FDS.

Practical NO - 01(A)

Title - A python program to store marks for N students.

Aim - Write a python program to store marks scored in subject "Fundamental of Data structure" by N students in the class. Write functions to compute following

a) The average score of class.
b) Highest score and lowest score of class.
c) Count of student who were absent for the test.
d) Display mark with highest frequency.

Prerequisite - Python programming.

Objectives - To understand the use function for N student Record.

Input - N number of students.

Output - Resulting average, highest and lowest mark operation.

Theory:

Arrays -

An array is a kind of data structure. that can store a fixed-size sequential collection

collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as collection of variables of the same type.

Instead of declaring individual values variables, such as number0, number1, ..., and number 99, you declare one array variable such as numbers and use umbers[0], numbers[1], and ... numbers[99] to represent individual variables, A specific element in an array is acessed by an index.

All array consits of contiguos memory location. The lowest address corrensponds to the first element and the highest address to last element.

Declaring Arrays.

- to declare an array in C, a programmer specifies the type of element and the number of elements required by an array as follows:-

type arrayName [arraysize];

This is called a single-dimensional array. The arraysize must be an integer constant greater than zero and type can be any valid C data-type. for example, to declare a 10-element array called balance of type double, use this element -
double balance [10];

Here balance is a variable array which is sufficient to hold up to 10 variable double numbers.

Initializing Arrays.
- You can initialize an array in C either one by one or using a single statement as follows-
double balance [5] = {1000.0 , 2.0 , 3.4 , 7.0 , 50.0}

Acessing Array Elements

An element is acessed by indexing the array name. This is done by placing the indexe of the element within squarebracket after the name of the array. for example -
double salary = balance [9];

Functions Used :
Write algorithm / pseudo code for each function
a) The average score of class.
```
sum = 0
for i in range(len(marks)):
        sum = sum + marks[i]
avg = sum/len(marks)
return avg
```

b) Highest score and lowest score of class.
```
min = 0
max = 0
for i in range(len(marks)):
        if (marks[i] < min):
            min = marks[i]
        if (marks[i] > max):
            max = marks[i]
print(min)
print(max)
```

c) Count of students who were absent for the test

```
count=0
for i in range(len(arr)):
        if ( ~~te~~ marks[i] < 0):
            count = count + 1
print(count)
```

d) Display marks with highest frequency.

```
max =0
f = marks[0]
for i in marks:
        freq = marks.count(i)
        if freq > max:
            max =freq
            f = i
print(str(f))
```

Algorithm -

Step ① Start

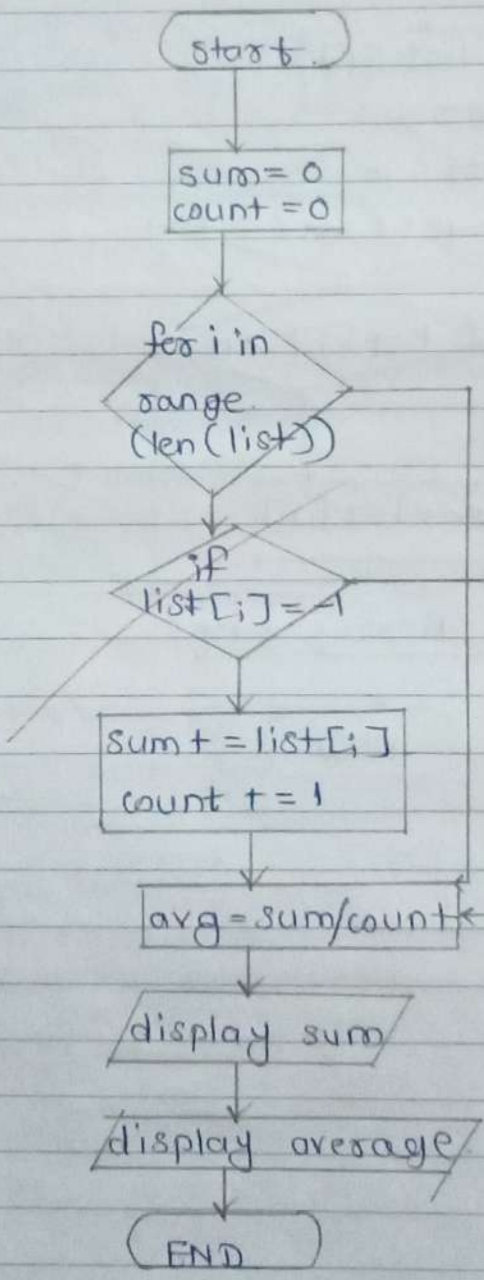
② declare integer input variable for no of student and take input.

③ declare array to store marks.

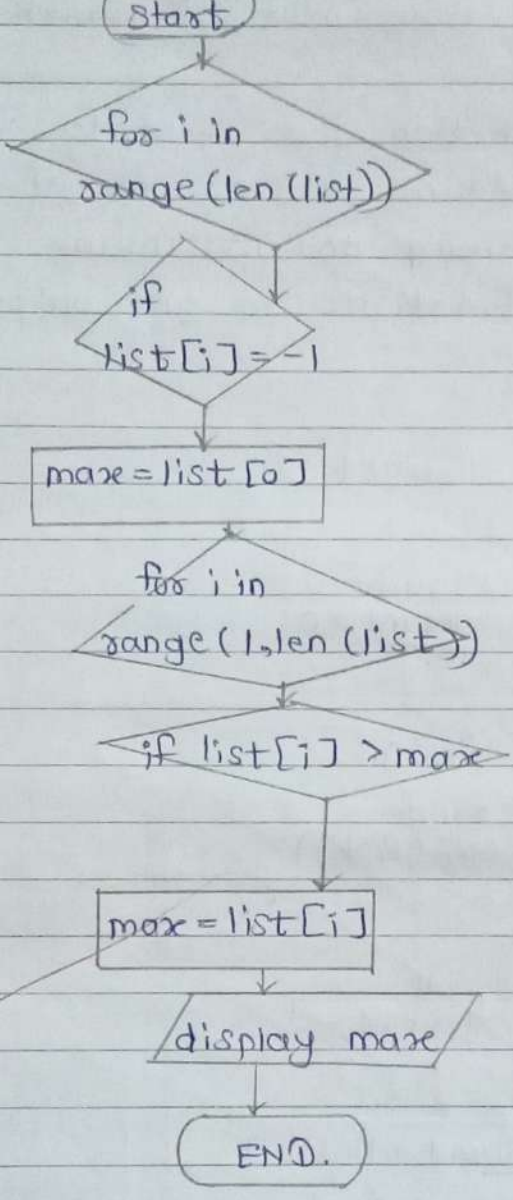
step ④ take input from user of marks and store it in array.
⑤ write a function for getdata, calculating average, checking absent no of student, check minimum ~~and~~ maximum.
⑥ call the functions to give output.
⑦ END.

