BCSE101E - Computer Programming: Python

Digital Footprint

Faculty In charge: JANANI T
Class No.: VL2021220107101

School of Computer Science and Engineering (SCOPE)
Vellore Institute of Technology
Vellore.

BCSE101E - Computer Programming: Python

Table of Contents

Ex.No.	Title	Date	Page No.
1	**M12_CSQ1**	19/10/2021	2
2	**M12_CSQ2**	19/10/2021	3
3	**M12_CSQ3**	19/10/2021	5
4	**M12_CSQ4**	19/10/2021	8
5	**M12_CSQ5**	19/10/2021	10
6	**M12_CSQ6**	19/10/2021	12
7	**M12_CSQ7**	19/10/2021	13
8	**M12_CSQ8**	19/10/2021	14
9	**M12_CSQ9**	19/10/2021	15
10	**M12_CSQ10**	19/10/2021	16
11	**M3_CSQ1**	19/10/2021	17
12	**M3_CSQ2**	19/10/2021	18
13	**M3_CSQ3**	19/10/2021	19
14	**M3_CSQ4**	20/19/2021	21
15	**M3_CSQ5**	20/10/2021	23
16	**M4_CSQ1**	16/11/2021	24
17	**M4_CSQ2**	16/11/2021	26
18	**M4_CSQ3**	16/11/2021	28
19	**M4_CSQ4**	17/11/2021	30
20	**M4_CSQ5**	18/11/2021	32
21	**M5_CSQ1**	23/11/2021	34
22	**M5_CSQ2**	26/11/2021	35
23	**M5_CSQ3**	26/11/2021	36
24	**M5_CSQ4**	30/11/2021	38
25	**M5_CSQ5**	30/11/2021	39
26	**M6_CSQ1**	8/12/2021	40
27	**M6_CSQ2**	10/12/2021	41
28	**M6_CSQ3**	22/12/2021	42
29	**M6_CSQ4**	31/12/2021	44
30	**M6_CSQ5**	31/12/2021	46
31	**M7_CSQ1**	4/1/2022	48
32	**M7_CSQ2**	5/1/2022	49
33	**M7_CSQ3**	5/1/2022	50
34	**M7_CSQ4**	7/1/2022	51
35	**M7_CSQ5**	7/1/2022	52
36	**M7_CSQ6**	7/1/2022	54

Signature of the student (Digital)
Abhinav Dinesh Srivatsa

[Ex. No. M12_CSQ1]

AIM

To calculate the age of a housefly in seconds, given the number of days the housefly lived.

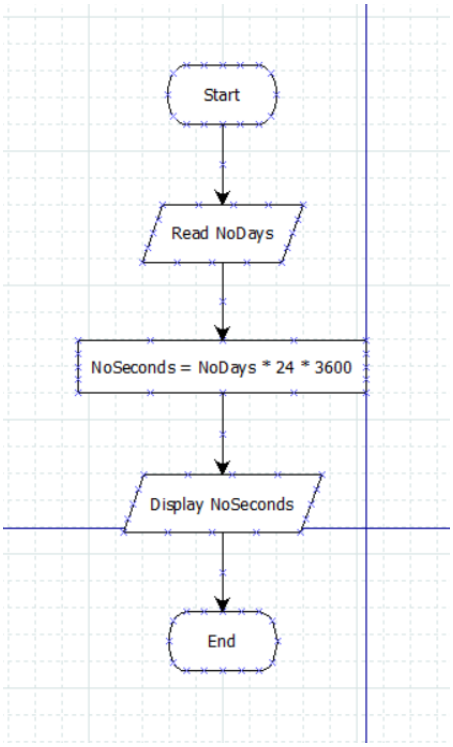
Algorithm / Pseudocode

Read NoDays

Calculate NoSeconds = NoDays * 24 * 3600

Display NoSeconds

Flowchart



Program Code

```
days = int(input("Enter days the fly has been alive: "))
seconds = days * 24 * 3600
print(f"The fly has lived for {seconds} seconds")
```

Output

```
Enter days the fly has been alive: 21
The fly has lived for 1814400 seconds
```

[Ex. No. M12_CSQ2]

AIM

To calculate the total quantity of milk from ‘n’ farm.

Algorithm / Pseudocode

Read NoFarms

Read L1, L2, …

Read M1, M2, …

Calculate TotalM = M1 + M2 + …

Calculate TotalL = L1 + L2 + …

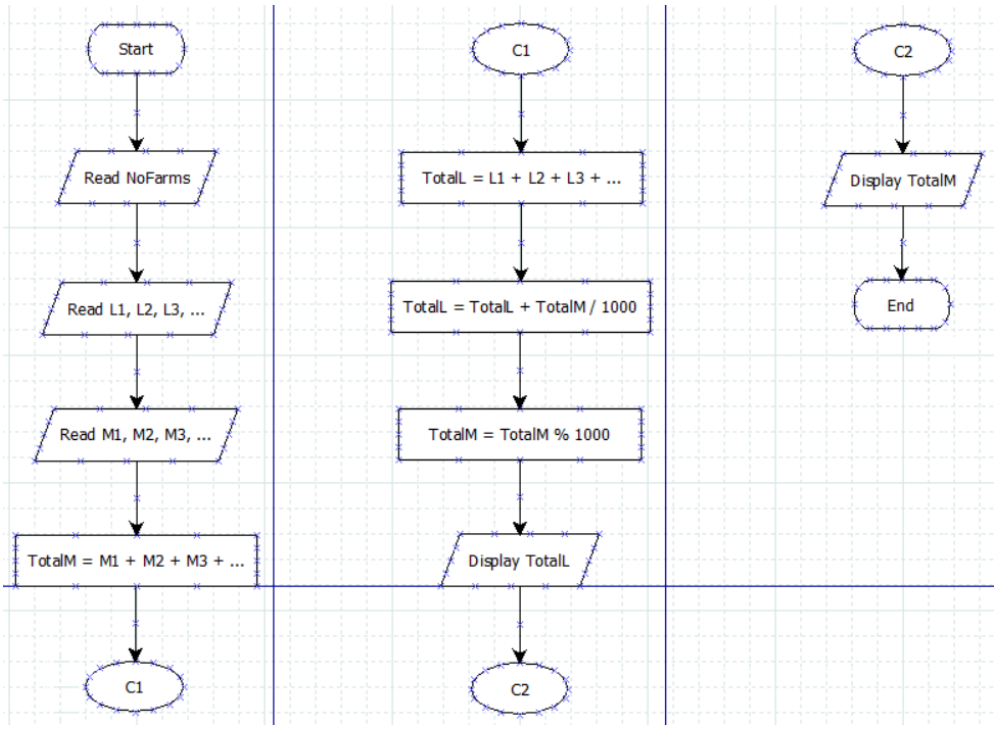
Calculate TotalL = TotalL + TotalM / 1000

