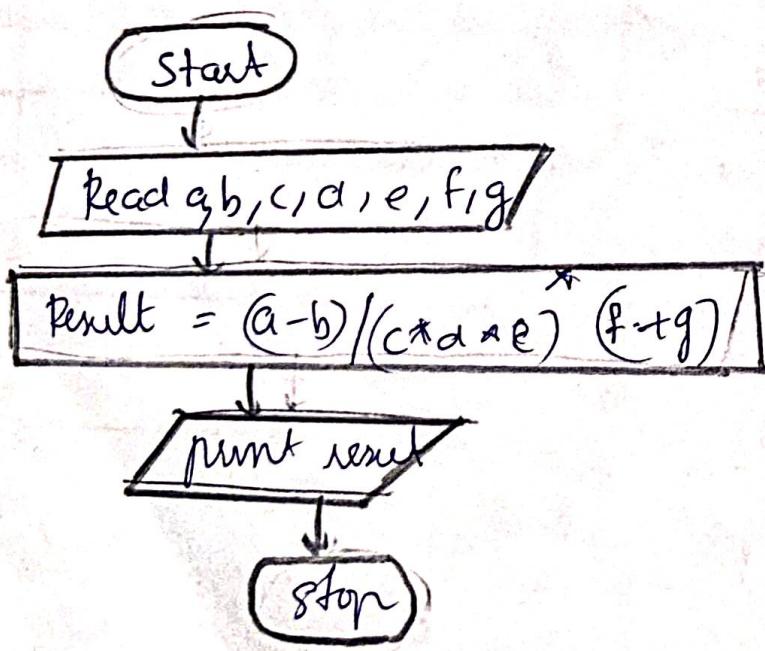
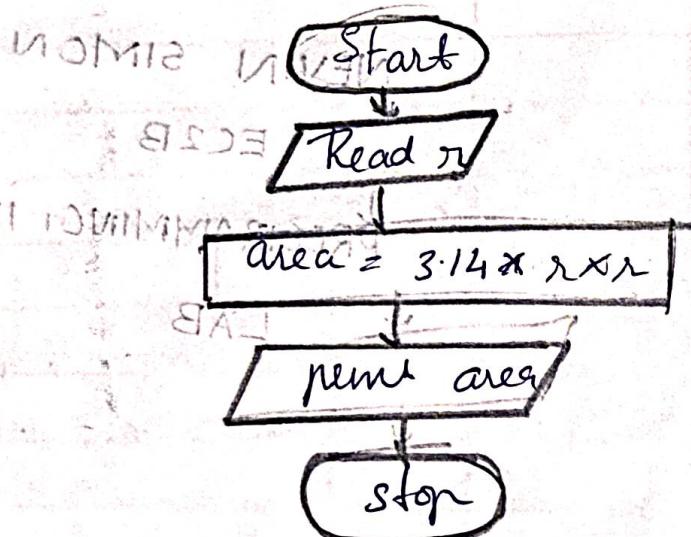
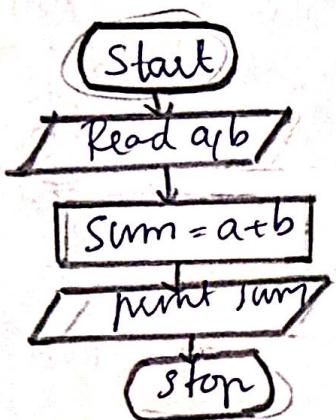
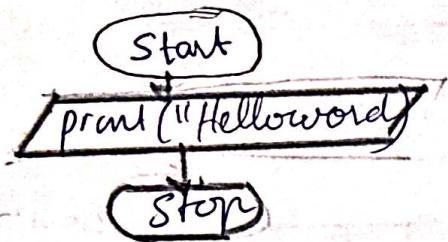


## FLOWCHART



# EXPERIMENT - 1

## FAMILIARIZATION OF CONSOLE I/O IN C

Aim:

Familiarization of console I/o and operation in C.

- (i) Display "Hello world".
- (ii) Read two number , add them and display their sum.
- (iii) Read the radius of a circle , calculate its area and display it .
- (iv) Evaluate the arithmetic expression and display its solution  $(a-b/c*d+e)* (f+g)$

### PROGRAM

```
(i) #include <stdio.h>
int main()
{
    printf("Hello world");
    return 0;
}
```

```
ii) #include <stdio.h>
int main()
```

```
{
    int a,b,sum;
    a=2;
    b = 7;
    sum=a+b;
```

```
printf("The sum of %d and %d : %d",a,b,sum)
```

## OUTPUT

(i) Hello world

ii) The sum of 2 and 7 is 9.

iii) Enter the radius : 10

area of the circle is 314.

