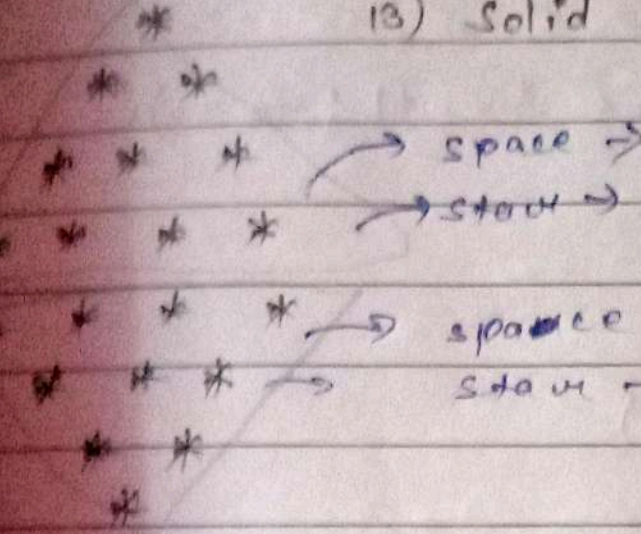
Lecture - 4

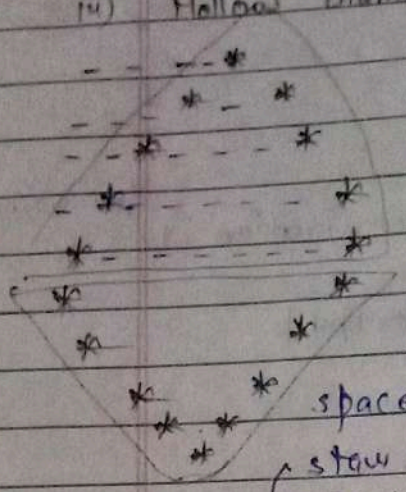
13) Solid Diamond



Done by vs code

* Even = 2*n
* Odd = 2*n-1
= 2*n+1

14) Hollow Diamond

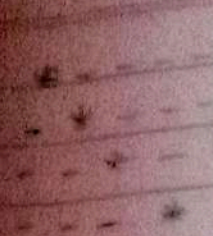


n=5

row 0 → - - - - *
row 1 → - - - * * - *
row 2 → - - * - - * *
row 3 → - * - - - * *
row 4 → * - - - - * - * - - *

space → n-row-1
star space → 2*row+1
if (col==0) → * → first char
else if(col==2*row) → * → last char
else ( ) → -

I^st part for(int row=0; row<n; row++){
for (int col=0; col<n-row-1; col++){
cout << " ";
}
for (int star=0; star<2*row+1; star++){
if (col==0)
{ cout << "*";
}
else if(col==2*row)
{ cout << "*";
}
else {
cout << " ";
}
}
}



row 0 → 0 space & 7 star
row 1 → 1 space & 5 star
row 2 → 2 space & 3 star
row 3 → 3 space & 1 star

space → row
col, star space 2n - 2*row - 1
→ if (col = 0 || col = 2n - 2*row - 2) → *
→ else ( ) → space

```
for(int row = 0; row < n; row++)
{ for(int space = 0; space < row; space++)
  { cout << " ";
  }
 for(int star = 0; star < (2*n - 2*row - 1); star++)
  { if (col = 0 || col = 2n - 2*row - 2)
    { cout << "*";
    }
   else{
    cout << " ";
   }
  }
 cout << endl;
}
```

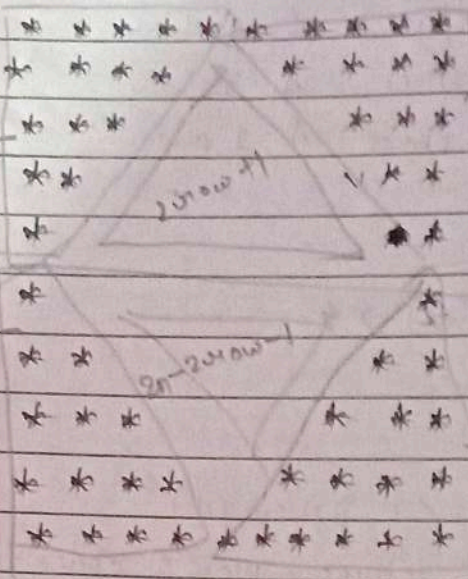
Done in vscode

Imp

1, 3, 5, 7 → 2*row+1

7, 5, 3, 1 → 2n - 2*row - 1

15) Flipped solid Diamond




```
for (int row=0; row<n; row+1){
    // half pyramid
    for (int col=0; col<n-row; col+1){
        cout << "*";
    }
    // space wise full pyramid
    for (int col=0; col<2*row+1; col++){
        cout << " ";
    }
    for (int col=0; col<n-row; col++){
        cout << "*";
    }
    cout << endl;
}
```

