Dr. D. Y. Patil Educational Enterprises Charitable Trust's

Dr D Y PATIL SCHOOL OF ENGINEERING

Dr. D. Y. Patil Technical Campus

Dr. D. Y. Patil Knowledge City, Charholi (Bk), Via-Lohgaon, Pune - 412 105

D Y PATIL GROUP

Department of : computer Engineering

INDEX

Sr. No.	Title	Page No.	Date	Sign of Faculty	Remark
1)	Write a python program using function to compute list of students who play	1-8	7/9/22		
a)	both cricket & badminton				
b)	either cricket or badminton but not both.				
c)	neither cricket nor badminton				
d)	cricket & football but not badminton.				
2)	Write a python program to compute the net amount of a bank account.	9-13	14/9/22		
3)	Write a python program to compute operations on string.	14-18	28/9/22		
4)	Write a python program to compute following computation on matrix.	19-25	14/10/22		
a)	Addition of 2 matrices				

Dr. D. Y. Patil Educational Enterprises Charitable Trust's
Dr D Y PATIL SCHOOL OF ENGINEERING
Dr. D. Y. Patil Technical Campus
Dr. D. Y. Patil Knowledge City, Charholi (Bk), Via-Lohgaon, Pune - 412 105

Department of : computer Engineering

INDEX

Sr. No.	Title	Page No.	Date	Sign of Faculty	Remark
b)	Substraction of 2 matrices				
c)	Multiplication of 2 matrices				
d)	Transpose of a matrix.				
5)	Write a python program to store names & mobile number using binary search.	26-29	30/10/22		
6)	Write a python program to store percentage of students for sorting array of floating point nos. in ascending order using selection sort & Bubble sort.	30-33	2/11/22		
7)	Write c++ program to store and compute with 2 sets using linked list.	34-41	11/11/22		
8)	Write c++ program for sorting binary no. using doubly lists. Write functions.	42-45	18/11/22		
a)	To compute 1's & 2's complem				

Dr. D. Y. Patil Educational Enterprises Charitable Trust's
Dr D Y PATIL SCHOOL OF ENGINEERING
Dr. D. Y. Patil Technical Campus
Dr. D. Y. Patil Knowledge City, Charholi (Bk), Via-Lohgaon, Pune - 412 105

D Y PATIL GROUP

Department of : computer Engineering

INDEX

Sr. No.	Title	Page No.	Date	Sign of Faculty	Remark
b)	Add 2 binary numbers.				
9).	Write ctt program using stack to check whether given expression is well parenthesized or not.	46-48	25/11/22		
10)	Implement ctt program for expn conversion as infix to postfix & its evaluation using stack.	49-53	2/12/22		
11).	Write ctt program for simulating job queue. Write functions to add job & delete job from queue.	54-57	5/12/22		
12)	Write ctt program to simulate deque with functions to add an delete elements from either end of thedqueue.	58-60	6/12/22		

Dr. D. Y. Patil Educational Enterprises Charitable Trust's
Dr D Y PATIL SCHOOL OF ENGINEERING
Dr. D. Y. Patil Technical Campus
Dr. D. Y. Patil Knowledge City, Charholi (Bk), Via-Lohgaon, Pune - 412 105

D Y PATIL GROUP

Department of : computer Engineering

INDEX

Sr. No.	Title	Page No.	Date	Sign of Faculty	Remark
13)	Pizza parlour accepting maximum M orders are served in first come first served basis. order once placed cannot be cancelled. write c++ program to simulate the system using circular queue using array.	61-63	9/12/22		

Assignment No - 1

- Problem statement :- To 2nd year computer engineering class of M students, set A of students play cricket and set B of students play badminton. Write c/c++ program to find and display 1) set of students who play either cricket or badminton or both. 2) set of stu. who play both ckt and badminton. 3) set of stu. who play neither cricket nor badminton.
- Objective :- To study the lists data type in python. To implement set operation in python.
- Theory

python list :- The list is a most versatile datatype available in python which can be written as list of comma-seperated value liten betn square brackets. Important thing about a list is that items in a list need not to be of the same type.

creating a list is as simple as putting diff. comma-seperated values betn square brackets. For ex :-

List 1 :- ['physics', 'chemistry', 1997, 2000];

List 2 :- [1, 2, 3, 4, 5].

similar to string indices, list indices start at 0 & lists can be sliced, concentranated and soon.

- Accessing values in lists :- To access value in lists, use the square brackets for slicing along with index as indices to obtain value available at that index. For ex -

```
list = [1, 2, 3, 4, 5, 6]
print "list [1:5] :", list [1:5]
```

When above code is executed, it produces following result. list [1:5] : [2, 3, 4, 5]

- updating lists :- you can update single or multiple elements of lists by giving slice on left-hand side of assignment operator. & you can add to elements in list. with apped().

```
ex:- list = ["physics", 'chemistry', 1997, 2000];
print "value available at index 2 :"
print list [2]
list [2] = 2001
```

- Built-in list function & methods.

Python includes following list function.

Sr No :-	function with description
1.	cmp (list1, list2) compare elements of both lists.
2.	len (list) lives the total length of list.
3.	Max (list) Returns item from list with max value.
4.	Min (list) Return item from list with min
5.	list (seq) converts tuple into list.

Python includes following list method.

Sr.No.	Method with description.
1.	list.append(obj). append obj to list.
2.	list.count(obj). returns count of obj.
3.	list.extend(seq). append content of seq list
4.	list.index(obj). returns largest index.
5.	list.insert(index, obj). Inserts object.
6.	list.pop(obj = list[1]). Remove.
7.	list.remove(obj) Removes objects
8.	list.reverse(). Reverse object.
9.	list.sort[(func)]. sorts object of list.

- Accepting input from user for list.

```
num_list = []
n = int(input("Enter list size"))
for i in range(0,n):
    print("Enter no. at index ", i)
    item = int(input())
    num_list.append(item)
```

- Python "in" operator :- Basically "in" operator in python checks whether a specified value is constituent element of sequence like string, array, list or tuple etc.

```
lis1 = [1,2,3,4,5]
string1 = "My name is python"
tuple1 = (11,22,33,44)
print(5 in list1) # True.
print("is" in string1) # True.
```

print (88 in tuple1) # False

- Python "not in" operator :- The "not in" operator in python moves exactly opposite way as in operator works. It is also checks the presence of specified value inside a given sequence but it's return values are totally opposite to that of in operator.

List 1 = [1, 2, 3, 4, 5].

string 1 = "My name is python"

tuple 1 = (11, 22, 33, 44)

print (5 is not in list 1) # False

print ("is" not in string 1) # False.

print (88 not in tuple 1) # True.

- SETS :- Set operations sets can be combined in a no. of diff. ways to produce another set. These 4 -basic operations are introduced.

1) union :- The union of set A & B, denoted by $A \cup B$. $A \cup B = \{x \mid x \in A \quad x \in B\}$

ex :- $A = \{1, 2, 3\}$ & $B = \{4, 5\}$ then

$A \cup B = \{1, 2, 3, 4, 5\}$

2) Intersection :- The intersection of sets A & B denoted by $A \cap B$ is.

$A \cap B = \{x \mid x \in A \quad x \in B\}$

ex :- $A = \{1, 2, 3\}$ & $B = \{1, 2, 4, 5\}$

$A \cap B = \{1, 2\}$

3) Difference :- The difference of set A from B denoted by A - B. A - B = { x | x ∈ A ^ x ∉ B }

ex :- A = { 1, 2, 3 } & B = { 1, 2, 4, 5 }

A - B = { 3 }

Note :- That in general A - B is not equal to B - A.

- Algorithm :-

1) Accept the students in list from user who play cricket, football, badminton in list criclist, footlist, badlist.

2) write the func. for union, intersection & difference of lists.

3) union of 2 sets :-

a) copy 1st list A in result list.

resultset = list A copy()

b) check whether elements of list B present in result list.

for in list B :

if i in list result set

continue

resultset.append(i)

c) if present : skip the element.

d) Not present : append element in result list.

4) Intersection of two sets :

a) check common elements of two lists using "in" operator & for loop.