iv) Enter seven numbers 10 20 20 20 20 20 20

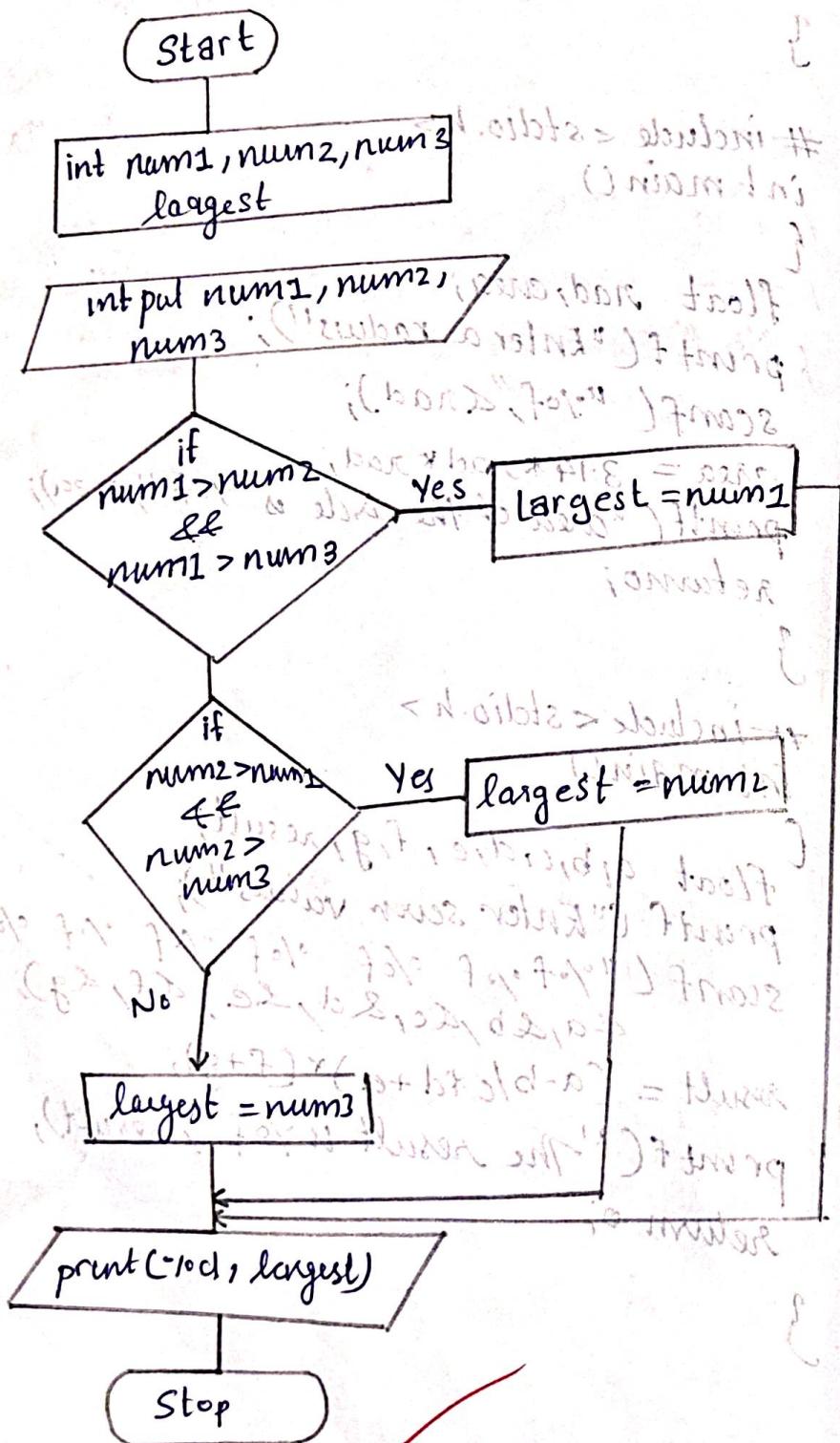
The result is 400.00

```
    return 0;  
}
```

```
iii) #include <stdio.h>  
int main()  
{  
    float rad, area;  
    printf("Enter a radius");  
    scanf("%f", &rad);  
    area = 3.14 * rad * rad;  
    printf("Area of the circle is %f", area);  
    return 0;  
}
```

```
iv) #include <stdio.h>  
int main()  
{  
    float a, b, c, d, e, f, g, result;  
    printf("Enter seven values");  
    scanf("%f %f %f %f %f %f %f",  
        &a, &b, &c, &d, &e, &f, &g);  
    result = (a - b / c * d + e) * (f + g);  
    printf("The result is %f", result);  
    return 0;  
}
```

## FLOWCHART



OUTPUT  
 Enter three numbers 10 20 30  
 The largest number is : 30

## EXPERIMENT-2

### LARGEST AMONG THREE NUMBERS

TUTORIAL

Aim:

Read 3 integer values and find the largest among them.

Program:

```
#include <stdio.h>
int main()
{
    int num1, num2, num3, largest;
    printf("Enter three numbers:");
    scanf("%d %d %d", &num1, &num2, &num3);
    if (num1 >= num2 & num1 >= num3)
    {
        largest = num1;
    }
    else if (num2 >= num1 & num2 >= num3)
    {
        largest = num2;
    }
    else
    {
        largest = num3;
    }
    printf("The largest number is : %d \n", largest);
    return 0;
}
```

Result: program to find largest among three number is executed and found successful