Calculate TotalM = TotalM % 1000

Display TotalL

Display TotalM

Flowchart



Program Code

```
farms = int(input("Enter number of farms: "))
litre_sum = 0
milli_sum = 0
for i in range(farms):
    milk = input(f"Farm {i + 1}: ").split(" ")
    litre_sum += int(milk[0])
    milli_sum += int(milk[1])

# carrying over the extra milliliters to liters
litre_sum += int(milli_sum / 1000)
milli_sum %= 1000
print(f"Total is {litre_sum} liters and {milli_sum} milliliters")
```

Output

```
Enter number of farms: 3
Farm 1: 1 100
Farm 2: 2 200
Farm 3: 3 300
Total is 6 liters and 600 milliliters
```

[Ex. No. M12_CSQ3]

AIM

To convert a two-digit number to its corresponding Roman numeral.

Algorithm / Pseudocode

Read Number

Tens = Number // 10

Units = Number % 10

If Tens is between 0 and 4, then concatenate Tens amount of 'X'

Else if Tens is 4, concatenate 'XL'

Else if Tens is between 4 and 9, then concatenate 'L' + (Tens – 5) amount of 'X'

Else if Tens is 9, then concatenate 'XC'

Else if Tens is 10, then concatenate 'C'

If Units is between 0 and 4, then concatenate Units amount of 'I'

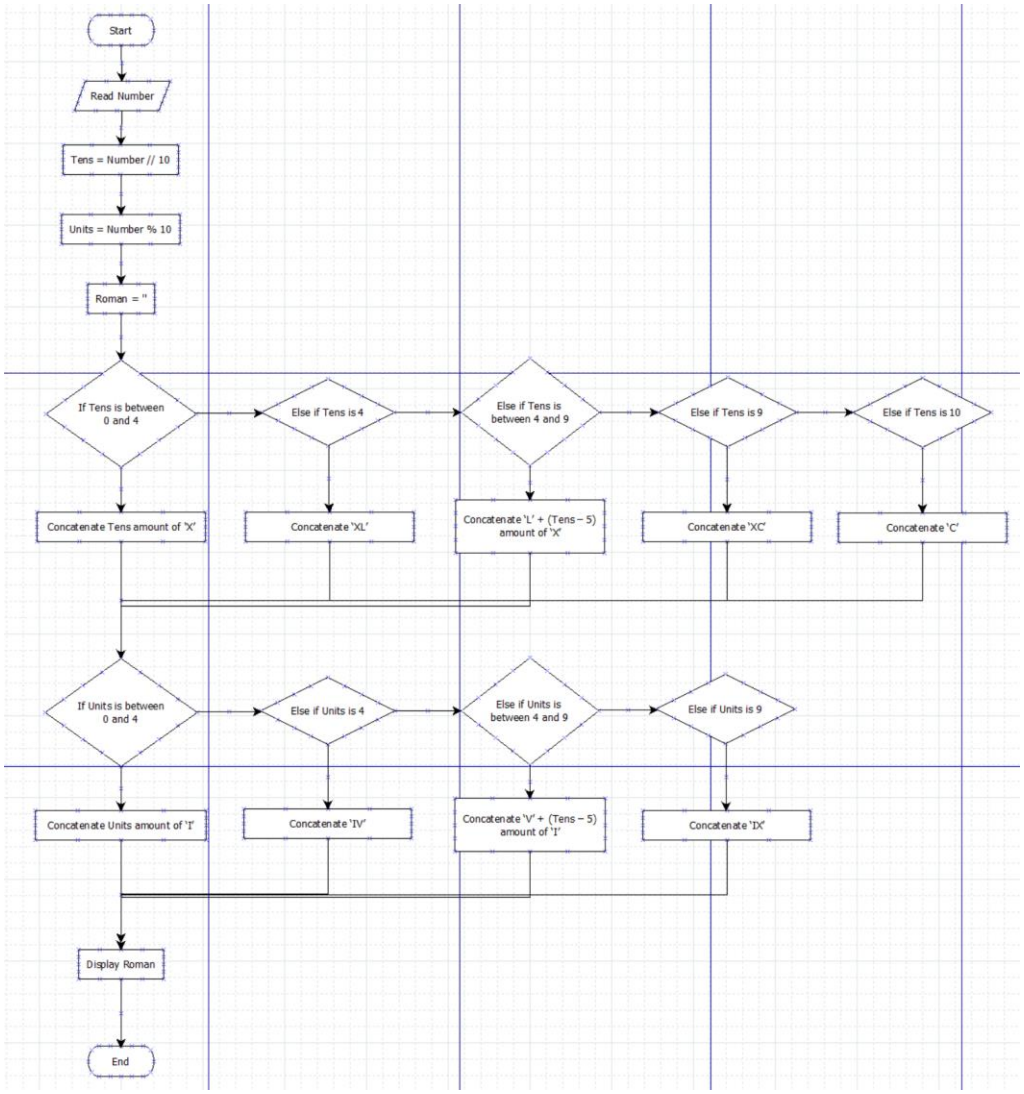
Else if Units is 4, concatenate 'IV'

