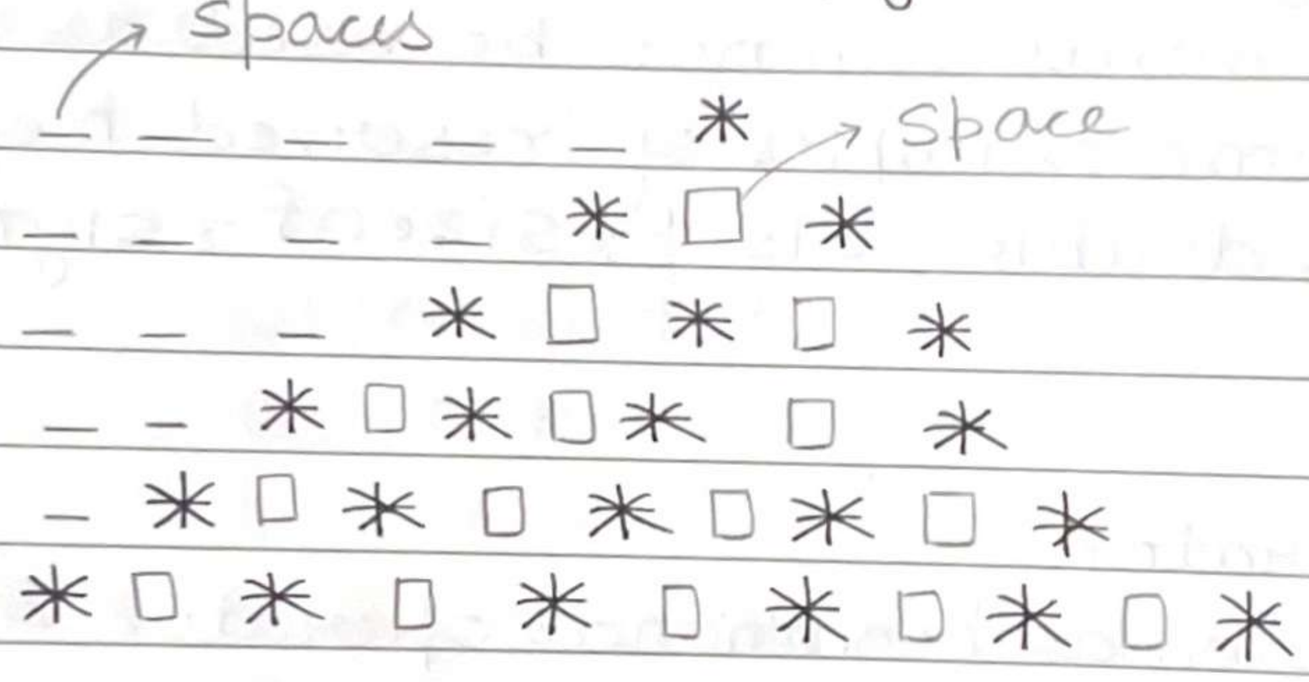
3/02/2023

Patterns continued

Questions of patterns won't be asked in the interview but we are studying patterns because it helps in logic building.

Pattern-1 Full pyramid



i=0	1st row → 5 space + 1 star
i=1	2nd row → 4 space + 2 star
i=2	3rd row → 3 space + 3 star
i=3	4th row → 2 space + 4 star
i=4	5th row → 1 space + 5 star
i=5	6th row → 0 space + 6 star

Outer for loop is for rows and hence it will run from i=0 to i<6.

Inside the inner for loop, first we will be printing the space & then we will be printing the stars.

Formulae for space = n - i - 1
↳ row number

$i = 0 \rightarrow$ row 1 $\Rightarrow$ print 5 spaces

$n - 0 - 1 = n - 1$, here n is 6

$6 - 1 = 5$ spaces

$i = 1 \rightarrow$ row 2 $\Rightarrow$ print 4 spaces

$n - i - 1 \rightarrow 6 - 1 - 1 = 4$ spaces

Hence the formulae of spaces works well.

