Size of Second List
3. First List elements
4. Second List elements
5. First List Element to be searched
6. Second List Element to be searched

Output:

1. Extended List or appended list
2. The index of the first list element (entered by user, Step 5)
3. The index of the second list element (entered by user, Step 6)

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output.

TEST CASE 1

INPUT

```
3

3

11

22

33

99
```

```
199

299

11

299
```

OUTPUT

```
The Extended List

[11, 22, 33, 99, 199, 299]

Index for 11 = 0

Index for 299 = 5
```

```
a=int(input())
b=int(input())
l1=[]
l2=[]

for i in range(a):
    c=int(input())
    l1.append(c)
for i in range(b):
    c=int(input())
    l2.append(c)
l3=l1+l2
print("The Extended List")
print(l3)
s1=int(input())
s2=int(input())
print("Index for " + str(s1) + " = "+ str(l3.index(s1)))
print("Index for " + str(s2) + " = "+ str(l3.index(s2)))
```

SESSION: Lists and Tuples

Q.1816110035: Longest word

QUESTION DESCRIPTION

Python Program to Read a List of Words and Return the Length of the Longest One

Input:
First Line has the Number of Words the followed by the words

Output:
Print the longest word in the list.

TEST CASE 1
INPUT

```
4

Apple

Ball

Cat
```

```
Dinosaur
```

OUTPUT

```
Dinosaur
```

```
a=int(input())
l=[]
for i in range(a):
    b=str(input())
    l.append(b)
print (max(l,key = len))
```

SESSION: Lists and Tuples

Q.1816110032: Swap first and last

QUESTION DESCRIPTION

Python Program to Swap the First and Last Value of a List

Input:

First Line:Number of elements in list
Second Line:Elements of the List

Output:

Print the new list after swapping

TEST CASE 1
INPUT

```
4

23

45

67

89
```

OUTPUT

```
New list is:

[89, 45, 67, 23]
```

```
a=int(input())
l=[]
for i in range(a):
    b=int(input())
    l.append(b)
(l[0],l[a-1])=(l[a-1],l[0])
print("New list is:")
print(l)
```

SESSION: Lists and Tuples

Q.1816110022: Reverse

QUESTION DESCRIPTION

Write a program to add the list and display the list in the reverse order

Input:

1. The size of the first list
2. The size of the second list
3. The List elements of first and second list

Output:

1. The appended or extended list
2. The reverse list

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output.

TEST CASE 1

INPUT

```
3

3

81

71

61

10

20

30
```

OUTPUT

```
The Extended List

[81, 71, 61, 10, 20, 30]

The Reverse List

[30, 20, 10, 61, 71, 81]
```

```
a=int(input())
b=int(input())
l1=[]
l2=[]
for i in range(a):
    c=int(input())
    l1.append(c)
for i in range(b):
```

```
    c=int(input())
    l2.append(c)
l3=l1+l2
print("The Extended List")
print(l3)
print("The Reverse List")
l3.reverse()
print(l3)
```

SESSION: Lists and Tuples

Q.1816110017: Maximum and Minimum

QUESTION DESCRIPTION

Write a program to find the length of the list and compute the maximum and minimum of the list

Input:

1. The Size of First List
2. The Size of Second List
3. First List elements
4. Second List elements

Output:
1. Length of First List
2. Length of Second List
3. First List Data
4. First List Data
5. Maximum in First List
6. Minimum in First List
7. Maximum in Second List
8. Minimum in Second List

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output.