Else if Units is between 4 and 9, then concatenate 'V' + (Tens – 5) amount of 'I'

Else if Units is 9, then concatenate 'IX'

Display Roman

Flowchart



Program Code

```
number = int(input("Enter number: "))
roman = ""

tens = number // 10
units = number % 10

if tens > 0 and tens < 4:
    roman += (tens * 'X')
elif number / 10 == 4:
    roman += 'XL'
elif tens > 4 and tens < 9:
    roman += 'L' + ((tens - 5) * 'X')
```

```
elif tens == 9:
    roman += 'XC'
elif tens == 10:
    roman += 'C'

if units > 0 and units < 4:
    roman += (units * 'I')
elif units == 4:
    roman += 'IV'
elif units > 4 and units < 9:
    roman += 'V' + ((units - 5) * 'I')
elif units == 9:
    roman += 'IX'

print(f"{number} in Roman numerals is {roman}")
```

Output

```
Enter number: 88
88 in Roman numerals is LXXXVIII
```

[Ex. No. M12_CSQ4]

AIM

To find the sum of the real and imaginary parts of two complex numbers.

Algorithm / Pseudocode

Read Vector1, Vector2

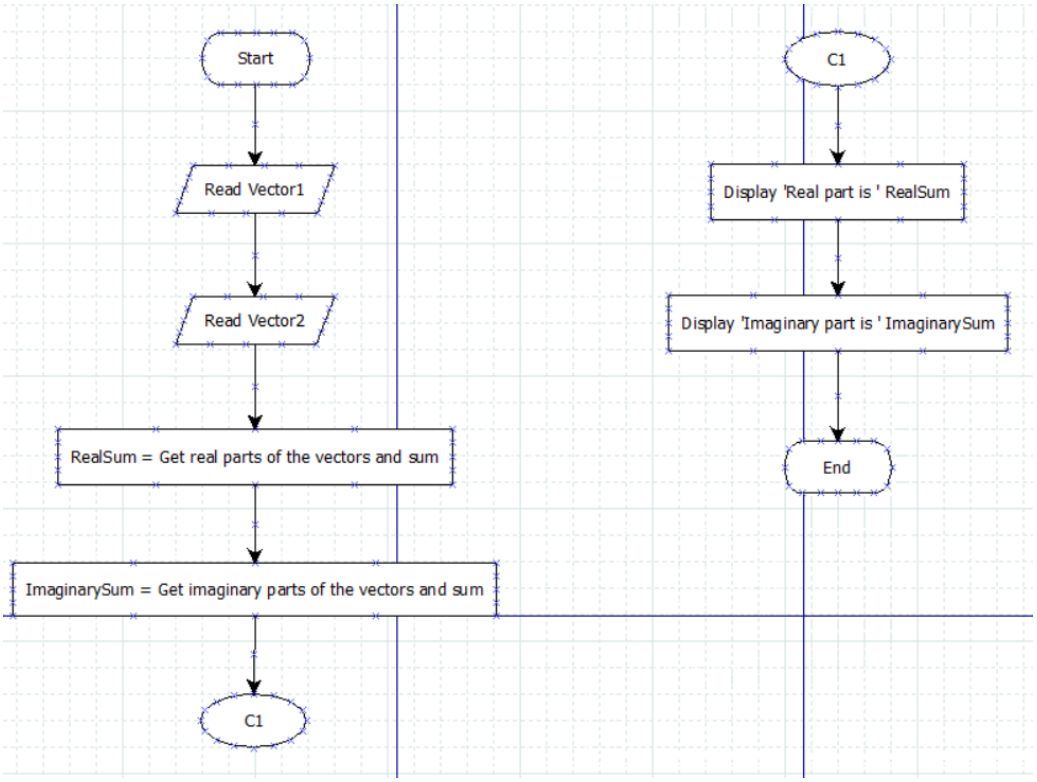
Calculate RealSum = Get real parts of the vectors and sum

Calculate ImaginarySum = Get imaginary parts of the vectors and sum

Display 'Real part is ' RealSum

Display 'Imaginary part is ' ImaginarySum

Flowchart



Program Code

```
vector1 = complex(input("Enter vector 1: "))
vector2 = complex(input("Enter vector 2: "))

real_sum = vector1.real + vector2.real
imaginary_sum = vector1.imag + vector2.imag

print(f"Real part is: {int(real_sum)}")
print(f"Imaginary part is: {int(imaginary_sum)}")
```

Output

```
Enter vector 1: 10j
Enter vector 2: 1+1j
Real part is: 1
Imaginary part is: 11
```

[Ex. No. M12_CSQ5]

AIM

To convert a given integer to binary, octal and hexadecimal.

Algorithm / Pseudocode

Read Number

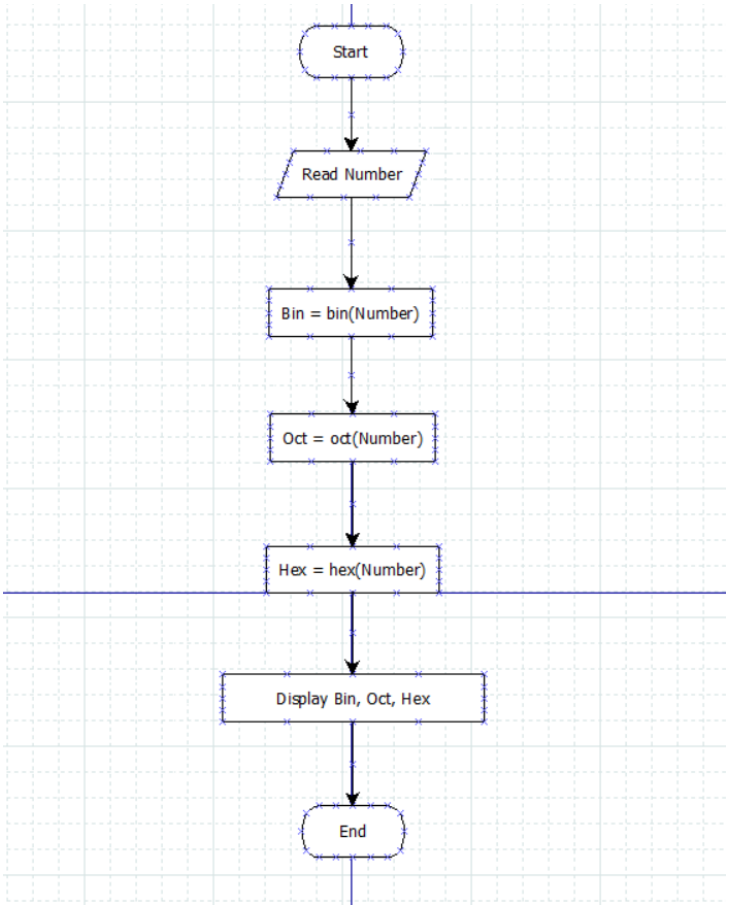
Calculate Bin = bin(Number)