Done

(6) Fancy Pattern 2

```
1
2*2
3*3*3
4*4*4*4
3*3*3
2*2
1
```

row 0 → 1 print
row 1 → 1 print *
row 2 → 3
row 4 → 4

```
for (int row=0; row<n; row++){
  for (int col=0; col<=row+1; col++){ → number
      cout << row+1;
      if (col != row){
        cout << "*";
      }
  }
  cout << endl;
}

for (int row=0; row<n; row++){
  for (int col=0; col<n-row; col++){
      cout << "n-row";
      if (col != n-row-1){
        cout << "*";
      }
  } cout << endl;
}
```

Done

(17) 1 row+1

2 2

3 3 3

4 4 4 4

5 5 5 5 5

```
for(int row = 0; row < n; row++){
for(int col = 0; col < row+1; col++){
  cout << row+1;
  }
  cout << endl;
 }
}
```

(18) Alphabetical Palindrome Pyramid:-

(19) A

A B A

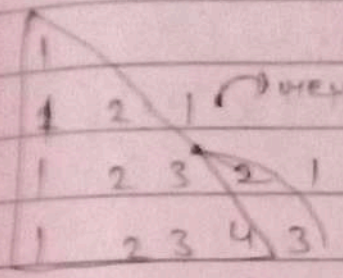
A B C B A

A B C D C B A

A B C D E D C B A

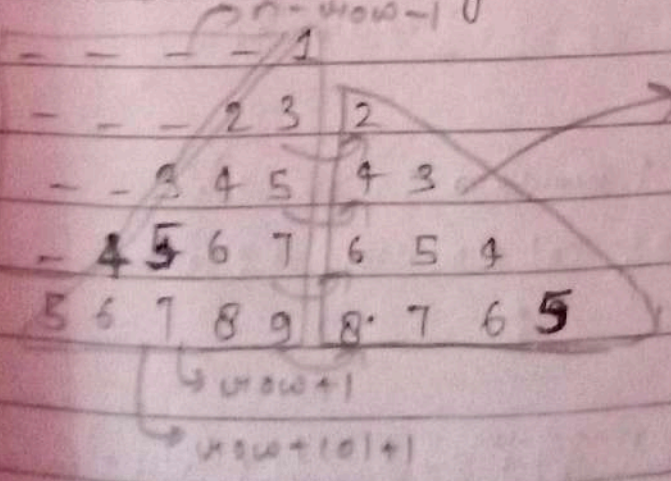
```
for(int row = 0; row < n; row++){
 for(int col = 0; col < row+1; col++){
   int ans = col+1;
   char ch = ans + 'A' - 1;
   cout << ch;
  }
for(int col = row; col >= 1; col--){
   int ans = col;
   char ch = ans + 'A' - 1;
   cout << ch;
  }
 cout << endl;
}
```

(9) 1 (done (Backside)
1 2
1 2 3
1 2 3 4

(10)



```
for(int row=0; row<n; row=row+1){
  int col;
  for(col=0; col<row+1; col=col+1){
    cout << col+1;
  }
  col=col-1; or (int col=row)
  for(; col>=1; col--)
  { cout << col;
  }
  cout << endl;
}
```

(11) Numeric full pyramid:-



```
for(int row=0; row<n ; row++){
  for(int col=0; col<n-row-1 ; col++){
    cout << " ";}
  for(int col=0; col<row+1 ; col++){
    cout << row+col+1;
    } int start = 2*row;
  for(int col=0; col<row ; col++){
    cout << start;
    start = start-1;
  }
  cout << endl;
}
```

(11) Numeric Hollow Pyramid (Do it 5 times)

```
    1          row 0 → 1 element
   1 2         row 1 → 3 elem
  1   3        row 2 → 5 elem
 1     4       row 3 → 7 elem
1 2 3 4 5      row 4 → 9 elemen
```