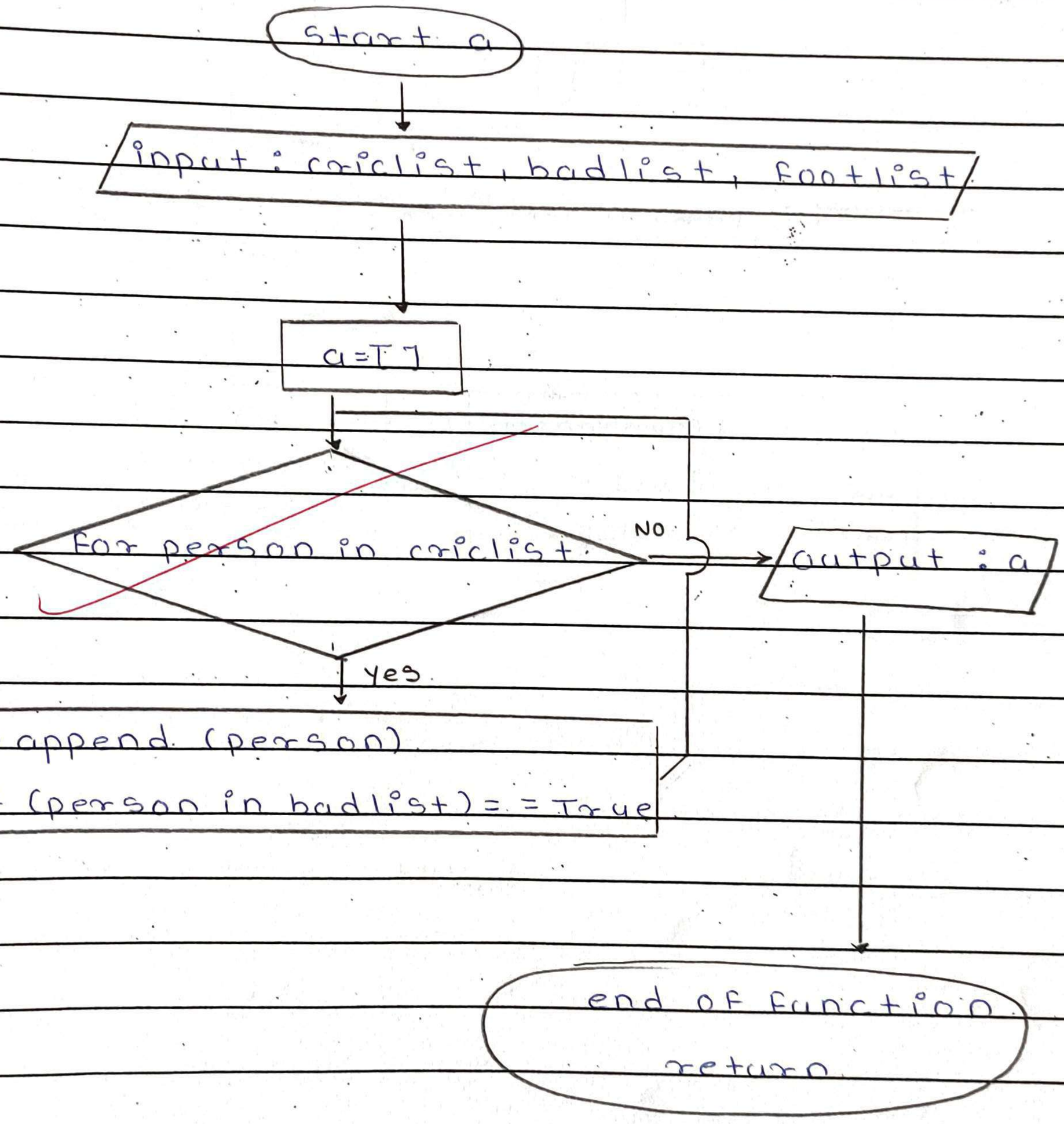
for i in list A :

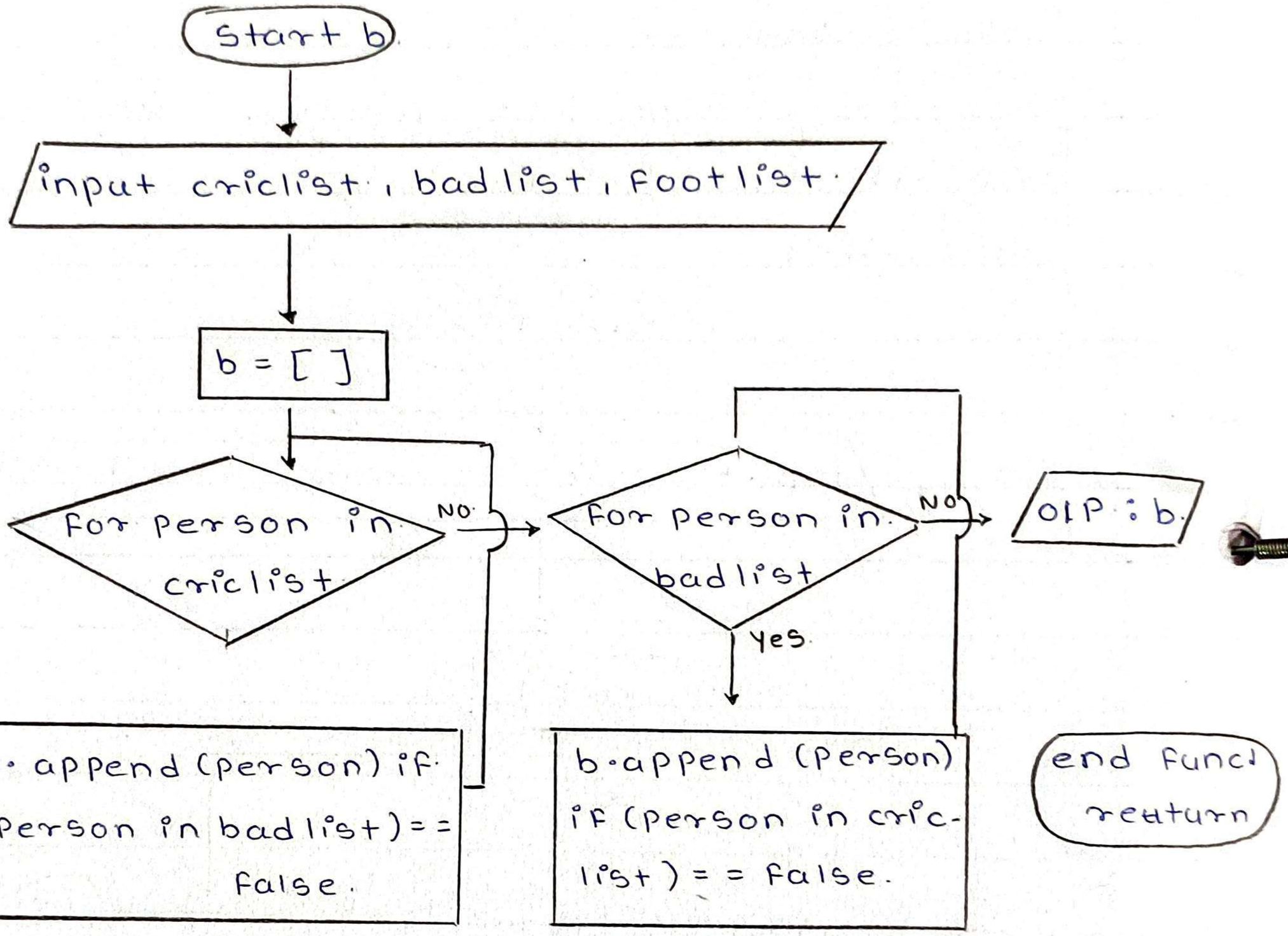
if in list B :
resultset (i)
b) append common elements in the result list.
5) Diff. of 2 sets :
a) Transverse 1st list A using For loop.
b) check common elements in list A & list B.
c) skip common element append the uncommon elements of list A in result list.
For i in list A :
if i in list B :
continue
resultset . append (i)
6) Now using the above functions find the solutions of given problem.
1) set of students who play either cricket or badminton or both union (criclist, badlist)
2) set of students who play both cricket & badminton.
intersection (criclist, badlist.
3) set of students who play only cricket difference (criclist, badlist) union diff. (criclist, footlist)
4) set of students who play only badminton difference (badlist, criclist) union difference (badlist, footlist)

