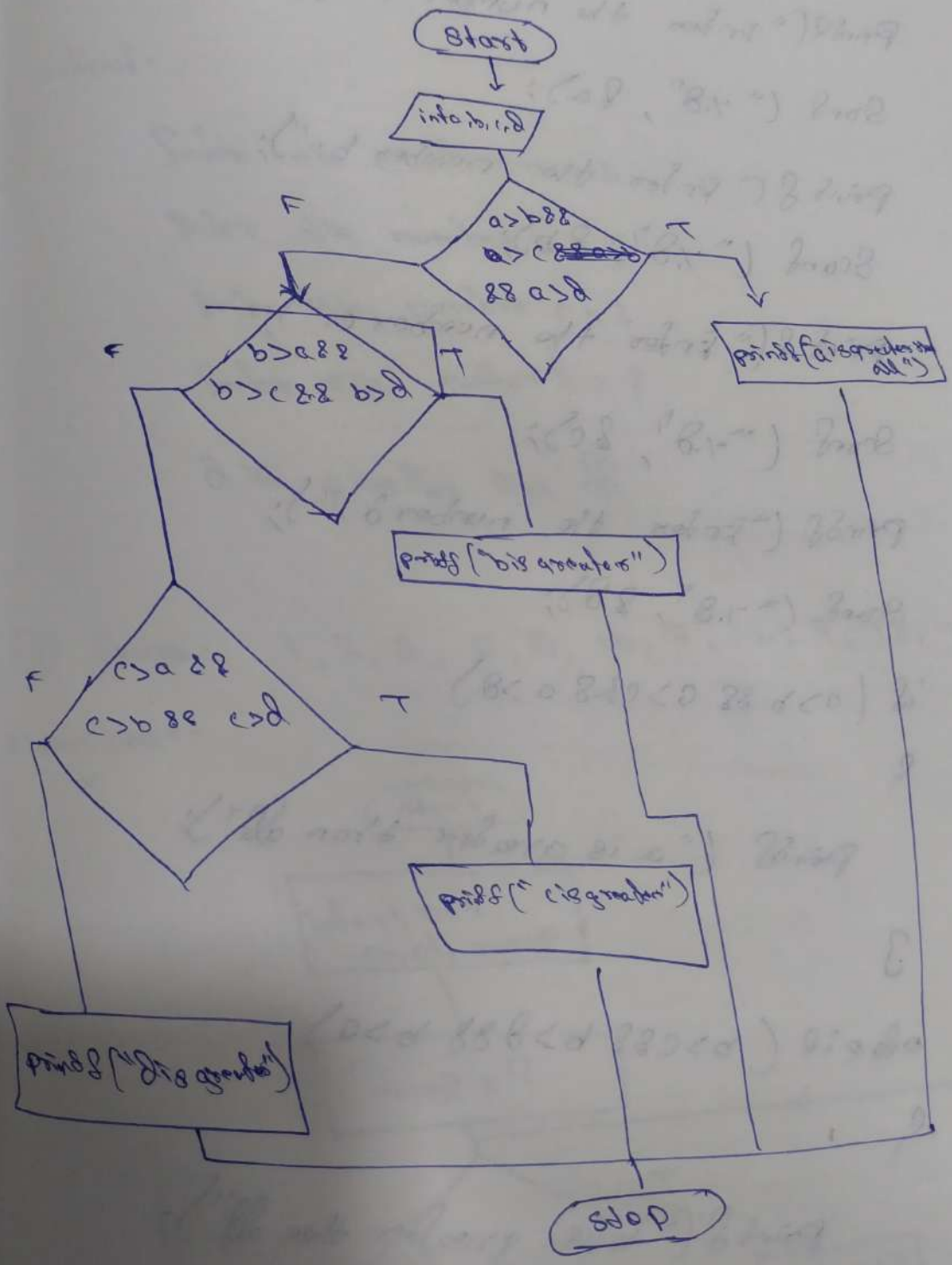
FLOWCHARTS

① Find largest number and smallest number among four numbers (user input)?

FLOWCHART:



code:

```
#include <stdio.h>
int main()
{
    int a,b,c,d;
    Printf("Enter the number a:");
    Scanf ("%d", &a);
    Printf("Enter the number b:");
    Scanf ("%d", &b);
    printf("Enter the number c:");
    Scanf ("%d", &c);
    printf ("Enter the number d:");
    Scanf ("%d", &d);
    if (a>b && a>c && a>d)
    {
        printf ("a is greater than all");
    }
    else if (b>c && b>d && b>a)
    {
        Printf("b is greater than all");
    }
    else if (c>d && c>a && c>b)
    {
```

else

{

printf("d is greater than dd");

}

return 0;

}

output:

Enter the number a = 5

Enter the number b = 6

Enter the number c = 7

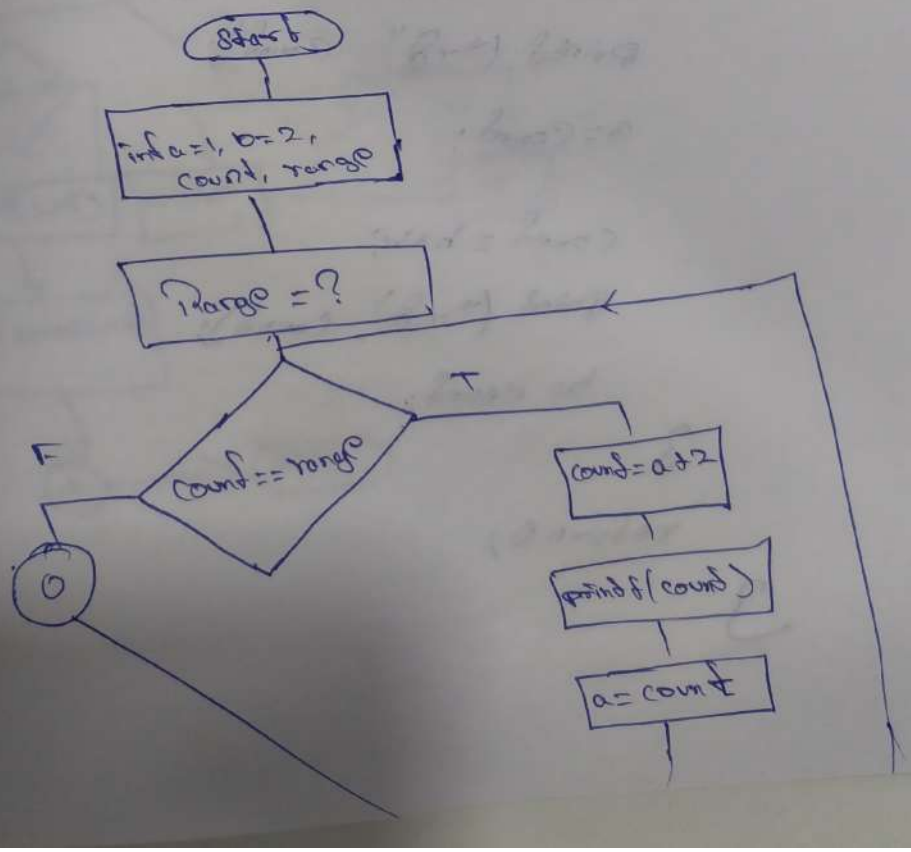
Enter the number d = 8

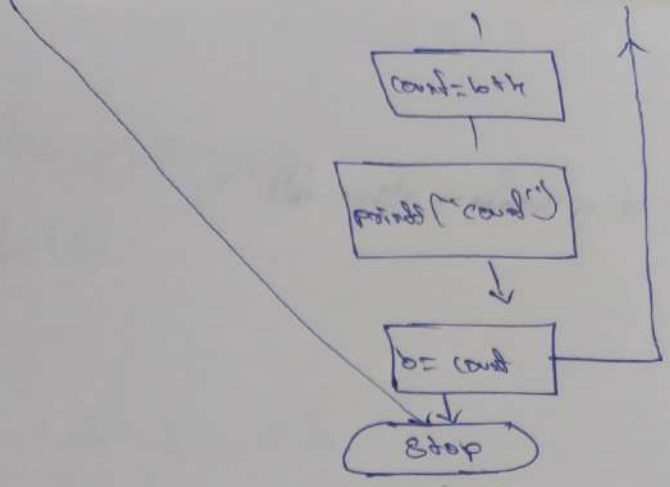
d is greater than dd

(2) Series: 1, 2, 3, 6, 5, 10, 7, 14, 9, 18, 11, ....

FLOWCHART:

=



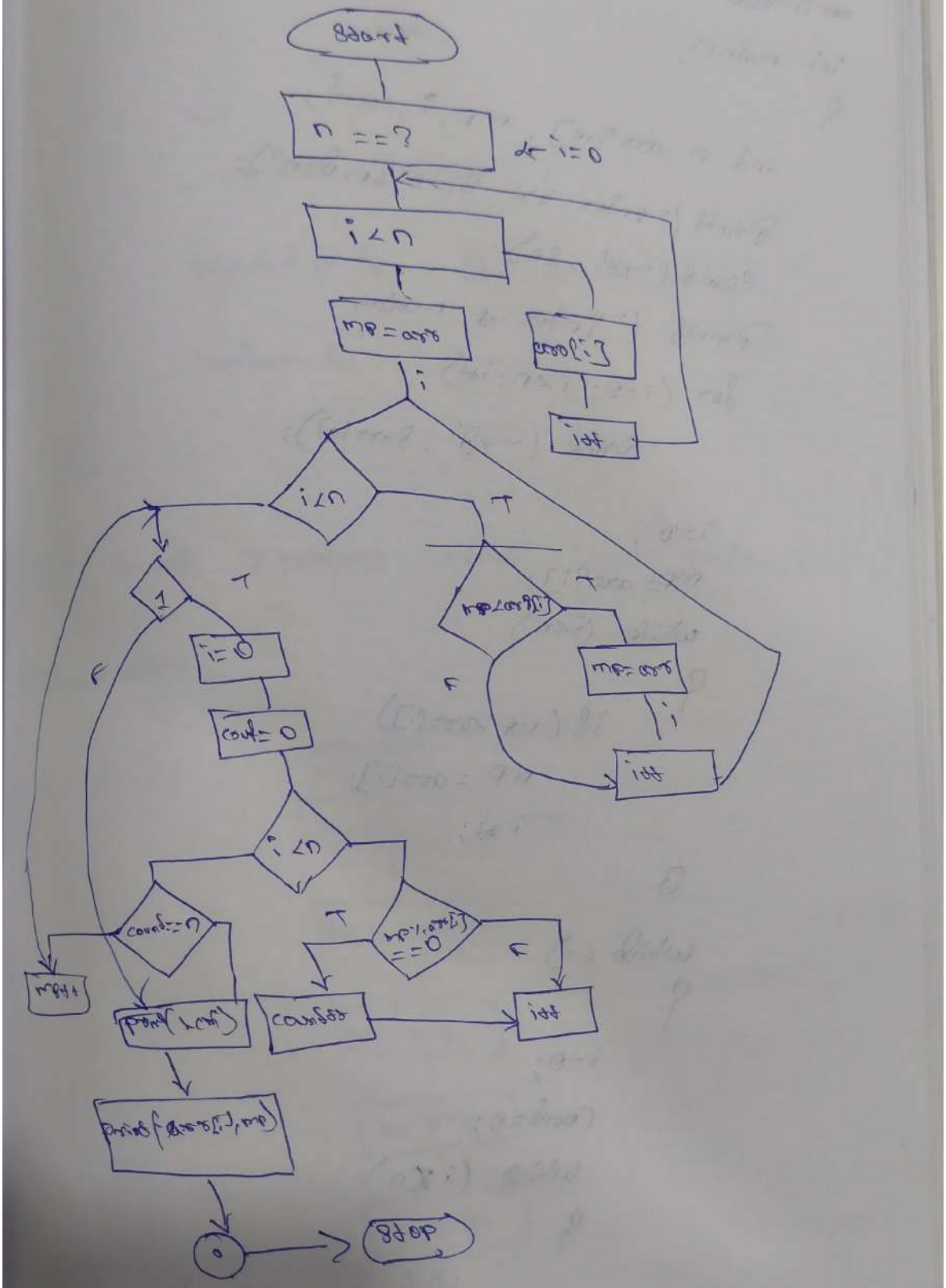



code:

```
#include <stdio.h>
int main ()
{
    int a=1, b=2, count, range;
    printf("Enter the range");
    scanf ("%d", &range);
    while (count == range)
    {
        count = a+2;
        printf ("%d", count);
        a=count;

        count = b+4;
        printf ("%d", count);
        b= count;
    }
    return 0;
}
```

③ LCM of numbers (user input)?

FLOWCHART:



code:

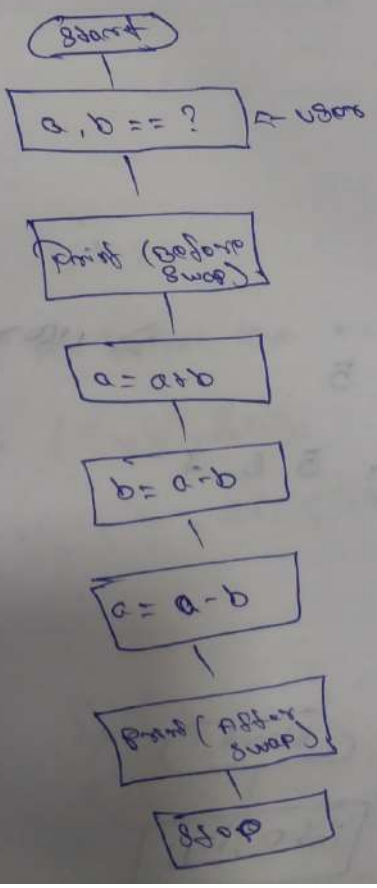
```
#include <stdio.h>
#include <math.h>
int main()
{
    int n, arr[10], mp, i, count;
    printf("Enter the size for d(n)");
    scanf("%d", &n);
    printf("Enter d number
    for (i=0; i<n; i++)
        scanf("%d", &arr[i]);

    i=0;
    mp= arr[i];
    while (i<n)
    {
        if (mp<arr[i])
            mp = arr[i];
            i++;
    }
    while (1)
    {
        i=0;
        count=0;
        while (i<n)
        {
            if(mp%arr[i]==0)

                count++;
                i++;
    }
```

```
if (count == n)
    break;
else
    mat++;
}
printf ("\n\tLCM (");
printf ("%d = %d", arr[i], mp);
return 0;
}
```

3

(4) Swap 2 numbers without temp variable?