## EXPERIMENT - 2

### OUTPUT

Enter a number : 5

5 is ~~not~~ a prime number

Enter a number : 12

12 is not a prime number

### EXPERIMENT - 3

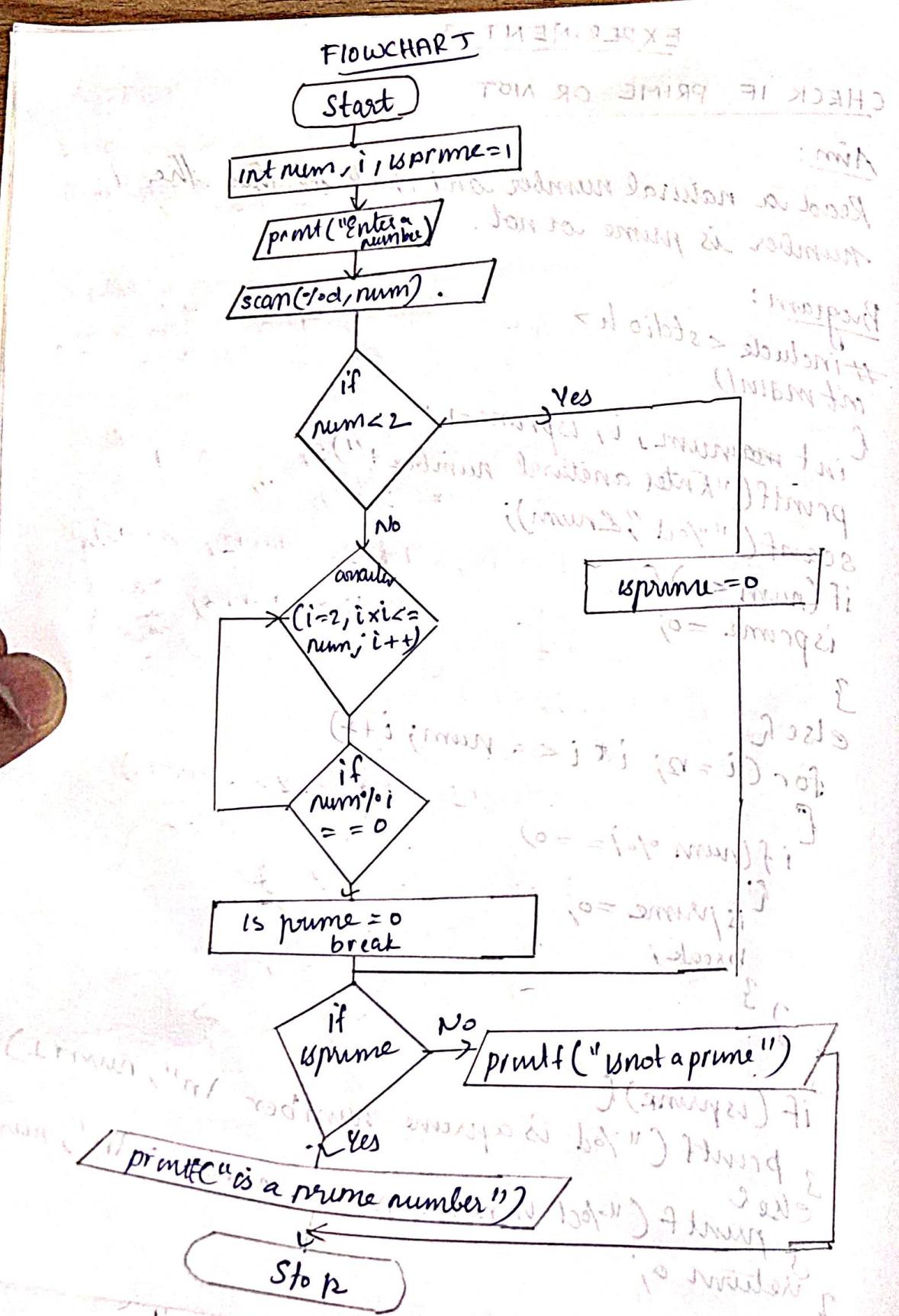
#### CHECK IF PRIME OR NOT

Aim:

Read a natural number and check whether the number is prime or not.

Program:

```
#include <stdio.h>
int main()
{
    int num, i, isprime = 1;
    printf("Enter a natural number : ");
    scanf("%d", &num);
    if (num <= 2)
        isprime = 0;
    else
        for (i = 2; i * i <= num; i++)
            if (num % i == 0)
                {
                    isprime = 0;
                    break;
                }
    if (isprime)
        printf("%d is a prime number\n", num);
    else
        printf("%d is not a prime number\n", num);
    return 0;
}
```



Result

program to check the number is prime or not

executed and found successful

("prime or not") Function

~~E~~

int prime (int num, int &ans)  
 {  
 if (num < 2) ans = 0; // not prime  
 for (int i = 2; i < num; i++)  
 {  
 if (num % i == 0) ans = 1; // not prime  
 else ans = 0; // prime  
 }  
 return ans;  
}

prime (19)

prime (7)

