Patterns :-

→ To get clarity in loops.
→ logic building.

How to solve pattern ?

(i) Write down the no. of rows.
(ii) Row breakdown
↳ Write down for each row.

Note:
* Outer loop in pattern printing are mainly considered for row printing.
* Inner loop for column printing in every row.

eg:

(1) Solid square :-

```
* * * * → row 0
* * * * → row 1
* * * * → row 2
* * * * → row 3
```

row 0 → 4 star
row 1 → 4 star
row 2 → 4 star
row 3 → 4 star

```
for(int i=0; i<4; i++) {
    for(int j=0; j<4; j++) {
        cout << "* ";
    }
    cout << endl;
}
```

(ii) Solid Rectangle :-

```
* * * * * → row 0
* * * * * → row 1
* * * * * → row 2
* * * * * → row 3
```

row 0 — 5 star
row 1 — 5 star
row 2 — 5 star
row 3 — 5 star

```
for(int i=0; i<4; i++) {
  for(int j=0; j<5; j++) {
    cout << "*";
  }
  cout << endl;
}
```

(iii) Hollow Rectangle :

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

row 0 — 5 star
row 1 → 1 star 3 space 1 star
row 2 → 1 star 3 space 1 star
row 3 → 5 star

```
for(int i=0; i<4; i++) {
  for(int j=0; j<5; j++){
    if(i==0 || i==4) {
      cout << "* ";
    }
    else {
      if(j==0 || j==4) {
        cout << "* ";
      }
      else{
        cout << " ";
      }
    }
  }
  cout << endl;
}
```

(IV) Half Pyramid :

```
*          -> row 0
* *        -> row 1
* * *      -> row 2
* * * *    -> row 3
* * * * *  -> row 4
```

row 0 - 1 star
row 1 - 2 star
row 2 - 3 star
row 3 - 4 star
row 4 - 5 star

row → (row + 1) star

```
for(int i=0; i<5; i++) {
    for(int j=0; j<i+1; j++) {
        cout << "* ";
    }
    cout << endl;
}
```

(V) Inverted half Pyramid :

```
* * * * * -> row 0
* * * *   -> row 1
* * *     -> row 2
* *       -> row 3
*         -> row 4
```

```
int row = 5;
for(int i = 0; i < row; i++)
{
    for(int j=0; j < row - i; j++)
    {
        cout << "* ";
    }
    cout << endl;
}
```

row 0 → 5 star
row 1 → 4 star
row 2 → 3 star
row 3 → 2 star
row 4 → 1 star

row → (Total row - 1) star

(vi) Numeric Half Pyramid :

```
1 → r0
1 2 → r1
1 2 3 → r2
1 2 3 4 → r3
1 2 3 4 5 → r4
```

r0 → 1 char
r1 → 2 char
r2 → 3 char
r3 → 4 char
r4 → 5 char

```
cin >> row = 5;
for(int i=0; i<row; i++)
{
  for(int j=0; j<row+1; j++)
  {
    cout << j+1 << " ";  // → This is to print number in the columns.
  }
  cout << endl;
}
```

* columns are printed with the help of current column + 1.

(vii) Inverted Numeric Half Pyramid :-

```
1 2 3 4 5 → row 0
1 2 3 4 → r1
1 2 3 → r2
1 2 → r3
1 → r4
```

r0 → 5 char
r1 → 4 char
r2 → 3 char
r3 → 2 char
r4 → 1 char

* To print number in the column we need to use (current column + 1)

```
for(int i=0; i<row; i++) {
  for(int j=0; j<row-i; j++) {
    cout << j+1 << " ";
  }
  cout << endl;
}
```

(8) Full Pyramid :-

```
- - - - *        → r0
- - - * *        → r1
- - * * *        → r2
- * * * *        → r3
* * * * *        → r4
```

Total row = 5

r0 → 4 space, 1 star
r1 → 3 space, 2 star
r2 → 2 space, 3 star
r3 → 1 space, 4 star
r4 → 0 space, 5 star

For star, (current row + 1)
For spaces, (Total row - current row - 1)

```
for(int i=0; i<row; i++) {
  for(int j=0; j<row-i-1; j++) { // To print spaces
    cout << " ";
  }
```

```
for(int j=0; j< i+1; j++) { // To print stars
    cout << "* ";
}
cout << endl;
}
```

(9) Inverted Full Pyramid :-

```
* * * * * → r0
- * * * * → r1
- -* * * → r2
- - - * * → r3
- - - -* → r4
```