FLOWCHART:



code:

```
#include <stdio.h>

int main()
{
        int a, b;
  printf("Enter the number a:");
  scanf("%d", &a);
  printf("Enter the number b:");
  scanf("%d", &b);
  printf("Before swap : %d %d", a,b);
  a = a+b;
  b = a-b;
  a = a-b;
  printf("After swap : %d %d", a,b);
  return 0;
}
```

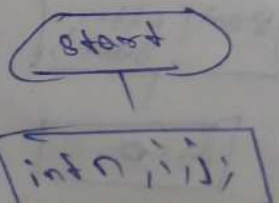
(5) Solve:

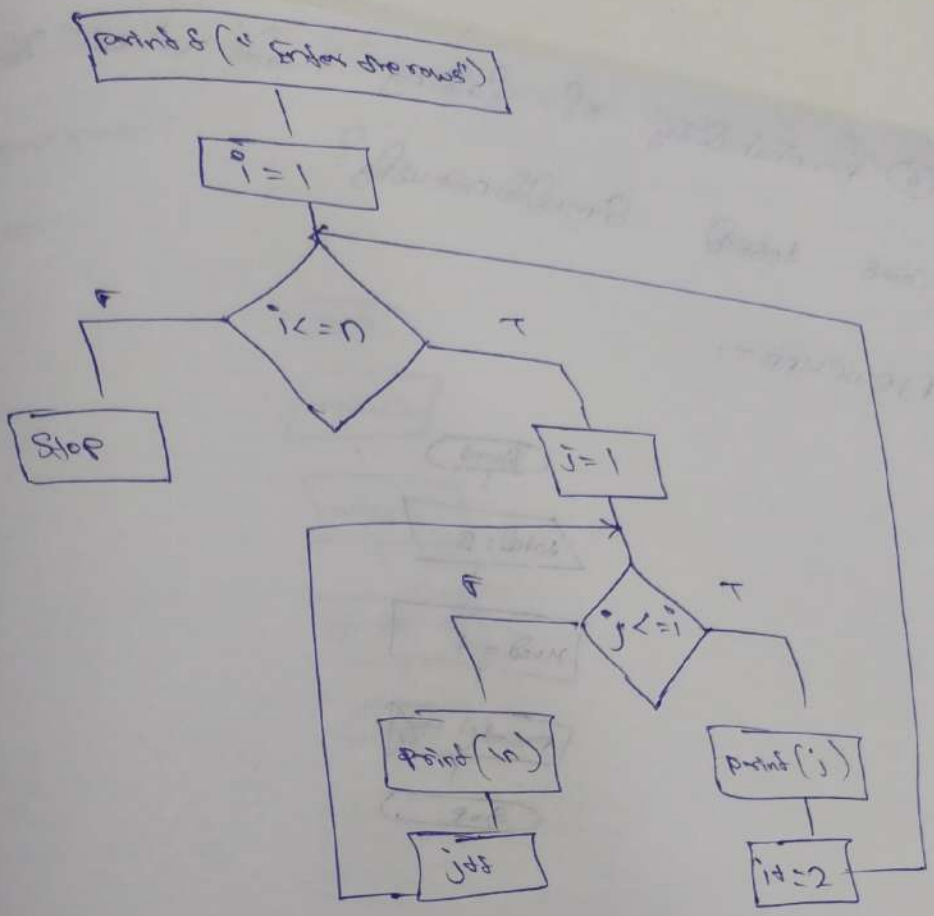
1 2 3 => User input (No. of rows)

1 2 3 4 5

1 2 3 4 5 6 7

FLOWCHART:



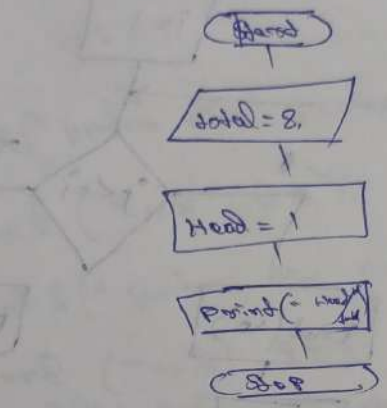



code:

```
#include <stdio.h>
int main()
{
    int n,i,j;
    printf("Enter the number of rows:");
    scanf("%d",&n);
    for(i=1;i<=n;i+=2)
    {
        for(j=1;j<=i;j++)
        {
            printf("%d",j);
        }
        printf("\n");
    }
```

⑥ probability of finding heads when 3 coins tossed simultaneously?

FLOWCHART:




code:

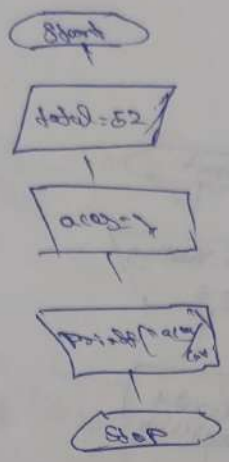
```
#include <stdio.h>
int main()
{
    int total = 8;
    int Head = 1;
    double Probability = Head / total;
    Printf("Probability of getting heads on all 3
            coins: %.2lf \n", Probability);
    return 0;
}
```

output:

=> 1/8;

(7) Probability of finding ace in pack of cards in C program?

Flowchart:




code:

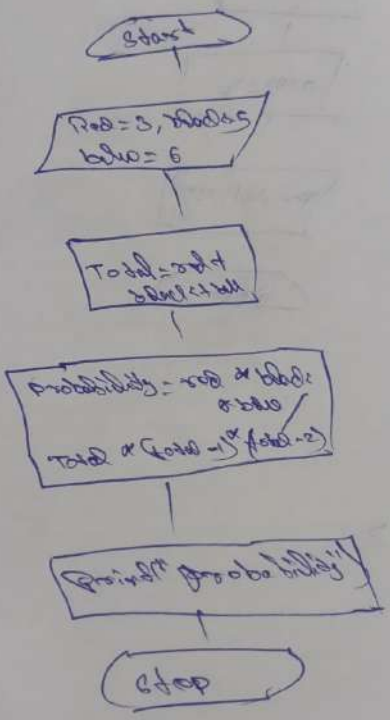
```
#include <stdio.h>

int main()
{
    int total=52;
    int aces=4;

    double Probability = aces/total;

    printf("Probability of getting aces: %.2lf\n",
           Probability);

    return 0;
}
```

output:

=> 4/52

(8) In a box we have 3 red, 5 black, 6 blue balls. If we draw 3 balls, find the probability of getting 1 red, 1 black, 1 blue in c program?

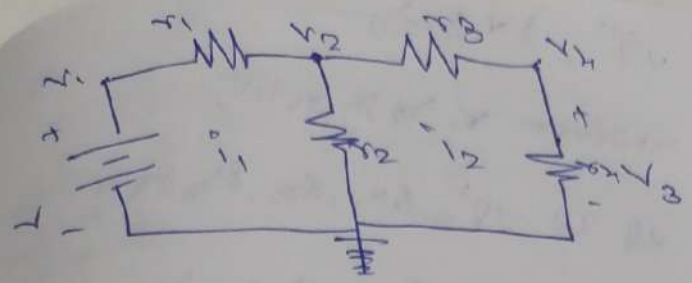
FLOWCHART:




Probability = Total ball / Total card

code:

```
#include <stdio.h>
int main()
{
    int red = 3, black = 5, blue = 6;
    int Total = red + black + blue;
    Probability = red * blue * black / Total * (Total-1) * (Total-2);
    Printf ("Probability of balls: %d \n", Probability);
    return 0;
}
```



Calculation:

$$\frac{V_2 - V}{R_1} + \frac{V_2}{R_2} + \frac{V_2}{R_3 + R_4} = 0$$

$$V = IR$$

$$I_2 = \frac{V_1}{R_3 + R_4}$$

$$V_3 = \frac{V_2}{R_3 + R_4} * R_4$$

$$P = \frac{V_3^2}{R_4}$$

$$\frac{V_2 - V}{R_1} = -\frac{V_2}{R_2} - \frac{V_2}{R_3 + R_4}$$

$$\frac{-V}{R_1} = \frac{V_2}{R_1} - \frac{V_2}{R_2} - \frac{V_2}{R_3 + R_4}$$

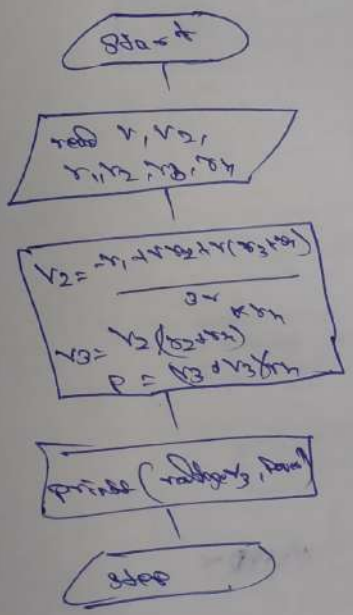
$$V = \frac{V_2}{R_1} + \frac{V_2}{R_2} + \frac{V_2}{R_3 + R_4} - \frac{V}{R_1}$$

$$V_2 (R_3 + R_4) = -R_1 + R_2 + R_3 + R_4$$

$$3V_2 = -R_1 + R_2 + R_3 + R_4$$

$$V_2 = \frac{-R_1 + R_2 + R_3 + R_4}{3V}$$

FLOWCHART:




Code:

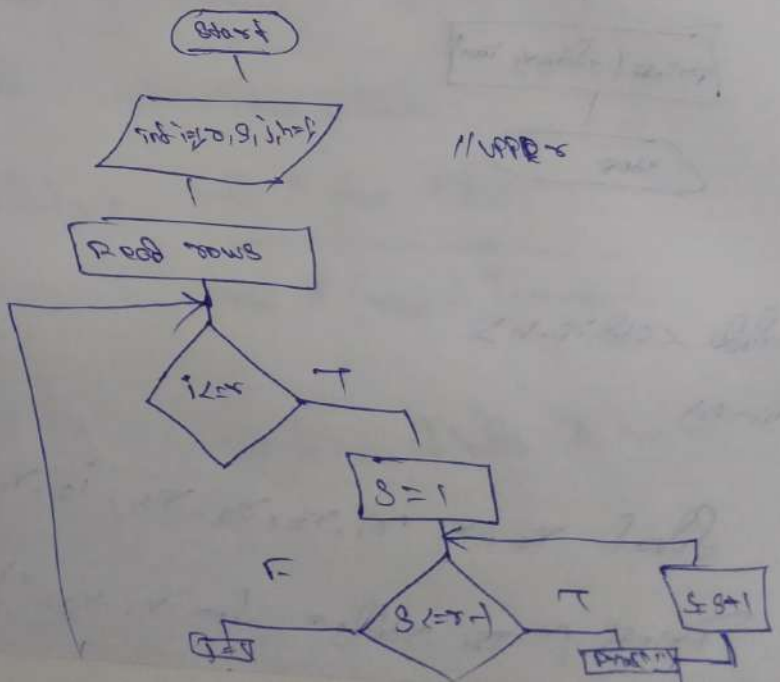
```
#include <stdio.h>
int main()
{
    float V2, V, R1, R2, R3, R4, I2, I1, V3, P;
    printf("Enter voltage for V, V2:");
```