(prime or not) Function  
 (prime or not) Function

ans = 0  
 ans = 1  
 ans = 0

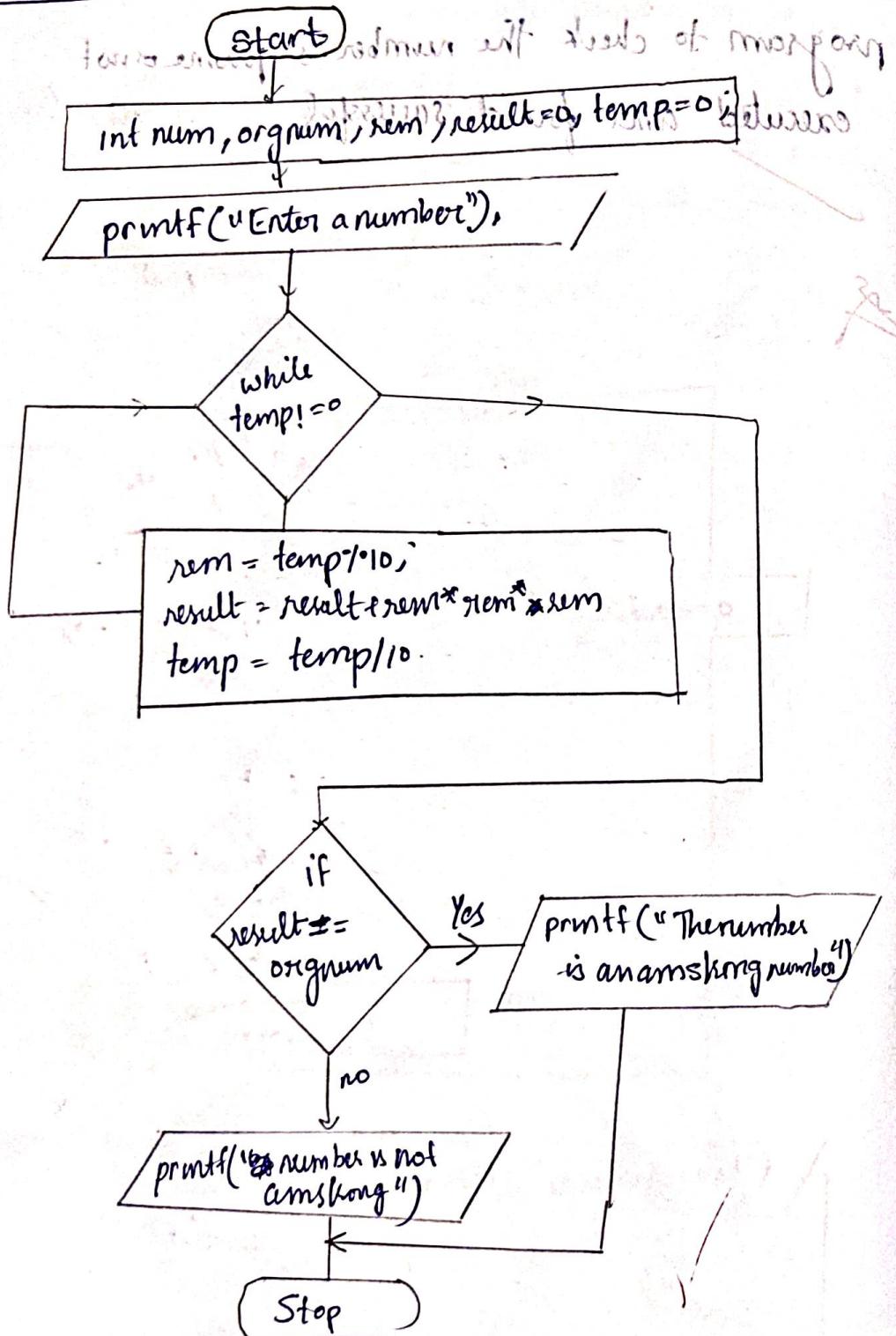
ans

prime (19)  
 ("prime or not") Function

ans = 1

## FLOWCHART

Urwa



## EXPERIMENT NO:4

### CHECK IF AMSTRONG NUMBER OR NOT

Aim: Read a number check if the number is amstrong or not

Program:

```
#include <stdio.h>
int main()
{
    int num, orgnum, rem, result=0, temp=0;
    printf("Enter a number");
    scanf("%d", &num);
    orgnum = num;
    temp = num;
    while(temp!=0)
    {
        rem = temp%10;
        result = result + rem*rem*rem;
        temp = temp/10;
    }
    if(result == orgnum)
    {
        printf("%d is an amstrong number\n", num);
    }
    else
    {
        printf("%d is not an amstrong number\n", num);
    }
}
```

## EXERCISE 4

output

enter a number 153

153 is an amstrong number

enter a number 22

22 is not an amstrong number

$\text{int } n = \text{input}();$   
if ( $n > 0$ ) {  
 int sum = 0;  
 for ( $i = 1$ ;  $i \leq n$ ;  $i++$ ) {  
 int r =  $n \% i$ ;  
 sum += r \* r \* r;  
 }  
 if ( $sum == n$ )  
 cout << "The number is an amstrong number";  
 else  
 cout << "The number is not an amstrong number";  
}

$\text{int } n = \text{input}();$   
int sum = 0;  
for ( $i = 1$ ;  $i \leq n$ ;  $i++$ ) {  
 int r =  $n \% i$ ;  
 sum += r \* r \* r;  
}  
if ( $sum == n$ )  
 cout << "The number is an amstrong number";  
else  
 cout << "The number is not an amstrong number";

$\text{int } n = \text{input}();$   
int sum = 0;  
for ( $i = 1$ ;  $i \leq n$ ;  $i++$ ) {  
 int r =  $n \% i$ ;  
 sum += r \* r \* r;  
}  
if ( $sum == n$ )  
 cout << "The number is an amstrong number";  
else  
 cout << "The number is not an amstrong number";

$\text{int } n = \text{input}();$   
int sum = 0;  
for ( $i = 1$ ;  $i \leq n$ ;  $i++$ ) {  
 int r =  $n \% i$ ;  
 sum += r \* r \* r;  
}  
if ( $sum == n$ )  
 cout << "The number is an amstrong number";  
else  
 cout << "The number is not an amstrong number";

return 0;  
}

Output

45042

45042

Result

The program to check the number is armstrong or not  
is executed and found successful