Formulae for stars $= i + 1$

↳ row number

$i = 0 \rightarrow$ row 1 $\Rightarrow$ print 1 star

$i + 1 = 0 + 1 = 1$ star is printed

$i = 1 \rightarrow$ row 2 $\Rightarrow$ print 2 stars

$i + 1 = 1 + 1 = 2$ stars

Hence the formulae for no. of stars also works fine.

Inner for loop

```
int space = n - i - 1;
for ( ; space >= 0 ; space = space - 1) {
        cout << " ";
}
for (int j = 0 ; j < i + 1 ; j++) {
        cout << "* ";
}
```

↳ This is space for □ in the pyramid.

Pattern - 2 Inverted full pyramid

space also

```
* □ * □ * □ *
_ * □ * □ *
_ _ * □ *
_ _ _ *
```

n = 4

i = 0 row - 1 → 0 space + 4 stars
i = 1 row - 2 → 1 space + 3 stars
i = 2 row - 3 → 2 space + 2 stars
i = 3 row - 4 → 3 space + 1 stars

→ Inner for loop

First we have to print the spaces & then print the stars.

Formulae for space = i

i = 0 → row - 1 has 0 space
i = 1 → row - 2 has 1 space

Hence the formulae for space works fine.

Formulae for stars = n - i

i = 0 → row - 1 ⇒ 4 - 0 = 4 stars
i = 1 → row - 2 ⇒ 4 - 1 = 3 stars

Hence formulae for stars works fine.

Code for inner for loop

```
int space = i;
for ( ; space >= 0 ; space = space - 1) {
        cout << " ";
}
```

```
for (int j=0 ; j < n-i ; j++) {
        cout << "* ";
}
```
↳ space (□)

Formulae practice ↑ n=5 ←

	stars	stars	stars	stars	stars
i=0	0	1	0	5	4
i=1	1	2	0	4	3
i=2	2	3	1	3	2
i=3	3	4	2	2	1
i=4	4	5	3	1	0
formulae	i	i+1	i-1	n-i	n-i-1

In i=0, i-1 = -1 but ↵
-1 stars does not exist
& hence 0 stars will be printed.

Pattern-3 Solid diamond

Full pyramid

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

Inverted Full pyramid

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

In this first print the full pyramid pattern & then the inverted full pyramid pattern.

Pattern - 4 Hollow diamond

```
- - - *
- - * - *
- * - - - *
* - - - - - *
```

} Hollow full pyramid

```
* - - - - - *
- * - - - *
- - * - *
- - - *
```

} Inverted full hollow pyramid

In this first we have to print spaces, then star, then space & then again star.

Step-1 Formulae for spaces $n = 4$

$i = 0 \rightarrow row - 1 \Rightarrow 3$ spaces
$i = 1 \rightarrow row - 2 \Rightarrow 2$ spaces
$i = 2 \rightarrow row - 3 \Rightarrow 1$ space
$i = 3 \rightarrow row - 4 \Rightarrow 0$ space

Formulae for space $= n - i - 1$

Step-2 Formulae for stars & inner spaces

	no. of characters
$i = 0 \rightarrow row - 1$	1 (star only)
$i = 1 \rightarrow row - 2$	3 (2 star + 1 space)
$i = 2 \rightarrow row - 3$	5 (2 star + 3 space)
$i = 3 \rightarrow row - 4$	7 (2 star + 5 space)