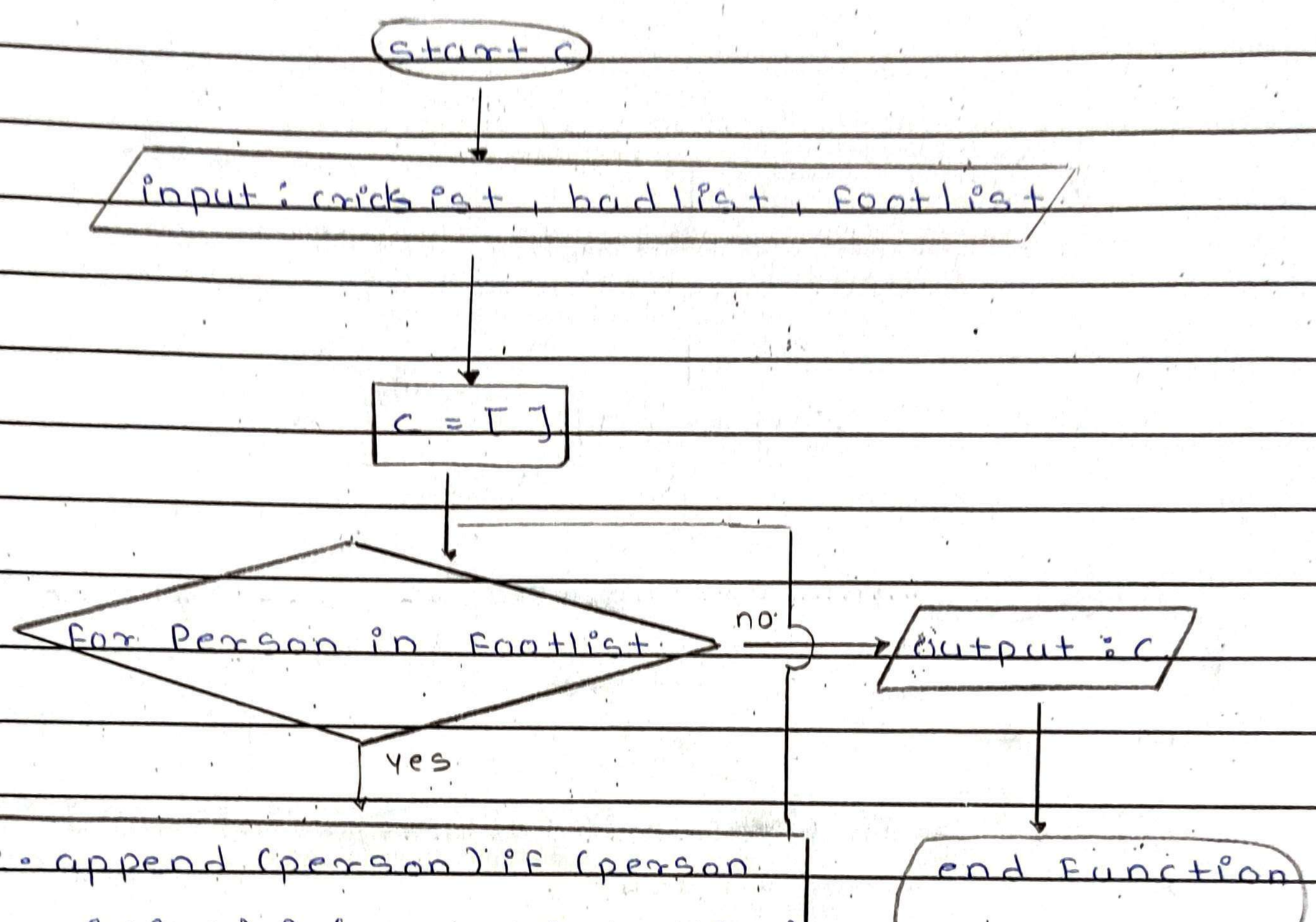
5) Number of students who play neither cricket nor badminton.

difference (Footlist - criclist) union difference (Footlist, badminton)

- Flowchart.



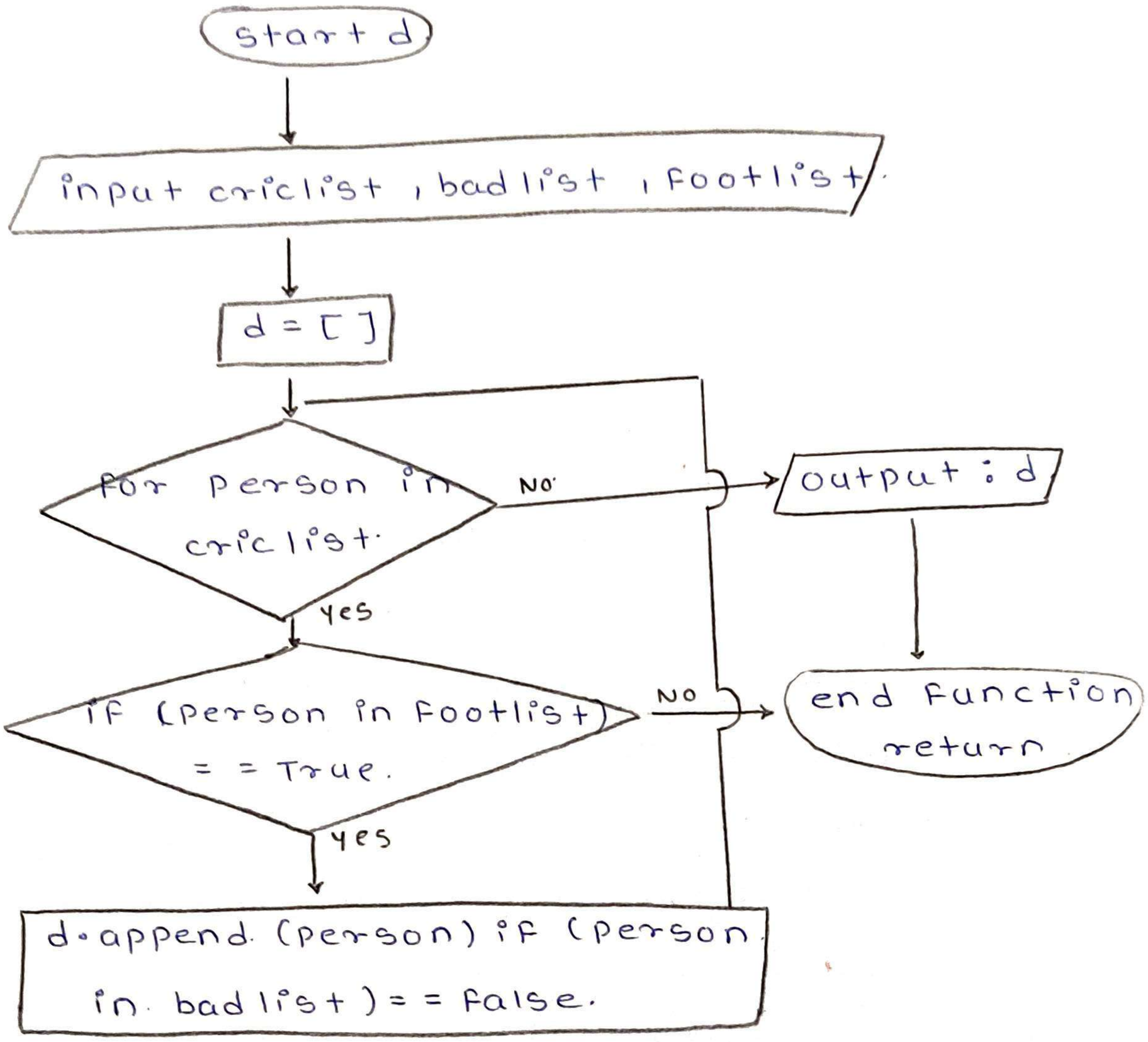

Start b
input criclist, badlist, Footlist.
b = [ ]
For person in criclist
NO
b.append (person) if (person in badlist) == false.
For person in badlist
NO
Yes
b.append (person) if (person in cric-list) == false.
O/P : b.
end Func
return



- conclusion :-

Thus, we have studied and implemented different set operations on lists successfully.

07/09/22


start d
input criclist , badlist , footlist
d = [ ]
for person in criclist.
NO
output : d
yes
if (person in footlist) = = True.
NO
end function return
yes
d.append (person) if (person in bad list) = = false.

Assigoment NO - 2

- Aim :- Program that computes the net amount of bank account based on transaction log from console input.

- Objective :- To study string operation.

- Problem statement :- Write a python program that computes the net amount of a bank account based transaction log from console input. The transaction log format is shown as following: D 100 W 200 (withdrawal is not allowed if balance is going negative. write functions for withdraw and deposite.) D means deposit while W means withdrawal. Suppose the following input is applied to the program.

1) D 300
2) D 300
3) W 200
4) D 100 Thus, the output should be : 500

- Theory :-

String are nothing but array of characters ended with null character. string is defined as an array of character or a pointer to character.

As string definition, we have two ways

to declare a string i.e.

1) char s[] = "string"

2) char str[20];

Strings can be declared using single, double or triple quotes.

input() function always return a string. Note that there are string which are strings but looks like a number or an integer.

Different string operation can be performed such as.

→ split();

→ strip();

→ Lstrip();

→ Rstrip();

1) split();

The split method splits a string into a list. one can specify the seperator, default seperator is blank space.

ex:- a = "hello world"

var = a.split(" ")

print(var)

o/p :- h, e, l, l, o, w, o, r, l, d

2) strip();

strip() method in python removes or truncates the given character from the begining and the

end of the original string. The default behaviour of the strip() method is to remove the white space from the beginning and at the end of the string.

Syntax :- string.strip([character])

ex :- x = " apple "

a = x.strip()

print (x)

o/p :- apple.

3) rstrip :-

The rstrip method removes any trailing character (characters at the end a string), space is the default trailing character to remove.