Flowchart :-

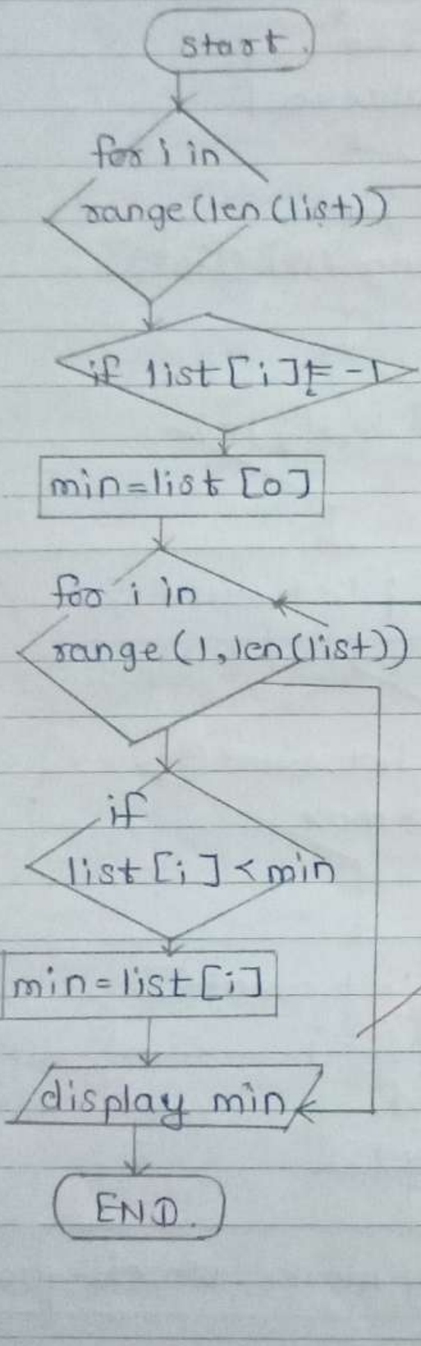
Flowchart for average



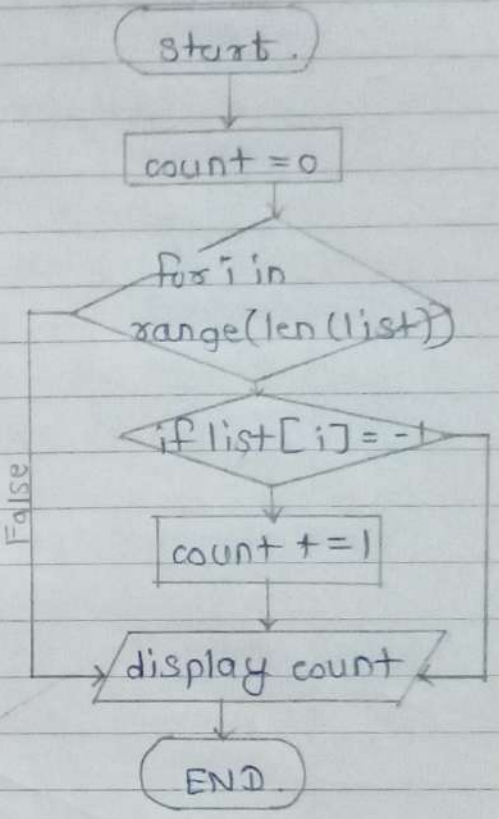
Flowchart for max marks.



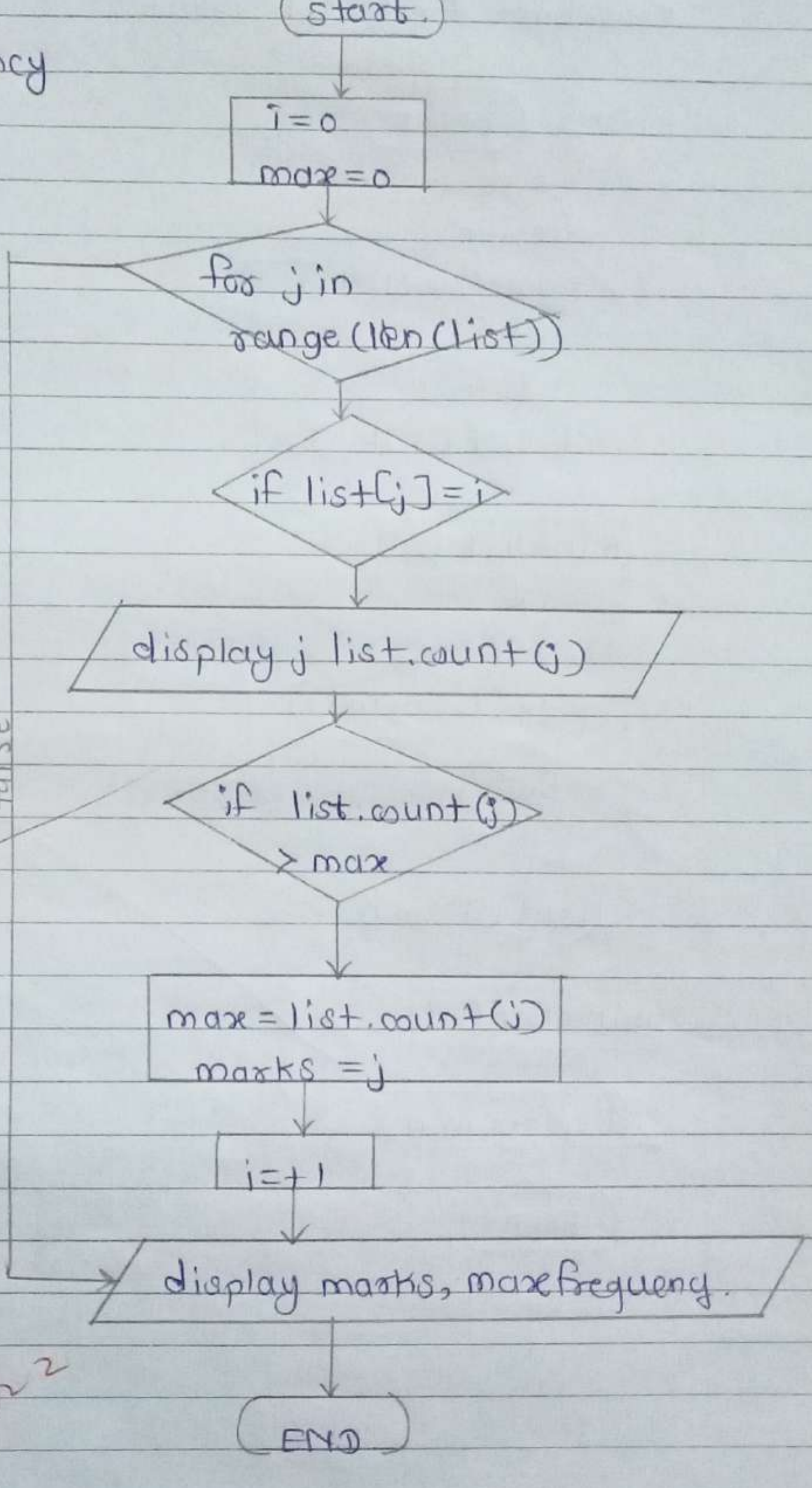
Flowchart for lowest count.



flowchart for absent



flowchart
for max frequency
of marks.



21/10/22

FDS

Practical NO - 2(A)

Title - Write a python program to perform string operations.

Aim - write a python program to compute following operations on string.
a) To display word with longest length.
b) to determine frequency of occurence of particular character in the string.
c) to check whether given string is palindrome is or not.
d) To display index of first appearance of the substring.
e) To count the occurences of each word in a given string.

Prequisite - Basic of string Operations.

Objectives - to understand the use standard library functions for string operations.
to perform the string operations.

Input - One or Two string

Output - Resulting string after performing string operations.

DPU

Theory -

String - string is defined as an array of characters or a pointer to characters.

Null-terminated string :-

string is terminated by a special character which is called as null terminator or null parameter (\0). So when you define a string you should be sure to have sufficient space for the null terminator. The null terminator has value 0.

Declaring string :

As in string definition, we have two ways to declare a string. The first way is, we declare an array of character as follows.

```
char s[] = "string"
    or
char str[20];
```

String operation (explain each operation in detail with example)

a) To display word with the longest length.

b) To determines the frequency of occurence of particular character in the string

c) To check whether given string is palindrome or not.

d) To display index of first appearance of the substring.

DPU

e) To count the occurence of each word in a given string.

Algorithm:
write algorithms for your program

Flowchart:
Draw flowchart for above algorithm

Conclusion:
By this way, we can perform string operations successfully.