TEST CASE 1
INPUT

```
3

3

125

19

15

225

23

199
```

OUTPUT

```
Length of First List 3

Length of Second 3

First List Data

125

19

15

Second List Data

225

23

199

Maximum in List 1= 125

Minimum in List 1= 15

Maximum in List 2= 225

Minimum in List 2= 23
```

```
a=int(input())
b=int(input())
l1=[]
l2=[]
for i in range(a):
    c=int(input())
    l1.append(c)
for i in range(b):
    c=int(input())
    l2.append(c)
print("Length of First List",len(l1))
print("Length of Second",len(l2))
print("First List Data")
for i in range(len(l1)):
    print(l1[i])
print("Second List Data")
for i in range(len(l2)):
    print(l2[i])
print("Maximum in List 1=",max(l1))
print("Minimum in List 1=",min(l1))
print("Maximum in List 2=",max(l2))
print("Minimum in List 2=",min(l2))
```

SESSION: Matrix Operations

Q.1817110009: Matrix Multiplication - 1

QUESTION DESCRIPTION
Write a program to perform matrix multiplication

TEST CASE 1
INPUT

```
3

3

12

7

3

4

5

6

7

8

9

3

4

5

8

1

2

6

7

3

0

4

5

9

1
```

OUTPUT

```
114

160

60

27

74
```

```
97

73

14

119

157

112

23
```

```
a=int(input())
b=int(input())
m=[[int(input()) for i in range(b)] for j in range(a)]
c=int(input())
d=int(input())
n=[[int(input()) for i in range(d)] for j in range(c)]
r=[[0 for i in range(d)] for j in range(a)]
for i in range(a):
    for j in range(d):
        for k in range(b):
            r[i][j]+=m[i][k]*n[k][j]
for i in range(a):
    for j in range(d):
        print(r[i][j])
```

SESSION: Matrix Operations

Q.1817110005: Print Diagonal Elements

QUESTION DESCRIPTION

Write a python program to create a NESTED LIST and print the diagonal elements of the matrix

Hint:

1. Input the number of rows First Matrix

2. Input the number of Columns for Second Matrix

3. Display the diagonal elements in the matrix

TEST CASE 1
INPUT

```
3

3

10

20

30

40

50
```

```
60

70

80

90
```

OUTPUT

```
10

50

90
```

```
a=int(input())
b=int(input())
m=[[int(input()) for x in range(a)] for y in range(b)]
for i in range(a):
    print (m[i][i])
```

SESSION: Matrix Operations

Q.1817110012: Seasoners

QUESTION DESCRIPTION

The year is divided into four seasons: spring, summer, fall and winter. While the exact dates that the seasons change vary a little bit from year to year because of the way that the calendar is constructed, we will use the following dates for this exercise:

Season First day
Summer March 20
Spring June 21
Fall September 22
Winter December 21

Create a program that reads a month and day from the user. The user will enter the name of the month as a string, followed by the day within the month as an integer.

Then your program should display the season associated with the date that was entered.

Note: Enter First three letter for month example: Jan for january, Feb for Feburary ans so on....and first letter of the month should be capital

TEST CASE 1
INPUT

```
Sep

22
```

OUTPUT

```
Fall
```

```
m=input()
d=int(input())
if m=="Jan" or m=="Feb":
    s="Winter"
elif m=="Mar":
    if d<20:
        s="Winter"
    else:
        s="Summer"
elif m=="Apr" or m=="May":
    s="Summer"
elif m=="Jun":
    if d<21:
        s="Summer"
    else:
        s="Spring"
elif m=="Jul" or m=="Aug":
    s="Spring"
elif m=="Sep":
    if d<22:
        s="Spring"
    else:
        s="Fall"
elif m=="Oct" or m=="Nov":
    s="Fall"
elif m=="Dec":
    if d<21:
        s="Fall"
    else:
        s="Winter"
print(s)
```

SESSION: Matrix Operations

Q.1817110008: Transpose of Matrix

QUESTION DESCRIPTION

Write a python program to create a NESTED LIST and print the diagonal elements of the matrix

Hint:

1. Input the number of rows First Matrix

2. Input the number of Columns for first Matrix

3. Display the elements of the matrix

4. Display the transpose of the matrix

TEST CASE 1

INPUT

```
2

2

1

2
```

```
3

4
```

OUTPUT

```
Given Matrix

[1, 2]

[3, 4]

Transpose of the matrix

[1, 3]

[2, 4]
```

```
a=int(input())
b=int(input())
m=[[int(input()) for x in range(a)] for y in range(b)]
t=[[0 for x in range(b)] for y in range(a)]
print ("Given Matrix")
for i in range(a):
    print (m[i])
for x in range(a):
    for y in range(b):
        t[y][x]=m[x][y]
print("Transpose of the matrix")
for x in range(b):
    print(t[x])
```

SESSION: Matrix Operations

Q.1817110010: Matrix Multiplication 2

QUESTION DESCRIPTION
Write a program for matrix multiplication

TEST CASE 1
INPUT

```
3

3

1

2

0

0

1

1

2
```

```
0

1

3

3

1

1

2

2

1

1

1

2

1
```

OUTPUT

```
[5, 3, 4]

[3, 3, 2]

[3, 4, 5]
```

```
a=int(input())
b=int(input())
m=[[int(input()) for x in range(a)] for y in range(b)]
c=int(input())
d=int(input())
n=[[int(input()) for x in range(c)] for y in range(d)]
r=[[0 for x in range(a)] for y in range(d)]
for x in range(a):
    for y in range(d):
        for z in range(c):
            r[x][y]+=m[x][z]*n[z][y]
for x in r:
    print (x)
```

SESSION: Matrix Operations

Q.1817110013: Seasoners

QUESTION DESCRIPTION

The year is divided into four seasons: spring, summer, fall and winter. While the exact dates that the seasons change vary a little bit from year to year because of the way that the calendar is constructed, we will use the following dates for this exercise:

Season First day
Summer March 20
Spring June 21

Fall September 22
Winter December 21

Create a program that reads a month and day from the user. The user will enter the name of the month as a string, followed by the day within the month as an integer.

Then your program should display the season associated with the date that was entered.

Note: Enter First three letter for month example: Jan for january, Feb for Feburary ans so on....and first letter of the month should be capital

TEST CASE 1
INPUT

```
Sep

22
```

OUTPUT

```
Fall
```

```
m=input()
d=int(input())
if m=="Jan" or m=="Feb":
    s="Winter"
elif m=="Mar":
    if d<20:
        s="Winter"
    else:
        s="Summer"
elif m=="Apr" or m=="May":
    s="Summer"
elif m=="Jun":
    if d<21:
        s="Summer"
    else:
        s="Spring"
elif m=="Jul" or m=="Aug":
    s="Spring"
elif m=="Sep":
    if d<22:
        s="Spring"
    else:
        s="Fall"
elif m=="Oct" or m=="Nov":
    s="Fall"
elif m=="Dec":
    if d<21:
        s="Fall"
    else:
        s="Winter"
print(s)
```

SESSION: Matrix Operations

Q.1817110003: Matrix Addition

QUESTION DESCRIPTION

Write a python program to create a NESTED LIST and add the two matrix. (Without Square Brackets)

Hint:

1. Input the number of rows and columns for First Matrix

2. Input the number of rows and columns for Second Matrix

3. Display the first matrix elements in Matrix format (Without commas and Square Bracket)

4. Display the first matrix elements in Matrix format (Without commas and Square Bracket)

5. Display the result of sum of two matrix (Without commas and Square Bracket)

TEST CASE 1

INPUT

```
2

2

10

20

30

40

100

110

120

130
```

OUTPUT

```
Matrix 1

10

20

30

40

Matrix 2

100

110

120

130
```

```
Sum of Matrix

110

130

150

170
```

```
a=int(input())
b=int(input())
m=[[int(input()) for x in range(a)] for y in range(b)]
n=[[int(input()) for x in range(a)] for y in range(b)]
r=[[0 for x in range(a)] for y in range(b)]
for x in range(a):
    for y in range(b):
        r[x][y]=m[x][y] + n[x][y]
print("Matrix 1")
for x in range(a):
    for y in range(b):
        print (m[x][y])
print("Matrix 2")
for x in range(a):
    for y in range(b):
        print (n[x][y])
print("Sum of Matrix")
for x in range(a):
    for y in range(b):
        print (r[x][y])
```