function to calculate sum

$n > i$

$n$

process of iteration

$c = 3$

$1+3=4$

$c$

$c$

[i] next step

$c = 1$

$i$

$n$

[i]  $(nmod + nmod^2) == n$

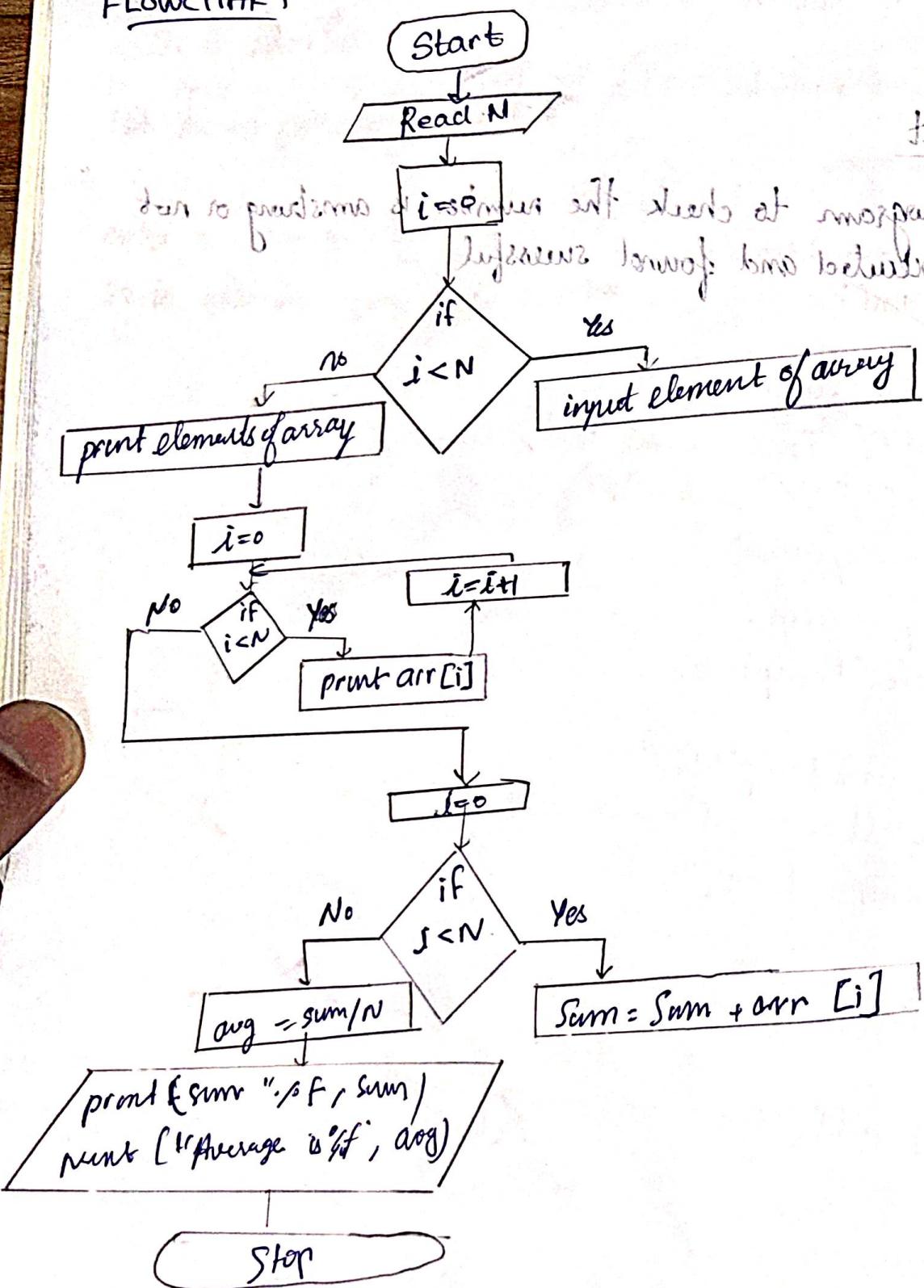
$n > 2$

if true = true

(true, if not Armstrong)  
(odd, if it's Armstrong) done

7012

## FLOWCHART



## EXPERIMENT NO: 5

### FIND THE SUM AND AVERAGE OF ELEMENTS IN AN ARRAY

Aim : Read n integers store them in an array find their sum and average.

#### Program

```
#include<stdio.h>
int main()
{
    float a[100], sum=0, avg;
    int i, n;
    printf("Enter the size of array: ");
    scanf("%d", &n);
    printf("Enter the elements of the array:\n");
    for(i=0; i<n; i++)
    {
        printf("a[%d] = ", i);
        scanf("%f", &a[i]);
    }
    for(i=0, i<n, i++)
    {
        sum = sum + a[i];
    }
    avg = sum/n;
    printf("Sum is %f\n", sum);
    printf("Average is %.f", avg);
    return 0;
}
```

## EXPERIMENT NO. 5

### OUTPUT

LINQ THE SUM OF THE ELEMENTS IN THE ARRAY : 5  
ENTER THE SIZE OF THE ARRAY : 5  
ENTER THE ELEMENTS OF THE ARRAY : 1 2 3 4 5

a[0] = 1

a[1] = 2

a[2] = 3

a[3] = 4

a[4] = 5

the sum of the elements in the array is 15.000000000000002

the average of the elements in the array is 3.0000000000000004

i((i < "greater than or equal to elements with value") ? true  
: ((i > "less than or equal to elements with value") ? false  
: (++i > i) ? true  
: (++i < i) ? false

i((i < "[value] & ")) ? true  
: i("[value] & ") ? false

i((i < "[value] & ")) ? true  
: i("[value] & [value] = [value] & ") ? false

i("value" & " for " & "value") ? true  
: i("for " & "is greater than or equal to value") ? true  
: i("value" & " is greater than or equal to value") ? true  
: i("value" & " is less than or equal to value") ? true

## Result

MHTGOSA

The program to read n elements in array and to find sum and average is executed and found successful.