Syntax :- string.rstrip (character)

ex :- a = " * * * people * * * "

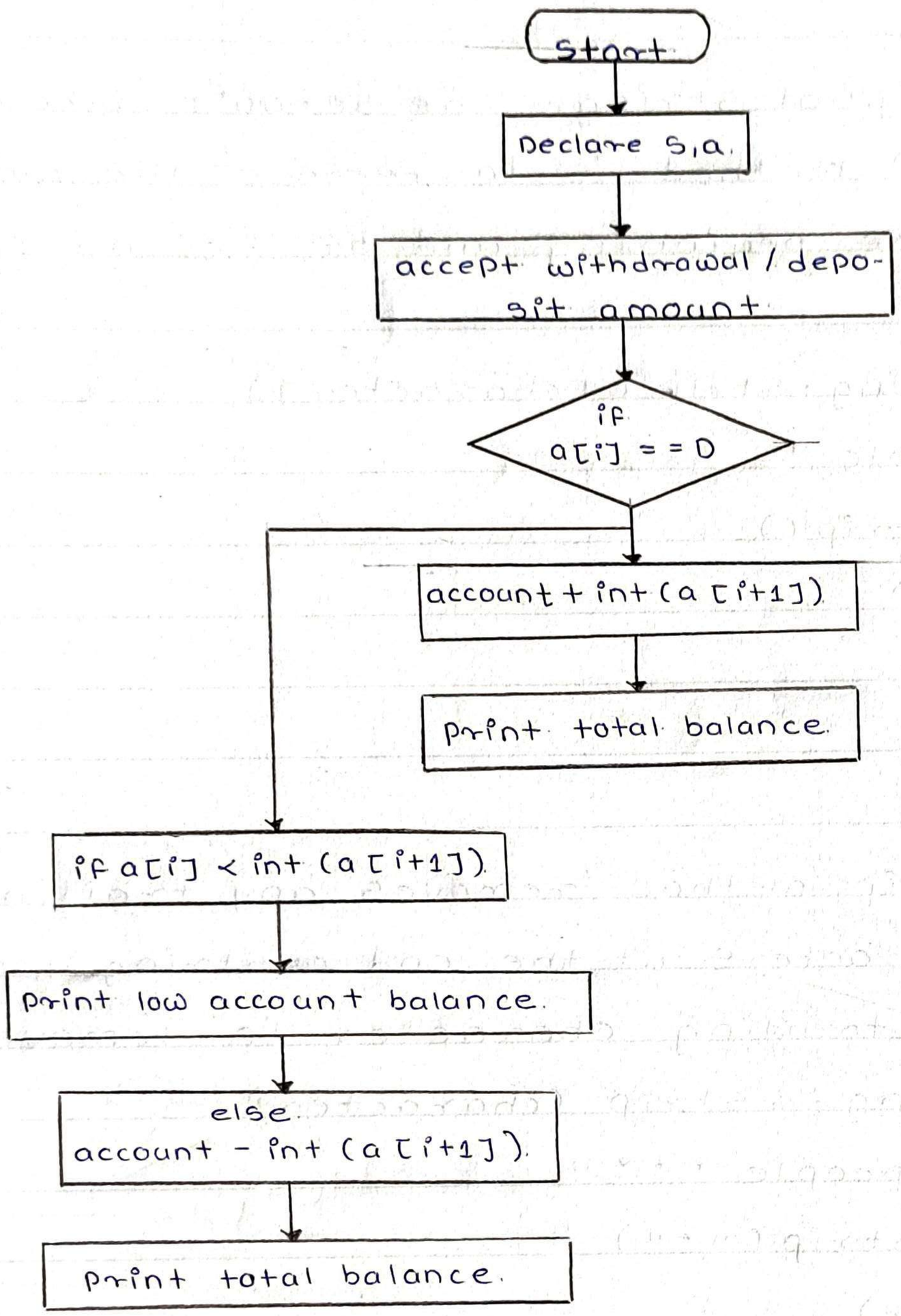
var = a.rstrip(" * ")

print (var)

o/p :- * * * people

4) lstrip :-

The lstrip() method return a copy of the string with leading characters removed (based on the strip argument passed). The lstrip() removes characters from the left base on the arguments (a string specifying the set of


Start
Declare s,a.
accept withdrawal / deposit amount
if
a[i] = = D
account + int (a [i+1])
print total balance
if a[i] < int (a [i+1])
print low account balance.
else.
account - int (a [i+1]).
print total balance.

characters to be removed).

Syntax :- string.lstrip([char])

ex :- a = " * * star "

var = a.lstrip(*)

print(var)

o/p :- star

To convert a string to integer in python use the int() function. This function takes two parameters: the initial string and the optional base to represent the data.

Syntax :- print(int("str"))

- Algorithm

1) Start.
2) Declare the variable as global s, a.
3) Input to enter the amount.
4) Input to seperate string by split function.

   a = s.split(" ")

5) print a.
6) Deposite withdrawal

→ declare variable as global amount.

→ account = 500.

→ 1) check if (a[i] = D).

```
if (a[i] = = D)
    account = account + int(a[i+1])
    print(("Rs" int(a[i+1]), "deposited")
```

print("account because is ", account)

2) check if (a[i] == w).

if (account i< int (a[i+1])).

print ("account balance is low").

else :

account = account - int (a[i+1]).

print("Rs", int(a[i+1]), "withdrawed")

print("account balance is ", account)

7) print ("total account balance is ", account)

8) stop.

- conclusion :-

Thus we have implemented the string operations on depositing and withdrawing the amount from an account.

14/09/22

Assignment NO- 43

- Title :- Program for string operation
- Objective :- To study & implement string operation
- Problem statement :- Write a python program to compute following operations on string :-

a) To display word with langest length.

b) To determines freq. of occurance of particular character in string.

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

d) To display index of 1st appearence of the substring.

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

- outcome :- student will be able to use array concepts for implementing string operations.
- Theory :- String are nothing but array of characters ended with null character ('\0'). This null character indicates end of strings. string are always enclosed by double quotes. Whereas, character is enclosed by single quotes.

ex :-

char string[20] = "Hello world";

char string [] = "Hello world";

- String function :- string.h header file supports all string functions. All the string functions are given below.

1) strcat() function is concatenates 2 given strings. It concatenates source string at end of destination string.

∴ Syntax :-

strcat(destination string, source string);

2) strcpy() function copies contents of one string into another string.

∴ Syntax :-

strcpy(destination string, source string)

ex :-

strcpy(str1, str2) → It copies contents of str2 into str1.

If destination str length is less than source string, entire source string value won't be copied into destination string.

3) strstr() function returns pointer to 1st occurrence of string in given string.

∴ Syntax :-

strstr(str1, str2);

→ strstr() func. is used to locate 1st occurrence of string "test" in the string.

4) strcmp() function in c compares 2 given string & return zero if they are same. If length of string 1 > string 2, it returns >0 value.

∴ Syntax :-

strcmp(str1, str2);

5) Strlen() function in c gives length of given string.

∴ Syntax :-

Strlen (str 1);

Strlen() function counts no. of character in a given string & returns integer value. bcoz, null character indicates the end of string in c.

- Algorithm :-

Algorithm (str 1, str 2)

- Input :- Two string

Begin consider str 1 & str 2 are 2 string

for i = 0 to str [i] ! = '\0'; //string length

print length of string (i)

for i = 0 to str1[i] ! = '\0' //string copy

str 2 = str 1

for i = 0 to str2 [i] ! = '\0'

print string (str2)

for i = 0 to str 1 [i] ! = '\0' //string concatinatio

for j = 0 to str 2 [j] ! = '\0'

str 1 [i] = str 2 [j]

str 2 [j] = '\0'

print string (str 1)

```
// string equal
j = 0
for i = 0 to str1[i] != '\0'
        if (str1[i] == str2[j])
                j++
        else
                string are not equal
// reverse of string
for i = 0 to str1[i] != '\0'
        1 == --i
for j = 1 to j >= 0
        print string (str1)
// substring of string
j = 0
for i = 0 to str1[i] != '\0'
        If (str2[j] == str1[i]) j++
        if (str2[j] != '\0')
        substring not found.
        else
        substring is found.
        }
End.
```