SESSION: Dictionary

Q.1818110015: Display Items

QUESTION DESCRIPTION

Write a program to get four dictionary items and display all the key-values format using update function and items() function

TEST CASE 1

INPUT

```
10

-10

20

-20

30

-30

40

-40
```

OUTPUT

```
[(10, -10)]

[(20, -20)]

[(30, -30)]

[(40, -40)]
```

```
d={}
for i in range(4):
    key=int(input())
    value=int(input())
    d[key]=value
for key,value in d.items():
    print("[(" + str(key) + ", " + str(value) + ")]")
```

SESSION: Dictionary

Q.1818110008: Check the Key

QUESTION DESCRIPTION

Write a program to check whether the given key is present in the dictionary.

If the key is present then display the value of the entered key and if the key is not present then display the appropriate message

Input:

1.Get two dictionaries key-value elements
2. Get the key value to be searched

Output:
1. Display the first dictionary
2. Display the second dictionary
3. Display the concatenated dictionary
4. Display whether the key is present or nor and the respective value of the key.

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output

TEST CASE 1
INPUT

```
81

9

64

8

81
```

OUTPUT

```
First Dictionary

{81: 9}

Second Dictionary

{64: 8}

Concatenated dictionary is

{81: 9, 64: 8}

Key is present and value of the key is

9
```

```
d1={}
a=int(input())
b=int(input())
d1[a]=b

d2={}
c=int(input())
d=int(input())
d2[c]=d

print('First Dictionary')
print(d1)
print('Second Dictionary')
print(d2)
print('Concatenated dictionary is')
d1.update(d2)
print(d1)
val=int(input())
c=0
for k,v in d1.items():
    if k==val:
        c=c+1
        print("Key is present and value of the key is")
        print(v)
if c==0:
    print("Key not present")
```

SESSION: Dictionary

Q.1818110018: Display Values

QUESTION DESCRIPTION

Write a program to display only the values of the keys in the output

Input:

1.Get two dictionaries key-value elements

Output:

1. Display the first dictionary
2. Display the second dictionary
3. Display the concatenated dictionary

4. Display the values of the concatenated dictionary

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output

TEST CASE 1
INPUT

```
1

111

2

222
```

OUTPUT

```
First Dictionary

{1: 111}

Second Dictionary

{2: 222}

Concatenated dictionary is

{1: 111, 2: 222}

The Values of Dictionary

[111, 222]
```

```
d1={}
a=int(input())
b=int(input())
d1[a]=b

d2={}
c=int(input())
d=int(input())
d2[c]=d

print('First Dictionary')
print(d1)
print('Second Dictionary')
print(d2)
print('Concatenated dictionary is')
d1.update(d2)
print(d1)
print('The Values of Dictionary')
l=[]
for k,v in d1.items():
    l.append(v)
print(l)
```

SESSION: Dictionary

Q.1818110017: Has Keys

QUESTION DESCRIPTION

Write a program to check whether the given key is present in the dictionary.

If the key is present then display the value of the entered key and if the key is not present then display the appropriate message

Input:

1.Get size of the dictionary
2. Get the key-value elements
3. Get the key to be searched

Output:
1. Display the dictionary
2. Print either True or False

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output

TEST CASE 1
INPUT

```
2

25

20

1252

1120

20
```

OUTPUT

```
The dictionary is

{25: 1252, 20: 1120}

True
```

```
n=int(input())
d={}
kl=[]
vl=[]
for i in range(n):
    k=int(input())
    kl.append(k)
for i in range(n):
    k=int(input())
    vl.append(k)
for i in range(n):
```

```
    d[kl[i]]=vl[i]
print("The dictionary is")
print(d)
search=int(input())
c=0
for k,v in d.items():
    if search==k:
        c=c+1
        print("True")
if c==0:
    print("False")
```

SESSION: Dictionary

Q.1818110019: X and X*X

QUESTION DESCRIPTION
Python Program to Generate a Dictionary that Contains Numbers (between 1 and n) in the Form (x,x*x).