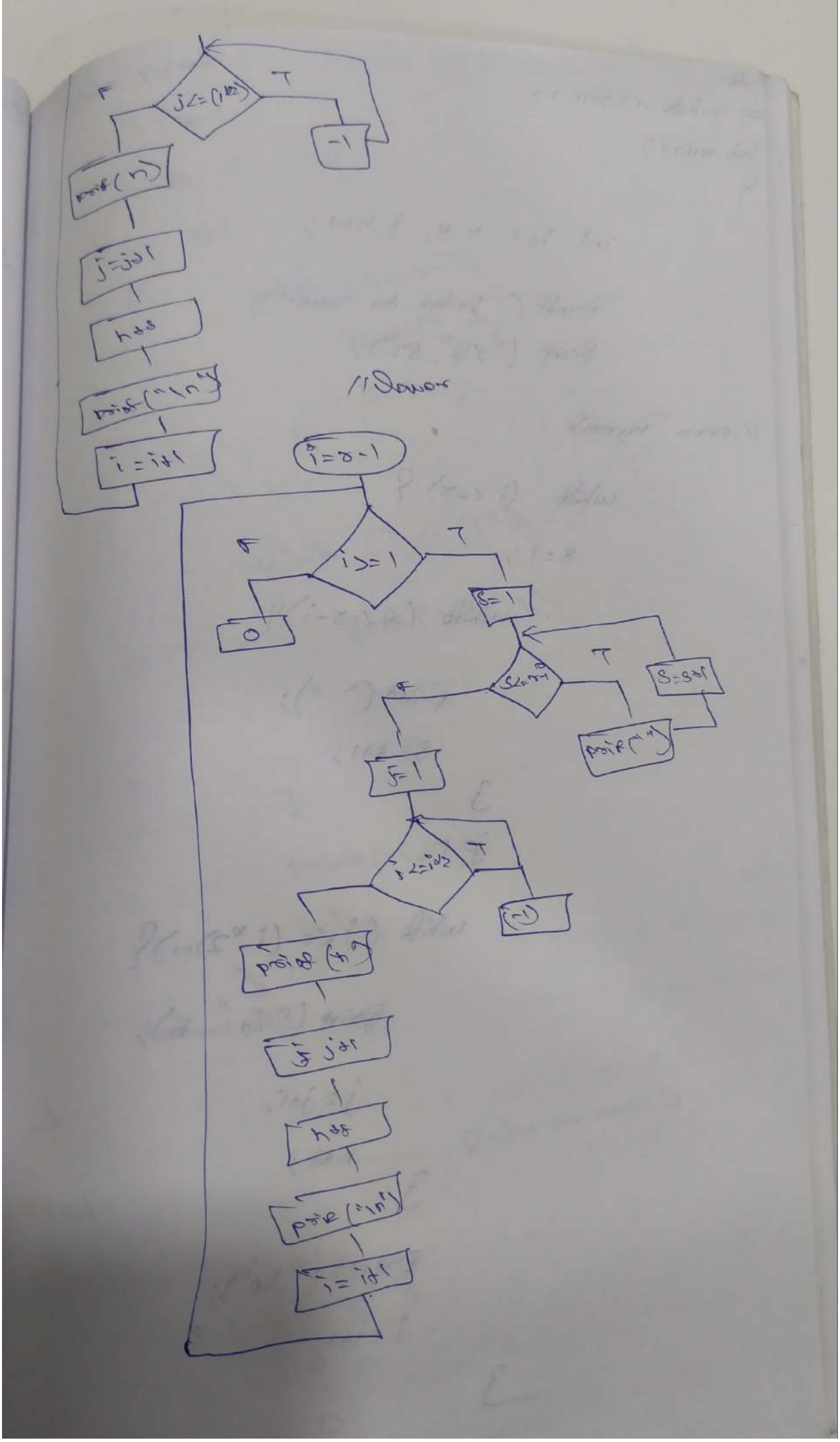
```
scanf("%f %f", &r1, &r2);
printf("The resistance r1, r2 r3, r4:");
scanf("%f %f %f %f", &r1, &r2, &r3, &r4);
r2 = ((-r1 + (r1*r2) + (r1*(r3+r4))) / (r3+r4));
v3 = (r2 (r3+r4) * r4);
P = (((v3) * (v3)) / r4);
printf("The voltage v3: %f", v3);
printf("The power: %f W", P);
return 0;
}
```

(10)

```
      1
   2  3  4
5  6  7  8  9
  10 11 12
     13
```

FLOWCHART:





```
#include <stdio.h>
int main()
{
    int i=1, n, s, j, h=1;
    printf("Enter the rows:");
    scanf("%d", &n);
//upper Pyramid
    while (i<=n){
    s=1;
    while (s<=n-i){
    printf(" ");
    s=s+1;
    }
    j=1;
    while (j<=(i*2)-1){
    printf("%d", h);
    j=j+1;
    h++;
    }
    printf("\n");
    i=i+1;
}
```

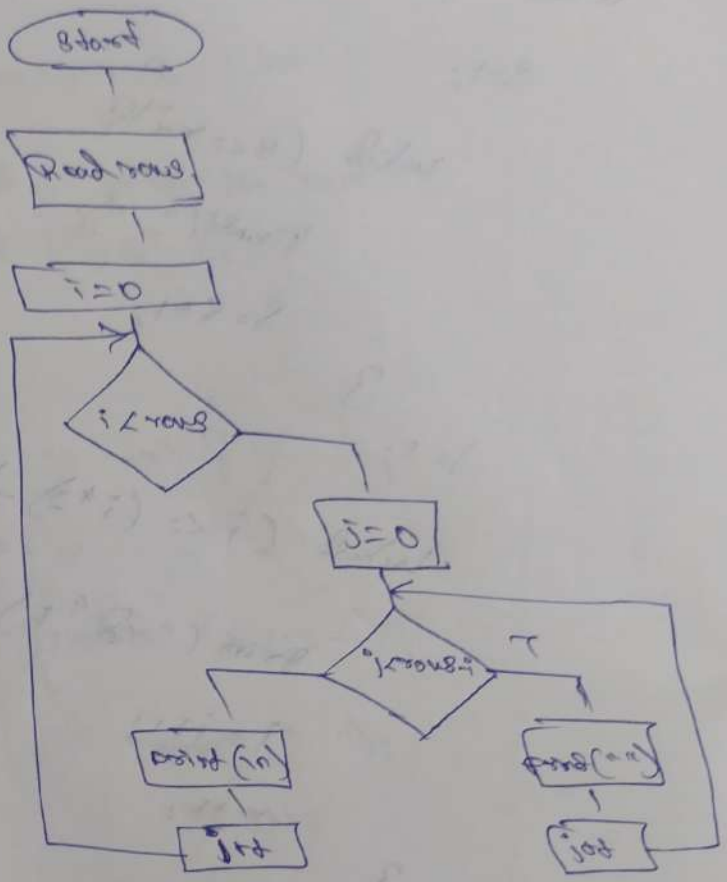
```
//Lower Pyramid
    i=n-1;
    while (i>=1){
        s=1;
        while (s<=n-i){
            printf(" ");
            s=s+1;
        }
    j=1;
    while (j<=(i*2)-1){
        printf("%d ",k);
        j=j+1;
        k++;
    }
    printf("\n");
    i=i-1;
}
return 0;
}
```

Output:

```
Enter the rows: 3
  1
 2 3 4
5 6 7 8 9
```

(11) Inverted right half pyramid pattern:

Flowchart:




Code:

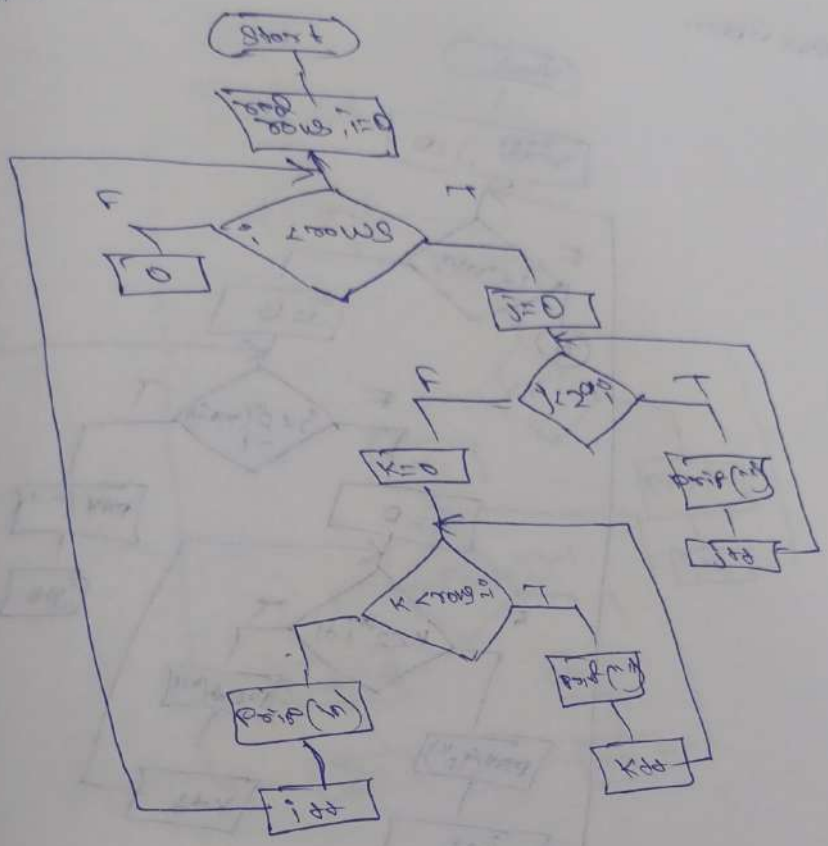
```
#include <stdio.h>
int main()
{
    int rows=5;
    for(int i=0; i<row; i++){
        for (int j=0; j<rows-i; j++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

Output:

```
*****
****
***
**
*
```

(12) Inverted left half pyramid pattern:

Flow chart:




code:

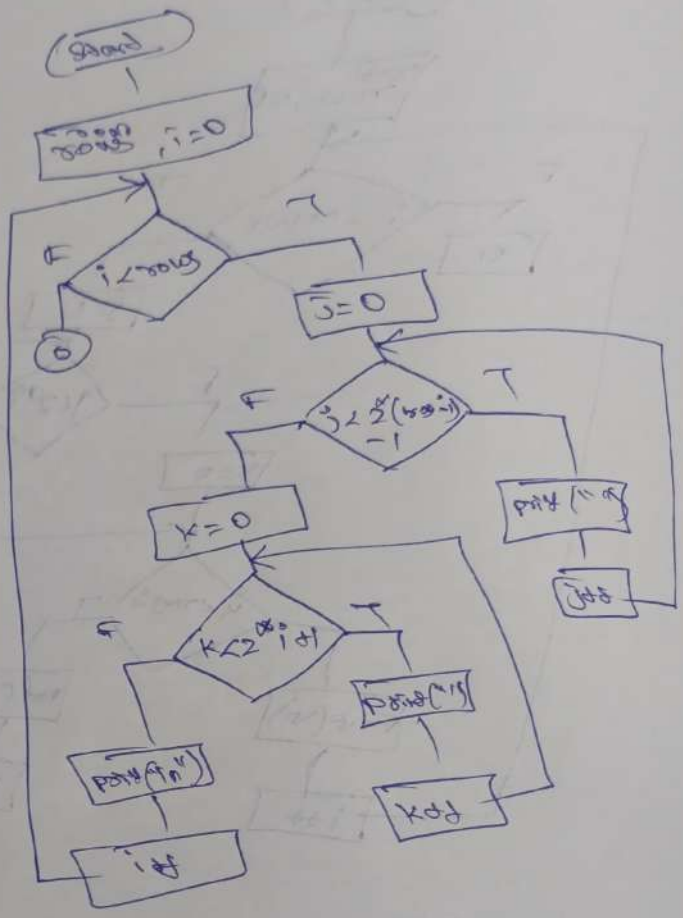
```
#include <stdio.h>
int main()
{
    int rows=5;
    for(int i=0; i<rows; i++){
        for(int j=0; j<2*i; j++){
            printf(" ");
        }
        for(int k=0; k<rows-i; k++){
            printf("%d", k+1);
        }
        printf("\n");
    }
    return 0;
}
```

Output:

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

(11) (1) Full pyramid pattern:

Flowchart:



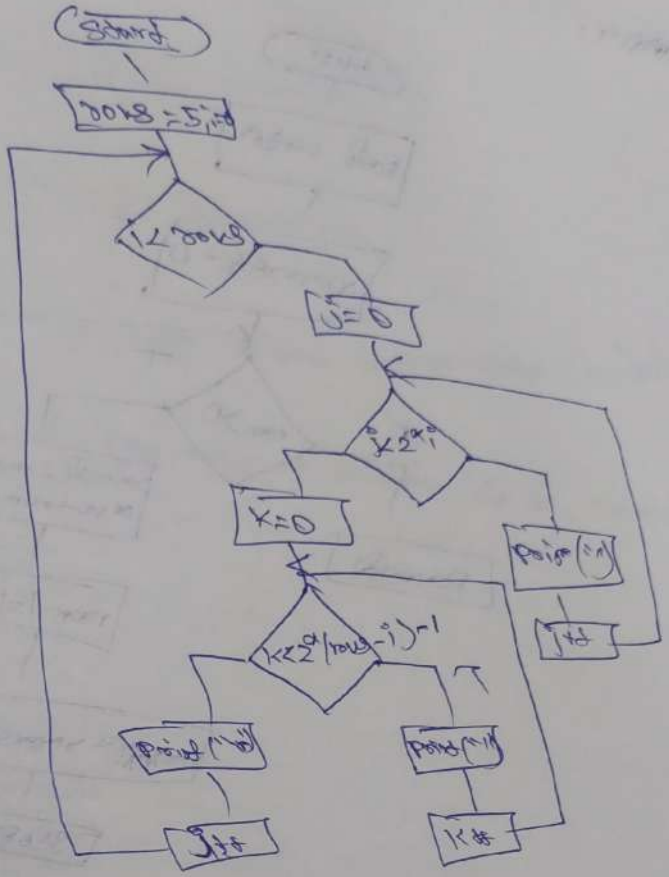

Code:

```
#include <stdio.h>
int main()
{
    int rows = 5;
    for (int i=0; i<rows; i++){
        for (int j=0; j<2*(rows-i)-1; j++){
            printf(" ");
        }
        for (int k=0; k<2*i+1; k++){
            printf("*");
        }
        printf("\n");
    }
    return 0;
}
```

*
* * *
* * * * *
* * * * * * *
* * * * * * * * *

(14) Inverted Full Pyramid Pattern:

Flowchart:




Code:

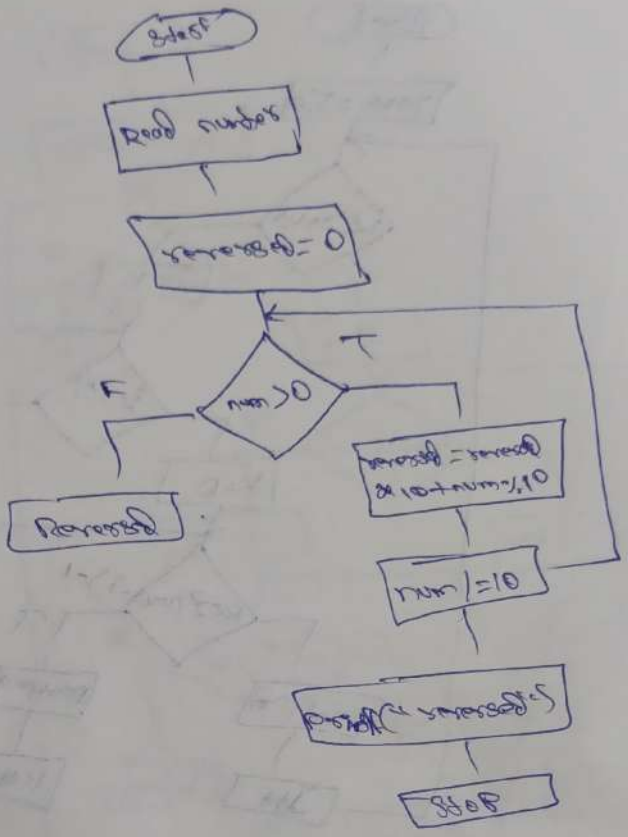
```
#include <stdio.h>
int main()
{
    int rows = 5;
    for (int i=0; i<rows; i++)
    {
        for (int j=0; j<2*i; j++)
        {
            printf(" ");
        }
        for (int k=0; k<2*(rows-i)-1; k++)
        {
            printf("*");
        }
        printf("\n");
    }
}
```

Output:

```
*********
 *******
  *****
   ***
    *
```

(13) Reverse the number:

FLOWCHART:




Code:

```
#include <stdio.h>

int reverseNumber(int num){

    int reversed = 0;

    while (num > 0){

        reversed = reversed * 10 + num % 10;

        num /= 10;

    }

    return reversed;

}
```

```
int main() {
    int number;
    printf("Enter a number: ");
    scanf("%d", &number);
    int reversed_number = reverseNumber(number);
    printf("Reversed number: %d\n", reversed_number);
    return 0;
}
```

3

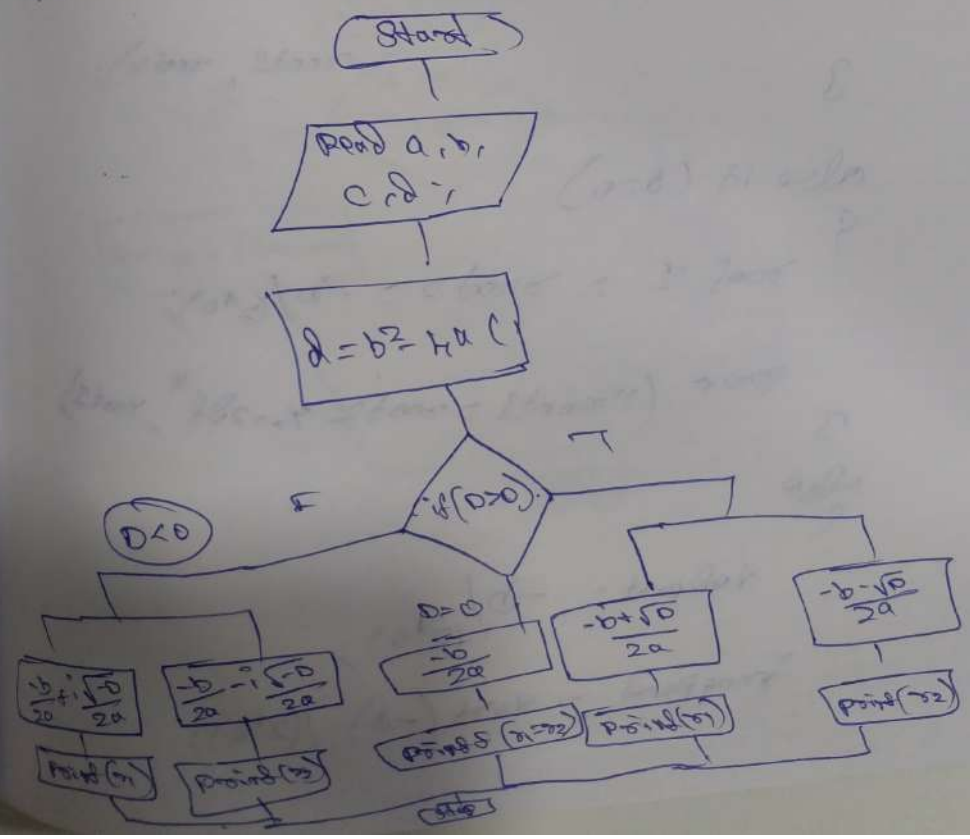
output :

⇒ 12345

⇒ 54321

(b) Quadratic equation :

Flowchart :



```
Ans:
#include <stdio.h>
#include <math.h>
int main()
{
    double a, b, c, d, root1, root2, realpart,
                                    imagpart;
    printf("Enter the quadratic numbers:");
    scanf("%lf %lf %lf", &a, &b, &c);
    d = (b*b) - 4*a*c;
    if(d>0)
    {
        root1 = (-b + sqrt(d))/(2*a);
        root2 = (-b - sqrt(d))/(2*a);
        printf("root1 = %.2lf and root2 = %.2lf",
                                    root1, root2);
    }
    else if (d==0)
    {
        root1 = root2 = -b/(2*a);
        printf("root1 = root2 = %.2lf", root1);
    }
    else
    {
        realpart = -b/(2*a);
        imagpart = sqrt(-d)/(2*a);
```

printf("root1 = %.2f + %.2fi and root2 = %.2f - %.2fi", realPart, imagPart, realPart, imagPart);

3

return 0;

->gcc source.c -lm

3

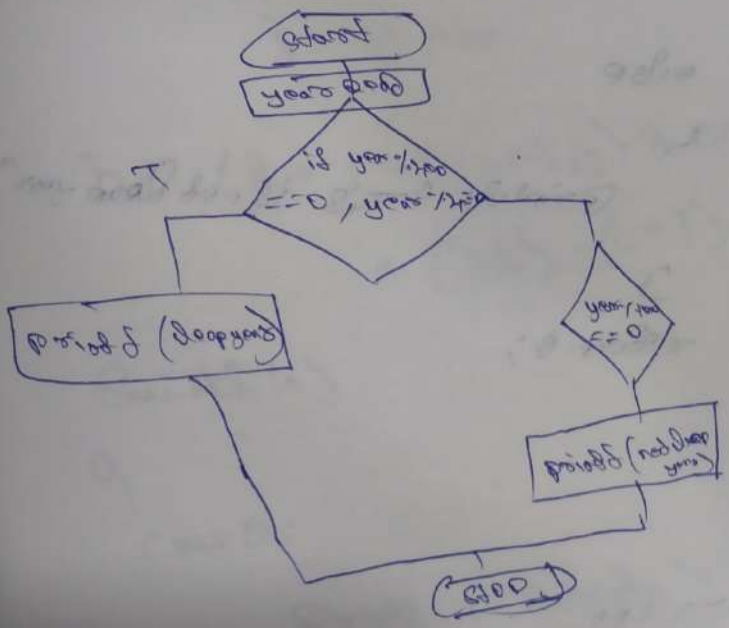
output:

a=3, b=3, c=2

root1 = 0.8 + 0.00i, root2 = -0.5*(+0.00i)

(17) Leap years (or not):

Flowchart:



code:

```
#include <stdio.h>
int main()
{
    int year;
    printf("Enter the year");
    scanf("%d", &year);
    if (year % 400==0)
        printf("This is leapyear");
    else if (year % 100==0)
        printf("This is not leap year");
    else if (year % 4==0)
        printf("This is leapyear");
    else
    {
        printf("This is not leap year");
    }
    return 0;
}
```

Output:

year = 100

This is not a leap year

(12) Find day from the date, month, year?

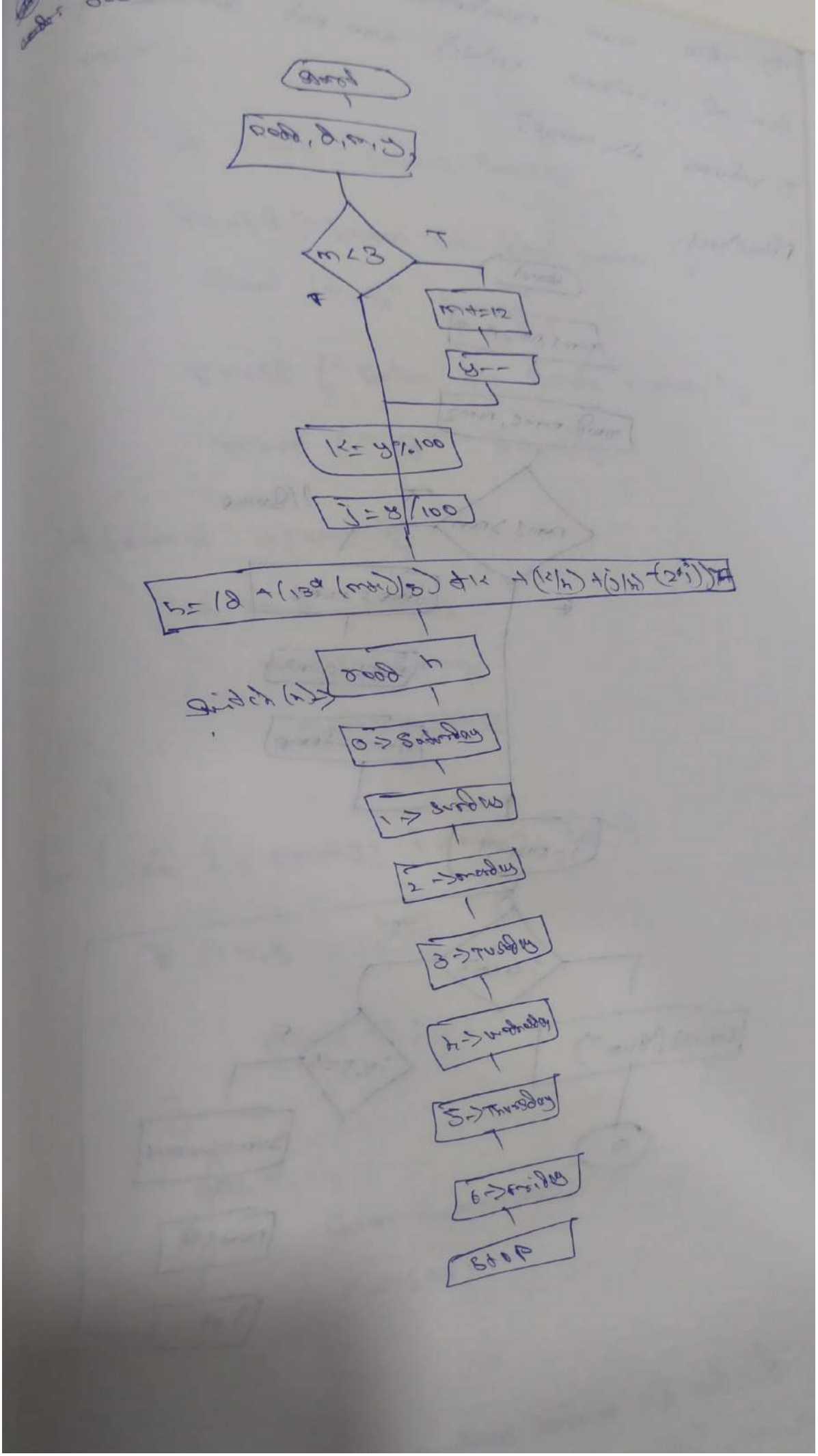
```
#include <stdio.h>
#include <math.h>
int main()
{
    int h, d, m, y, j, k;
    printf("Enter the date month year:");
    scanf("%d %d %d", &d, &m, &y);
    if (m<3)
    {
        m += 12;
        y--;
    }
    k = y%100;
    j = y/100;
    h = (d + (13*(m+1))/5) + (k) +(k/4)
        + (j/4) -(2*j)) %7;
    switch(h)
    {
    case 0:
        printf("Saturday\n");
        break;
    case 1:
        printf("Sunday\n");
        break;
```