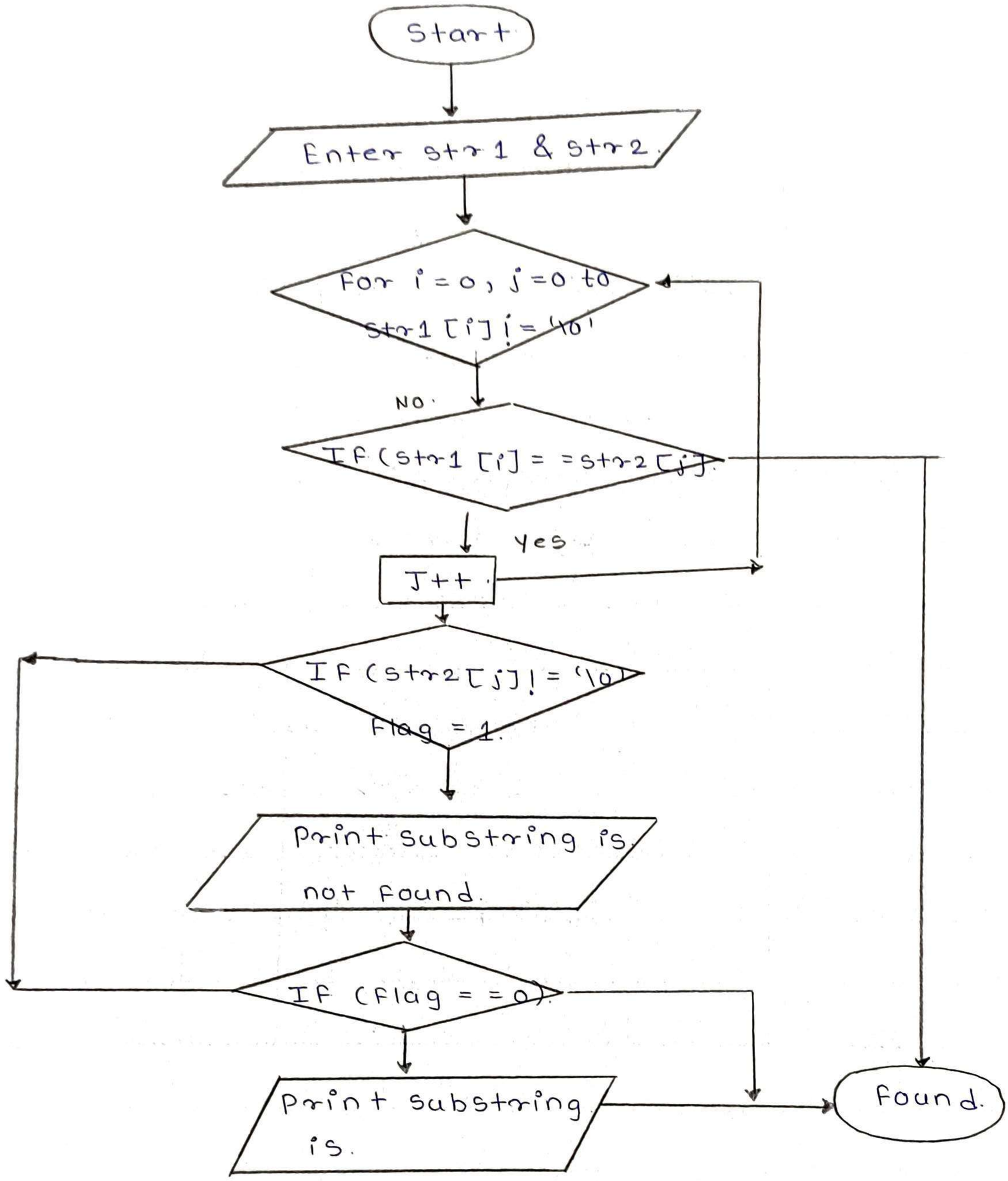
- Test cases :-

	Steps	Input	Expected result	Actual result	status
TD1	If character array not there	char c;	It must return complete string	But it return only 1st character from string	Fail
TD2	Length is in format of integer	Int l; l = str len(str1)	It must return length in form of integer	It is not printing in integer format	Fail
TD3	At time of concatenation after completion of 1st string it must append next str	Strcat (str1, str2)	1st string will read then str2 & then concat string. Display string.	yes it displayed concatenated string (str1 then str2).	pass

- conclusion :-

  Successfully implemented all string operations.

28/09/22


Start
Enter str1 & str2.
For i=0, j=0 to
str1 [i] != '\0'
NO.
If (str1 [i] = = str2 [j]).
Yes
J++.
If (str2[j] != '\0')
Flag = 1.
Print substring is not found.
If (Flag = = 0).
Print substring is.
Found.

Assignment No :- 4

- Title :- Program for matrix operations.
- Objective :- To study & implement matrix operations.
- Outcome :- Student will be able to use array data structure for implementation of matrix operations.
- Problem statement :- Write a python program for compute following computation on matrix : a) addition of 2 matrices.
  b) Substraction of 2 matrices.
  c) Multiplication of 2 matrices
  d) Transpose of a matrix.
- Theory :-

What are Multi-dimentional arrays?

C programming language allows multidimentional arrays. Here is the general form of a multidimentional array declaration -

type name [size 1] [size 2] --- [size N];

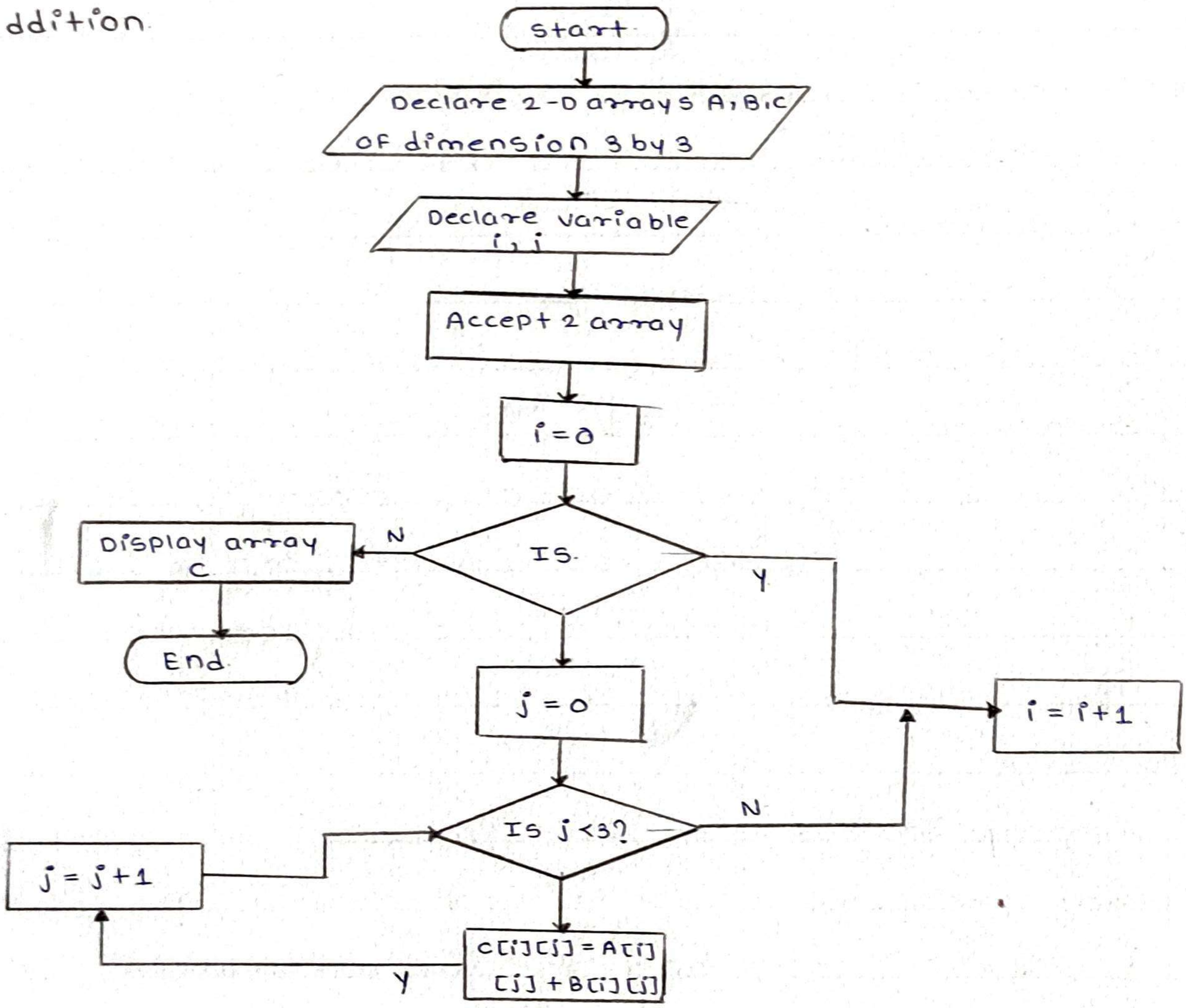
ex :-

int threedim [5] [10] [4];

2-D Arrays :- The simplest form of multi-dimentional array is the two-dimentional array. A 2D array is, in essence, a list of 1-D array

	column 0	column 1	column 2	column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

- Addition



To declare a 2-D array of size [x][y], you would write something as follows-

type arrayName [x][y];

Where type can be any valid c data type & arrayName will be valid c identifier. A 2-D array can be considered as table which will have x number of rows & y number of columns. A 2-D array a, which contains 3 rows & 4 columns can be shown as follows-

Thus, every element in the array a is identified by an element name of form a[i][j] where 'a' is the name of array & 'i' & 'j' are the subscripts that uniquely identify each element in 'a'.