Calculate Oct = oct(Number)

Calculate Hex = hex(number)

Display Bin, Oct, Hex

Flowchart



Program Code

```
num = int(input())

print(bin(num), oct(num), hex(num))
```

Output

```
35
0b100011 0o43 0x23
```

[Ex. No. M12_CSQ6]

AIM

To calculate the area of a triangle given its three sides.

Algorithm / Pseudocode

Read S1, S2, S3

Calculate Semiper = (S1 + S2 + S3) / 2

Calculate Area = (Semiper * (Semi per – S1) * (Semiper – S2) * (Semiper – S3)) ^ 0.5

Display Area

Program Code

```
s1 = int(input())
s2 = int(input())
s3 = int(input())

semi_per = (s1 + s2 + s3) / 2
area = (semi_per * (semi_per - s1) * (semi_per - s2) * (semi_per - s3)) **
0.5

print("{:.2f}".format(area))
```

Output

```
5
6
7
14.70
```

[Ex. No. M12_CSQ7]

AIM

To convert a string to an integer.

Algorithm / Pseudocode

Read String

Calculate Integer = int(Number)

Display Integer

Program Code

```
string = input()

integer = int(string)

print(integer)
```

Output

```
345
345
```

[Ex. No. M12_CSQ8]

AIM

To convert an integer to a string.

Algorithm / Pseudocode

Read Integer

Calculate String = str(Integer)

Display String

Program Code

```
integer = int(input())

string = str(integer)

print(string)
```

Output

```
467
467
```

[Ex. No. M12_CSQ9]

AIM

To convert Fahrenheit to Celsius using a lambda function.

Algorithm / Pseudocode

Celsius = lambda function of Fahrenheit

Read Fahrenheit

Display Celsius(Fahrenheit)

Program Code

```
c = lambda f: (f - 32) * 5 / 9

f = float(input())

print("{:.2f}".format(c(f)))
```

Output

```
212
100.00
```

[Ex. No. M12_CSQ10]

AIM

To swap the values of two given variables.

Algorithm / Pseudocode

Read Number1

Read Number2

Number1, Number2 = Number2, Number1

Display Number1, Number2

Program Code

```
number1 = input()
number2 = input()

number1, number2 = number2, number1

print(number1, number2)
```

Output

```
5
10
10 5
```

[Ex. No. M3_CSQ1]

AIM

To calculate the amount of tax given a yearly salary.

Algorithm / Pseudocode

Read Income

If Income > 1000000, then display Income * 0.04

Else if Income > 500000, then display Income * 0.02

Else display 0

Program Code

```
income = float(input())

if income > 1000000:
    print(int(income * 0.04))
elif income > 500000:
    print(int(income * 0.02))
else:
    print(0)
```

Output

```
1800000
72000
```

[Ex. No. M3_CSQ2]

AIM

To calculate the price of a taxi ride based on the amount of distance driven.

Algorithm / Pseudocode

Read Distance

If Distance < 0, then display 'Enter Positive Number Only'

Else,

If Distance <= 10, then display Distance * 15

Else if Distance <= 90, then display 150 + Distance * 8

Else display 150 + 720 + Distance * 6

Program Code

```
s = int(input())
if s < 0:
    print('Enter Positive Number Only')
else:
    if s <= 10:
        print(s * 15)
    elif s <= 90:
        print(150 + (s - 10) * 8)
    else:
        print(150 + 720 + (s - 100) * 6)
```

Output

```
120
990
```

[Ex. No. M3_CSQ3]

AIM

To calculate the amount of marks a student scores based on their original marks.

Algorithm / Pseudocode

Read Marks

If Marks <= 0, then display 'Enter appropriate Mark'

Else,

    Read Session

    If marks >= 80, then

        If Session is 'Theory', then display Marks * 1.08

        If Session is 'Lab', then display Marks * 1.06

    If marks >= 60, then

        If Session is 'Theory', then display Marks * 1.06

        If Session is 'Lab', then display Marks * 1.04

    If marks >= 40, then

        If Session is 'Theory', then display Marks * 1.04

        If Session is 'Lab', then display Marks * 1.02

    Else,

        If Session is 'Theory', then display Marks

        If Session is 'Lab', then display Marks

Program Code

```
marks = float(input())
if marks <= 0:
    print('Enter appropriate Mark')
else:
    session = input()
    if marks >= 80:
        if session == 'T':
            print('{:.2f}'.format(marks * 1.08))
        if session == 'L':
            print('{:.2f}'.format(marks * 1.06))
    elif marks >= 60:
        if session == 'T':
            print('{:.2f}'.format(marks * 1.06))
        if session == 'L':
            print('{:.2f}'.format(marks * 1.04))
    elif marks >= 40:
        if session == 'T':
            print('{:.2f}'.format(marks * 1.04))
        if session == 'L':
            print('{:.2f}'.format(marks * 1.02))
    else:
        if session == 'T':
            print('{:.2f}'.format(marks))
        if session == 'L':
            print('{:.2f}'.format(marks))
```

Output

```
92
T
99.36
```

[Ex. No. M3_CSQ4]

AIM

To display all alternate number factorials below the number given.