Each function upto sum & avg is used once.

1 - main application

2 - user input . 2

Input is given by terminal. It takes space separated  
and each terminal itself may contain more or  
less of digits followed by terminal mark .  
For two decimal terms terminal will be return - return

-return

After two decimal marks, if digits in return then  
it goes on

return : returns to a "main" , digit ends . p

opt2 . o1

## ALGORITHM

1. Start
2. Read num
3. Using for loop, iteratively read elements of array num times.
4. Initialize index = -1
5. Read search.
6. Using for loop, check if element of array is equal to search starting from first element num times
7. If an element becomes equal to search  
order = index of the element and break out of the loop
8. If index is equal to -1, search does not exist in array
9. else print, "Search" is at order : index + 1
10. Stop

## EXPERIMENT NO: 6

### FIND AN ELEMENT OF ARRAY USING LINEAR SEARCH

Aim: Read n integers, store them in an array and search for an element in the array using an algorithm for linear search.

Program:

```
#include <stdio.h>
int main()
{
    int array[100], search, i, num;
    printf("Enter the number of elements in array \n");
    scanf("%d", &num);
    printf("Enter %d numbers \n", num);
    for (i=0; i<num; i++)
        scanf ("%d", &array[i]);
    printf("Enter the number to search \n");
    scanf ("%d", &search);
    for (i=0; i<num; i++)
    {
        if (array[i] == search)
        {
            printf("%d is present at location %d \n", search, i+1);
            break;
        }
    }
    if (i == num)
        printf("%d is not present in array \n", search);
}
return 0;
```

## EXPERIMENTAL

L1D AN ELEMENT OF ARRAY WHICH INDEX IS

### OUTPO

Enter the no of elements : 5  
enter 5 elements

1  
2  
3  
4  
5

enter the element to search : 4

element 4 found at position 4

```

if(i < min) {
    cout << "Element not found" << endl;
}
else if(i == max) {
    cout << "Element found" << endl;
}
else {
    cout << "Element found" << endl;
}

```

```

cout << "Element found" << endl;
}
else if(i == max) {
    cout << "Element found" << endl;
}
else {
    cout << "Element found" << endl;
}

```

Digitized by srujanika@gmail.com

## Result

The program to read a integers on array and to search element is executed and found successful

~~alpha~~ one of leaves (now i understand  
it's ~~alpha~~ ~~beta~~ C<sub>2</sub>H<sub>5</sub> CH<sub>2</sub> Cl) - > i. same C<sub>2</sub>H<sub>5</sub> + Cl  
form before and during  
9012

## ALGORITHM

1. Start
2. Declare integer  $n, j, i, \text{swaps}$  and array [100] ~~array element~~
3. input value of  $n$  ~~and values of elements in array~~
4. for  $i < n$ ; input value for array[i] ( $i=0$ ) ~~information~~
5. Repeat step 4 until  $i=n$
6. Initialise  $i$  and  $j$  equal to zero
7. for  $i < n-1$  and  $j < n-i-1$ ; if array[j] > array[i+1]
8. print the sorted array
9. Stop.

## EXPERIMENT - 7

### BUBBLE SORT

AIM :  
To read n integers, store them in an array and elements using bubble sort algorithm

#### PROGRAM

```
#include < stdio.h >
int main()
{
    int n, j, i, array[100], swap;
    printf("Enter no of elements : \n");
    scanf("%d", &n);
    printf("Enter %d integers : \n", n);
    for (i = 0; i < n; i++)
    {
        scanf("%d", &array[i]);
    }
    for (i = 0; i < n - 1; i++)
    {
        for (j = 0; j < n - i - 1; j++)
        {
            if (array[j] > array[j + 1])
            {
                swap = array[j];
                array[j] = array[j + 1];
                array[j + 1] = swap;
            }
        }
    }
}
```

EXPERIMENT

BASIC SORTING

OUTPUT

enter the no of elements of array in base of  
multiple for sorted form : NVA

5

enter the integers:

1

45

23

5

7

sorted list in ascending order:

1

5

7

23

45

METHODS

$\leftarrow$  No. of elements (N)

(Initialising

(pow, loc1, loc2, i, j) + memory

variables from main() + memory

(n, "b.o.") + memory

(n, "r") : register to ("return") + memory

(++i; (n > i) (i = i)) not

{([i] <= [i] + 1) + memory}

(++i; (i - n > i) (i = i)) not

(++i; i - n > i) (i = i) not

{([i] <= [i] + 1) + memory}

{[i] <= [i] + 1} + memory

{[i] <= [i] + 1} + memory

{[i] <= [i] + 1} + memory

```

printf("Enter the sorted list in ascending order: ");
for (i=0; i<n; i++)
    printf("%d \n", array[i]);
return 0;
}

```

Result

The program to read  $n$  integers and store them using bubble sort algorithm is executed and found successful.

*[Handwritten notes]*

et loops  $\text{f}_{\text{w}}[i] \text{ visit}$ ; left grid = 3 sets

left grid = 3 sets

"marking top w/ green" true

"marking green" many sets

got a

## ALGORITHM

1. Start
2. Include <stdio.h> and <string.h>
3. Declare character, char - string [20], arr [20], is greater than or less than, flag = 0
4. Declare integer, int i, length, flag = 0
5. Input the string
6. Find the length of string using strlen
7. Copy string from string to arr using strcpy
8. Initialize j = 0
9. For i < length / 2 ; if arr[i] not equal to arr[length - i]  
for all i = length / 2
10. Print "string is not palindrome"
11. Else, print "string is palindrome"
12. Stop.