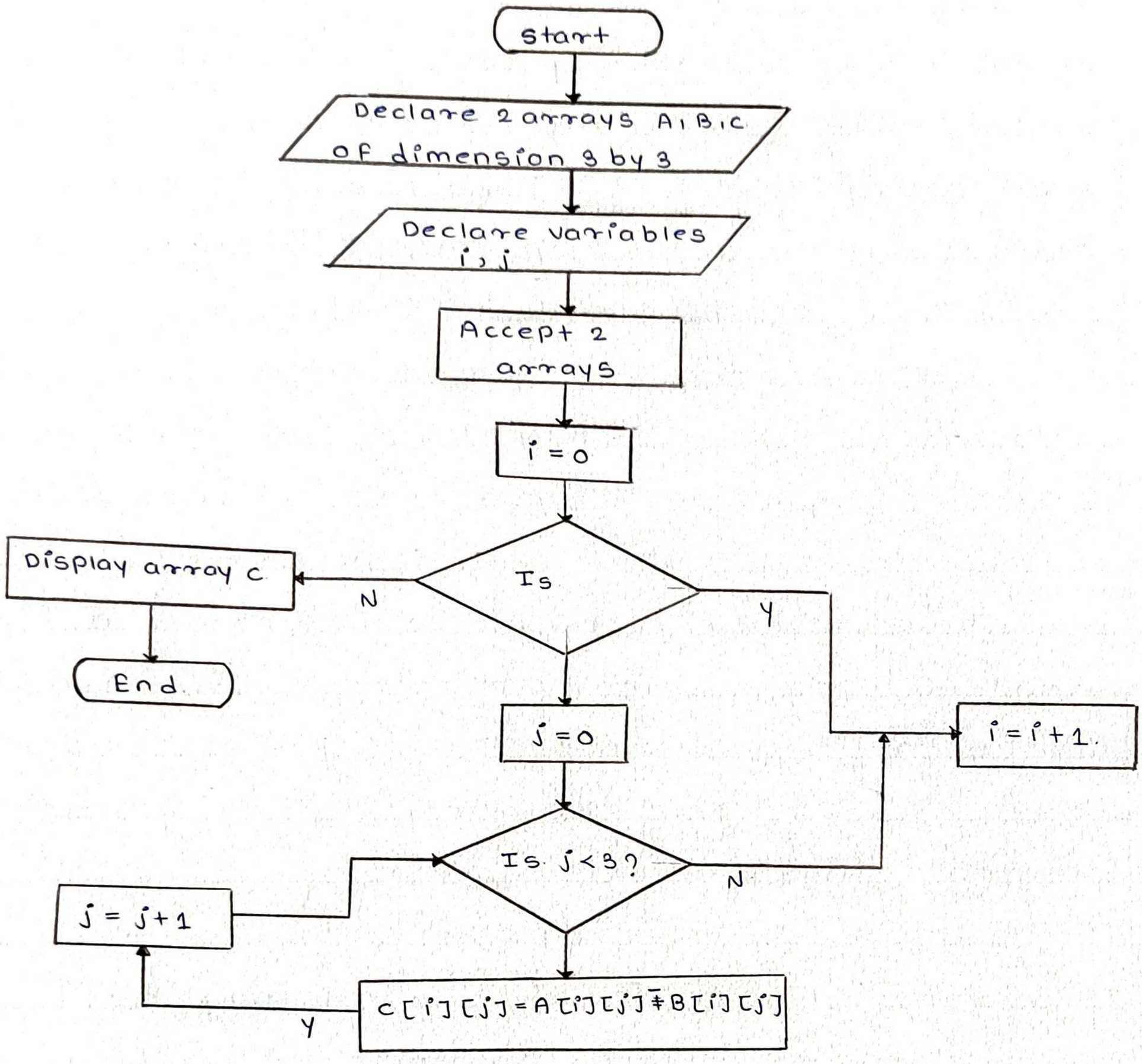
Initializing 2-D Arrays :-

Multi-dimentional arrays may be initialized by specifying bracted values for each row. Following is an array with 3 rows & each row has 4 columns.

```
int a[3][4] = {
  {0,1,2,3} , /* initializers for row indexed by 0*/
  {4,5,6,7} , /* initializers for row indexed by 1 */
  {8,9,10,11} , /* initializer for row indexed by 2 */
```

The nested braces, which indicates the intended row, are optional. The following initialization is equivalent to previous example-

- Substraction



int a[3][4] = {0,1,2,3,4,5,6,7,8,9,10,11};

Accessing 2-D Array elements

An element in a 2-D array is accessed by using subscripts, i.e, row index & column index of array. for example -

int val = a[2][3];

The above statement will take 4th element from the 3rd row of the array.

```
#include <stdio.h>
int main() {
    int a[5][2] = {{0,0},{1,2},{2,4},{3,6},{4,8}};
    int i,j;
    for (i=0; i<5; i++)
    {
        for (j=0; j<2; j++)
        {
            printf("a[%d][%d] = %d\n", i,j, a[i][j]);
        }
    }
    return 0;
}
```

When the above code is completed and executed, it produces following result -

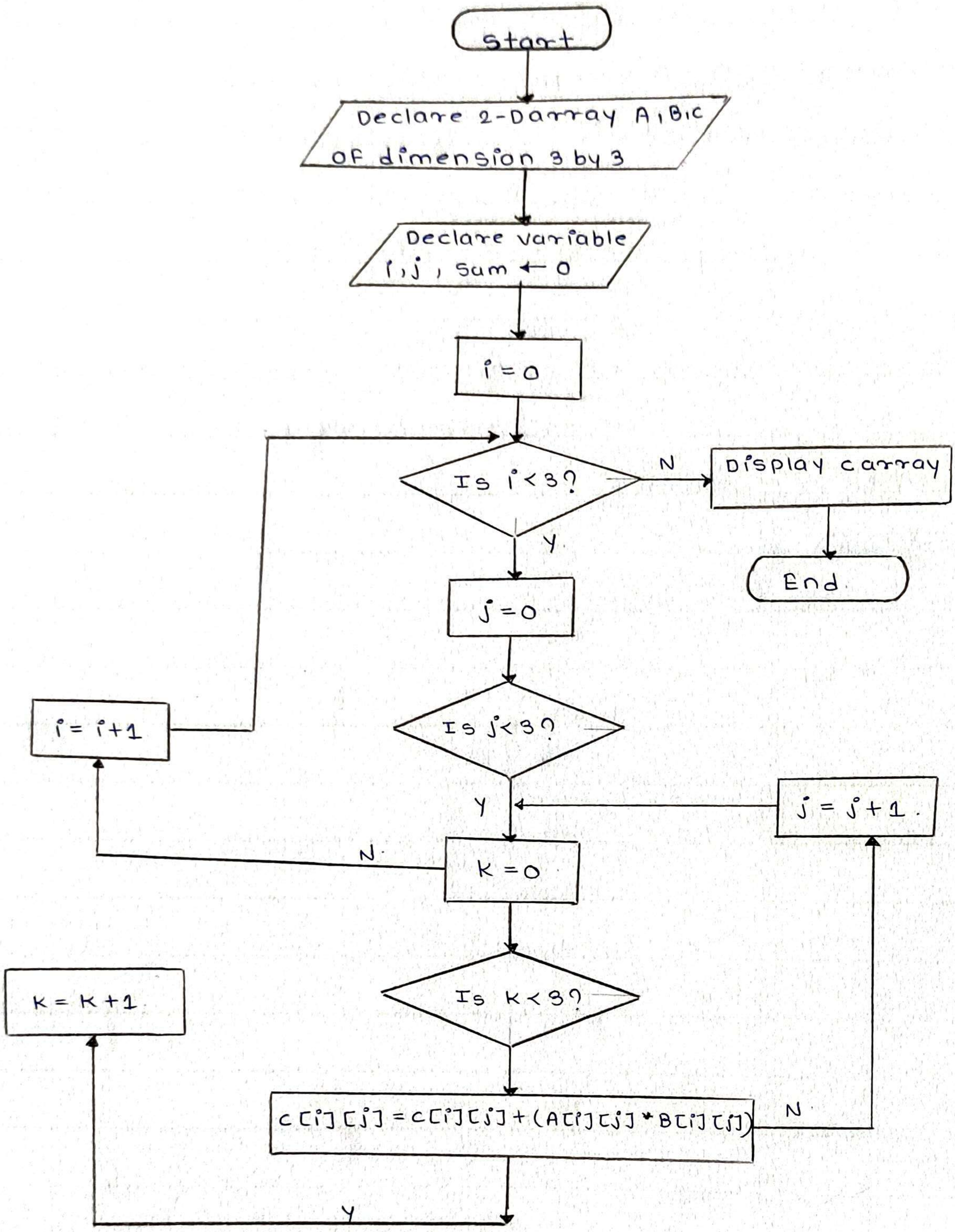
a[0][0] : 0

a[0][1] : 0

a[1][0] : 1

a[1][1] : 2

- Multiplication



a [2] [0] : 2

a [2] [1] : 4

a [3] [0] : 3

a [3] [1] : 6

a [4] [0] : 4

a [4] [1] : 8

- Implementation :-

Inputs :- 1) Elements of first 3x3 Matrix.

2) Elements of second 3x3 Matrix.

- Algorithms :-

Addition of 2 Matrices

1) let A [3] [3] & B [3] [3] be input matrices & c [3] [3] be resultant Matrix.

2) For i = 0 to 3

begin

For j = 0 to 3

begin

c [i] [j] = A [i] [j] + B [i] [j]

end

3) Display resultant Matrix.

For i = 0 to 3

begin

For j = 0 to 3

begin

print c [i] [j]

end

Substraction of 2 Matrices

1) let A[3][3] & B[3][3] be input matrices & c[3][3] be resultant matrix.

2) For i := 0 to 3
begin
For j := 0 to 3
begin
c[i][j] = A[i][j] - B[i][j]
end.

3) Display Resultant Matrix.
For i := 0 to 3.
begin
For j := 0 to 3
begin
print c[i][j]
end.

Multiplication of 2 Matrices.

1) let A[3][3] & B[3][3] be input matrices & c[3][3] be resultant matrix.

2) For i := 0 to 3
begin
For j := 0 to 3
begin
c[i][j] = 0
For k = 0 to 3
begin
c[i][j] = c[i][j] + A[i][k] * B[k][j]

end.