Algorithm / Pseudocode

Read Number

If Number <= 0, then display 'Enter only positive number'

Else,

    If Number is even, then Start = 2

    Else Start = 1

    While Start <= Number

        Find factorial of Start and assign to Fact

        Increment Start by 2

        Display Start, Fact

Program Code

```
num = int(input())

if num < 0:
    print("Enter only positive number")
else:
    if num % 2 == 0:
        i = 2
    else:
        i = 1

    while i <= num:
        fact = 1
        j = 2
        while j <= i:
            fact *= j
            j += 1
        print(i, fact)
        i += 2
```

Output

```
6
2 2
4 24
6 720
```

[Ex. No. M3_CSQ5]

AIM

To find the sum of the digits of a given number as a single digit.

Algorithm / Pseudocode

Read Number

While Number % 10 != 0

    Tens = Number // 10

    Units = Number % 10

    Number = Tens + Units

Display Number

Program Code

```
num = int(input())

while num // 10 != 0:
    num = num // 10 + num % 10

print(num)
```

Output

```
123456789
9
```

[Ex. No. M4_CSQ1]

AIM

To find the point in a list where the right elements and left elements sums are equal.

Algorithm / Pseudocode

Read Number

Create list Numbers

Read inputs in for loop and assign to Numbers

Loop I from 1 to Number – 1

    Initialise Left as the left side in Numbers

    Initialise Right as the right side in Numbers

    If the sum of Left = sum of Right

        Display I (Loop Variable)

If nothing has been displayed

    Display 0

Program Code

```
n = int(input())
numbers = []

for i in range(n):
    numbers.append(int(input()))

flag = False
for i in range(1, n - 1):
    left = numbers[:i]
    right = numbers[i + 1:]

    if sum(left) == sum(right):
        flag = True
        print(i)

if(not flag):
    print(0)
```

Output

```
6
4
3
100
2
3
2
2
```

[Ex. No. M4_CSQ2]

AIM

To find the answer of an RPN (postfix) notation expression.

Algorithm / Pseudocode

Read Number

Create list RPN

Read inputs in for loop and assign to RPN, as integer if number and string if character

Loop while RPN's length is not 1

Initialise Triple as the i^{th}, $i+1^{th}$ and $i+2^{th}$ elements in RPN

If the third element is an integer, then take the next triplet and continue

Else,

Initialise Op = the third element in Triple

If Op = '/', then divide the first and second elements of Triple

If Op = '*', then multiply the first and second elements of Triple

If Op = '+', then add the first and second elements of Triple

If Op = '-', then subtract the first and second elements of Triple

Replace the corresponding triple in with the calculated number

Start from the first index again and continue

Print the first index of RPN

Program Code

```
n = int(input())
rpn = []

for i in range(n):
    inp = input()
    try:
        inp = int(inp)
    except:
```

```
            pass
        finally:
            rpn.append(inp)

    i = 0
    while len(rpn) != 1:
        triple = rpn[i: i + 3]

        if type(triple[2]) is int:
            i += 1
            continue
        else:
            op = triple[2]
            if op == '/':
                sum = triple[0] / triple[1]
            elif op == '*':
                sum = triple[0] * triple[1]
            elif op == '+':
                sum = triple[0] + triple[1]
            elif op == '-':
                sum = triple[0] - triple[1]

            rpn[i: i+3] = [sum]
            i = 0

    print(int(rpn[0]))
```

Output

```
5
2
1
+
3
*
9
```

[Ex. No. M4_CSQ3]

AIM

To display a list with the primary key and the corresponding matching value from a list of lists.

Algorithm / Pseudocode

Read Number

Create list Matrix

Read inputs in for loop and assign to a list New_Mat, as integer if number and string if character

Add this New_Mat to Matrix

Read Primary_Index

Read Primary_Value

Try converting Primary_Value to string, if not continue

Loop through matrix with indices as i being row elements and j being column indices in i

    If j = Primary_Index and i[j] = Primary_Value, then display i

Program Code

```
n = int(input())
m = int(input())
matrix = []

for i in range(n):
    new_mat = []
    for j in range(m):
        inp = input()
        try:
            inp = int(inp)
        except:
            pass
        finally:
            new_mat.append(inp)
    matrix.append(new_mat)

primary_index = int(input())
```

```
primary_value = input()
try:
    primary_value = int(primary_value)
except:
    pass

for i in matrix:
    for j in range(len(i)):
        if j == primary_index and i[j] == primary_value:
            print(i)
```

Output

```
2
5
John
smith
1234
B+
10.03
Rockey
Jr
6789
A+
40.03
2
1234
['John', 'smith', 1234, 'B+', '10.03']
```

[Ex. No. M4_CSQ4]

AIM

To display a tuple summarizing the costs per department entered.

Algorithm / Pseudocode

Read Number

Initialise Costs as ()

Loop i from 0 to Number

    Read No_Items

    Initialise Info as ()

    Loop j from 0 to No_Items and record values in Info

    Initialise Sum as 0

    Loop through Items and add all integers to sum

    Add the department and cost to Costs

Display Costs

Program Code

```
n = int(input())
costs = ()
for i in range(n):
    items = int(input())
    info = ()
    for j in range(items):
        info += (input(), )
    sum = 0
    for j in info:
        try:
            sum += int(j)
        except:
            pass
    if i == n - 1:
        costs += ((info[0], sum))
    else:
        costs += ((info[0], sum), )
```

```
print(costs)
```

Output

```
2
7
Education
Primary
50
Secondary
25
Higher
20
7
Defense
Army
25
AirForce
40
Navy
45
('Education', 95, 'Defense', 110)
```

[Ex. No. M4_CSQ5]

AIM

To find which students failed in a particular subject.