EXPERIMENT - 8

PALINDROME

AIM  
To read a string, store it in an array and check if  
palindrome or not

PROGRAM

```
#include <stdio.h>
#include <string.h>
int main()
{
    char string[20], arr[20];
    int i, length, flag = 0;
    printf ("Enter a string : ");
    scanf ("%s", string);
    length = strlen (string);
    strcpy (arr, string);
    for (i=0; i<length/2; i++)
    {
        if (arr[i] != arr [length-i-1])
            flag = 1;
        break;
    }
    if (flag)
        printf ("%s is not a palindrome \n", arr);
    else
        printf ("%s is a palindrome", arr);
}
```

OUTPUT

Enter the string:

MALAYALAM

MALAYALAM is a palindrome

Enter the string

Red

Red is not a palindrome

{printf("%s is a palindrome\n", arr); } //function

3 return;

RESULT

The program to check palindrome is executed and found successful.

Abhishek  
14/12/2025

## ALGORITHM

1. Start
2. include <stdio.h>
3. Declare character str1[100], str2[100], final[200]
4. Declare integers i=0, j=0, k=0, n=0
5. Input str1 and str2
6. While str1[i] and str2[j] != \$  
    increment them by i++ and j++ respectively
7. for both n < i and n > j, final [k++] = str1[n]  
    and ~~final~~ final [k++] = str2[n] respectively
8. Repeat until i=n and j=n
9. Print concatenated string
10. Stop.

## EXPERIMENT - 9

### STRING CONCATENATION

#### AIM

To read two strings (each ending with \$ symbol)  
store them in arrays and concatenate them  
with library function.

#### PROGRAMS

```
# include <stdio.h>
int main()
{
    char str1[100], str2[100], final[100];
    int i=0, j=0, k=0;
    printf ("Enter the first string (ending with $): ");
    fgets (str1, sizeof(str1), stdin);
    printf ("Enter the second string (ending with $): ");
    fgets (str2, sizeof(str2), stdin);
    while (str1[i] != '$')
    {
        i++;
    }
    str1[i] = '\0';
    while (str2[j] != '$')
    {
        j++;
    }
```

P-THEOREM

DETERMINISTIC NFA

OUTPUT

enter the first string (ending with \$) :  $s_1$  &  $s_2$   
 enter the second string (ending with \$) :  $s_3$  &  $s_4$   
 concatenated string :  $s_1 s_2 s_3 s_4$  parallel to  $s_1 s_2 s_3 s_4$