3) Display Resultant Matrix.

```
For i: = 0 to 3
begin
For j: = 0 to 3
begin
    print c[i][j]
end
```

Transpose of Matrix

1) let A[3][3] be a input matrix & c[3][3] be a resultant matrix.

```
2) For i: = 0 to 3
begin
For j: = 0 to 3
begin
    c[i][j] = A[j][i]
end
3) Display resultant matrix
For i:= 0 to 3
begin
    For j: = 0 to 3
    begin
        print c[i][j]
    end.
end.
```

Test cases

	Steps	Input	Expected Result	Actual Result	Status
TD1	Input Array	Matrices A & B	Addition of A & B	Addition of A & B	pass
TD2	Input Array	Matrices A & B	subtraction of A from B	Subtraction A from B	pass
TD3	Input Array	Matrices A & B	Multiplication of A & B	Multiplication of A & B	pass

- conclusion :-

Thus, we have studied two dimentional arrays and implemented different operations on matrices successfully.

14/10/22

Assigment NO - 5

- Aim :- Program for searching purpose using binary search.

- objective :- To study searching using Binary searching techniques.

- Problem statement :- Write a python program to store roll numbers of students array who attended training program in sorted order. Write function for searching whether particular student attended training program or not using Binary search.

- Outcome :- student will be able to use binary search concepts and technique.

- Theory :-

Binary search :- Binary search is a searching techniques that works on sorted array. Instead of comparing each element of the array with the required element the binary search algorithm repeatedly divides the array into sub array and then searches for the required element in the sub-array.

Binary search is a searching algorithm in which the list of elements is divided

into two sublists and the key element is compared with the middle element. If the match is found then, the location of middle element is returned otherwise, we search into either of the halves (sublists) depending upon the result produced through the match.

This algorithm is considered as an efficient searching algorithm.

Binary search method involves computation for finding the middle element.

- Advantages of Binary search

1) It is efficient technique.
2) It is better than a linear search algorithm since its run time complexity is O(log N).
3) At each iteration, the binary search algorithm eliminates half of the list and significantly reduces the search space.
4) The binary search algorithm works even when the array is rotated by some position and finds the target element.

- Disadvantages of Binary search

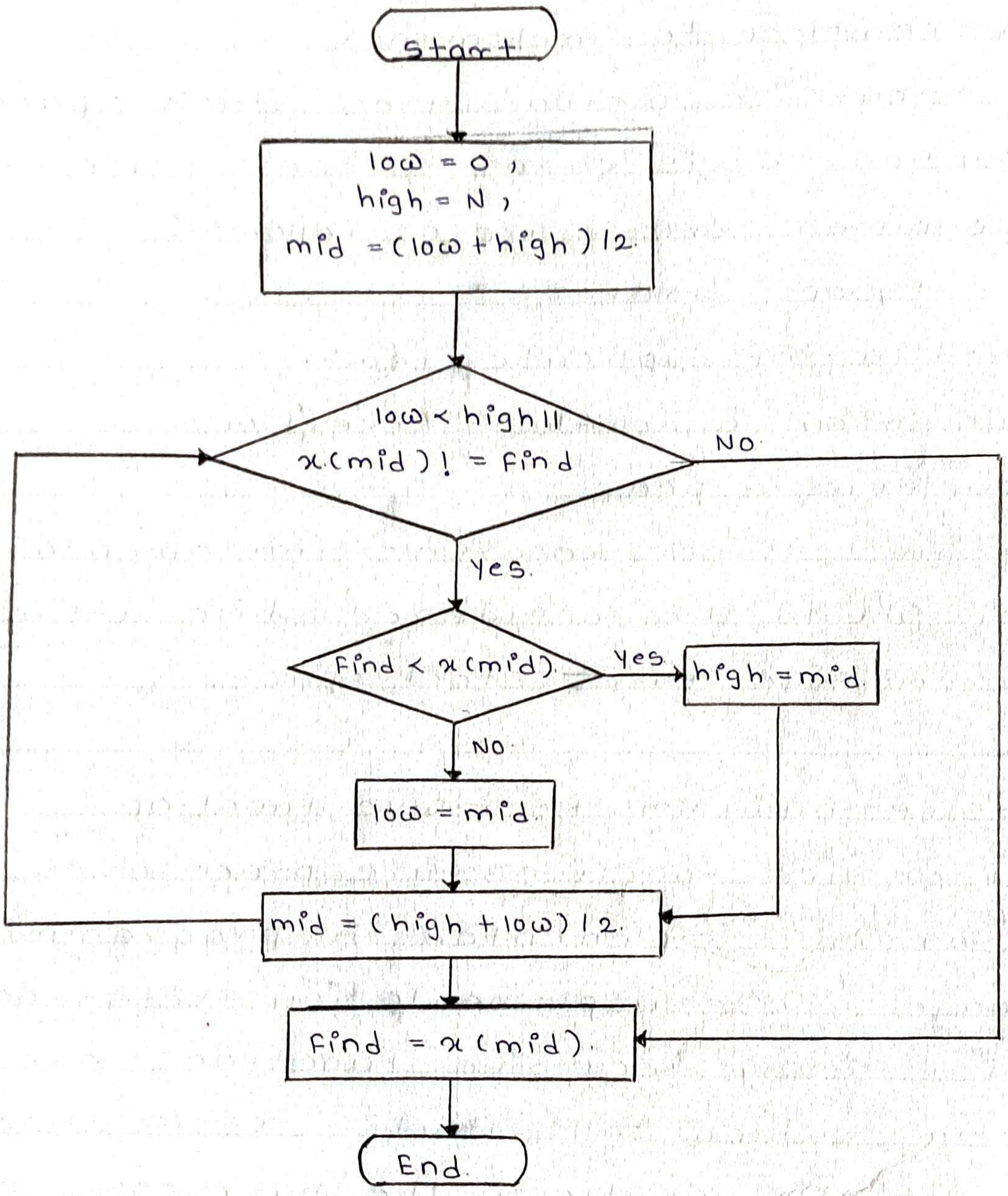
1) It requires specific ordering before applying the method.

2) It is complex to implement.
3) The recursive method uses stack space.
4) Binary search is error-prone. Some of the common errors are as follows:
a) off-by one error.
b) Handling of duplicate items.
c) Numerical underflows / overflows.
d) coaching is poor.
5) Recursive vs non-recursive implementation, which should be considered while desiging as recursive takes stack space.

- Recursive and Non-Recursive function.

Recursive functions are procedures or sub-routines implemented in programming language whose implementation reference itself. Simply Recursive functions are functions which calls itself. It is always made up of 2 portions the base case and recursive case. The base case is the condition to stop the recursion. The recursive case is the part where the function calls itself.

Non-Recursive function are procedures or sub-routines implemented in a programming


Start
low = 0 ,
high = N ,
mid = (low + high) / 2.
low < high ||
x(mid) ! = find
No.
Yes.
Find < x(mid).
Yes.
high = mid.
No.
low = mid.
mid = (high + low) / 2.
find = x(mid).
End.

language whose implementation does not reffer itself. Non-recursive function defined from a particular formal model of computable functions using primitive recursion and 'μ' operator.

- Algorithm for Binary search

```
1) if (low > high)
2) return ;
3) mid = (low + high) / 2 ;
4) if (x = = a [mid]).
5) return (mid) ;
6) if (x < a [mid]).
7) search for x in a[low] to a [mid - 1] ;
8) else ;
9) search for x in a [mid + 1] to a [high] ;
```

- Conclusion :-

Successfully implemented Binary search concept and techniques.

30/10/22

DYPSOETP, PUNE

Assignment NO- 6

- Title :- Program for sorting techniques.

- Objective :- To study sorting algorithm and its complexity.

- Problem statement :- 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 scores.

- Outcome :- Student will understand and implement sorting techniques.

- Theory :-

Sorting -

→ Refers to arranging data in particular format. Sorting algorithm specifies the way to arrange data in particular order.

→ Most common orders are numerical or lexicographical (also known as lexial order, dictionary order, alphabetical) order.

Bubble sort

→ This sorting algorithm is comparison based

algorithm in which each pair of adjacent elements is compared and elements are swapped if they are not in order.

→ This algorithm is not suitable for large data sets as its average and worst case complexity are of $O(n^2)$ where n are no. of items.

complexity

Worst case performance $O(n^2)$

Best case performance $O(n)$

Average case performance $O(n^2)$

Algorithm

We assume list is an array of n elements. We further assume that swap function, swaps the values of given array elements.

Input :- Unsorted elements.

output :- Sorted elements.

```
begin Bubble sort (list)
    for all elements of list if
        list [i] > list [i+1]
        swap (list [i], list [i+1])
        end if
    end for
return list.
end Bubble sort.
```

Selection sort

→ This sorting algorithm is in-place comparison based algorithm in which the list is divided into two parts, sorted part at left and unsorted part at right end.

→ Initially sorted part is empty and unsorted part is intire list.

→ Smallest element is selected from the unsorted array and swapped with leftmost element and that element becomes part of sorted array.

→ This process continues moving unsorted array boundary by one element to the right.