Algorithm / Pseudocode

Read No_Students

Initialise Students as list

Loop i from 0 to No_Students

    Initialise Marks as dictionary

    Read Subject and Marks and assign to dictionary Marks

    Append the Marks to Students list

Display Students

Initialise Failed as dictionary

Initialise Total to calculate total failures

Loop through Students and check if any score is less than 50

    If true, then add the student and subject to Failed and increment Total

Display Failed and Total

Program Code

```
no_students = int(input())
students = []
for i in range(no_students):
    marks = {}
    no_marks = int(input())
    for j in range(no_marks):
        m = input()
        marks[m] = int(input())
    students.append(marks)

print(students)

failed = {}
total = 0
```

```
#getting fails
for i in students:
    flag = True
    for j in i:
        if j not in failed.keys():
            failed[j] = 0
        if i[j] < 50:
            if(flag):
                total += 1
                flag = False
            failed[j] += 1

for i in failed:
    print(i)
    print(failed[i])
print(total)
```

Output

```
3
3
m1
50
m2
40
m3
75
3
m2
49
m3
35
m4
54
3
m1
77
m2
84
m4
51
[{'m1': 50, 'm2': 40, 'm3': 75}, {'m2': 49, 'm3': 35, 'm4': 54}, {'m1': 77, 'm2': 84, 'm4': 51}]
m1
0
m2
2
m3
1
m4
0
2
```

[Ex. No. M5_CSQ1]

AIM

To swap a string uppercase letters to lower case and vice versa.

Algorithm / Pseudocode

Read Sentence

Loop through characters in Sentence as i

If i is lower case, then display it's upper case

Else if i is upper case, then display it's lower case

Else, display i

Program Code

```
sen = input()

for i in sen:
    if i.islower():
        print(i.upper(), end = '')
    elif i.upper():
        print(i.lower(), end = '')
    else:
        print(i, end = '')
```

Output

```
aBcDeFghiJKlM
AbCdEfGHIjkLm
```

[Ex. No. M5_CSQ2]

AIM

To check if a string is a good word or a bad word. Good words have no repeat characters, while bad words do.

Algorithm / Pseudocode

Read Word and convert it to lower case

Initialise Letters as the characters of Word

Initialise Verdict as 'GOOD'

Loop from 0 to length of Letters as i

    Initialize Letter1 as Letters[i]

    Loop Letter2 as Letters past Letter1

        If Letter1 equals Letter2, the assign verdict as'BAD'

        Break

Display Verdict

Program Code

```
word = input().lower()
letters = list(word)
verdict = 'GOOD'

for i in range(len(letters)):
    letter1 = letters[i]
    for letter2 in letters[i + 1:]:
        if letter1 == letter2:
            verdict = 'BAD'
            break

print(verdict)
```

Output

```
START
BAD
```

[Ex. No. M5_CSQ3]

AIM

To check if a word exists in a sentence.

Algorithm / Pseudocode

Read Sentence and convert it to lower case

Read Word and convert it to lower case

Initialise Length as length of Word

Initialise Found as False

Loop through 0 to length of Sentence as i

    Initialise Replace as replacing Sentence of i to i + Length with Word

    If Replace and Sentence are equal, then

        Assign Found as True

        Display i

        Display i + Length

        Break

If Found is False, the display 'Not Found'

Program Code

```
sen = input().lower()
word = input().lower()
length = len(word)
found = False

for i in range(len(sen)):
    if sen.replace(sen[i:i + length], word) == sen:
        found = True
        print(i)
        print(i + length)
        break

if not found:
```

```
        print('Not Found')
```

Output

```
The quick brown fox jumps over the lazy dog
Fox
16
19
```

[Ex. No. M5_CSQ4]

AIM

To find all the students names and marks from an input string.

Algorithm / Pseudocode

Read Sentence

Initialise Words by taking out commas and splitting Sentence by spaces

Initialise Students and Marks as a list

Loop through Word in Words

    If Word starts with a capital letter, then append Word to Students

    Else if Word is a number, then append Word to Marks

Display Marks

Display Students

Program Code

```
import re

sen = input()
words = sen.replace(',', '').split(' ')
students = []
marks = []

for word in words:
    if re.match('^[A-Z]', word):
        students.append(word)
    elif re.match('[0-9]', word):
        marks.append(word)

print(marks)
print(students)
```

Output

```
Rahul got 75 marks, Vijay got 55 marks, whereas Subbu got 98 marks
['75', '55', '98']
['Rahul', 'Vijay', 'Subbu']
```

[Ex. No. M5_CSQ5]

AIM

To find all the words starting with vowels in a sentence.

Algorithm / Pseudocode

Read Sentence

Initialise Words by splitting Sentence by spaces

Loop through Word in Words

    If Word starts with a vowel (a, e, i, o, u, A, E, I, O, U), then display Word

Program Code

```
import re

sen = input()
words = sen.split(' ')

for word in words:
    if re.match('^[aeiouAEIOU]', word):
        print(word)
```

Output

```
An apple for a day keeps the doctor away
An
apple
a
away
```

[Ex. No. M6_CSQ1]

AIM

To find the substring of a word relative to a position given.

Algorithm / Pseudocode

Substring_Position(String S1, String S2, Integer N)

    Loop I from 0 to length of S1

        If slice of S1 of I to I + length of S2 equals S2, then return I – N

    Return 'NotAvailable'

Read S1

Read S2

Read position N

Display Substring_Position(S1, S2, N)

Program Code

```
def substring_position(s1, s2, n):
    for i in range(len(s1)):
        if s1[i : i + len(s2)] == s2:
            return i - n

    return 'NotAvailable'

s1 = input()
s2 = input()
n = int(input())

print(substring_position(s1, s2, n))
```

Output

```
mask
ask
3
-2
```

[Ex. No. M6_CSQ2]

AIM

To find the number of chicke3ns and rabbits being given the number of heads and tails.