2*row+1

```
for(int row=0 ; row<n ; row++){
  for(int col=0; col<n-row-1 ; col++){
    cout<<" ";
  } else {
    for(int col=0; col<2*row+1 ; col++){
      if (col==0)
```

space

if ← condition

next page

```
* [illegible] {
    cout << 1; }
else if (col == 2 * row) {
    cout << row + 1;
else {
    cout << " ";
}}}
cout << endl;
}
}
```

```
if (row == n-1) {
for (int col = 1;
  col <= n; col++) {
cout << col << " ";
}
}
```

(20) Hollow Square Pattern

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

n=5

row 0 → 5 *

row 1 → 3 space, 2 star

row 2 → 3 space, 2 star

row 3 → 3 space, 2 star

row 4 → 5 *

```
for (int row = 0; row < n; row++) {
  for (int col = 0; col < n; col++) {
    if (col == 0 || col == n-1) {
      cout << "*";
    }
    else if (row == 0 || row == n-1) {
      cout << "*";
    }
    else {
      cout << " ";
    }
  }
  cout << endl;
}
```

(21) Hollow Inverted Half Pyramid:-

```
* * * * *   row -> 5 *
*       *   row 1 -> 2 *, 2 space   col=0, 1
*   *       row 2 -> 2 *, 1 space   col=0, 2
* *         row 3 -> 2 *            col=0, 2
*           row 4 -> 1 *
```

```
for(int row = 0; row < n ; row++){
  for(int col=0; col < n ; col++){
    if( row == 0 || col == 0 || col = n-row-1)
      cout << "*";
    }
  else {
    cout << " ";
  }
}
```

Done

(22) 4 4 4 4
3 3 3
2 2
1

```
for(int row = 0; row < n; row++
  for(int col = 0; col < n-row; col++)
    {
      cout << n-row << " ";
    }
  cout << endl;
}
```

Done

Same as Question 23 but code diff^h

(23) Hollow Full Pyramid :- n = 5

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

row 0 → 4 space, 1 star, 1 elem
row 1 → 3 space, 2 star, 3 elem
row 2 → 2 space, 2 star, 5 ele
row 3 → 1 space, 2 star, 7 ele
row 4 → 0 space, 5 star, 9 ele

n - row - 1 | 2 * row + 1

Space → n - row - 1
if (row == n-1) → last line
for (2*row+1)
if (col % 2 == 0) → *
else → space
else if (2*row+1) → first 2 line
if (col == 0 || col == 2*row) → *
else → space

(24) Numeric Hollow Half Pyramid :-

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

```
for (int row=0; row<n; row++){
    for (int col=0; col<row+1; col++){
        if (col==0 || col==row || row==n-1){
            cout << col+1;
        }
        else{
            cout << " ";
        }
    }
    cout << endl;
}
```

```
else {
    cout << " ";
  }
}
cout << endl;
}
```

(25)

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

```
for (int row = 0; row < n; row++){
for (int col = 0; col < row+1; col++){
    if ( col == 0 || col == row ||
                row == n-1){
    cout << "*";
    }
    else {
    cout << " ";
    }
}
cout << endl;
}
```

Numeric Hollow Inverted Half Pyramid

(26)

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

```
for (int row = 0; row < n; row++){
    for (int col=0; col < n-row; col++){
        if (row==0 || col==0 || col==n-row-1){
        cout << col+row+1;
        }
        else {
            cout << " ";
        }
    }
    cout << endl;
}
```

(27) Numeric Palindrome Equilateral Pyramid:-

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

```
for (int row=0; row<n; row++){
    for (int col=0; col<n-row-1; col++){
        cout << " ";
    }
    for (int col=0; col<row+1; col++){
        cout << col+1;
```

```
// reverse printing
for (int col = row; col >= 1; col--)
    cout << col;
}
cout << endl;
}
```

Pause

Tea

vs

Code

(22) Fancy Pattern 1

n=5

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

```
for (int row = 0; row < n; row++){
    for (int col = 0; col < n + 3 - row; col
    cout << "*";
}
for (int col = 0; col < (2*row+1); col++){
    if (col % 2 == 0){
        cout << row+1;
}
```

```
else{
    cout << " * ";
    }
}
for(int col = 0; col < n+8-row; col++){
    cout << " * ";
}
cout << endl;
}

}
```

Done

In

vs code

(2) Solid Half Diamond :-

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

row 0 → 2 (row+1)

row 1 → 2

row 2 → 3

row 3 → 4

row 0 → 4 (n-row)

row 1 → 3

row 2 → 2

row 3 → 1

```
for(int row = 0; row < n ; row++){
    for(int col = 0; col < row+1 ; col++){
        cout << " * ";
    }
    cout << endl;
}
```

```
for (int vrow = 0; vrow < n ; vrow++){
    for (int col = 0; col < n - vrow; col++){
        cout << "*";
    }
    cout << endl;
}

}
```