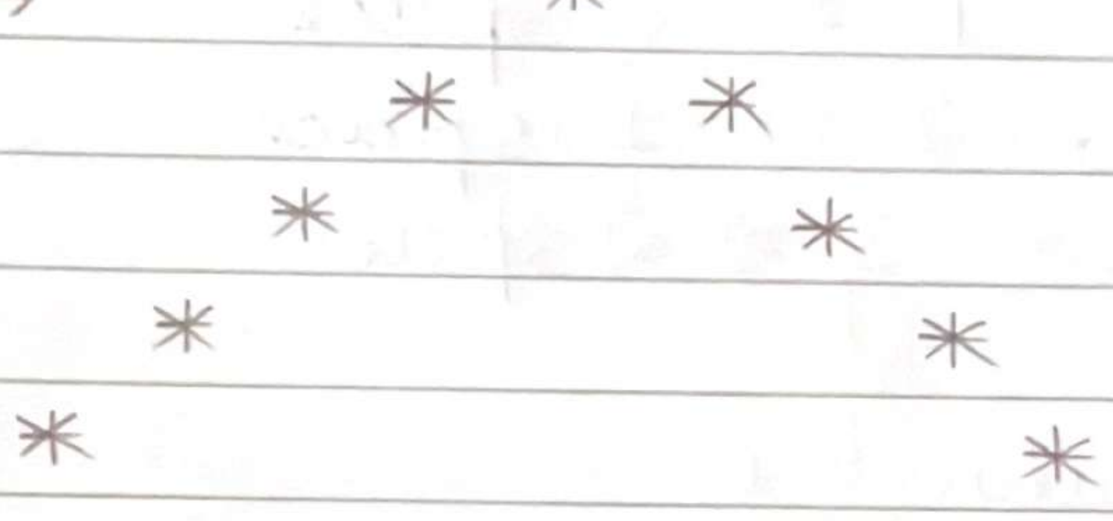
↳ Odd numbers

Formulae for number of characters $= 2 * i + 1$

Now we can also make an observation that the stars will be printed in first & last column of that respective row & in between there will be spaces.

Till now we will be able to print the below pattern.

Pattern →



Code for inner for loops

```
int space = n-i-1 ;
for ( ; space >=0 ; space--) {
        cout << " ";
}
for (int j=0 ; j < 2*i +1 ; j++) {
        //Print star if character is first or last
        if (j==0 || j==2*i) {
                cout << "*";
        }
        else {
                cout << " ";
        }
}
```

By this code we will be able to print the above pattern. (Pattern →)

Now we will be doing the next part i.e inverted hollow full pyramid

Step-1 Formulae for spaces

```
*_ _ _ _ _ _ *
_ * _ _ _ _ * _
_ _ * _ _ _ * _ _
_ _ _ * _ * _ _ _
_ _ _ _ * _ _ _ _
```

i = 0 → row - 1 ⇒ 0 space
i = 1 → row - 2 ⇒ 1 space
i = 2 → row - 3 ⇒ 2 space
:

Space formulae = i

Step-2 Formulae for stars & inner spaces

	No. of characters	
i = 0 → row - 1	9	(2 stars + 7 space)
i = 1 → row - 2	7	(2 stars + 5 space)
i = 2 → row - 3	5	(2 stars + 3 space)

↳ Odd number

Formulae for characters = $2*n - (2*i + 1)$

$= 2*(n-i) - 1$

We have to print stars only for first & last column of that respective row.

Code for inner for loop

```
int space = i;
for ( ; space >= 0 ; space--) {
        cout << " ";
    }
    for (int j=0; j < 2*n - 2*i - 1 ; j++) {
```

```
                    ↳ 1st column          ↳ Last column
if (j == 0 || j == 2*n - 2*i - 2) {
        cout << "*";
    }
else {
        cout << " ";
    }
}
```

Note → The 2nd for loop is running from $j = 0$ to $j < 2*n - 2*i - 1$, this also means $j = 0$ to $j = 2*n - 2*i - 2$ & hence this is the last column in that respective row.

Pattern - 5 Flipped solid diamond

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

Upper part

Lower part

Upper part

Pattern - 1

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

Pattern - 2

```
*     *
* _ *
* _ _ *
```

Also in between the above 2 patterns there will be spaces.

Pattern - 1 and 2 $\boxed{n = 3}$

* * *	$i = 0$	3 stars
* *	$i = 1$	2 stars
*	$i = 2$	1 star

Formulae for stars = $n - i$

Spaces
$i = 0 \rightarrow$ 1 space
$i = 1 \rightarrow$ 3 space
$i = 2 \rightarrow$ 5 space