Algorithm / Pseudocode

Get_Animals(Integer M, Integer N)

Return N/2 – M, 2M – N/2

Read Count as a tuple

Initialise M = Count[0]

Initialise N = Count[1]

Display Get_Animals(M, N)

Program Code

```
def get_animals(m, n):
    # equations:
    #    x  +  y = m
    #    4x + 2y = n
    #
    # =>  x      = n/2 - m
    # =>  y      = m - x = 2m - n/2

    return n//2 - m, 2*m - n//2

count = eval(input())
m = count[0]
n = count[1]

print(get_animals(m, n))
```

Output

```
(35,94)
(12, 23)
```

[Ex. No. M6_CSQ3]

AIM

To find the mean marks of students and categorise them into groups of scoring more, equal, or less to the average.

Algorithm / Pseudocode

Scores(Dictionary Student_Marks)

Calculate mean marks of Student_Marks and assign to Mean_Marks

Initialise lists Above, Mean, Below

Loop through items in Student_Marks with I as students and J as marks

If J > Mean_Marks, then append I to Above

Else if J = Mean_Marks, append I to Mean

Else, append I to Below

Return Mean, Above, Below

Read Student_Marks

Mean, Above, Below = Scores(Student_Marks)

If Mean contains more than 0 elements, then print Mean joined by commas

If Above contains more than 0 elements, then print Above joined by commas

If Below contains more than 0 elements, then print Below joined by commas

Program Code

```
def scores(student_marks):
    marks = student_marks.values()
    mean_marks = sum(marks) // len(marks)
    above = []
    mean = []
    below = []
    for i, j in student_marks.items():
        if j > mean_marks:
            above.append(i)
```

```
            elif j == mean_marks:
                mean.append(i)
            else:
                below.append(i)
        return mean, above, below

student_marks = eval(input())
mean, above, below = scores(student_marks)
if len(mean) != 0:
    print(', '.join(mean))
if len(above) != 0:
    print(', '.join(above))
if len(below) != 0:
    print(', '.join(below))
```

Output

```
{'stud1':40,'stud2':60,'stud3':80}
stud2
stud3
stud1
```

[Ex. No. M6_CSQ4]

AIM

To create a quiz with two files – Questions.txt and Answers.txt

Algorithm / Pseudocode

Initialise Questions by reading Questions.txt

Initialise Answers by reading Answers.txt

Display Questions

Read User_Answers as input

Initialise I as 0

Initialise Sum as 0

Loop through lines in Answer as Line

    If the corresponding User_Answer and Line are the same, then add 50 to Sum

    Increment I by 1

Display Sum

Program Code

```
questions = open('Questions.txt', 'r')
answers = open('Answers.txt', 'r')
print(questions.read())
user_ans = input().split(' ')
i = 0
sum = 0
for line in answers.readlines():
    if user_ans[i] == line.strip()[-1].lower():
        sum += 50
    i += 1
print(sum)
```

Output

```
1. Who is the prime minister of India?
A. Narendra Modi B. Shivraj Patil
2. Who is the President of USA?
A. Donald Trump B. Joe Biden
a b
100
```

[Ex. No. M6_CSQ5]

AIM

To find the number of unique words in a sentence

Algorithm / Pseudocode

Duplicate_Characters(String Sen):

    Initialise Words by splitting Sen by spaces

    Initialise Unique_Words as list

    Loop through Words with Word

        If Word is not in Unique_Words, then add Word to Unique_Words

    Initialise Word_Count as dict

    Loop through a sorted Unique_Words as Unique

        Initialise Count as 0

        Loop through Words as Word

            If Word = Unique, then increment count by 1

        Word_Count of Unique = Count

    Return Word_Count

Read Sen as input

Initialise Dict as Duplicate_Characters(Sen)

Loop through Dict and print the word counts

Program Code

```
def duplicate_characters(sen):
    words = sen.split(' ')
    unique_words = []
    for word in words:
        if word not in unique_words:
            unique_words.append(word)
    word_count = {}
    for unique in sorted(unique_words):
```

```
            count = 0
            for word in words:
                if unique == word:
                    count += 1
            word_count[unique] = count
            count = 0
        return word_count


sen = input()
dict = duplicate_characters(sen)
for i, j in dict.items():
    print(f'{i}:{j}')
```

Output

```
It is true for all that that that that refers to is not the same that that that to refers to
It:1
all:1
for:1
is:2
not:1
refers:2
same:1
that:7
the:1
to:3
true:1
```

[Ex. No. M7_CSQ1]

AIM

To find the mean, standard deviation and variance of a list given

Algorithm / Pseudocode

Import numpy as NP

Initialise Array as NP array of evaluated input

Calculate Mean as NP mean of Array

Calculate Standard_Deviation as NP std of Array

Calculate Variance as NP var of Array

Display Mean

Display Standard_Deviation

Display Variance

Program Code

```
import numpy as np

array = np.array(eval(input()))
mean = np.mean(array)
std_dev = np.std(array)
var = np.var(array)
print(mean)
print(std_dev)
print(var)
```

Output

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
4.5
2.8722813232690143
8.25
```

[Ex. No. M7_CSQ2]

AIM

To sort an array by class then height for given data

Algorithm / Pseudocode

Import numpy as NP

Initialise Students as given data

Initialise Data_Types as the data types of students

Reinitialise Students as a structured array of students with datatypes as Data_Types

Reinitialise Students as NP sort of Students, while ordering by class and then height

Display Students

Program Code

```
import numpy as np

students = [('john', 164.5, 'XA'), ('mark', 188.3, 'XB'), ('zack', 178.7,
'XB')]
data_types = [('name', 'S10'), ('height', float), ('class', 'S4')]
students = np.array(students, dtype = data_types)
students = np.sort(students, order = ['class', 'height'])
print(students)
```

Output

```
[(b'john', 164.5, b'XA') (b'zack', 178.7, b'XB') (b'mark', 188.3, b'XB')]
```

[Ex. No. M7_CSQ3]

AIM

To develop a NumPy program to sort the student id with increasing height of the students from given students id and height. Print the integer indices that describes the sort order by multiple columns and the sorted data

Algorithm / Pseudocode

Import numpy as NP

Initialise Names as NP array of random names

Initialise Heights as NP array of random heights with same length as Names

Initialise Sort as NP lexsort of (Names, Heights)

Loop through Sort as I

    Display Names of I, Heights of I

Program Code