(30) Fancy Pattern 3

```
*                   vrow 0 → 1
* 1 *               vrow 1 → 3
* 1 2 1 *           vrow 2 → 5
* 1 2 3 2 1 *       vrow 3 → 7
* 1 2 1 *           vrow 0 → 5
* 1 *               vrow 2 → 3
*                   vrow 2 → 1
```

2vrow+1

2n - 2vrow

Solution

```
for (int vrow = 0; vrow < n ; vrow++){
    int count = 0;
    for (int col ; col < 2vrow+1; col++){
        if (col == 0 || col == 2vrow){
            cout << "*";
            if (count < 1){
                for (int col1 = 0; col1 < vrow; col1++){
                    cout << col+1;
                    col++;
                }
            }
        }
    }
```

```
if (count < 2){
for (int col2=0; col2 < row-1; col2++){
    cout << row-1-col2;
    col++;
}}
count++;
}
```

cout << endl;
}

(33) Fancy Pattern 2

```
1                   row 0 → 1
2 * 3               row 1 → 3
4 * 5 * 6           row 2 → 5
7 * 8 * 9 * 10      row 3 → 7
```

2*row+1

```
7 * 8 * 9 * 10      row 0 → 7
9 * 5 * 6           row 1 → 5
2 * 3               row 2 → 3
1                   row 4 → 1
```

2n-2*row-1

```
int count = 1;
for(int row = 0; row < n; row++){
for (int col = 0; col < 2*row+1; col++){
    if (col%2 == 0){
        cout << count;
        count++;
    }
    else{
        cout << "*";
    }} cout << endl;
}
int count2 = count-1;
for (int row=0; row < n; row++){
    int count2 = count2-1;
for (int col=0; col < 2n-2*row-1; col++){
    if (col%2 == 0){
        cout << count2;
```

```
count 2 ++; }
else { cout << "*"; }
}}
count 1 = count 2 - (n - row - 1);
cout << endl;
}
```

(33) Floyd's Triangle Pattern

	0	1	2	3	4	5	6	7
row 0 →	1							
row 1 →	2	3						
row 2 →	4	5	6					
row 3 →	7	8	9	10				
row 4 →	11	12	13	14	15			
row 5 →	16	17	18	19	20	21		
row 6 →	22	23	24	25	26	27	28	

n = 7

(row + 1)

```
int count = 1
for (int row = 0; row < n ; row ++){
    for (int col = 0 ; col < row+1 ; col++)
    {
        cout << count << " ";
        count ++;
    }
    cout << endl;
    }
}
```

(5) Pascal's Triangle Pattern

row 0: 1
row 1: 1 1
row 2: 1 2 1
row 3: 1 3 3 1
row 4: 1 4 6 4 1
row 5: 1 5 10 10 5 1
row 6: 1 6 15 20 15 6 1

n = 7

Pascal Triangle make Binomical Cofficient formula: c = c * (row - col) / col;

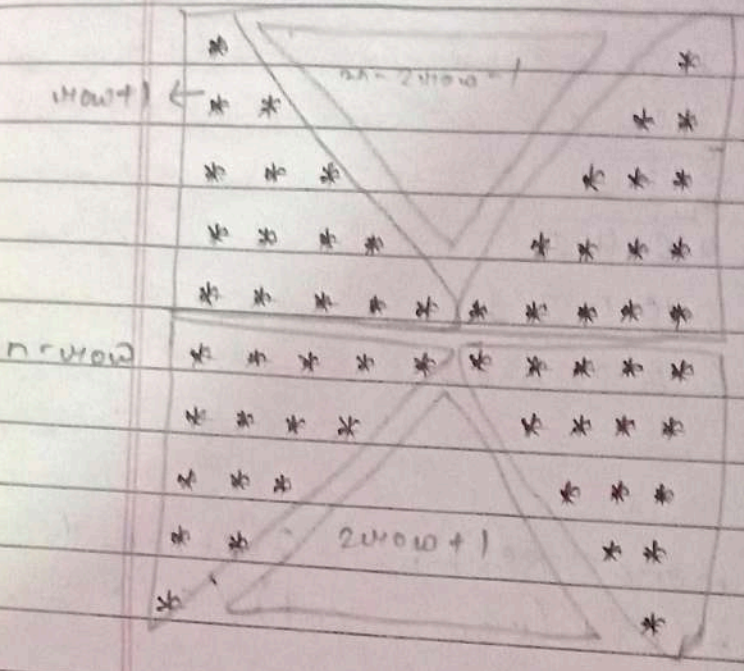
count

```
for (int row = 1; row <= n; row++) {
    int count = 1;
    for (int col = 1; col <= row; col++)
    {
        cout << count << " ";
        count = count * (row - col) / col;
    }
    cout << endl;
}
```