Formulae for spaces = $2 * i + 1$

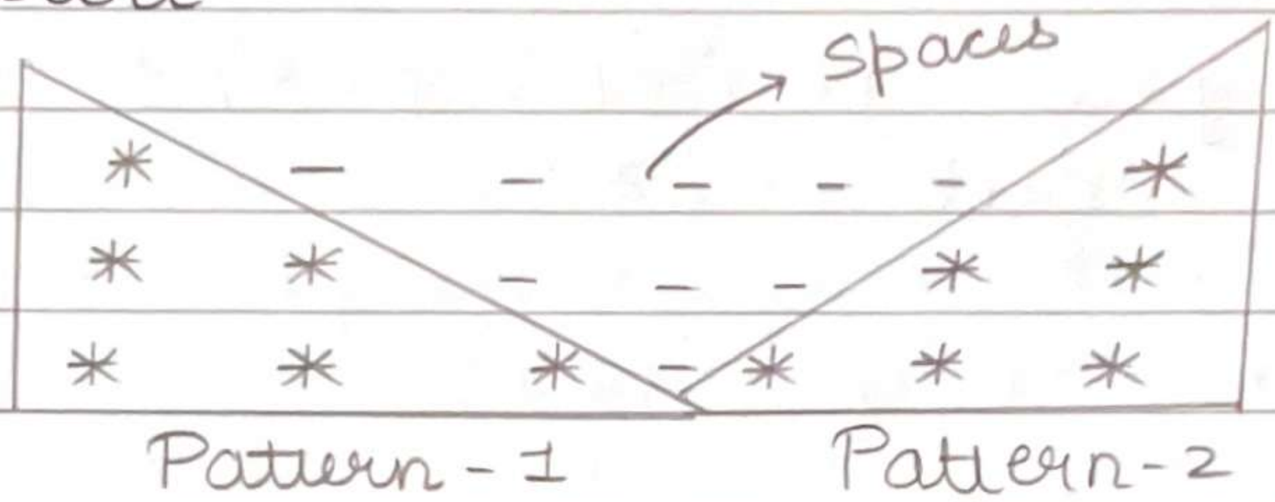
Code for upper part

```
// Pattern - 1
for (int j = 0 ; j < n - i ; j++) {
        cout << "*";
    }
// Spaces
for (int j = 0 ; j < 2* i + 1 ; j++) {
        cout << " ";
    }
// Pattern - 2
for (int j = 0 ; j < n - i ; j++ ) {
        cout << "*";
    }
}
```

By the above code, we will be able to print the following pattern

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

Lower part



In between pattern-1 & pattern-2, we have to print the spaces.

Pattern - 1 and 2

*	$i = 0$	1 star
* *	$i = 1$	2 star
* * *	$i = 2$	3 star

Formulae for star = $i + 1$

Spaces

$i = 0 \rightarrow$ 5 space
$i = 1 \rightarrow$ 3 space
$i = 2 \rightarrow$ 1 space

Formulae for space = $2 * n - 2 * i - 1$

Code for lower part

```
// Pattern - 1
for (int j = 0 ; j < i+1 ; j++) {
```

```
        cout << "* ";
    }
Spaces for (int j=0 ; j< 2*n - 2*i -1 ; j++){
            cout << " ";
    }
Pattern-2 for (int j=0 ; j< i+1 ; j++) {
            cout << "*";
    }
}
```

Pattern - 6 Fancy pattern #2

```
1                                   } 1st half
2 * 2
3 * 3 * 3
4 * 4 * 4 * 4
4 * 4 * 4 * 4                       } 2nd half
3 * 3 * 3
2 * 2
1
```

This is similar to the pattern

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

The logic will be that we have to print numbers & then stars.

1st half

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

i = 0 , 1 is getting printed
i = 1 , 2 is getting printed
i = 3 , 3 is getting printed
i = 3 , 4 is getting printed

Formulae is that for ith row i+1 is getting printed with a star after it but star is only printed when the column is not the last one for the respective row.