Algorithm :-
step① take input string from user.
② calculate length of string.
③ write a function to calculate frequency of substring in a string
④ function to check palindrome
⑤ function to check index of substring.
⑥ function to count occurence of substring in a string.
⑦ call all functions.
⑧ STOP.

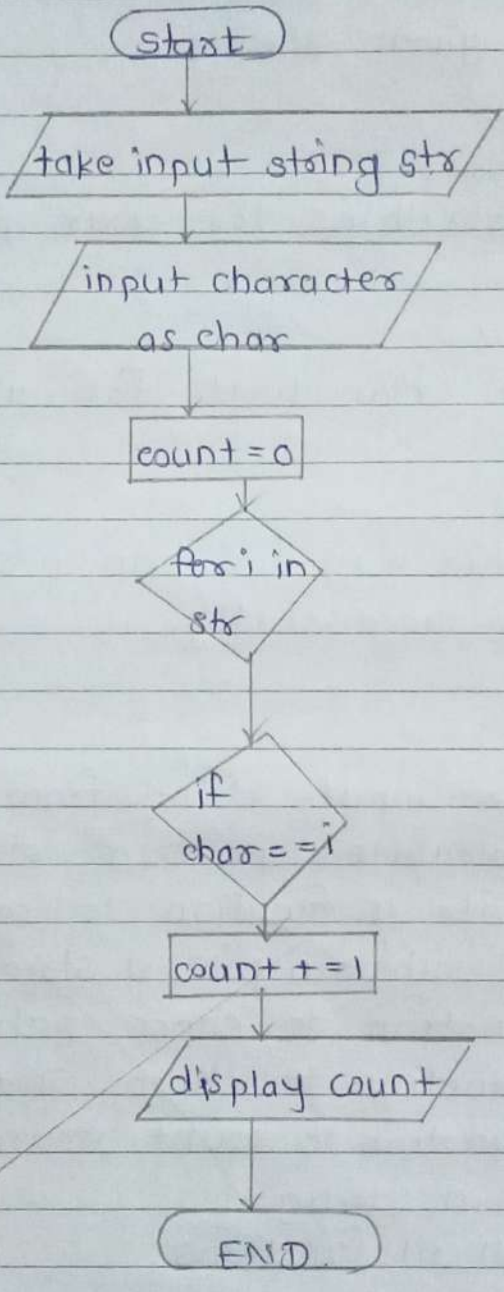
flowchart :-

for particular character in string.

