Week-4 Graded Assignment (Programming)

Problem 1 [10]

Question

Calculate the standard deviation with respect to two significant decimals from the multiline numerical value obtained from the user. If the user gives the value `END` then that is the end of the data points. X_i be the data points and $\bar{X}$ be the average of data points.

$$\sigma = \sqrt{\frac{\sum_i^n (X_i - \bar{X})^2}{n-1}}$$

Answer

```
S, l = 0, []
x = input() # Getting the first input
while x != 'END': # Evaluating for the first and further inputs
    l.append(float(x))
    x = input()

if len(l) > 1: # Only one input leads to zero division error in calculation standard deviation
    avg = sum(l) / len(l)
    for i in l:
        S += (i-avg)**2 # Summation of square of difference with mean
    SD = (S / (len(l)-1))**0.5 # Evaluation for standard deviation
    print(f'{SD:.2f}')
```

Test Cases

Public

Input 1

```
1
2
3
4
5
6
7
8
9
END
```

Output 1

```
2.74
```

Input 2

```
124
1124
-1342
-214
-153
-215
-15
END
```

Output 2

```
721.94
```

Private

Input 1

```
1
1
1
1
1
1
1
END
```

Output 1

```
0.00
```

Input 2

```
10
-5
2
-1
0
0
0
0
0
7
END
```

Output 2

```
4.24
```

Input 3

```
1.531
1.32523
4.32143
9.524
8.2452
END
```

Output 3

```
3.77
```

Input 4

```
1.234e5
-9.234e4
-73e3
-43332
98245
4e3
END
```

Output 4

```
90089.92
```

Solution

Tags

while, list

Problem 2 [10]

Question

Write a program to accept a string from the user that contains `(` and `)` brackets. If the string has properly matched parentheses, then print the maximum nesting depth. If the brackets are not properly matched, print `Not matched`.

Note:

- Parentheses `(` and `)` are matched if every `(` has a matching `)` after it.
- `!(a)b,` has a nesting depth of 1. `a1(ad(d4)2)a4` has a depth of 2 and so on.

Input	Output
`(7)(a`	Not matched
`a)*(?`	Not matched
`((jkl)78(A)&l(8(dd(FJI:),):)?)`	4

Answer

```
# Getting input and initialize max_depth = 0 and match =False
s = input()
max_depth = 0
match = True
# Match counting of open brackets and close brackets
if s.count("(") == s.count(")"):
    depth = 0
    for i in s:
# Check each character from s,if character is "(" then increase depth by 1
        if i =="(":
            depth = depth + 1
# if depth value become greater than max_depth then assign depth = max_depth
            if depth > max_depth:
                max_depth = depth
# if character is ")" then decrease depth by 1
        elif i == ")":
            depth = depth - 1
# if depth value become -1, means brackets are not matched, assign match =
false and break the loop
            if depth == -1:
                match = False
                break
# if character is other than "(" or ")" then skip without any operation
        else:
            pass
#if brackets counts are not matched
else:
    match = False
# print the output according to match value
if match == True:
    print(max_depth)
else:
```

```
32      print("Not matched")
```

Test Cases

Public

Input	Output
`(7)(a`	Not matched
`a)*(?`	Not matched
`((jkl)78(A)&l(8(dd(FJI:),):)?)`	4

Private

Input	Output
`(hhfgfhh(ffff))9())`	Not matched
`a)(*)(?`	Not matched
`((((aaaaa)AA)AA)A)SS(S(S(S(D(D(D(D)))))))`	7

Solution

Tags

Problem 3 [15]

Question

Write a program to obtain integers in multiple lines and print all pairs where the sum of any two integers is present in the obtained input.