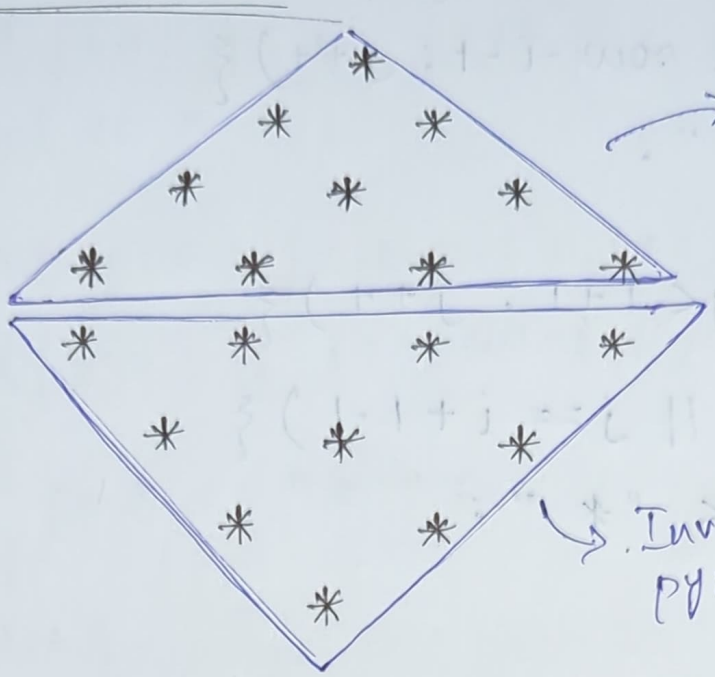
Total row = 5

r0 - 0 space, 5 star
r1 - 1 space, 4 star
r2 - 2 space, 3 star
r3 - 3 space, 2 star
r4 - 4 space, 1 star

For space = current row = no. of space
for star = Total row - current row

```
for(int i=0; i< row; i++) {
    for(int j=0; j< i; j++) {
        cout << " ";
    }
    for(int j=0 j< row-i; j++){
        cout << "*";
    }
    cout << endl;
}
```

(10) Diamond Pattern :-



→ full pyramid.

→ Inverted full pyramid.

→ This pattern is the combination of both full pyramid and inverted full pyramid.

(11) Hollow Pyramid :-

```
- - - * → r0
- - * - * → r1
- * - - - * → r2
* - - - - - * → r3
```

Total row = 4

r0 – 3 space 1 star
r1 – 2 space 1 star 1 space 1 star
r2 – 1 space 3 star 3 space 1 star
r3 – 1 star 5 space 1 star

→ By this we are not finding any pattern in it.

* → So the pattern we can find that only first and last column star otherwise space.

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

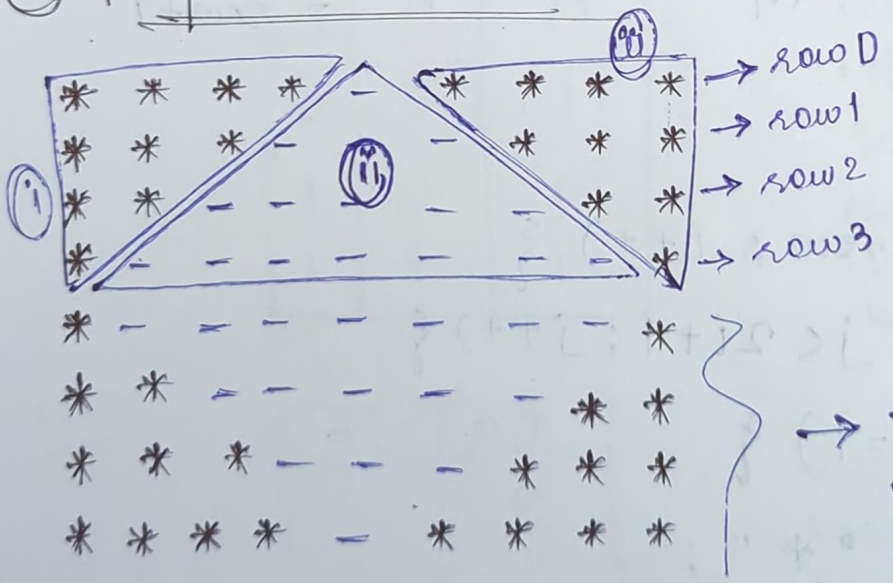
(12) Hollow Inverted Pyramid :-

```
* - - - - * * → r0
- * - - * * → r1
- - * - * → r2
- - - * → r3
```

r0 - 0 space
r1 - 1 space
r2 - 2 space
r3 - 3 space

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

(13) Flip Solid Diamond :-



Total row = 4

→ This is the mirror image of above pattern.

row 0 - 4 star
row 1 - 3 star
row 2 - 2 star
row 3 - 1 star } - for (i) & (iii)

row 0 – 1 space
row 1 – 3 space
row 2 – 5 space
row 3 – 7 space
} – for (ii)

(14) Fancy pattern

1 → r0

2 * 2 → r1

3 * 3 * 3 → r2

4 * 4 * 4 * 4 → r3

col0 col1 col2 col3 col4 col5 col6

r0 → 1 char
r1 → 3 char
r2 → 5 char
r3 → 7 char

odd col" → col1, col3, col5 = "*"

even col" → col0, col2, col4, col6 = row+1

formula 2r+1

```
for(int i=0; i<row; i++) {
    for(int j=0; j<2i+1; j++) {
        if(j%2==1) {
            cout << "*";
        }
        else {
            cout << i+1;
        }
    }
}
```