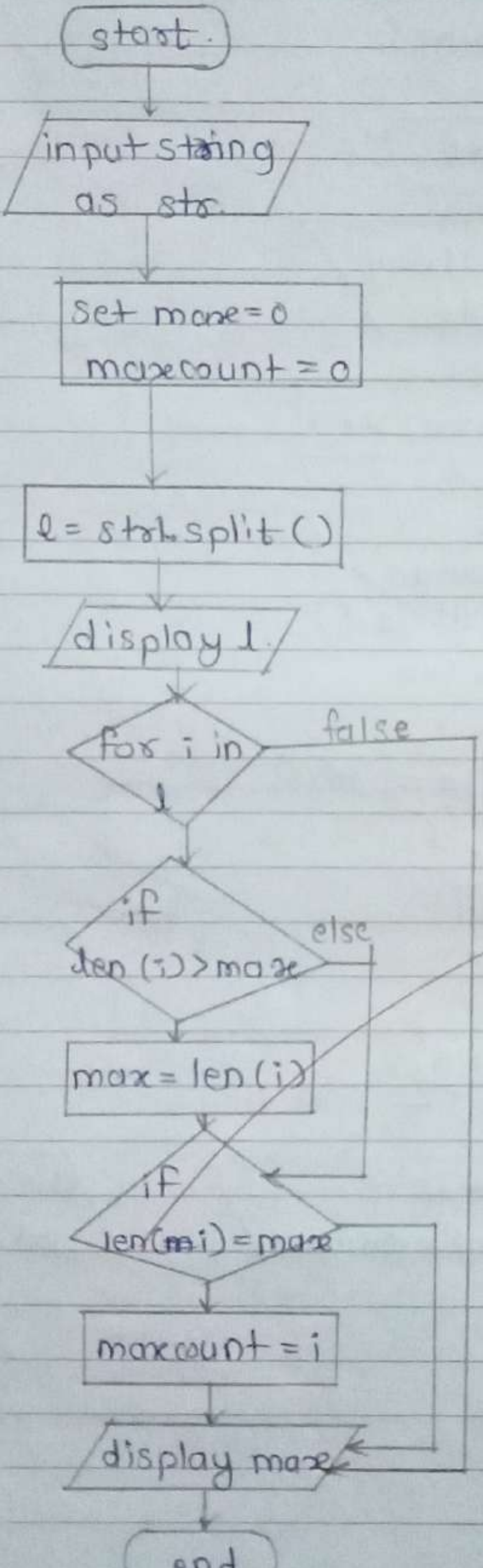
~~if s[::-1]==s.~~

~~print palindrome.~~

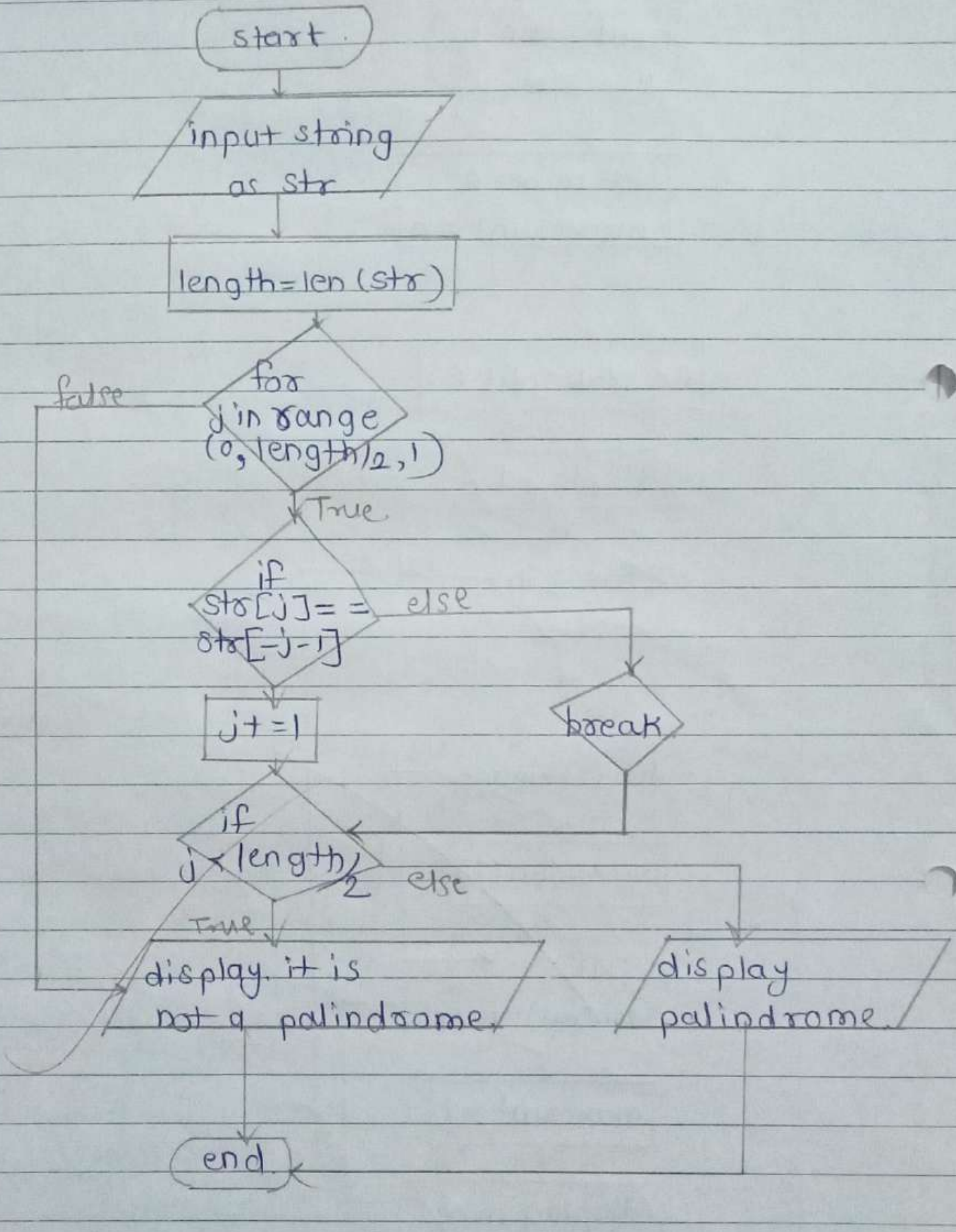
~~print Not palindrome~~



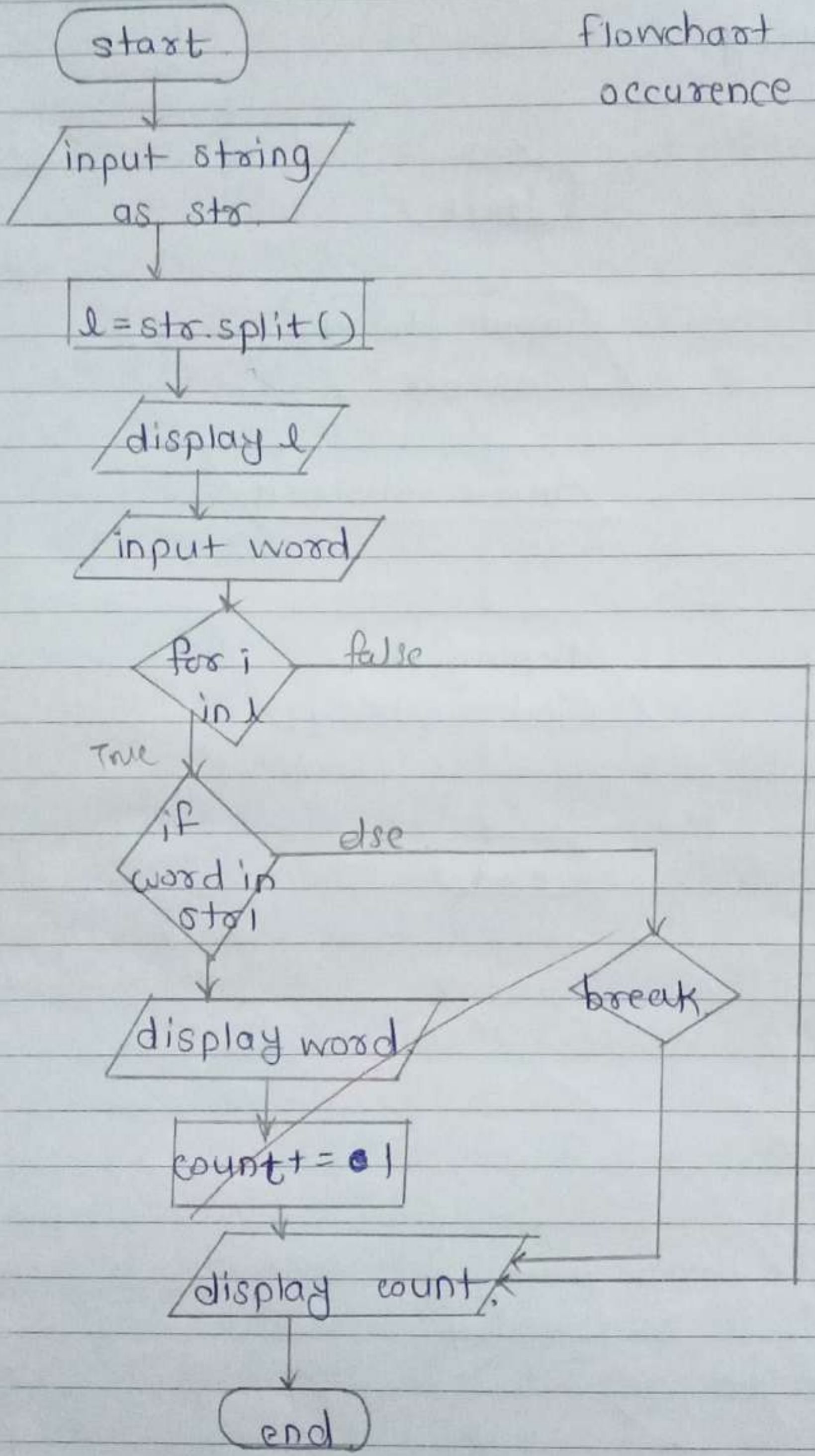
flowchart for word with longest length.



flowchart for palindrome :-

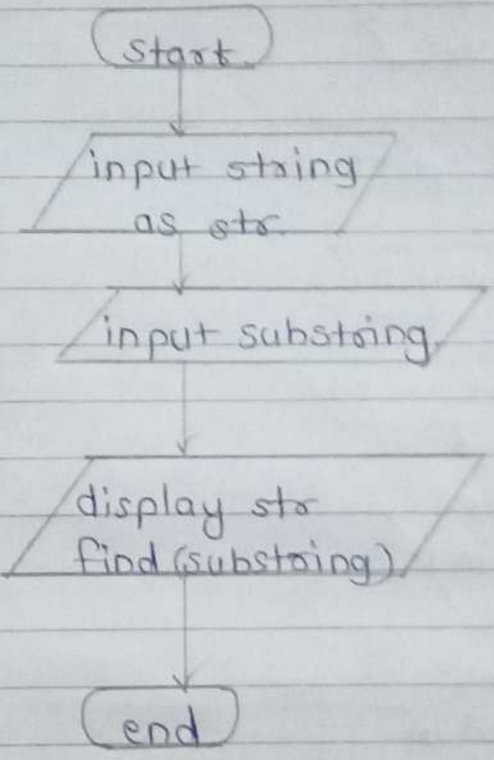


flowchart to count occurence of each word.



DPU

displaying.



[illegible] 2/10/22

Practical No. 3 (A)

Title - Perform different operations on Matrix.

Aim - Write a python program to compute following compatation on matrix.
A) Addition of two matrix.
B) substraction of two matrix.
c) Multiplication of two matrix.
D) Transpose of Matrix.

Prerequisite
- knowledge of representing matrix in python.
- knowledge of different operations that can be performed on matrix.

Objectives -
- compute the transpose of matrix.
- Perform addition, substraction and multiplication of two matrix.

Input - Number of rows and columns of two matrices.
- Elements of both the matrix.

Outcome - Transpose of matrix.
- Result of addition, substraction and multiplication of both matrices.

DPU

Outcome -

Theory - * 2-dimension array -
In a 2D array, multiple arrays are inserted as elements in an outer array. Eeach element in a 2D array is represented by two indices, the row and the column, 2D array in python are zero indexed, which means counting of indices starts from zero rather than 1 and thus zero is the first index in an array in python.