Note:

- The output should be in non-descending order with respect to the first printed number in the line.
- Final line of input will be an empty line

Answer

```
# Getting the input and appending into the list
l = []
n = input()
while n:
    l.append(int(n))
    n = input()
l.sort() # sorting to maintain a non-descending order

# A two level loop for comparision of two element with each other
for i in range(len(l)):
    for j in range(len(l)):
        if l[i] + l[j] in l and i != j: # required condition and prevention of comparison of same element twice.
            print(l[i], l[j])
```

Test Cases

Public

Input 1

```
1
6
8
9

```

Output 1

```
1 8
8 1
```

Input 2

```
6
2
9
4
8
4
6
3
7
2

```

Output 2

```
2 2
2 4
2 4
2 6
2 6
2 7
2 2
2 4
2 4
2 6
2 6
2 7
3 4
3 4
3 6
3 6
4 2
4 2
4 3
4 4
4 2
4 2
4 3
4 4
6 2
6 2
6 3
6 2
6 2
6 3
7 2
7 2
```

Private

Input 1

```
2
-10
9
-10
6
-5
6
-8
-3
8

```

Output 1

```
-10 2
-10 2
-5 -3
-5 2
-3 -5
-3 9
2 -10
2 -10
2 -5
2 6
2 6
6 2
6 2
9 -3
```

Input 2

```
2
4
7
6
4

```

Output 2

```
2 4
2 4
4 2
4 2
```

Input 3

```
624
620
566
623
340
693
333
446
827
728

```

Output 3

```

```

Solution

Problem 4 [15]

Question

Write a program to obtain a matrix from the user and rotate it in the anti-clockwise direction by 90 degrees.

$$\begin{pmatrix} a & b & c \\ e & f & g \end{pmatrix} \rightarrow \begin{pmatrix} c & g \\ b & f \\ a & e \end{pmatrix}$$

Note:

- The user input will be in multiple lines.
- Each line represents the elements of the rows where the number will be separated by spaces.
- Final line of input will be an empty line.
- Inputs need not be integers.
- No space at the end of line.

Answer 1

```
# Creating a nested list from the input
M = []
row = input()
while row:
    t = []
    for i in row.strip().split(' '):
        t.append(i)
    M.append(t)
    row = input()


# New nested list for the rotated matrix
M_ = []
for i in range(len(M[0])):
    M_.append([])
    for j in range(len(M)):
        M_[i].append(0)

# Transformation
## Transpose
for i in range(len(M)):
    for j in range(len(M[0])):
        M_[j][i] = M[i][j]

## Flipping the rows of the transposed matrix
M_ = M_[::-1]

# Printing Rotated Matrix
for i in range(len(M_)):
    for j in range(len(M_[0])):
        if j != len(M_[0])-1:
            print(M_[i][j], end=' ')
        else:
            print(M_[i][j], end='')
```

```
    print()
```

Answer 2

```
# Some parts of the code used will be covered in the later weeks
M, row = [], input()
while row:
    M.append([int(i) for i in row.strip().split(' ')])
    row = input()
M_ = [[M[j][i] for j in range(len(M))] for i in range(len(M[0]))][::-1]
for i in range(len(M_)):
        print(*M_[i])
    print()
```

Test Cases

Public

Input 1

```
1 2 3
4 5 6

```

Output 1

```
3 6
2 5
1 4
```

Input 2

```
a b
c d

```

Output 2

```
b d
a c
```

Private

Input 1

```
1 2 0 0 0
9 8 7 6 1

```

Output 1

```
0 1
0 6
0 7
2 8
1 9
```

Input 2

```
0.987 0.1 0.0
0.0 0.0 0.0
1.1 567 43
0 9.8 -7.3

```

Output 2

```
0.0 0.0 43 -7.3
0.1 0.0 567 9.8
0.987 0.0 1.1 0
```

Input 3

```
ab bc ce ef
gh hi ij jk
lm mn no pq

```

Output 3

```
ef jk pq
ce ij no
bc hi mn
ab gh lm
```

Input 4

```
* * * *
* * . .

```

Output 4

```
* .
* .
* *
* *
```

Solution

Tags