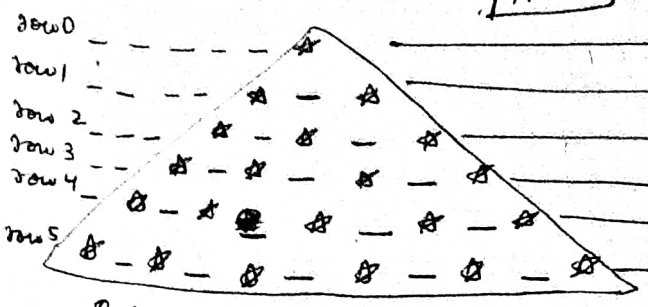
03/Feb/2023

Lec-4
Some more Patterns

① Full Pyramid →

$n = 6$



5 space 1 star
4 space 2 ★
3 space 3 ★
2 space 4 ★
1 space 5 ★
0 space 6 ★

- row 0 → 5 spaces ~~(n-row-1)~~
- row 1 → 4 spaces
- row 2 → 3 spaces
- row 3 → 2 spaces
- row 4 → 1 spaces
- row 5 → 0 spaces

~~(n-row-1)~~

(n-row-1)

```
int n;
cin >> n;
for(int row = 0; row < 5; i++ = 1){
    for (int col=0; col< n-row-1; col++)   // spaces (depends on rows as n-row-1)
    {
        cout << " ";
    }

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

row 0 → 1 ★ → row+1
row 1 → 2 ★
row 2 → 3 ★
and so on n → n+1 ★

without space after each star it become half triangle.

and space after each star.

→ $n = 5$

row = 0 → 4 space & 1 star
row = 1 → 3 space, 2 star
row = 2 → 2 space, 3 star
row = 3 → 1 space, 4 star
row = 4 → 0 space, 5 star

0 space = n-row-1
5 star = row+1

⊙ Inverted Full Pyramid →

$n = 4$

row 0 → ★ ★ ★ ★ → 0 space, 4 ★
row 1 → _ ★ ★ ★ → 1 space, 3 ★
row 2 → _ _ ★ ★ → 2 space, 2 ★
row 3 → _ _ _ ★ → 3 space, 1 ★

$n = 4$

row	space	star
0	0	4
1	1	3
2	2	2
3	3	1

→ $n-0=4$
$n-1=3$
$n-2=2$
$n-3=1$

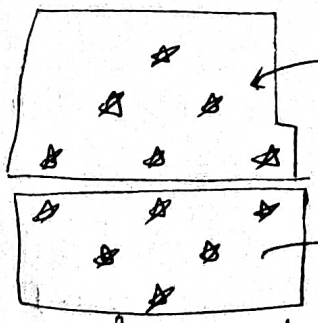
row n-row

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

$n=5$

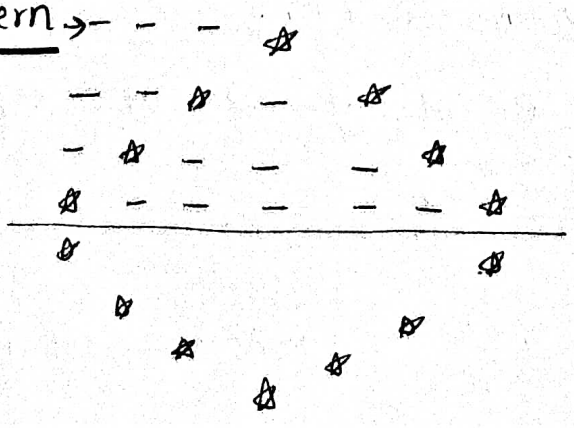
	Star	Star	Star	Star	Star
row=0	0	1	0	5	4
row=1	1	2	0	4	3
row=2	2	3	1	3	2
row=3	3	4	2	2	1
row=4	4	5	3	1	0
	row	row+1	row-1	n-row	n-row-1

③ **Solid Diamond** → $n=3$

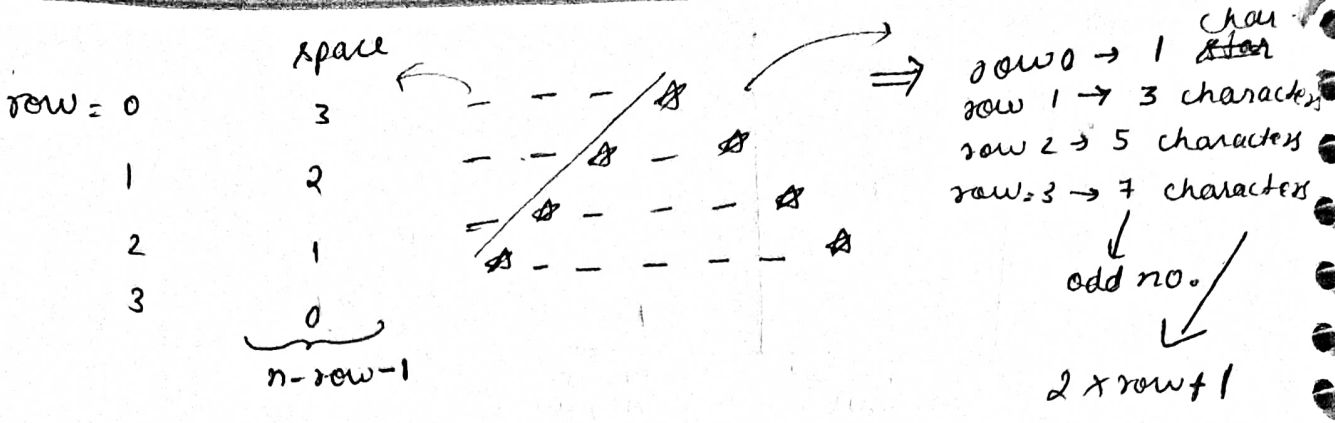


~~Inverted~~ Full pyramid and Inverted pyramid together.

④ **Hollow Pattern** →

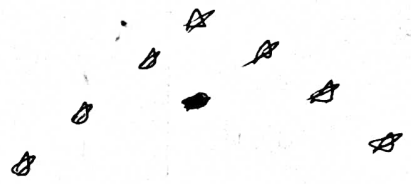


n = 4



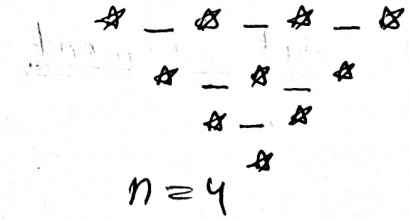