* Matrix Operations.
A) concept of matrix
- Matrix, a set of numbers arranged in rows and columns so as to form a rectangular array. The numbers are called the elements, on entries, of the matrix.
B) Addition of Matrix
for e.g. x = [[1, 2], [4, 5]] would represent 2x2 matrix. first row can be selected as x[0] & the element in first row, first column can be selected as x[0][0].
x + x = [[2, 4], [8, 10]]
c) Substraction
same as above addition. substraction of two matrices done if & only if order of both matrix is same
e.g. x − x = [[0, 0], [0, 0]]

DPU

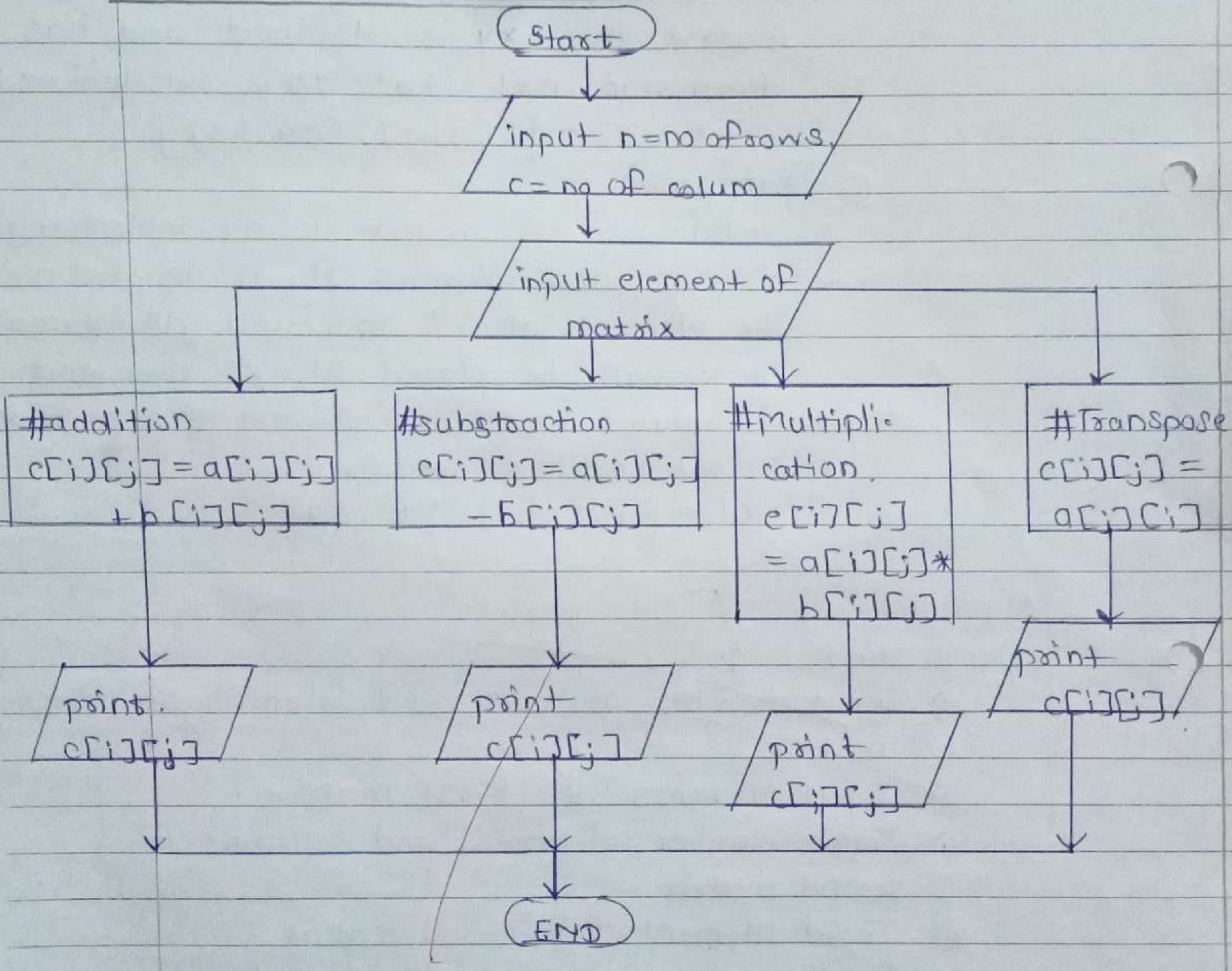
D) Multiplication:
multiplication of 2 matrices x and y is defined only if the number of columns in X is equal to the number of rows y.
If x is n×m matrix and y is a m×l matrix then xy is defined and has dimension n×l (but yx is not defined)
$X \times X = [[9,12], [24,33]]$

E) Transpose:
transpose of matrix is the interchanging of rows and columns. It is denoted as 'X''. The element at ith row and jth column in X will be placed at jth row and ith column in 'X''. so if X is a 3X2 matrix, 'X' will be a 2×3 matrix.
$X^T = [[9, 24], [12, 33]$

Algorithm -
step ① start
② Input number of rows and columns of first matrix.
③ Input elements of first matrix.
④ Input number of rows and columns of second matrix.
⑤ Input elements of second matrix.
⑥ Functions to transpose first matrix i.e. the element at row R column C in the original is placed at row C column R of the transpose

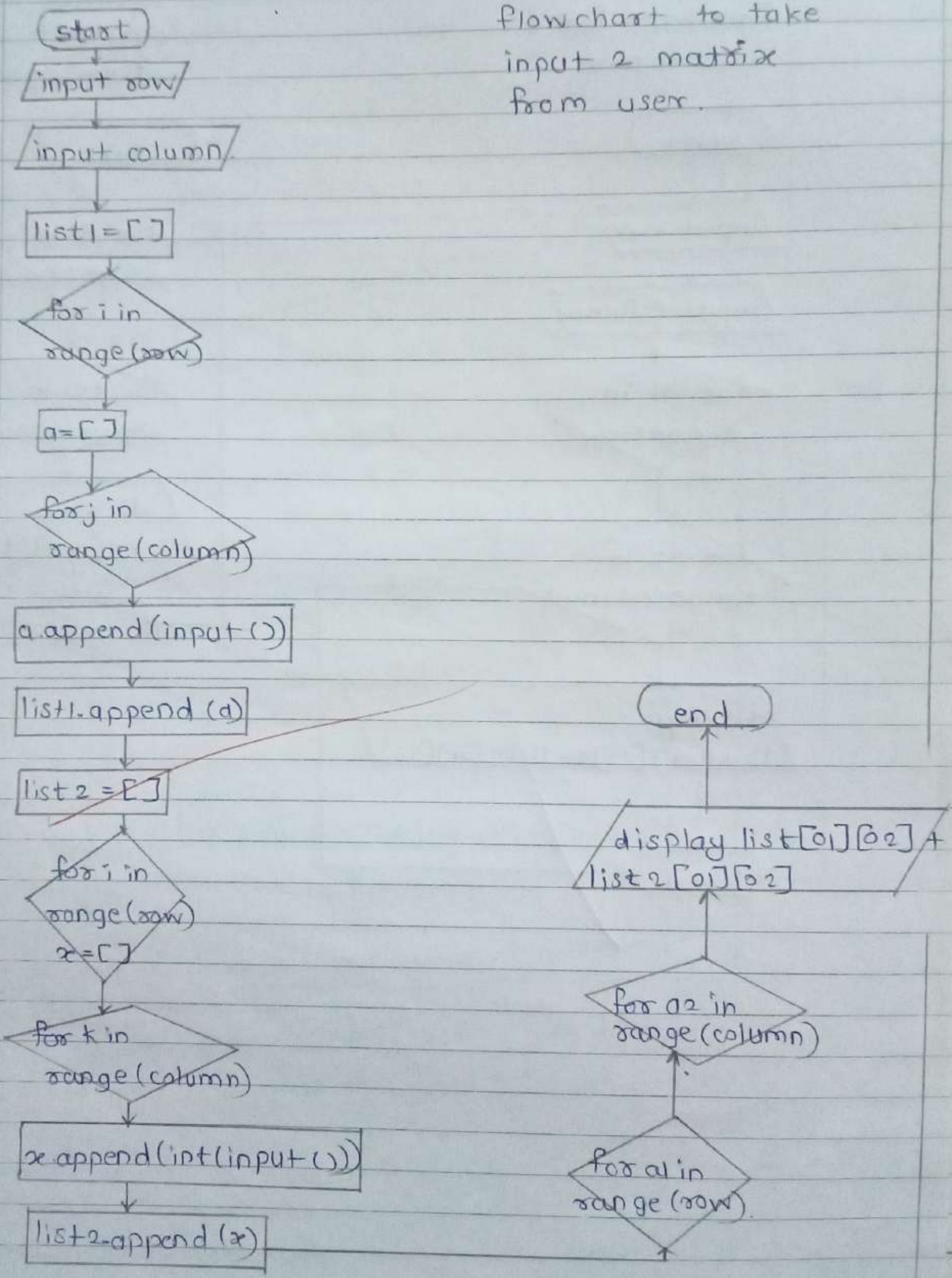
7. Functions to add, substroact and multiply two matrices.
8. call the functions.
9. STOP