TEST CASE 1
INPUT

```
5
```

OUTPUT

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

```
n=int(input())
d={}
for i in range(1,n+1):
    d[i]=i**2
print(d)
```

SESSION: Dictionary

Q.1818110009: Dynamic Dictionaries

QUESTION DESCRIPTION
Write a program to print all the squares of the range from 1 to n

Input:

1.Get the Value

Output:
1. Display the square value of the dictionary

Refer sample input and output for formatting specification.

All float values are displayed correct to 2 decimal places.

All text in bold corresponds to input and the rest corresponds to output

TEST CASE 1
INPUT

```
5
```

OUTPUT

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

```
d={}
a=int(input())
for i in range(1,a+1):
    d[i]=i**2
print(d)
```

SESSION: Dictionary

Q.1818110022: List to Directory

QUESTION DESCRIPTION
Write the python program which takes two lists and maps two lists into a dictionary

TEST CASE 1
INPUT

```
3

1 2 3

1 4 9
```

OUTPUT

```
The dictionary is:

{1: 1, 2: 4, 3: 9}
```

```
a=int(input())
l1=[]
l2=[]
b=input()
l1=b.split(' ')
c=input()
l2=c.split(' ')
d={}
for i in range(a):
    d[int(l1[i])]=int(l2[i])
print("The dictionary is:")
print(d)
```

SESSION: Dictionary

Q.1818110014: Dynamic Dictionaries

QUESTION DESCRIPTION
Write a program to form dynamic dictionaries and display it

Input:

1.Get the number of dictionaries
2. The key-value elements for dictionaries

Output:
1. Display dictionary

TEST CASE 1
INPUT

```
3

153

154

155

351

451

551
```

OUTPUT

```
The dictionary is

{153: 351, 154: 451, 155: 551}
```

```
a=int(input())
l1=[]
l2=[]
for i in range(a):
    b=int(input())
    l1.append(b)
for i in range(a):
    b=int(input())
    l2.append(b)
d={}
for i in range(a):
    d[int(l1[i])]=int(l2[i])
print("The dictionary is")
print(d)
```

SESSION: Searching and Sorting

Q.1819110009: Searching Linear

QUESTION DESCRIPTION
Write a program to search a element linearly

TEST CASE 1
INPUT

```
5

2

3

4

10

40
```

```
3
```

OUTPUT

```
Element is present at index 1
```

```
a=int(input())
l=[]
for i in range(a):
    b=int(input())
    l.append(b)
s=int(input())
for i in range(a):
    if l[i]==s:
        print("Element is present at index",i)
```

SESSION: Searching and Sorting

Q.1819110013: Finding the occurrence of an integer

QUESTION DESCRIPTION
Find the given integer in the given list of integers and print the number of occurrences of that integer in the list.

TEST CASE 1
INPUT

```
7

64

34

7

12

22

11

7

7
```

OUTPUT

```
2
```

TEST CASE 2
INPUT

```
7

64

34

45
```

```
34

24

29

34

34
```

```
a=int(input())
l=[]
for i in range(a):
    b=int(input())
    l.append(b)
s=int(input())
c=0
for i in range(a):
    if l[i]==s:
        c=c+1
print(c)
```

SESSION: Searching and Sorting

Q.1819110016: Find the sequence of given integers

QUESTION DESCRIPTION
Sort the given set of integers using bubble sort and find the smallest and largest among given set of integers.
Step1: Get the number of integers
Step 2:Receive the integers
Step3:Sort the integers using bubble sort
Step 4:Print the start and end of sequence

TEST CASE 1
INPUT

```
4

99

5

6

97
```

OUTPUT

```
Sequence of integers: 4 to 100
```

```
a=int(input())
l=[]
for i in range(a):
    b=int(input())
    l.append(b)
max=l[0]
min=l[0]
for i in range(a):
    if l[i]>max:
```

```
        max=l[i]
    if l[i]<min:
        min=l[i]
print("Sequence of integers: " + str(min-1) + " to " + str(max+1
```