```
int n;
cin >> n;
for (int row = 0; row < n; row++){        // run for each row
    // spaces
    for (int col = 0; col < n-row-1; col++)
        cout << " ";
    }
    // stars
    for (int col = 0; col < 2*row+1; col++)
    {
        // if first character or if last character
        if (col == 0){
            cout << "*";
        }
        else if (col == 2*row)
            cout << "*";
        else
            cout << " ";
    }
    cout << endl;
}
```

Part 1



row → 0 → 0 space
row → 1 → 1 "
row → 2 → 2 "
row → 3 → 3 "
↓
row

row 0 → 7 ch
row 1 → 5 ch
row 2 → 3 ch
row 1 → 1 ch
2n − [2 × row + 1]

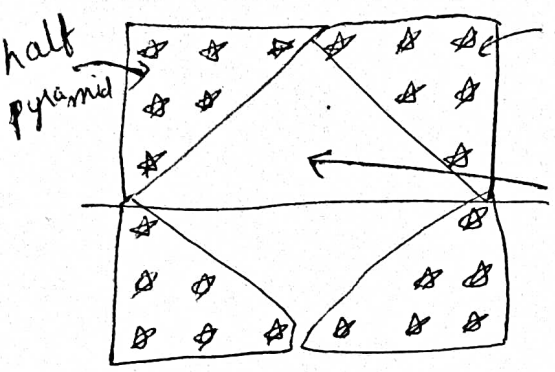



```
for (int row = 0; row < n; row++){
    // spaces
    for (int col = 0; col < row; col++){
        cout << " ";
    }
    // star
    for (int col = 0; col < 2*n-2*row-1; col++)
```

// part 2

```
    {
        if(col==0 || col==2*n-2*row-2){
            cout<<"*";
        }
        else{
            cout<<" ";
        }
    }
    cout<<endl;
}
```

⊙ Flipped Solid Diamond →

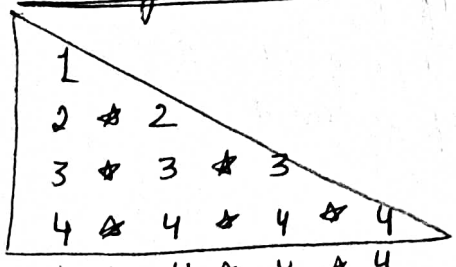


half pyramid
full pyramid (space)
half pyramid

```
int n;
cin>>n;
//I st part
for(int row=0; row<n; row++){
    // half pyramid
    for(int col=0; col<n-row; col++){
        cout<<"*";
    }
    // full pyramid
    for(int col=0; col<2*row+1; col=col+1){
        cout<<" ";
    }
    // half pyramid
    for(int col=0; col<n-row; col++){
        cout<<"*";
    }
    cout<<endl;
}
// 2nd part
for(int row=0; row<n; row++){
```

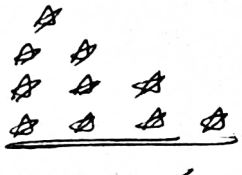
```
                pyramid
    // half pattern
    for( int col =0; col< row+1; col++){
            cout << "* ";

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

    } // half pyramid
    for (int col= 0; col< row+1; col++) {
            cout << "* ";

    }
    cout << endl;
}
```

① Fancy Pattern →




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

// print only stars



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

        }
        cout << endl;
}
```

now print numbers →

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

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

    }
    cout << endl;
}
```

now printing the upper half of our pattern :-

```
for (int row = 0; row < n; row++){
    for (int col = 0; col < row + 1; col++){
        cout << row+1;
        if (col != row)          → if not last character.
            cout << "*";
    }
    cout << endl;
}
```

// lower half :-

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

⊙ counting m numbers

1, 2, 3, 4, - - - - (m)

0, 1, 2, 3, — - - - (m-1)

⊙

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

for n = 5

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

```
1                          for n = 4
1 2 1
1 2 3 2 1
1 2 3 4 3 2 1   <-- reverse counting
```

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

⑤ Alphabet Palindrome Pyramid →

```
A
A B A                    for n = 4
A B C B A
A B C D C B A
```

any way to map
1 → A
2 → B
3 → C
4 → D

```
int n;
cin>>n;
for (int row = 0 ; row < n; row++){
    ~~int col=0;~~ char ch = 'A';
    for (int col = 0; col < row+1; col++){
        cout << char (ch + col);      // printing in increasing order
    }
    // decreasing order.
    for (int col = row; col > 0 ; col--){
        cout << char (ch + col - 1);
    }
    cout << endl;
}
```