Flowchart -

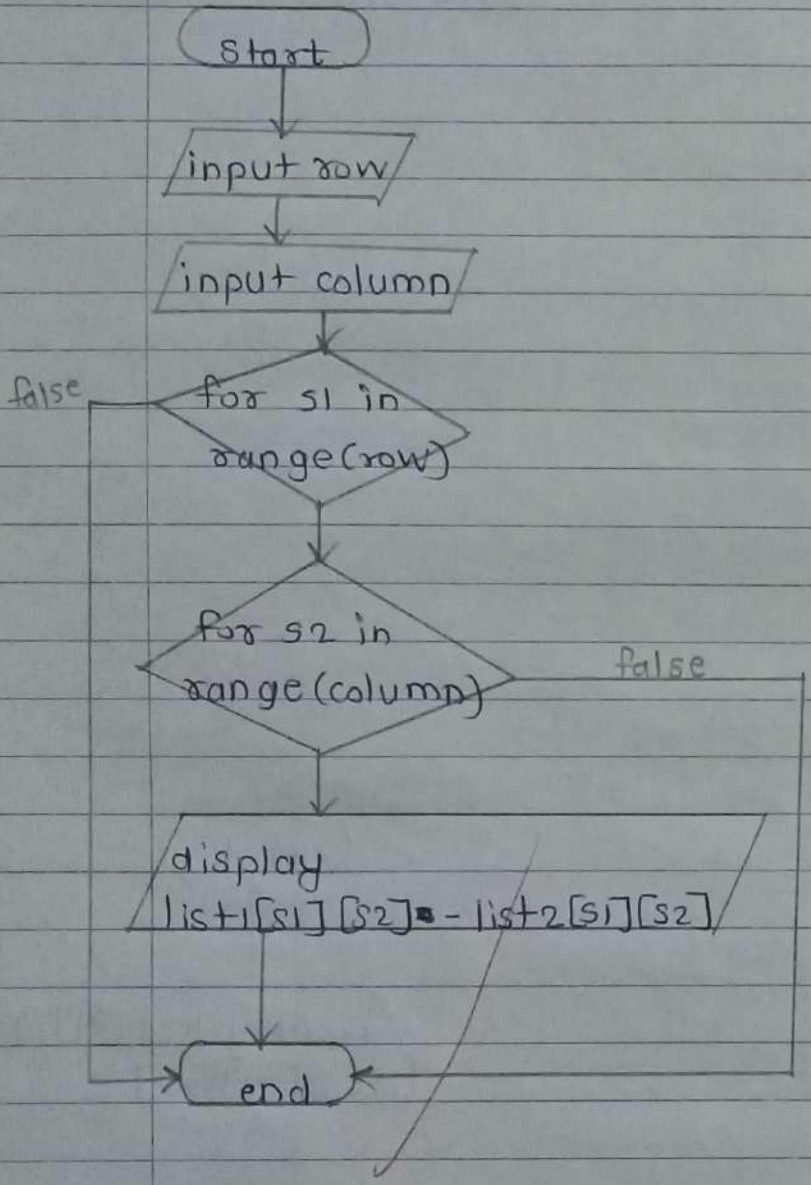

Start
input n=no of rows, c= no of colum
input element of matrix
#addition c[i][j] = a[i][j] + b[i][j]
#substraction c[i][j] = a[i][j] - b[i][j]
#Multiplication c[i][j] = a[i][j] * b[i][j]
#Transpose c[i][j] = a[j][i]
print c[i][j]
print c[i][j]
print c[i][j]
print c[i][j]
END

Conclusion - By this way, we can perform various operations on matrix successfully.

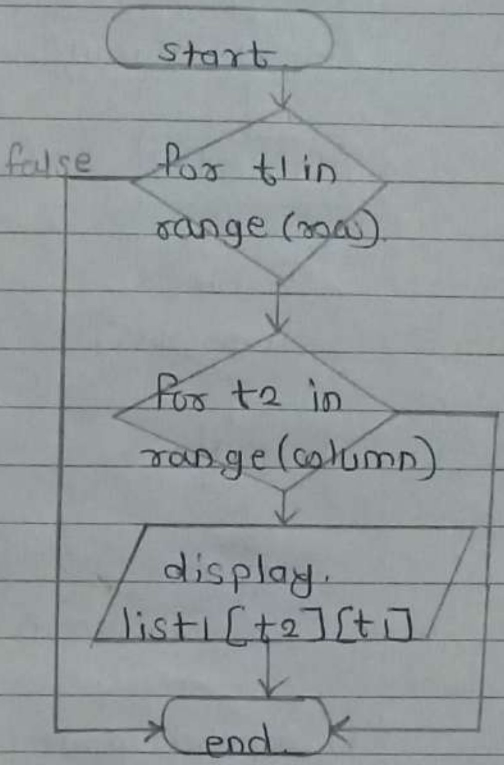
flowchart to take
input 2 matrix
from user.