```
import numpy as np

names = np.array(['Mark', 'Sam', 'Henry', 'John', 'Xander', 'Hornato',
'Wade', 'Xenos', 'Barry', 'Rich'])
heights = np.array([145.5, 193.2, 178.4, 135.9, 169.8, 159.3, 178.4, 183.2,
176.7, 165.3])
sort = np.lexsort((names, heights))
for i in sort:
    print(f'{names[i]}, {heights[i]}')
```

Output

```
John, 135.9
Mark, 145.5
Hornato, 159.3
Rich, 165.3
Xander, 169.8
Barry, 176.7
Henry, 178.4
Wade, 178.4
Xenos, 183.2
Sam, 193.2
```

[Ex. No. M7_CSQ4]

AIM

To design a Pandas program to join the two given data frames along rows and assign all data

Algorithm / Pseudocode

Import pandas as PD

Initialise Data1 as dictionary of random data

Initialise Data2 as dictionary of random data

Initialise DF1 as PD dataframe of Data1

Initialise DF2 as PD dataframe of Data2

Initialise DF3 as PD concat of [DF1, DF2]

Display DF3

Program Code

```
import pandas as pd

data1 = {'regno': ['21BDS0342', '21BDS0343'], 'studname': ['ghi', 'jkl'],
'cgpa': [5.6, 10]}
data2 = {'regno': ['21BDS0340', '21BDS0341'], 'studname': ['abc', 'def'],
'cgpa': [8.5, 9.2]}
DF1 = pd.DataFrame(data1)
DF2 = pd.DataFrame(data2)
DF3 = pd.concat([DF1, DF2])
print(DF3)
```

Output

```
       regno studname  cgpa
0  21BDS0342      ghi   5.6
1  21BDS0343      jkl  10.0
0  21BDS0340      abc   8.5
1  21BDS0341      def   9.2
```

[Ex. No. M7_CSQ5]

AIM

To write a Pandas program to convert all the string values to upper, lower cases in each pandas series. Also find the length of the string values

Algorithm / Pseudocode

Import pandas as PD

Read Strings as input

Reinitialise Strings as Strings split by space

Reinitialise Strings as PD series of Strings

Set Strings' name as 'Text'

Initialise Upper as Strings' strings as upper case

Set Upper's name as 'Upper Case'

Initialise Lower as Strings' strings as lower case

Set Lower's name as 'Lower Case'

Initialise Length as Strings' strings length

Set Length's name as 'Length'

Initialise DF as PD concat of [Strings, Upper, Lower, Length] concatenated horizontally

Display DF

Program Code

```
import pandas as pd

strings = input()
strings = strings.split(' ')
strings = pd.Series(strings)
strings.name = 'Text'

upper = strings.str.upper()
upper.name = 'Upper Case'

lower = strings.str.lower()
```

```
lower.name ='Lower Case'

length = strings.str.len()
length.name = 'Length'

DF = pd.concat([strings, upper, lower, length], axis = 1)
print(DF)
```

Output

Lorem ipsum dolor sit amet. Aut rerum eaque est quisquam deleniti cum quia consequuntur et saepe dolor et galisum voluptas aut harum galisum ex sequi voluptatibus. Sed rerum voluptas est quia consequatur vel nihil architecto galisum iure aut sint explicabo. Ea mollitia amet aut exercitationem internos sed dolorem quaerat id earum suscipit. Et mollitia repudiandae in omnis totam id ratione voluptatem ex dolores delectus nam commodi provident.

	Text	Upper Case	Lower Case	Length
0	Lorem	LOREM	lorem	5
1	ipsum	IPSUM	ipsum	5
2	dolor	DOLOR	dolor	5
3	sit	SIT	sit	3
4	amet.	AMET.	amet.	5
...	...	...	...	...
62	dolores	DOLORES	dolores	7
63	delectus	DELECTUS	delectus	8
64	nam	NAM	nam	3
65	commodi	COMMODI	commodi	7
66	provident.	PROVIDENT.	provident.	10

67 rows × 4 columns

[Ex. No. M7_CSQ6]

AIM

To create a pandas program to find

- Datetime object for Jan 12, 2022
- Specific date and time of 10:00 pm
- Local date and time
- A date without time
- Current date
- Time from a datetime
- Current local time

Algorithm / Pseudocode

Import pandas as PD

Initialise Timestamp as list

Append a PD timestamp of "12/1/2022 00:00:00.00" to Timestamp

Append a PD timestamp of "7/1/2022 22:00:00.00" to Timestamp

Append a PD timestamp of now to Timestamp

Append a PD timestamp of "30/12/2022 00:00:00.00" date to Timestamp

Append a PD timestamp of current date to Timestamp

Append a PD timestamp of "12/1/2022 12:34:56.00" time to Timestamp

Append a PD timestamp of current time to Timestamp

Loop through Timestamp as I

    Display I

Program Code

```
import pandas as pd

timestamp = []
```

```
timestamp.append(pd.Timestamp(year = 2022, month = 1, day = 12))
timestamp.append(pd.Timestamp(year = 2022, month = 1, day = 7, hour = 22))
timestamp.append(pd.Timestamp.now())
timestamp.append(pd.Timestamp(year = 2022, month = 12, day = 30).date())
timestamp.append(pd.Timestamp.now().date())
timestamp.append(pd.Timestamp(year = 2022, month = 1, day = 12, hour = 12,
minute = 34, second = 56).time())
timestamp.append(pd.Timestamp.now().time())
for i in timestamp:
    print(i)
```

Output

```
2022-01-12 00:00:00
2022-01-07 22:00:00
2022-01-07 18:45:19.838828
2022-12-30
2022-01-07
12:34:56
18:45:19.838828
```