```
    case 2:
        printf ("monday\n");
        break;
    case 3:
        printf ("tuesday\n");
        break;
    case 4:
        printf ("wednesday\n");
        break;
    case 5:
        printf ("thursday\n");
        break;
    case 6:
        printf("friday\n");
        break;
}
return 0;
}
```

output:

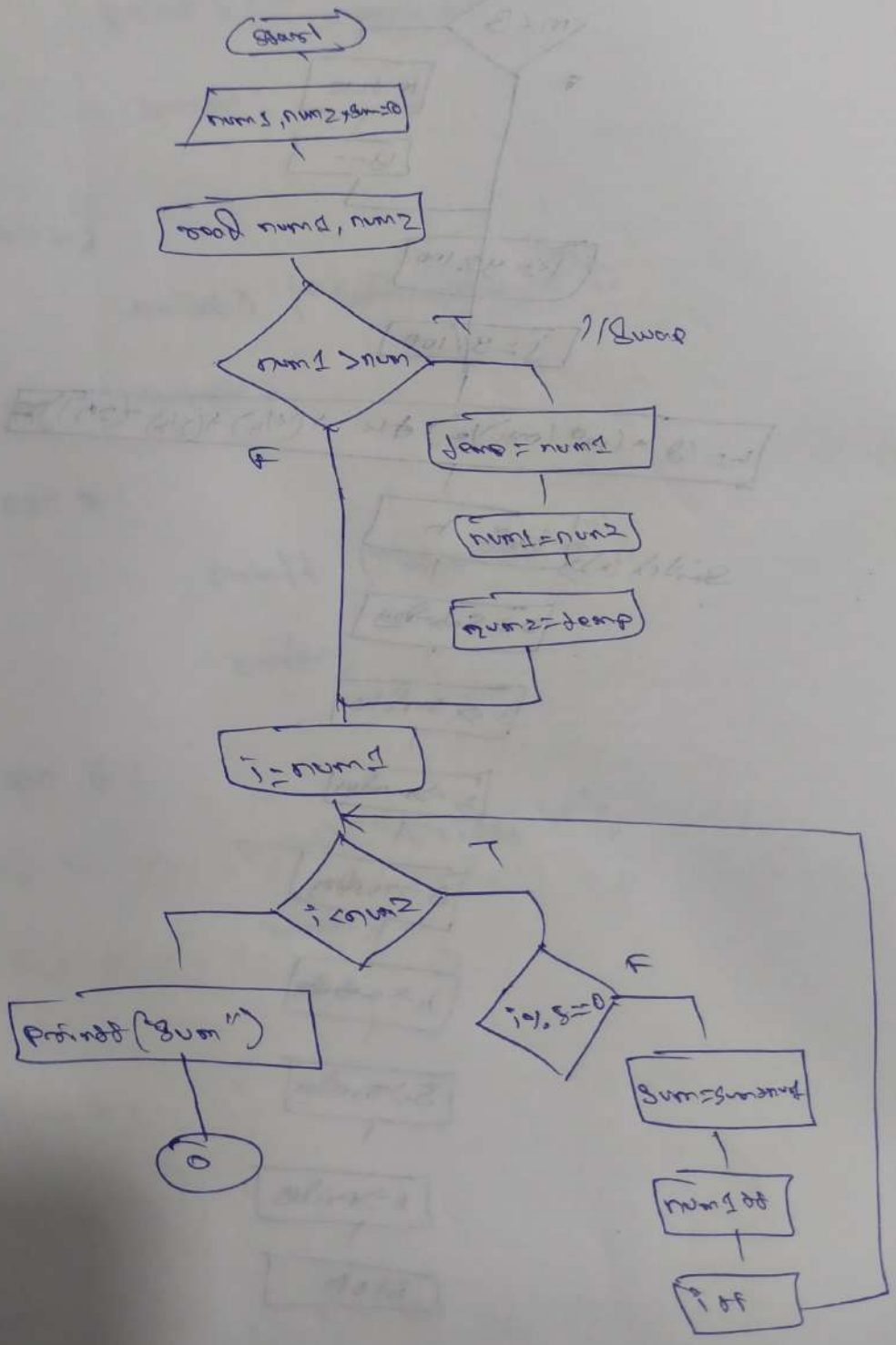
-> 26/11/1999

-> friday



(19) Take two numbers from user & find the sum of numbers which are not divisible by 5 between the range?

Flowchart:



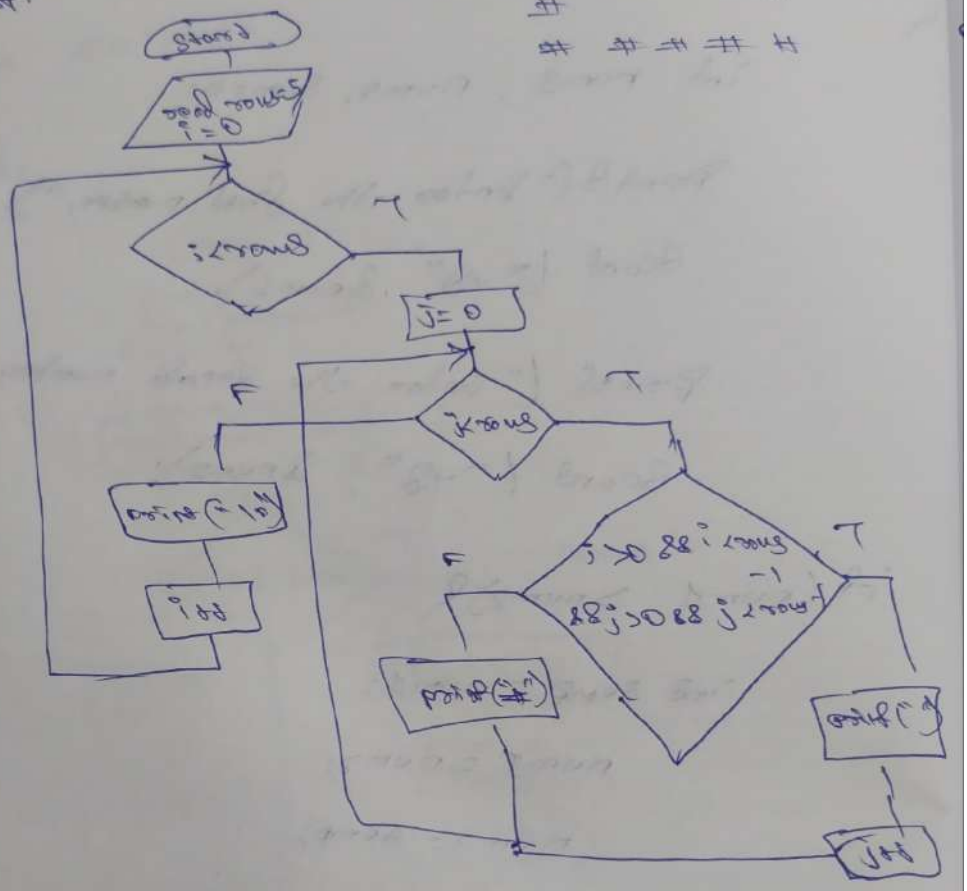
```
#include <stdio.h>
int main()
{
    int num1, num2, sum=0;
    printf("Enter the first number:");
    scanf("%d", &num1);
    printf("Enter the second number:");
    scanf("%d", &num2);
if(num1 > num2){
    int temp = num1;
    num1 = num2;
    num2 = temp;
}
for(int i = num1; i < num2; i++){
    if(i%5 == 0){
        num1++;
}
else
{
    sum = sum + num1;
    num1++;
}
}
printf("Sum of numbers between %d and %d
(not divisible by 5) : %d\n", num1, num2,
sum);
}
```

(2) Hollow Square:

Output:

# # # # #
# #
# #
# #
# # # # #

Flowchart:

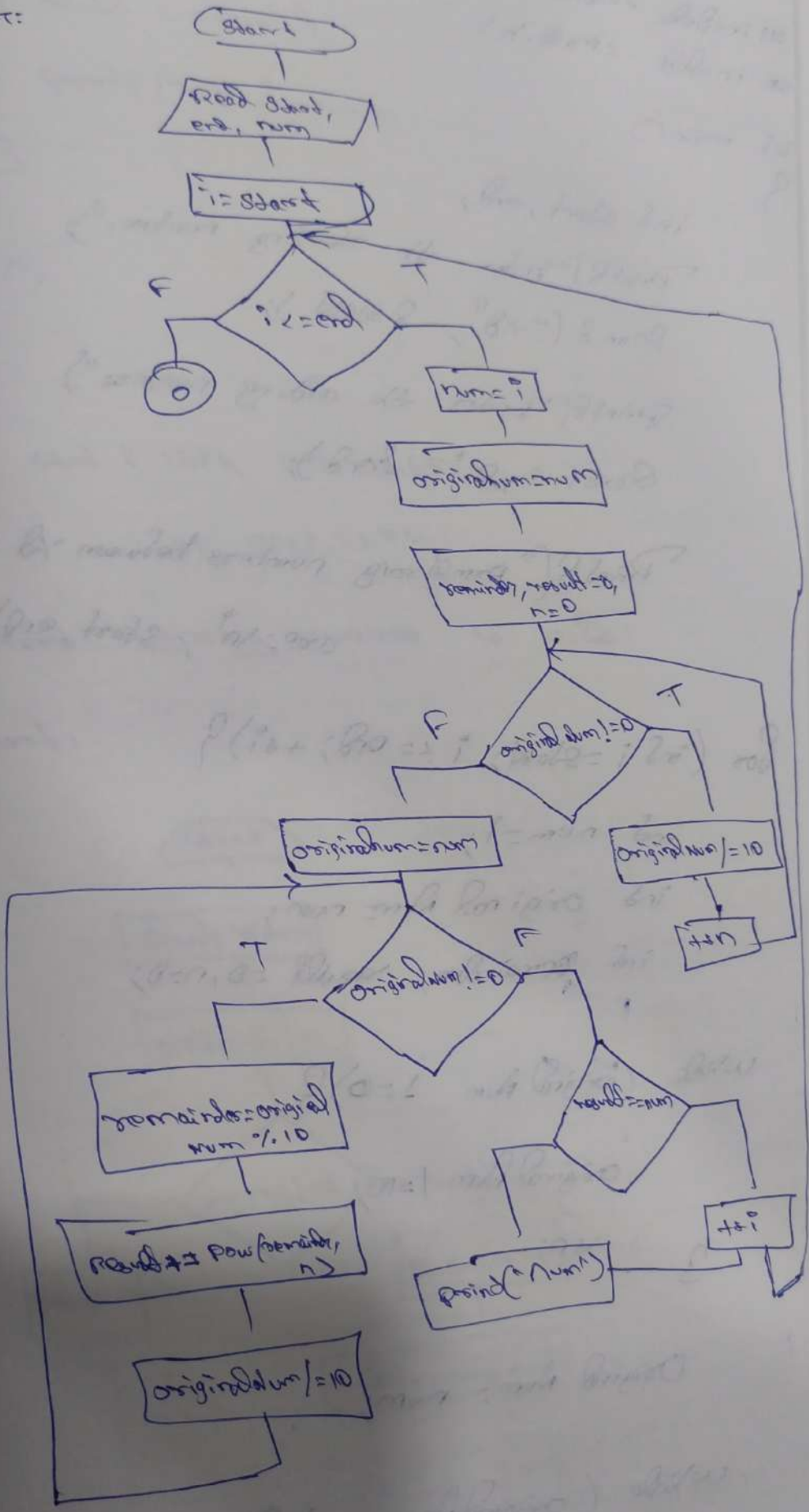



Code:

```
#include <stdio.h>
int main()
{
    int rows = 5;
    for (int i = 0; i < rows; i++)
    {
        for (int j = 0; j < rows; j++)
        {
            if (i > 0 && i < rows - 1 &&
                j > 0 && j < rows - 1)
                printf(" ");
            else
                printf("#");
        }
        printf("\n");
    }
    return 0;
}
```

(10) find Armstrong numbers between two intervals. (User input)

FLOWCHART:



code:

```
#include <stdio.h>
#include <math.h>

int main()
{
    int start, end;
    Printf("Enter the starting number:");
    Scanf("%d", &start);
    Printf("Enter the ending number:")
    Scanf("%d", &end);
    Printf("Armstrong numbers between %d and %d
            are:\n", start, end);

for (int i = start; i <= end; ++i) {
    int num = i;
    int originalNum = num;
    int remainder, result = 0, n = 0;

while (originalNum != 0) {
    originalNum /= 10;
    ++n;
}

originalNum = num;

while (originalNum != 0) {
    remainder = originalNum % 10;
    result += pow(remainder, n);
```

```
}
if(result == num){
    printf("%d\n", num);
}
}
return 0;
}
```

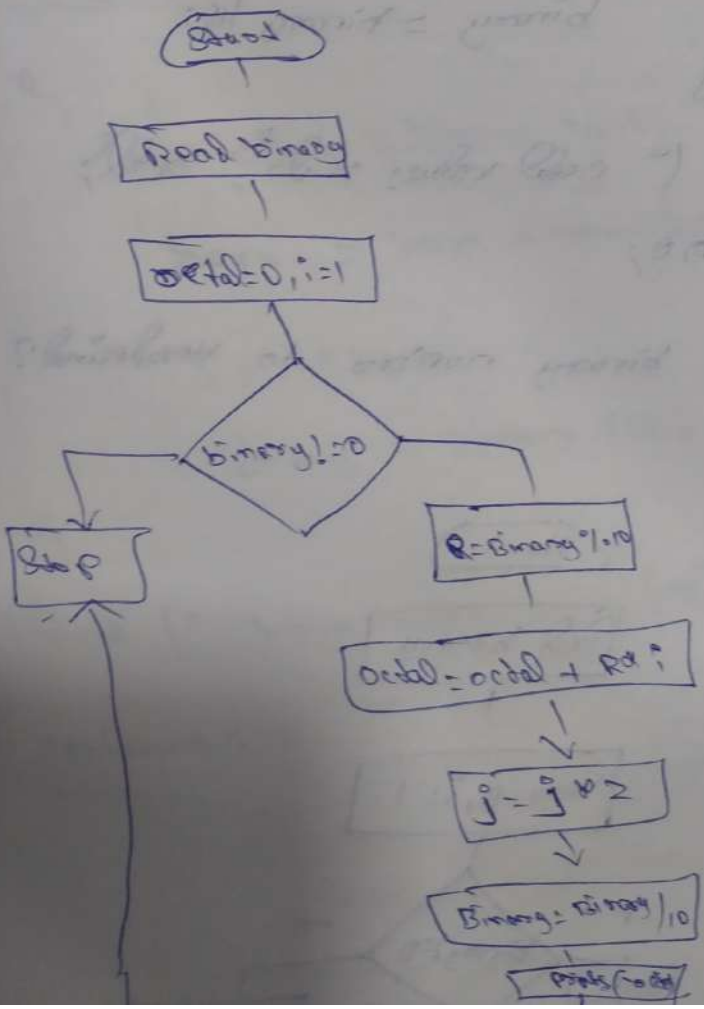
output:

start : 133x  end : x65x6

163x, 8208, 947x

(22) Convert binary number to octal:

flowchart:

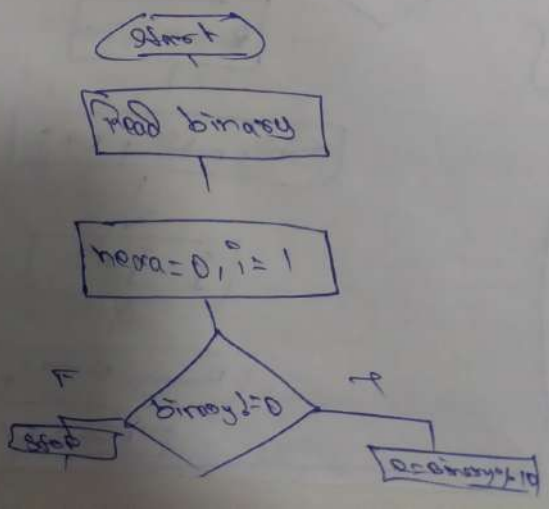


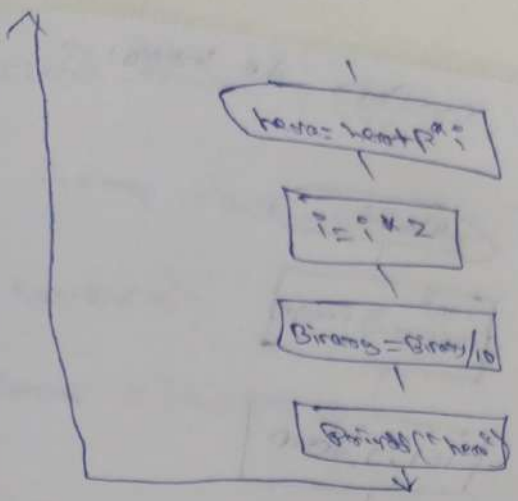
code:

```
#include <stdio.h>

int main()
{
    long int binary, octal = 0, j = 1, remainder;
    printf("Enter the value for binary number: ");
    scanf("%ld", &binary);
    while (binary != 0)
    {
        remainder = binary % 10;
        octal = octal + remainder * j;
        j = j * 2;
        binary = binary / 10;
    }
    printf("octal value: %ld", octal);
    return 0;
}
```

(23) Convert binary number to hexadecimal?

Flowchart:



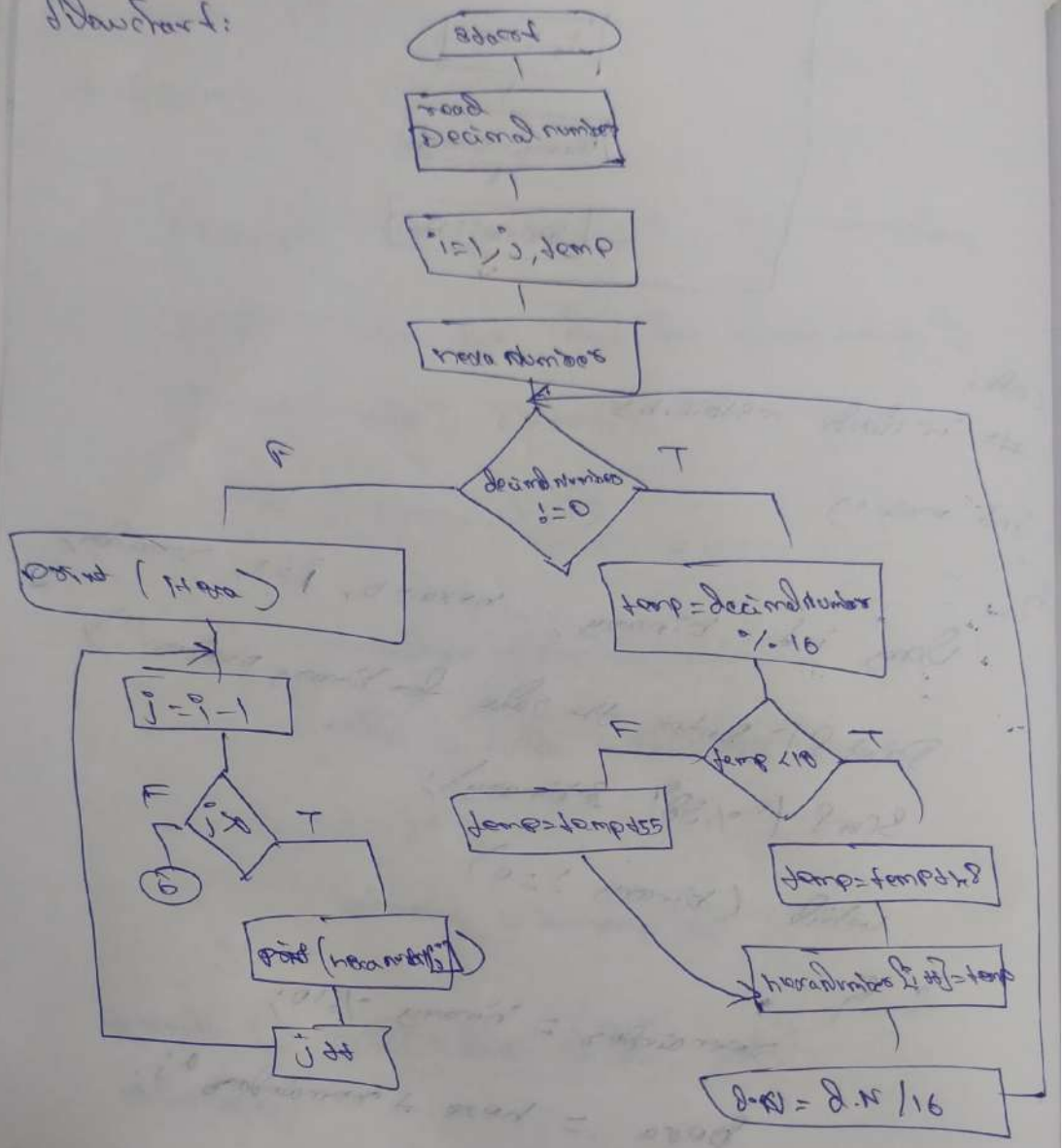



code:

```
#include <stdio.h>

int main()

{
    long int binary, hexa=0, i=1, remainder;
    printf("Enter the value for binary number: ");
    scanf("%ld", &binary);
    while (binary != 0)
    {
        remainder = binary %10;
        hexa = hexa + remainder * i;
        i = i*2;
        binary = binary /10;
    }
    printf("Hexa value : %ld", hexa);
    return 0;
}
```

(2) convert decimal number to hexa:

flowchart:




code:

```
#include <stdio.h>
int main()
{
    int decimal_Number;
    Printf("Enter the decimal number: ");
    Scanf("%d", &decimal_Number);
    int i=1, j, temp;
    Char hexa_Number[100];
```

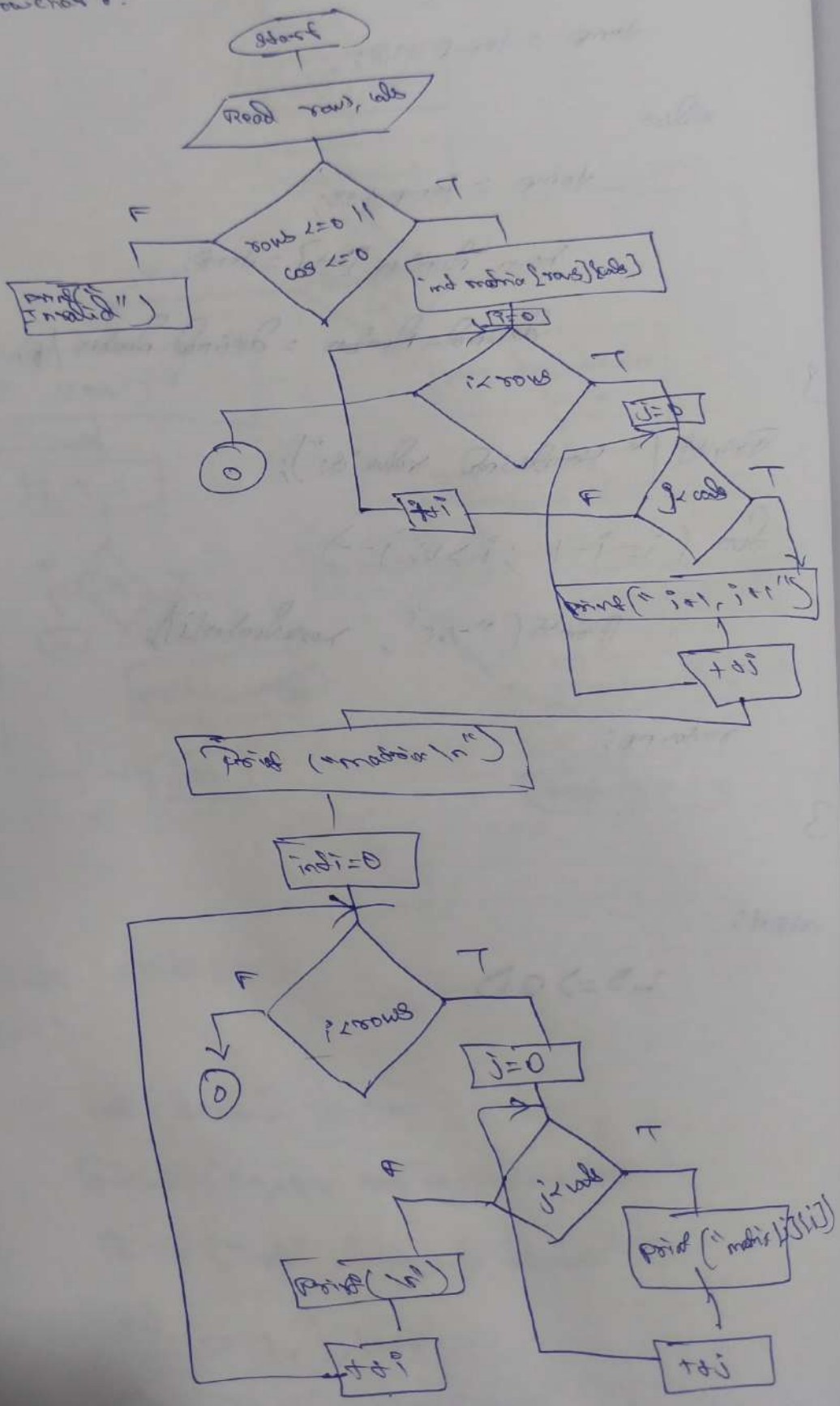
```
while (decimal_number != 0) {
        temp = decimal_number % 16;
if (temp < 10)
        temp = temp + 48;
else
        temp = temp + 55;
        hexa_number[i++] = temp;
        decimal_number = decimal_number / 16;
}
printf("Hexadecimal value is: ");
for (j = i - 1 ; j > 0; j--)
        printf("%c", hexadecimal[j]);
return 0;
}
```

output:

45 => 2D

(25) Take matrix elements from the User and display the matrix?

Flowchart:



code:

```
#include <stdio.h>
int main()
{
    int rows, cols;
    printf("Enter the number of rows:");
    scanf("%d", &rows);
    printf("Enter the number of columns:");
    scanf("%d", &cols);
    if (rows <=0 || cols<=0){
    printf("Invalid dimensions");
    return 1;
    }
int matrix[rows][cols];
printf("Enter matrix elements:\n");
for (int i=0; i<rows; ++i){
    for(int j=0; j<cols; ++j){
    printf("Enter element at position(%d, %d):"
                i+1, j+1);
    scanf("%d", &matrix[i][j]);
}}
```

```
// Display the matrix
printf("matrix:\n");
for (int i=0; i<rows; ++i){
    for (int j=0; j<cols; ++j){
        printf("%d ", matrix[i][j]);
    }
    printf("\n");
}
return 0;
}
```

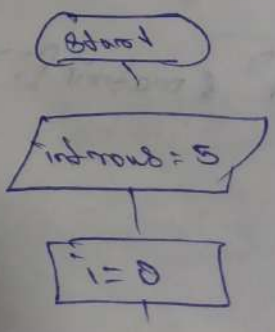
output:

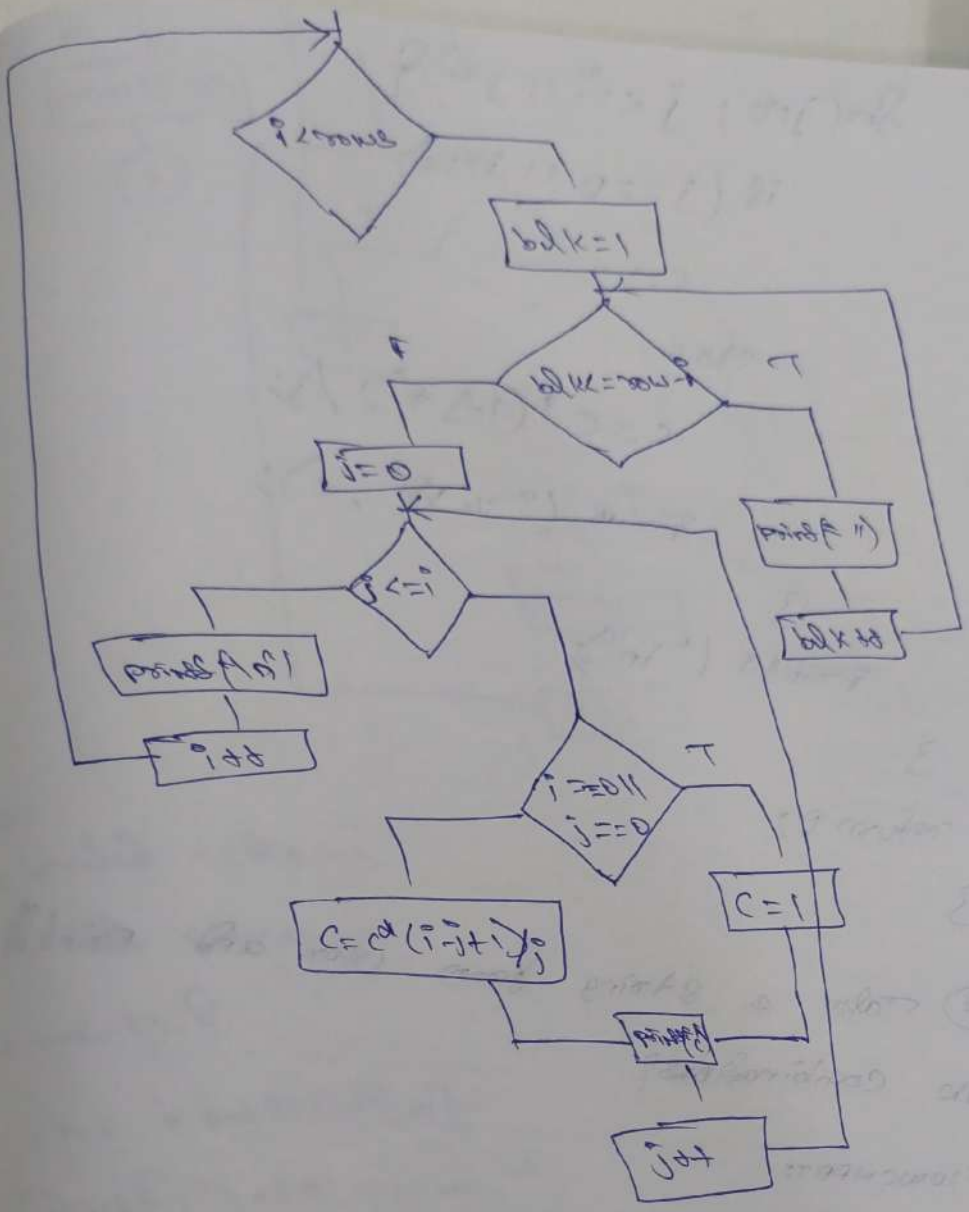
```
1 2 | 1 2
3 4 | 3 4
```

(26) Pascal's triangle:

```
        1
      1   1
    1   2   1
  1   3   4   3   1
1   4   7   7   4   1
```

Flowchart:





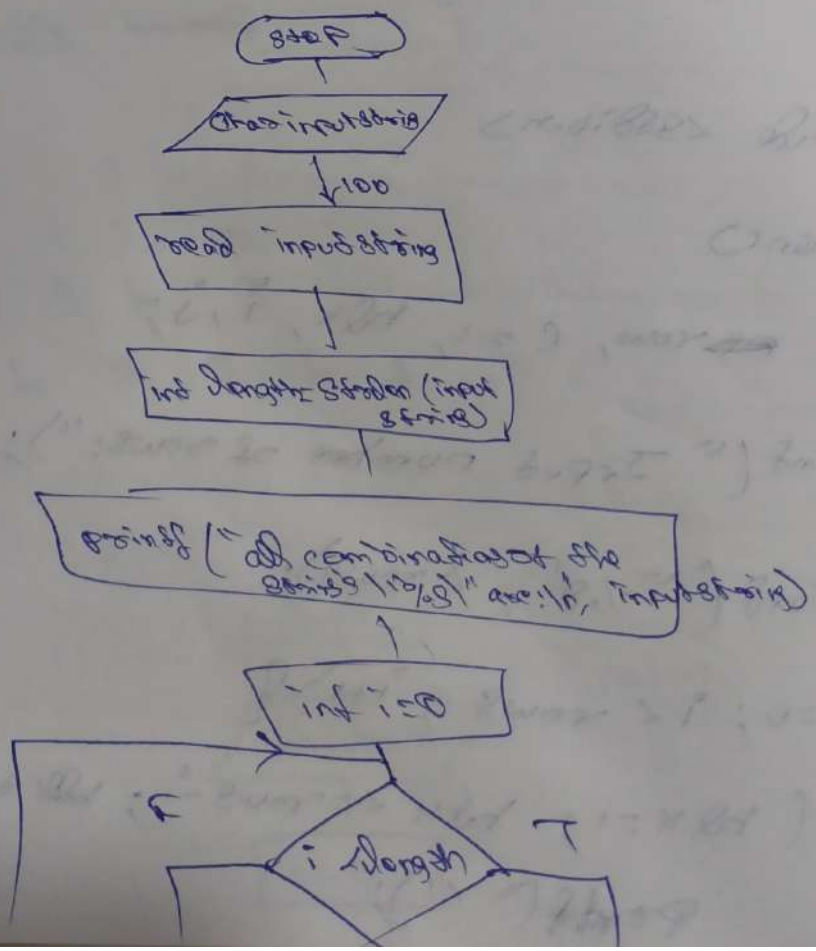

code:

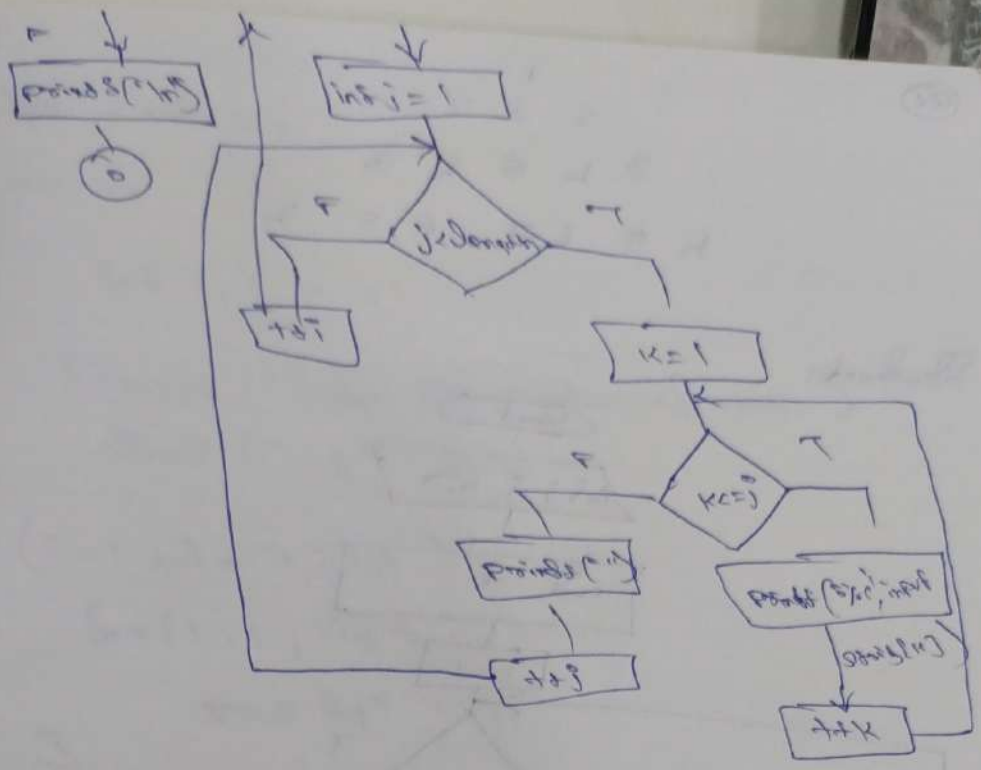
```
#include <stdio.h>

int main()

  int row, c=1, blk, i, j;

  Printf(" Input number of rows: ");

    Scanf("%d", &rows);

for(i=0; i<rows; i++){

  for( blk=1; blk<=rows-i; blk++)

    Printf(" ");
```

```
for(j=0; j<=i; j++){
    if(j==0 || i==0)
        c=1;
    else
        c=c*(i-j+1)/j;
    printf("%d ", c);
}
printf("\n");
}
return 0;
}
```

(27) Take a string from user and printf all the combinations?

FLOWCHART:






Code:

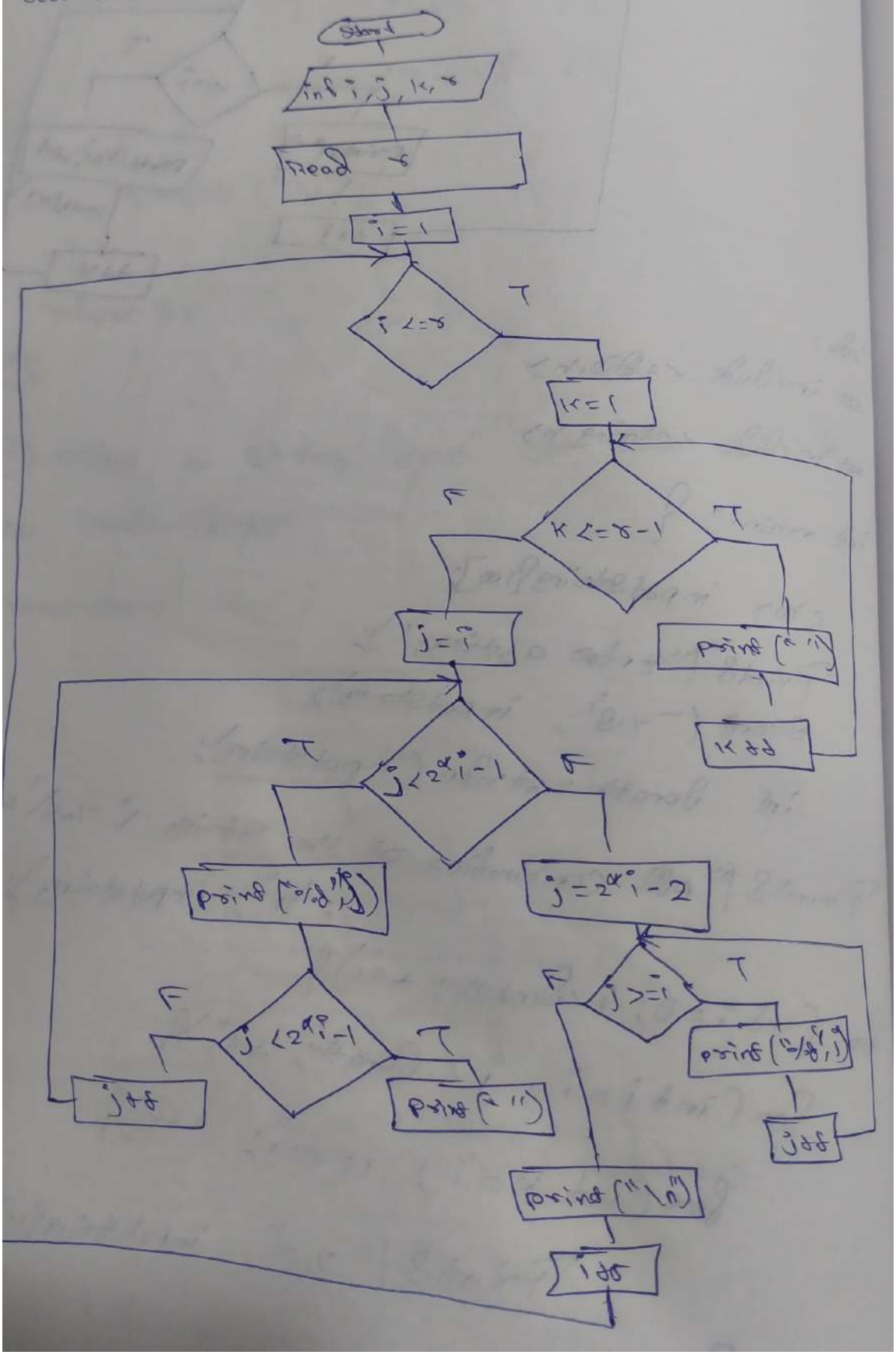
```
#include <stdio.h>
#include <string.h>
int main() {
    char inputstring[10];
    printf("Enter a string:");
    scanf("%s", inputstring);
    int length = strlen(inputstring);
printf("All combinations of the string \"%s\" are
                    :\n", inputstring);
for (int i=0; i<length; i++){
    for(int j=i; j<length; j++){
        for(int k=i; k<=j; k++){
            printf("%c", inputstring[k]);
        }
        printf("\n");
    }
}
```

3 3 3

(24)

```
        1
      2 3 2
    3 4 5 4 3
  4 5 6 7 6 5 4
```

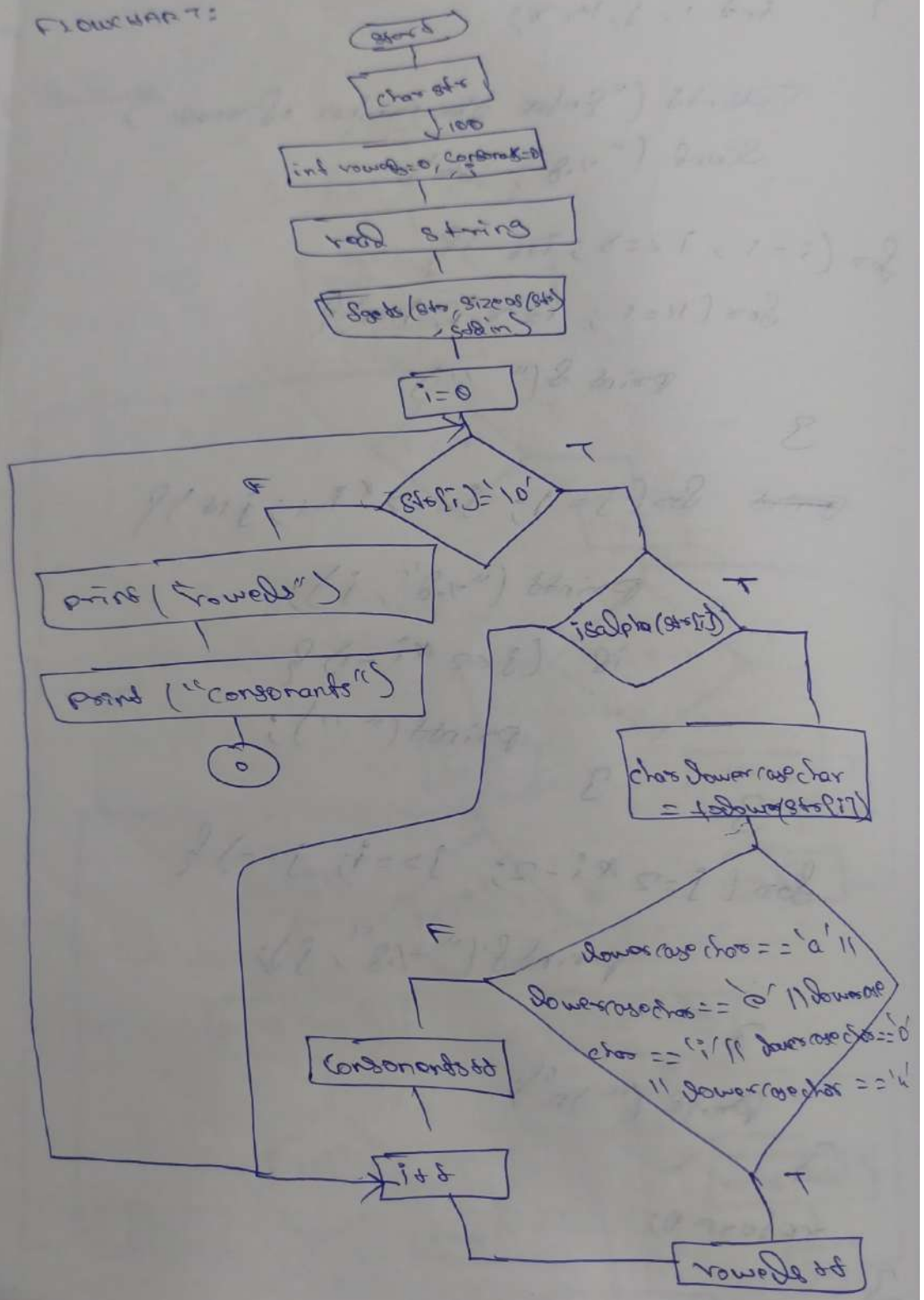
Flowchart:



```
#include <stdio.h>
int main()
{
    int i, j, k, r;
    printf("Enter the number of rows: ");
    scanf("%d", &r);
    for (i = 1; i <= r; i++) {
        for (k = 1; k <= r - i; k++) {
            printf(" ");
        }
        for (j = 1; j <= 2 * i - 1; j++) {
            printf("%d", j);
            if (j < 2 * i - 1) {
                printf(" ");
            }
        }
        for (j = 2 * i - 2; j >= 1; j--) {
            printf("%d", j);
        }
        printf("\n");
    }
    return 0;
}
```

(29) find no. of vowels and consonants in a string

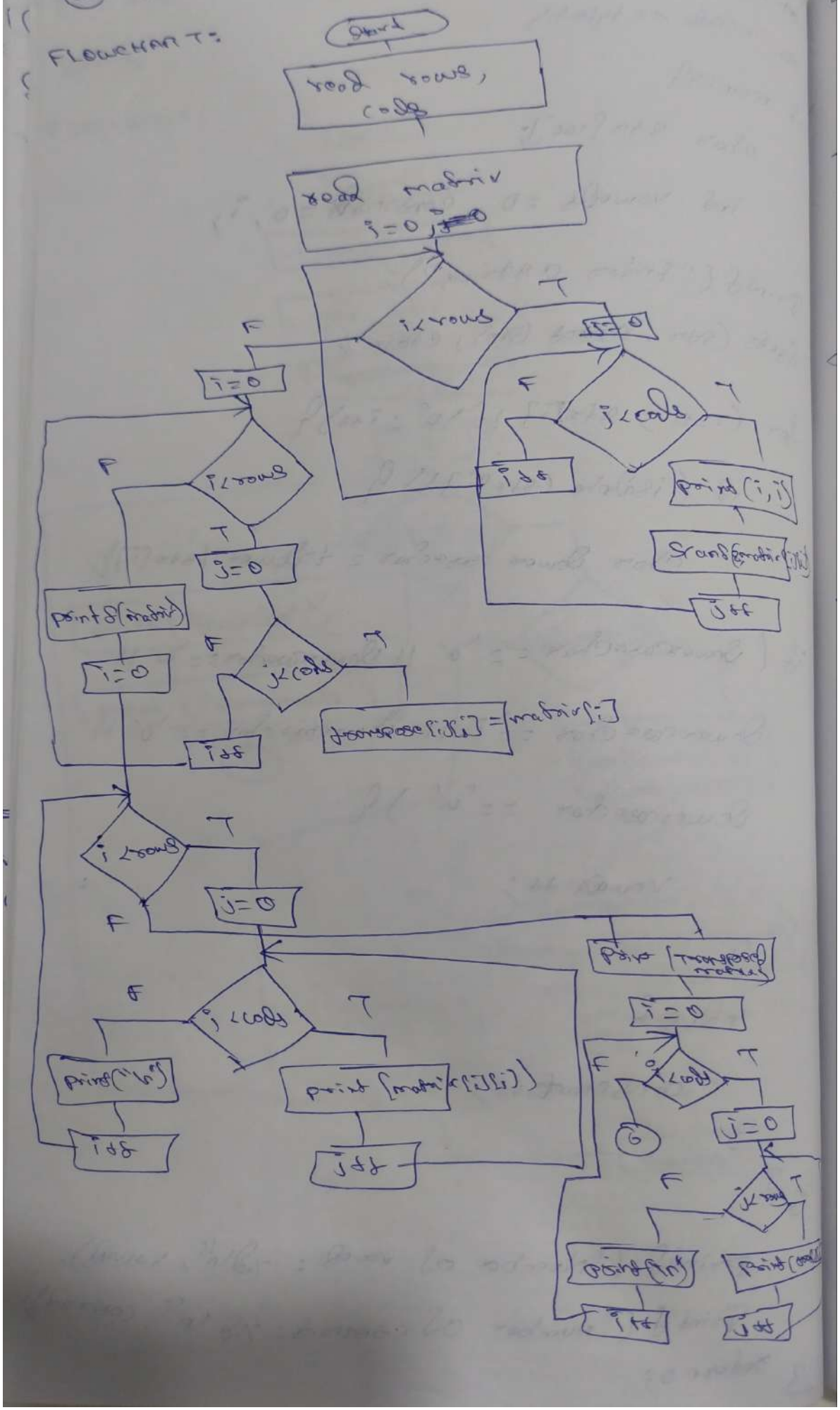
Flowchart:



```
#include <stdio.h>
#include <ctype.h>
int main(){
    char str[100];
    int vowels = 0, consonants = 0, i;
printf("Enter a string");
fgets(str, sizeof(str), stdin);
for (i=0; str[i] != '\0'; i++){
    if (isalpha(str[i])){
        char lowercasechar = tolower(str[i]);
if (lowercasechar == 'a' || lowercasechar == 'e' ||
    lowercasechar == 'i' || lowercasechar == 'o' ||
    lowercasechar == 'u' ){
        vowels ++;
    }
    else{
        consonants ++;
    }
}
    printf("Number of vowels : %d\n", vowels);
    printf("Number of consonants : %d\n", consonants);
} return 0;
```

(30) Transpose a matrix?

FLOWCHART:



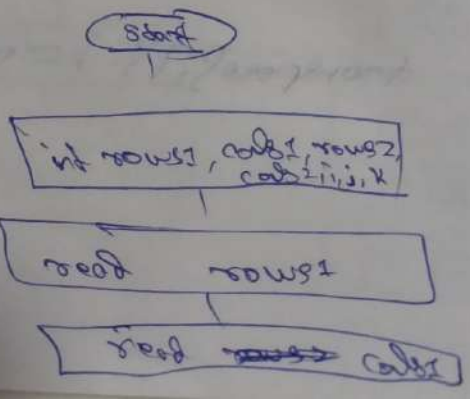
code:

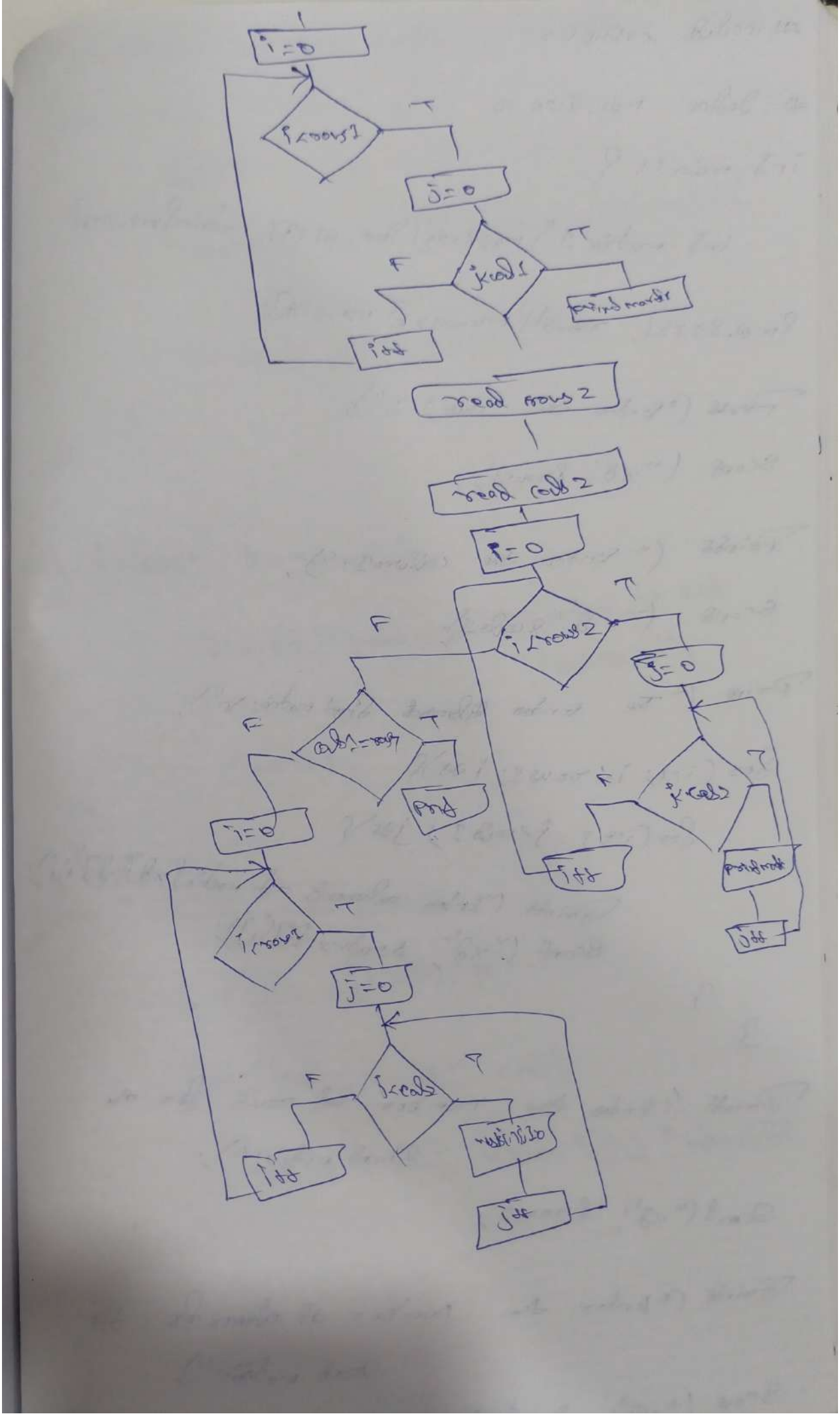
```
#include <stdio.h>
#define max_size 10
int main(){
    int matrix[max_size][max_size], transpose[max_size]
                                        [max_size];
    int rows, cols, i, j;
printf("Enter the number of rows: ");
scanf("%d", &rows);
Printf("Enter the number of colums: ");
scanf("%d", &cols);
Printf("Enter the elements of the matrix: \n");
for(i=0; i<rows; i++){
    for(j=0; i<cols; j++){
        Printf("Enter element of matrix[%d][%d]: ",
                                        i, j);
        scanf("%d", &matrix[i][j]);
}}
for(i=0; i<rows; i++){
    for(j=0; j<cols; j++){
        transpose[j][i] = matrix[i][j];
}}
```

```
printf ("\n original matrix:\n");
for (i=0; i<rows; i++){
    for(j=0; j<cols; j++){
        printf ("%d", matrix[i][j]);
    }
    printf ("\n");
}

printf ("\n Transposed matrix :\n");
for (i=0; i<cols; i++){
    for (j=0; j<rows; j++){
        printf ("%d\t", transpose[i][j]);
    }
}
printf("\n");
return 0;
}
```

(3) multiplication of matrix:

FLOW CHART:





```
#include <stdio.h>
#define max_size 10
int main() {
    int matrix1[max_size][max_size], matrix[max_size][max_size],
    result[max_size][max_size];
    printf("Enter the rows 1 : ");
    scanf("%d", &row1);
    printf("Enter the columns 1 : ");
    scanf("%d", &col1);
    printf("The Enter elements first matrix: \n");
    for(i=0; i<row1; i++){
        for(j=0; j<col1; j++){
            printf("Enter elements at matrix[%d][%d]:",
            scanf("%d", &matrix1[i][j]);
    }
}
```

(3)
```
printf("Enter the number of rows for the
        second matrix: ");
scanf("%d", &row2);
printf("Enter the number of columns for the
        second matrix: ");
```

```
printf("\nEnter the second matrix:\n");
for (i=0; i<row2; i++){
    for (j=0 ; j<col2; j++){
        printf("Enter element at ,matrix[i+1][j+1]
                i, j);
        scanf("%d", &matrix2[i][j];
    }
}

if (col1 != row2) {
    printf("matrix multiplication is not
            possible\n");
    return 1;
}

for (i=0; i<row1; i++){
    for (j=0 ; j<col2; j++)
    for (k=0 ; k<col1; k++){
        result[i][j] += matrix1[i][k] * matrix2[k][j];
    }
}
```