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

1+1 = 2
1+2 = 3

(34) Butterfly Pattern




```
for(int row = 0; row < n ; row++){
    for(int col1 = 0; col1 < row+1 ; col1++)
      cout << "*" ; }
for(int col2 = 0 ; col2 < 2n-2row-1; col2++)
{ cout << " " ; }
for(int col3 = 0 ; col3 < row+1 ; col3++){
    cout << "*" ; }
cout << endl;
}
```

```
for(int row = 0; row < n ; row++){
    for(int col1 = 0; col1 < n-row ; col1++){
      cout << "*" ; }
for(int col2 = 0 ; col2 < 2row+1 ; col2++)
    cout << " " ; }
for(int col3 = 0 ; col3 < n-row; col3++)
    cout << "*" ; }
cout << endl;
}
```

~~Lecture 26~~

operator new, function for dot move

(35)
```
1 1 1
2 2 2
3 3 3
```

```
for(int row=1; row<=n; row++){
for(int col=1; col<=n; col++){
 cout << row;
}
cout << endl;
}
```

(36)
```
1 2 3 4
1 2 3 4
1 2 3 4
1 2 3 4
```

```
for(int row=1; row<=n; row++)
{
for(int col=1; col<=n; col++){
 cout << col;
}
cout << endl;
}
```

(37)
```
3 2 1
3 2 1
3 2 1
```

```
for(int row=1; row<=n; row++){
for(int col=n; col>=1; col--){
 cout << col;
}
cout << endl;
}
```

(38)
```
1 2 3
4 5 6
7 8 9
```

```
int count=1;
for(int row=1; row<=n; row++){
for(int col=1; col<=n; col++){
cout << count;
count++;
}
cout << endl;
}
```

(39)
```
1
2 2
3 3 3
4 4 4 4
```

0 → (row+1)

```
for(int row=1; row<=n; row++){
for(int col=1; col<=row; col++)
{
cout << row;
}
cout << endl;
}
```

(40)
```
1
2 3
4 5 6
7 8 9 10
```

```
int count=1;
for(int row=1; row<=n;
{
for(int col=1; col<=row; col++
{ cout << count << " ";
count++;
}
cout << endl;
}
}
```

(41)
```
1
2 3
3 4 5
4 5 6 7
```

0 -> row+1

```
for (int row=0; row<n; row++){
  for (int col=0; col<row+1; col++){
    cout << col+row+1 << " ";
  }
  cout << endl;
}
}
```

(42)
```
1
2 1
3 2 1
4 3 2 1
```

```
#include <iostream>
using namespace std;

int main(){
int n;
cout << "Enter number ";
cin >> n;
for (int row=1; row<=n; row++){
  for (int col=row; col>=1;
    col--){
    cout << col << " ";
  }
  cout << endl;
}
```

(13)
```
A A A
B B B
C C C
```

```
for(int row = 0; row < n; row++)
{
    for(int col = 0; col < n; col++)
    {
        char ans = row + 'A';
        cout << ans;
    }
    cout << endl;
}
```

(14)
```
A B C
A B C
A B C
```

```
for(int row = 0; row < n; row++)
{
    for(int col = 0; col < n; col++)
    {
        char ans = col + 'A';
        cout << ans;
    }
    cout << endl;
}
```

(15)
```
A B C
D E F
G H I
```

```
int count = 0;
for(int row = 0; row < n; row++)
{
    for(int col = 0; col < n; col++)
    {
        char ans = count + 'A';
        cout << ans;
        count++;
    }
    cout << endl;
}
```

i^{th} row + col - 2

(16)
```
A B C
B C D
C D E
```

```
for (int row = 1 ; row <= n ; row++){
  for (int col = 1 ; col <= n ; col++) {
    char ans = 'A' + col + row - 2;
    cout << ans;
  }
  cout << endl;
}
}
```