MIA

MIA

(1) from L1

(2) from L2

(3) from L3

(4) from L4

(5) from L5

(6) from L6

(7) from L7

(8) from L8

(9) from L9

(10) from L10

(11) from L11

(12) from L12

(13) from L13

(14) from L14

(15) from L15

(16) from L16

(17) from L17

(18) from L18

(19) from L19

(20) from L20

(21) from L21

(22) from L22

(23) from L23

(24) from L24

(25) from L25

(26) from L26

(27) from L27

(28) from L28

(29) from L29

(30) from L30

(31) from L31

(32) from L32

(33) from L33

(34) from L34

(35) from L35

(36) from L36

(37) from L37

(38) from L38

(39) from L39

(40) from L40

(41) from L41

(42) from L42

(43) from L43

(44) from L44

(45) from L45

(46) from L46

(47) from L47

(48) from L48

(49) from L49

(50) from L50

(51) from L51

(52) from L52

(53) from L53

(54) from L54

(55) from L55

(56) from L56

(57) from L57

(58) from L58

(59) from L59

(60) from L60

(61) from L61

(62) from L62

(63) from L63

(64) from L64

(65) from L65

(66) from L66

(67) from L67

(68) from L68

(69) from L69

(70) from L70

(71) from L71

(72) from L72

(73) from L73

(74) from L74

(75) from L75

(76) from L76

(77) from L77

(78) from L78

(79) from L79

(80) from L80

(81) from L81

(82) from L82

(83) from L83

(84) from L84

(85) from L85

(86) from L86

(87) from L87

(88) from L88

(89) from L89

(90) from L90

(91) from L91

(92) from L92

(93) from L93

(94) from L94

(95) from L95

(96) from L96

(97) from L97

(98) from L98

(99) from L99

(100) from L100

(101) from L101

(102) from L102

(103) from L103

(104) from L104

(105) from L105

(106) from L106

(107) from L107

(108) from L108

(109) from L109

(110) from L110

(111) from L111

(112) from L112

(113) from L113

(114) from L114

(115) from L115

(116) from L116

(117) from L117

(118) from L118

(119) from L119

(120) from L120

(121) from L121

(122) from L122

(123) from L123

(124) from L124

(125) from L125

(126) from L126

(127) from L127

(128) from L128

(129) from L129

(130) from L130

(131) from L131

(132) from L132

(133) from L133

(134) from L134

(135) from L135

(136) from L136

(137) from L137

(138) from L138

(139) from L139

(140) from L140

(141) from L141

(142) from L142

(143) from L143

(144) from L144

(145) from L145

(146) from L146

(147) from L147

(148) from L148

(149) from L149

(150) from L150

(151) from L151

(152) from L152

(153) from L153

(154) from L154

(155) from L155

(156) from L156

(157) from L157

(158) from L158

(159) from L159

(160) from L160

(161) from L161

(162) from L162

(163) from L163

(164) from L164

(165) from L165

(166) from L166

(167) from L167

(168) from L168

(169) from L169

(170) from L170

(171) from L171

(172) from L172

(173) from L173

(174) from L174

(175) from L175

(176) from L176

(177) from L177

(178) from L178

(179) from L179

(180) from L180

(181) from L181

(182) from L182

(183) from L183

(184) from L184

(185) from L185

(186) from L186

(187) from L187

(188) from L188

(189) from L189

(190) from L190

(191) from L191

(192) from L192

(193) from L193

(194) from L194

(195) from L195

(196) from L196

(197) from L197

(198) from L198

(199) from L199

(200) from L200

(201) from L201

(202) from L202

(203) from L203

(204) from L204

(205) from L205

(206) from L206

(207) from L207

(208) from L208

(209) from L209

(210) from L210

(211) from L211

(212) from L212

(213) from L213

(214) from L214

(215) from L215

(216) from L216

(217) from L217

(218) from L218

(219) from L219

(220) from L220

(221) from L221

(222) from L222

(223) from L223

(224) from L224

(225) from L225

(226) from L226

(227) from L227

(228) from L228

(229) from L229

(230) from L230

(231) from L231

(232) from L232

(233) from L233

(234) from L234

(235) from L235

(236) from L236

(237) from L237

(238) from L238

(239) from L239

(240) from L240

(241) from L241

(242) from L242

(243) from L243

(244) from L244

(245) from L245

(246) from L246

(247) from L247

(248) from L248

(249) from L249

(250) from L250

(251) from L251

(252) from L252

```
str2[j] = '0';
```

```
for (int n=0; n < i; j++, n++)
```

```
{ final[k++] = str1[n]; }
```

```
} for (int n=0; n < j; n++)
```

```
{ final[k-1-j] = str2[n]; }
```

```
}
```

```
final[k] = '\0';
```

```
cout << "concatenated string : " << final;
```

```
return 0;
```

```
}
```

### RESULT

The program for string concatenation is executed and found successful

Johnith A.  
12/2020

## ALGORITHM

1. Start
2. Include < stdio.h > and < ctype.h >
3. declare char = line [50].
4. Declare int vowels, constants, digits and spaces.
5. Initialize vowels = consonants = digits = space = 0.
6. Read string
7. Transform uppercase to lowercase with tolower.
8. For line [i] check vowels are i, o, u. if found + vowels
9. for line[i] = 'a' and line[i] <= z if found + digit
10. for line [i] > = '0' and line [i] <= '9' if found + digit
11. for line [i] = ' ' if found + space
12. print vowels, consonants, digits, spaces.
13. Stop

## EXPERIMENT - 10

Vowels consonants and digits

TOP 10

### AIM

Read a string (ending with \$ symbol) store it in an array and count the number of vowels, consonants, digits and spaces in them.

### PROGRAM

```
#include <stdio.h>
//include <ctype.h>
int main()
{
    char line[150];
    int vowels, consonants, digits, spaces, i=0;
    vowels = consonants = digits = spaces = 0;
    printf("Enter string:");
    fgets(line, sizeof(line), stdin);
    while (line[i] != '$')
    {
        i++;
    }
    for (int i=0; line[i] != '\0'; i++)
    {
        if (line[i] == 'a' || line[i] == 'e' || line[i] == 'i'
            || line[i] == 'o' || line[i] == 'u' || line[i] == 'A'
            || line[i] == 'E' || line[i] == 'I' || line[i] == 'O'
            || line[i] == 'U')
        {
            ++vowels;
        }
    }
}
```

01-THE MUSEUM

## OUTPUT

days from December - now

enter va. string:

computer to write (odometer flow pointers) prints itself  
Differences always be reflected by changes from previous

vowels : 3

Consonants : 5

digits : 2

whales pass : °

MIA

~~Nicht erlaubt~~  
<Nichter> nicht -n-  
~~erlaubt~~ -n-

Union Bay

(Locality and name)

$\beta = 3$ , single objects, known now, shown by

$i^o = \omega_0 q_2 = \Delta p_{\text{lab}} + \Delta m_{\text{resonas}} + \Delta v_{\text{vel}}$

(1: grande velvete) + tronco  
:(diblitz e se)

(All but 2, (and 3) are) steps

( $\frac{d}{dt} + \text{IC}) \text{and } \text{dW}_A$ )

四百三

$$(\text{Eff}(\sigma))' = \text{Eff}' \text{ and } (\sigma = \text{Eff}(\sigma)) \text{ if}$$

~~(B) and (C) are not true.~~  
~~(D) and (E) are not true.~~

$\{i\} = \{i\}$  and  $\{i\}' = \{i\}$  and  $\{i\}'' = \{i\}$ .

W = (Jewel)  $\cdot$  (Jewel)  $\cdot$  (Jewel)  $\cdot$  (Jewel)

100-172000-110000

1800 Vol. I.

```

else if (line[i] >= 'a' && line[i] <= 'z' || line[i]
{
    ++consonants;
}

else if (line[i] >= '0' && line[i] <= '9')
{
    ++digit;
}

else if (line[i] == ' ')
{
    ++space;
}

printf("vowels : %d", vowels);
printf("In consonants : %d", consonants);
printf("In digits : %d", digits);
printf("In white spaces : %d", spaces);
return 0;
}

```

## RESULT

Program to count the numbers of vowels, consonants, digits and whitespaces is executed and found successful

ABhishek A.  
12/2/2025