flowchart for
substraction

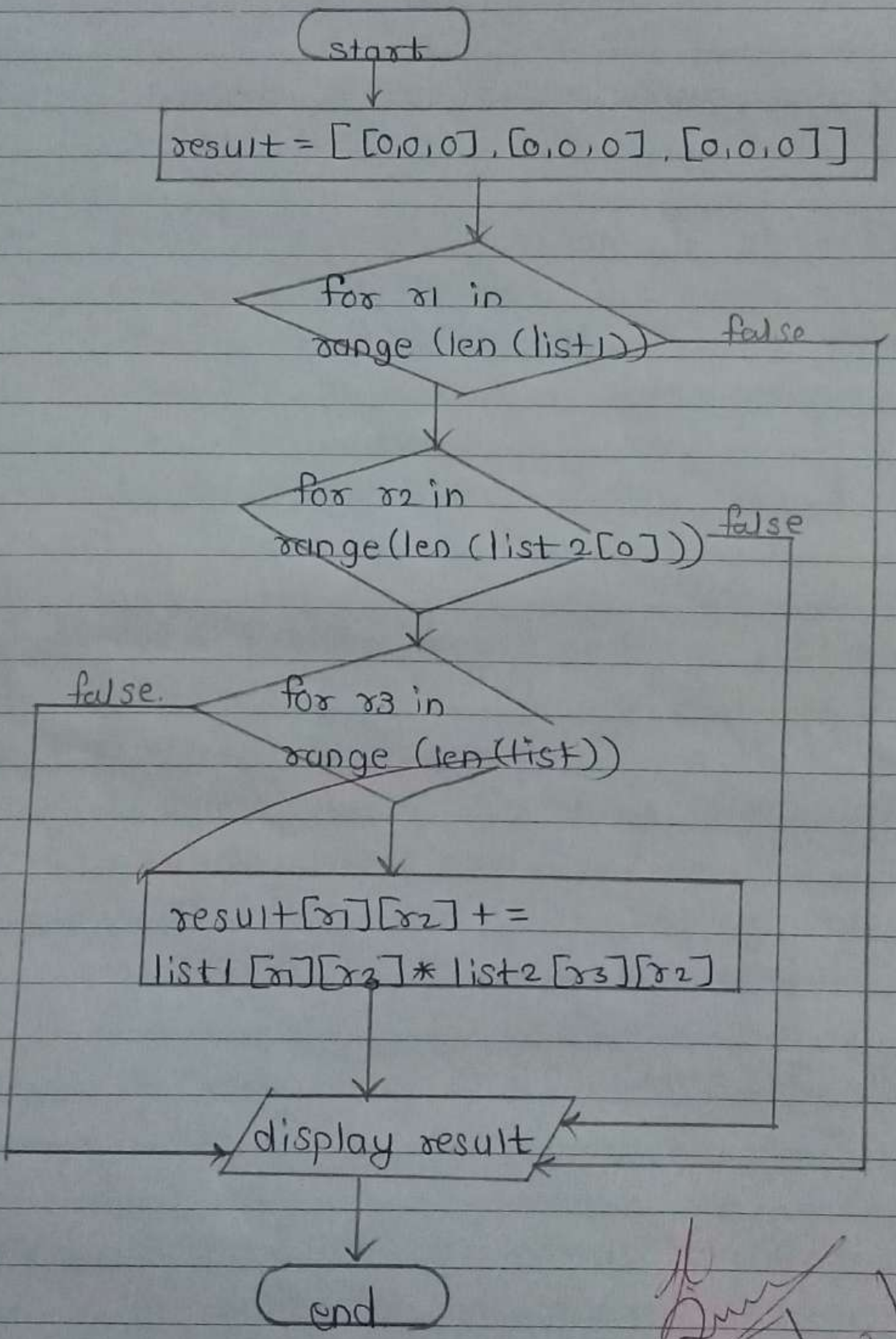


flowchart for
transpose



DPU

Flowchart for Multiplication.



start
result = [[0,0,0], [0,0,0], [0,0,0]]
for r1 in range (len (list1))
false
for r2 in range(len (list 2[0]))
false
for r3 in range (len (list))
false.
result[r1][r2] + =
list1 [r1][r3] * list2 [r3][r2]
display result
end

[illegible] 21/10/22

FDS.

Practical Assignment NO - 04

Title - Sorting of an array using selection and bubble sort.

Aim - Write a python program to store first year percentage of student in array. write function for sorting array of floating point numbers in ascending order using
a) Selection sort
b) Bubble sort and display top five score of club.

prerequisite - knowledge of sorting technique.

Objective - To sort array of floating point numbers in ascending order using
a) selection sort
b) Bubble sort and display top five scores.

Input - size of array and Elements of array.

Theory - Sorting of an array means putting elements in an ordered sequence.
- Ordered sequence is any sequence that has an order coresponding to elements

like numeric or alphabetical, ascending or descending.

Advantages of sorted array

- The advantage of sorted array is that it takes logarithmic time to search for an element since you can use a divide and conquer method.
- In an unordered array, unsorted array searches takes linear time.
- Deletion takes linear time in both types of arrays because of need to shift element blocks back to fill in the gap left by the deleted element.

Disadvantages of sorted array

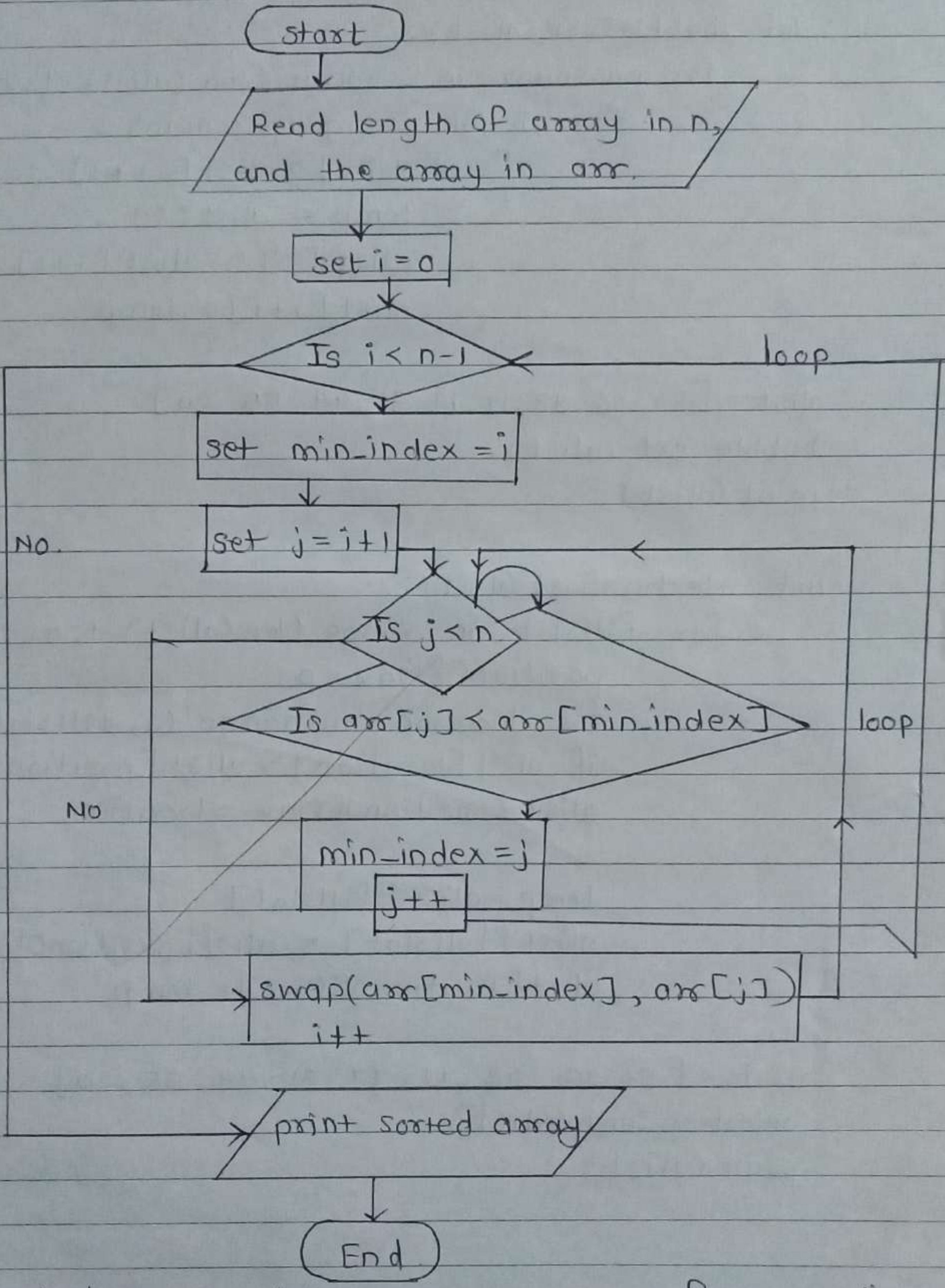
- The disadvantage is that insertion in an ordered array takes linear time because you have to shift elements over to make room for, for inserted element.
- In an unsorted array, insertion take constant time because you can just track the new element on to the end of the array.