→ This algorithm is not suitable for large data sets as its average and worst case complexity are of $O(n^2)$ where n are no. of items.

complexity

Worst case performance $O(n^2)$

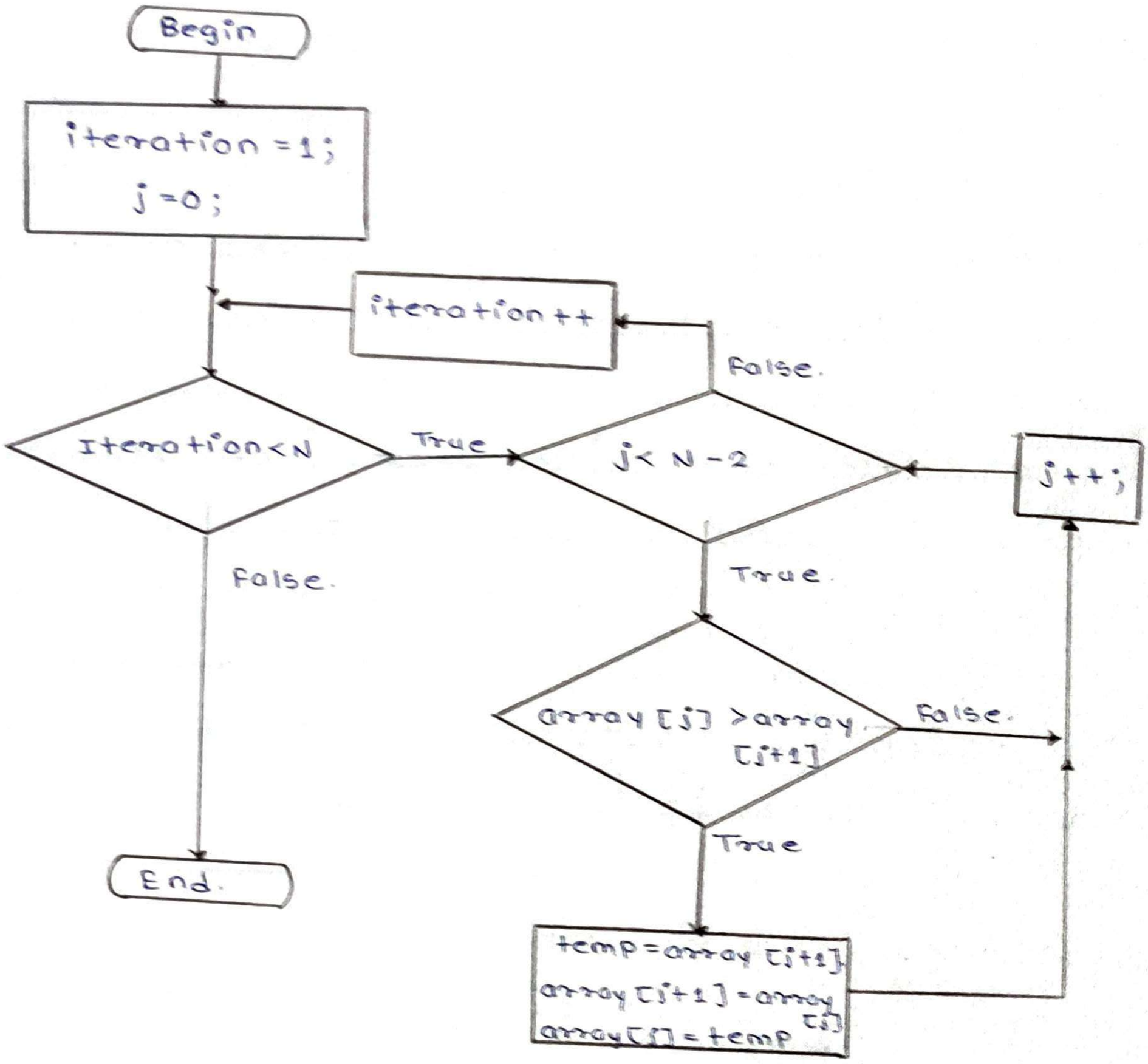
Best case performance $O(n^2)$

Average case performance $O(n^2)$

Algorithm

Step 1 :- Set MIN to location 0.

Bubble sort.


Begin
iteration = 1;
j = 0;
iteration++
False.
Iteration<N
True
j< N - 2
j++;
False.
True.
array [j] >array [j+1]
False.
True
End.
temp = array [j+1]
array [j+1] = array [j]
array [j] = temp

Step 2 :- Search the minimum element in the list.

Step 3 :- Swap with value at location MIN

Step 4 :- Increment MIN to point to next element.

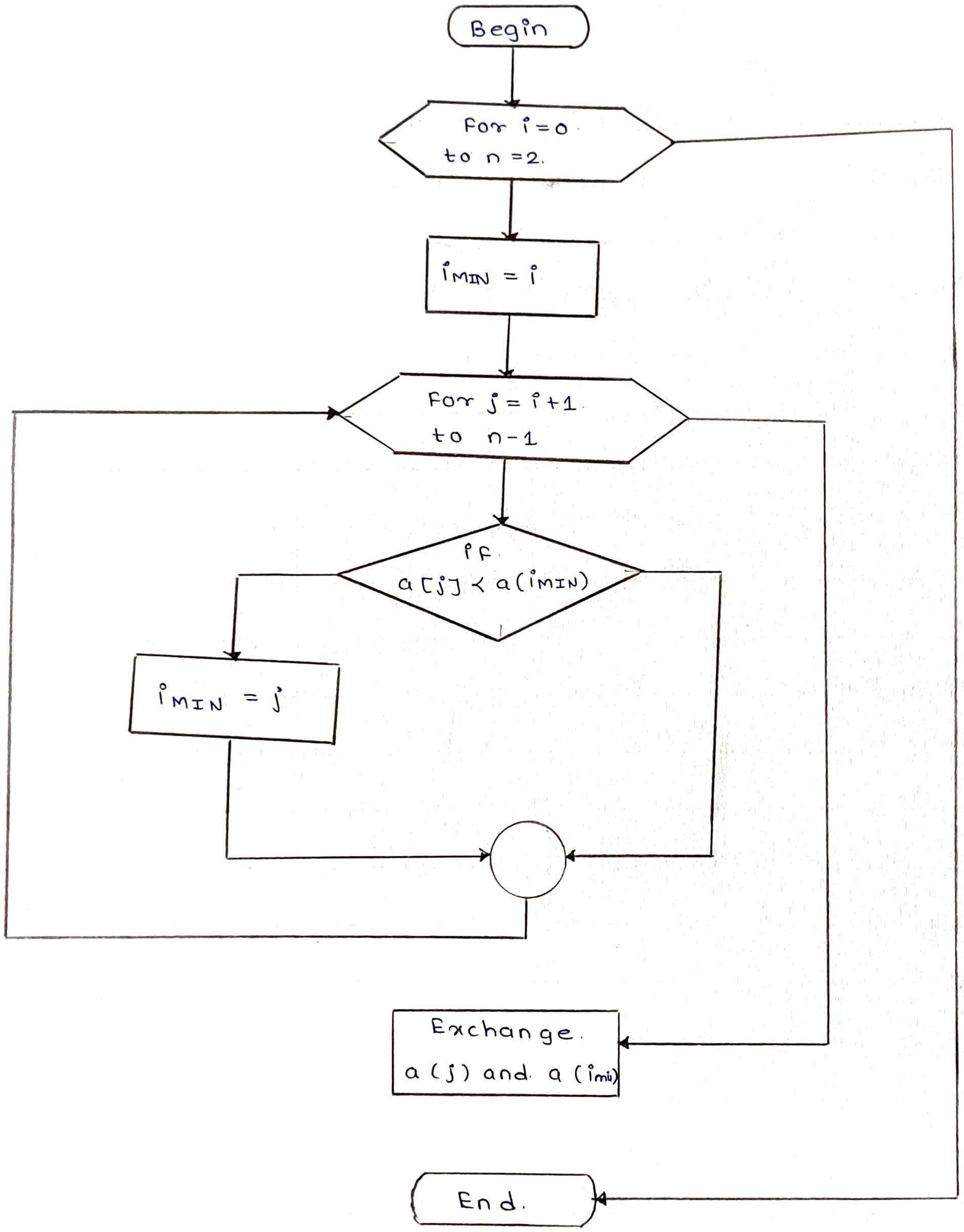
Step 5 :- Repeat until list is sorted.

- Test cases

	Steps	Input	Expected Result	Actual result	Status
ID1	Apply bubble sort	{5, 1, 6, 2, 4, 3}	{1, 2, 3, 4, 5, 6}	{1, 2, 3, 4, 5, 6}	Pass
ID2	Apply Bubble sort	{5, 1, 6, 2, 4, 3}	{1, 2, 3, 4, 5, 6}	{1, 6, 3, 2, 5, 4}	Fail
ID3	Apply selection sort	{3, 6, 1, 8, 4, 5}	{1, 3, 4, 5, 6, 8}	{1, 3, 4, 5, 6, 8}	pass

- conclusion :-

Successfully implemented and analyzed selection sort and Bubble sort.

02/11/22

- Selection sort