(17)
```
1 2 3
2 3 4
3 4 5
```

```
for (int row = 0; row < n; row++){
  for (int col = 1; col <= n ; col++){
    cout << col + row;
  }
  cout << endl;
}
}
```

(18)
```
A
B B
C C C
```

```
for (int row = 0; row < n; row++){
  for (int col = 0; col < row + 1;
                    col++) {
    char ans = 'A' + row;
    cout << ans;
  }
  cout << endl;
}
}
```

(49)

```
A
B C
D E F
G H I J
int count = 0;
for(int row = 1; row <= n ; row++){
  for (int col = 1; col <= row ; col++){
    char ans = 'A' + count;
      cout << ans;
      count++;
    }
    cout << endl; }}
```

(50)

```
A
B C
C D E
D E F G
E F G H I

for (int row = 0; row < n ; row++){
  for (int col = 0; col < 2*row+1 ; col++){
    char ans = 'A' + col + row;
      cout << ans;
    }
    cout << endl;
  }
```

(51)
```
D
C D
B C D
A B C D
```

```
for(int row=0; row<=n; row++){
    for(int col=1; col<=row; col++)
    { cout << ans << " ";
      ans++;
    }
    cout << endl;
}
```

```
char ans =
'A' + n - row;
```

(52)
```
_ _ _ *
_ _ * *
_ * * *
* * * *
```

```
for(int row=0; row<n; row++)
{ for(int col=0; col<n-row-1;
                  col++)
  { cout << " ";
  }
  for(int col1=0; col1<row+1; col1++){
    cout << "*";
  }
  cout << endl;
}
```

(53)
```
* * * *
_ * * *
_ _ * *
_ _ _ *
```

```
for(int row=0; row<n; row++){
    for(int col=0; col<row; col++){
        cout << " ";
    }
    for(int col1=0; col1<n-row; col1++){
        cout << "*";
    }
    cout << endl;
}
```

(54)

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

```
for(int row=0; row<n; row++)
{
    for(int col=0; col<row; col++)
    {
        cout << " ";
    }
    for(int col1=0; col1<n-row; col1++)
    {
        cout << row+1;
    }
    cout << endl;
}
```

(55)

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

```
for(int row=0; row<n; row++)
{
    for(int col=0; col<n-row-1; col++)
    {
        cout << " ";
    }
    for(int col1=0; col1<row+1; col1++)
    {
        cout << row+1;
    }
    cout << endl;
}
```

(56)

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

```
for(int row=0; row<n; row++)
{
    for(int col=0; col<row; col++)
    {
        cout << " ";
    }
    for(int col1=1; col1<=n-row; col1++)
    {
        cout << col1+row;
    }
    cout << endl;
}
```

(57)

```
1
2 3
4 5 6
7 8 9 10
```

```
int count = 1;
for(int row=0; row<n; row++){
for (int col=0; col<n-row-1; col++){
cout << " ";
}
for(int col1=1; col1<=row+1; col1++){
cout << count;
count++;
}
cout << endl;
}
```

(58)

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

```
for(int row=0; row<n; row++){
for (int col=0; col<n*row; col++){
cout << col+1;
}
for(int col=0; col<2*row+1; col++){
cout << "*";
}
for (int col3=n-row; col3>=1; col3--){
cout << col3;
}
cout << endl;
}
```

(50)

9 8 7
6 5 4
3 2 1

```
int count = n*n;
for(int row = 0; row < n; row++){
  for(int col = 0; col < n; col++){
    cout << count << " ";
    count--;
  }
  cout << endl;
}
```

(51)

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

```
for(int row = 0; row < n; row++){
  for(int col = 0; col < n; col++){
    cout << "* ";
  }
  for(int col = 0; col < n; col++){
    cout << "  ";
  }
}
```

```
for (int col=0; col<n; col++){
    cout << " * ";
} cout << endl; }
for (int row=0; row<n-1; row++){
    for (int col=0; col<n; col++){
        cout << " * ";
}
for (int col=0; col<n; col++){
    cout << " * "; }
for (int col=0; col<n; col++){
    cout << " * "; }

for (int row=0; row<n; row++){
    for (int col=0; col<n; col++){
        cout << "*";
}
```

(6) 3 3 3 3 3
3 2 2 2 3
3 2 1 2 3
3 2 2 2 3
3 3 3 3 3

n=3

In vs code
code
Done

```
for (int col=0; col<n;
col++){
cout << " ";
}
```

```
for (int col=0; col<n; col++){
    cout << " * ";
}
cout << endl;
}
```