Algorithm -

```
def bubbleSort (alist):
        for passnum in range (len (alist)-1, 0, -1):
                for i in range(passnum) :
                        if alist[i] > alist[i+1]:
                                temp = alist[i]
                                alist [i] = alist[i+1]
                                alist[i+1] = temp.

alist = [54, 26, 93, 17, 77, 31, 44, 55, 20]
bubbleSort (alist)
print (alist)

def selectionSort (alist):
        for fillslot in range (len(alist)-1, 0,-1):
                positionOfMax = 0
                for location in range (1, fillslot +1);
                if alist[location] > alist[positionofMax]
                ~~alist [~~PositionOfMax = location.

                temp = alist[fillslot]
                alist[fillslot] = alist[positionOfMax]
                alist[PositionofMax] = temp.

alist = [54, 26, 93, 17, 77, 31, 44, 55, 20]
selection Sort (alist)
print (alist).
```

DPU

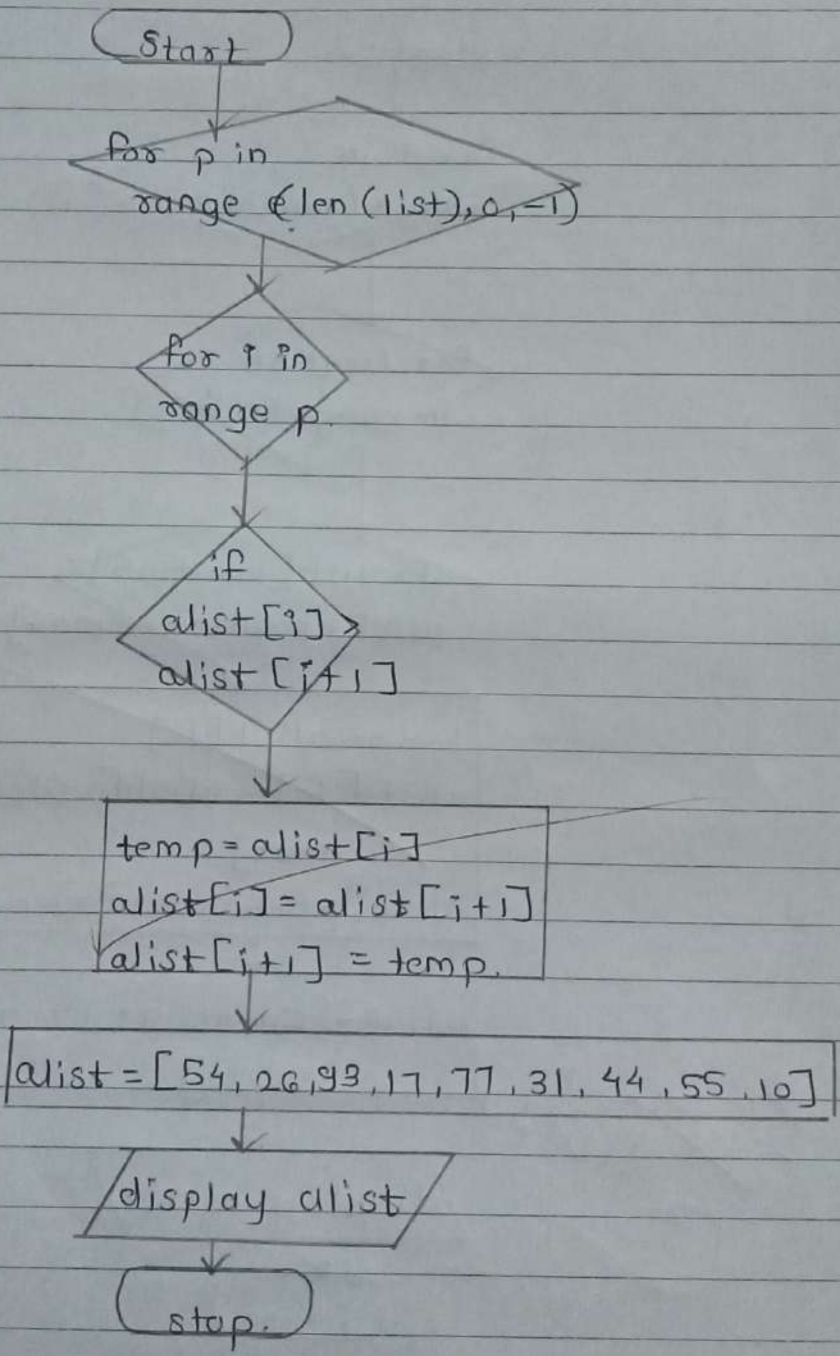
Flowchart :-



Conclusion - By this way, we can perform sorting of an array using selection and bubble sort.

DPU

flowchart for Bubblesort.

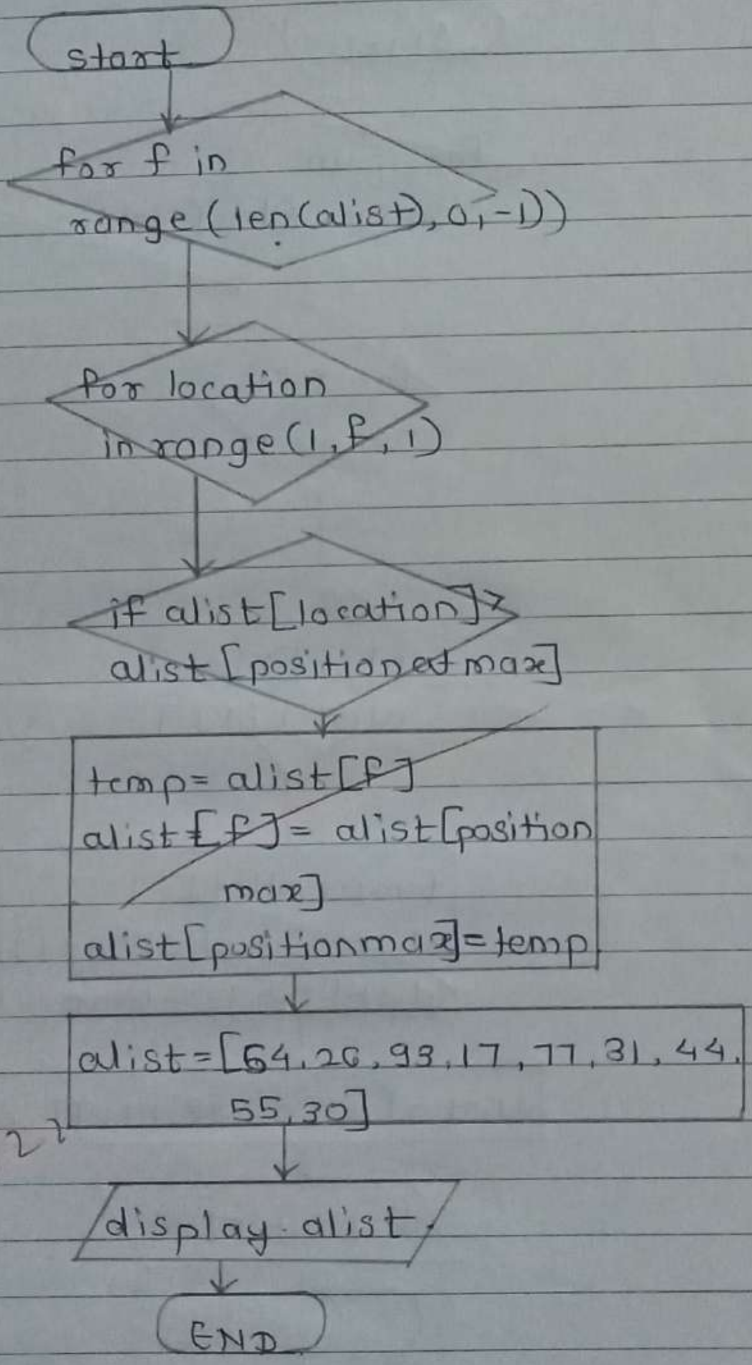


Start
for p in range (len (list), 0, -1)
for i in range p.
if alist[i] > alist [i+1]
temp = alist[i]
alist[i] = alist[i+1]
alist[i+1] = temp.
alist = [54, 26, 93, 17, 77, 31, 44, 55, 10]
display alist
stop.

DPU

DPU

Flowchart for selection sort.


start
for f in range(len(alist),0,-1))
for location in range(1,f,1)
if alist[location]> alist[positionofmax]
temp= alist[f]
alist[f]= alist[position max]
alist[positionmax]=temp
alist=[64,26,93,17,77,31,44,55,30]
display alist
END

[illegible] 21/10/22