Code

```
for (int j = 0 ; j < i + 1 ; j + + ) {
        cout << i + 1 ;
        if (j ! = i) { → Last col condition
            cout << " * " ;
    }
}
```

2nd half

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

i = 0 , 4 is getting printed
i = 1 , 3 is getting printed
i = 2 , 2 is getting printed
i = 3 , 1 is getting printed

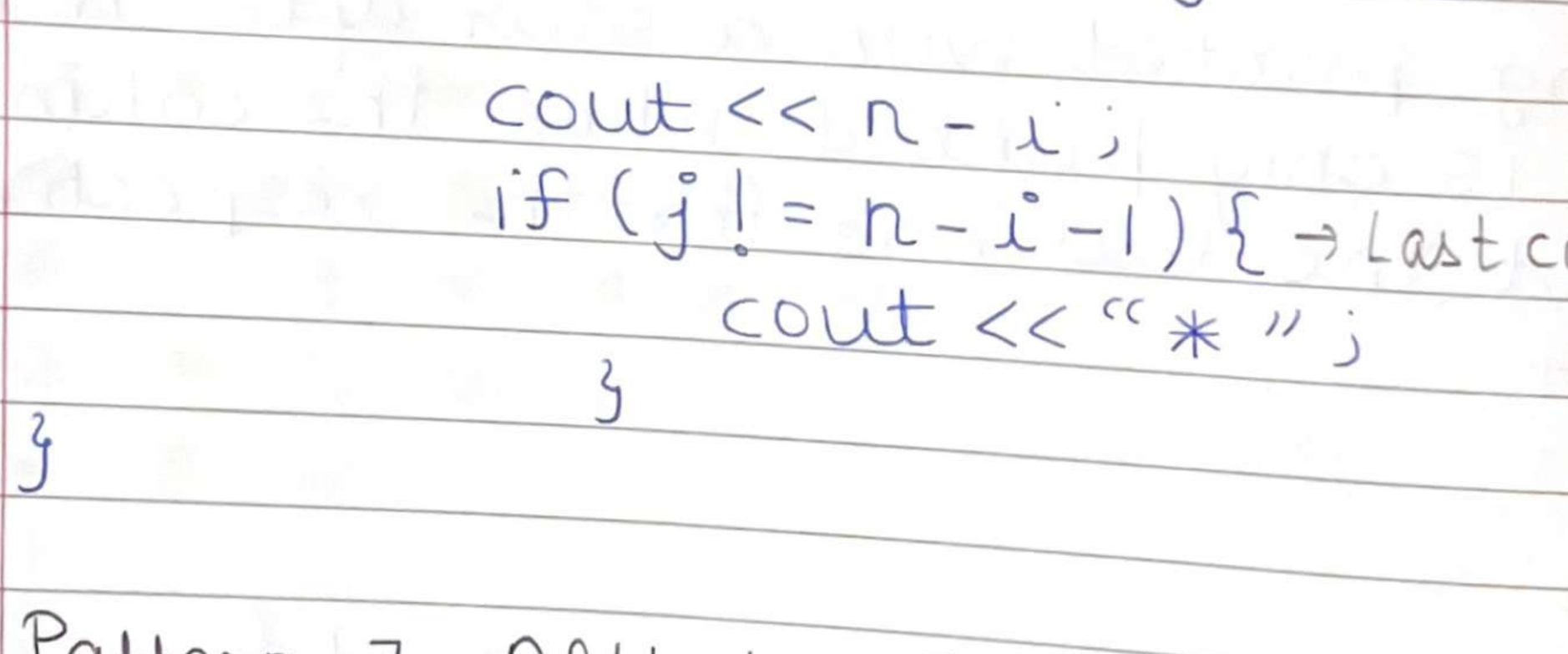
Hence we can say that in i^{th} row, n-i number is getting printed with * & for last column of that respective row, * will not be printed.

Code

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

    cout << n - i ;
    if (j != n-i-1) { → Last col conditn
        cout << "*" ;
    }
}
```

Pattern-7 Alphabet Palindrome Pyramid



First we have to print the following pattern

A

A B

A B C

We always have to start from character A.

Code

```
for (int i=0 ; i<n; i++) {
        char ch = 'A'; //Always start from 'A'
    for(int j=0; j<i+1; j++) {
            cout << ch;
            ch++;
    }
    ch = ch-2;
    if (i!=0){//No reverse count for 1st row
        for(; ch>='A'; ch=ch-1){
            cout <<ch;
        }
    }
    cout << endl;
}
```