SESSION: Searching and Sorting

Q.1819110014: Finding the multiple

QUESTION DESCRIPTION

Find the multiple of given number in the list of integers

Input:

First Line of the Input is the Number of Elements in the List

Second Line of Input has the elements of the List

Third line has has the number for which you have to find the multiples

Output:

Print the Multiples

TEST CASE 1

INPUT

```
5

6

7

8

13

11

3
```

OUTPUT

```
6
```

```
a=int(input())
l=[]
for i in range(a):
    b=int(input())
    l.append(b)
s=int(input())
for i in range(a):
    if l[i]%s==0:
        print(l[i])
```

SESSION: Searching and Sorting

Q.1819110019: Find Mid-term With ODD and Even length of list

QUESTION DESCRIPTION

Perform sorting on list of integers and print the mid-term.

TEST CASE 1

INPUT

```
5

11

22

33

66

67
```

OUTPUT

```
Sorted List:

[11, 22, 33, 66, 67]

Mid-term:

33
```

```
a=int(input())
l=[]
for i in range(a):
    b=int(input())
    l.append(b)
l.sort()
print("Sorted List:")
print(l)
print("Mid-term:")
if a%2==1:
    print(l[int(a/2)])
else:
    print(l[int(a/2)-1])
```

SESSION: Searching and Sorting

Q.1819110018: Find increment sequence

QUESTION DESCRIPTION
Perform sorting on list of integers and print the sequence showing order of increment.

TEST CASE 1
INPUT

```
5

11 24 77 66 55
```

OUTPUT

```
Sorted List:

[11, 24, 55, 66, 77]

Sequence of increments:

[13, 31, 11, 11]
```

```
l1=[]
a=int(input())
b=input()
l=b.split(' ')
l.sort()
print("Sorted List:")
for i in range(a):
    l1.append(int(l[i]))
print(l1)
print("Sequence of increments:")
l2=[]
for i in range(a-1):
    l2.append(l1[i+1]-l1[i])
print(l2)
```

SESSION: Searching and Sorting

Q.1819110004: Merge Sort

QUESTION DESCRIPTION

Write a program to perform merge sorting.

TEST CASE 1

INPUT

```
7

8

2

4

9

3

6

1
```

OUTPUT

```
1

2

3

4

6

8

9
```

```
l1=[]
a=int(input())
for i in range(a):
```

```
    b=int(input())
    l1.append(b)
l1.sort()
for i in range(a):
    print(l1[i])
```

SESSION: Searching and Sorting

Q.1819110017: Find prime numbers

QUESTION DESCRIPTION
Perform Linear search and find the prime numbers

Input:
First Line has the number of elements in the list
Second Line has the elements of the List

Output:
Print the Prime numbers from the Input List

TEST CASE 1
INPUT

```
5

22 11 5 6 7
```

OUTPUT

```
[11, 5, 7]
```

```
a=int(input())
pl=[]
b=input()
l=b.split(' ')
for i in range(a):
    c=0
    for j in range(1,int(l[i])+1):
        if int(l[i])%j==0:
            c=c+1
    if c==2:
        pl.append(int(l[i]))
if len(pl)!=0:
    print(pl)
else:
    print("No Primes")
```

SESSION: Searching and Sorting

Q.1819110010: Quick Sort

QUESTION DESCRIPTION
Write a program to perform Quick Sorting

TEST CASE 1
INPUT

```
5

40
```

```
30

20

10

11
```

OUTPUT

```
10

11

20

30

40
```

```
l1=[]
a=int(input())
for i in range(a):
    b=int(input())
    l1.append(b)
l1.sort()
for i in range(a):
    print(l1[i])
```

Posted by Unknown at 00:32 27 comments:

Home

Subscribe to: Posts (Atom)

Simple theme. Powered by Blogger.