Welcome,

PROGRAMMERS

Let’s see **With Space pattern** in detail with an example…

01

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

Break down into row & column

		Columns				
		1	2	3	4	5
Rows	1					1
	2				1	2
	3			1	2	3
	4		1	2	3	4
	5	1	2	3	4	5

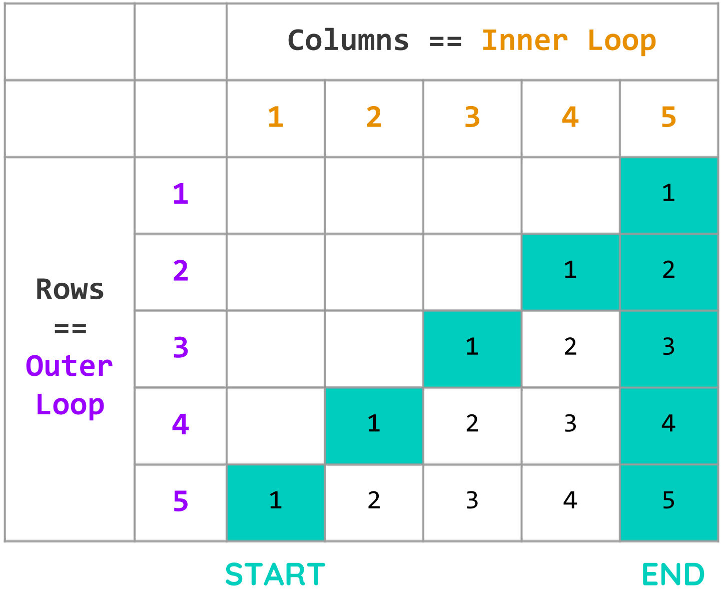
Analysis the given **pattern** & **Make a Code** ○ ○ ●

1. Initialization of inner loop

a. Highlight the START & END pillars Initialize

b. inner loop control variable with a value from a pillar which have same value

```
for ( ) {
        for ( j=1; ; ) {
                printf("%d ", j);
        }
        printf("\n");
}
```



		Columns == Inner Loop				
		1	2	3	4	5
Rows == Outer Loop	1					1
	2				1	2
	3			1	2	3
	4		1	2	3	4
	5	1	2	3	4	5

START

END

Analysis the given pattern & Make a Code

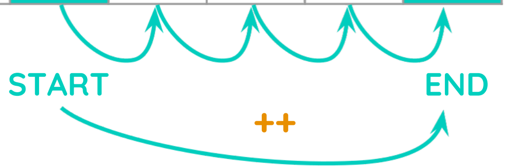
2. Decide condition & increment/decrement of inner loop

a. Mark from START to END and see whether its increasing or decreasing

b. If one is value then another is always variable

```
for ( ) {
        for ( j=1; j<=i ; j++) {
                printf(“%d ”, j);
        }
        printf(“\n”);
}
```

		Columns == Inner Loop				
		1	2	3	4	5
Rows == Outer Loop	1					1
	2				1	2
	3			1	2	3
	4		1	2	3	4
	5	1	2	3	4	5



Analysis the given **pattern** & **Make a Code**

3. Fill out outer loop

a. Outer loop always iterates for no. of rows, so we have two possibilities:

   i. From **1 to 5**

   ii. From **5 to 1**

b. Put any of the one value, and iterate a whole loop at least one time for finalize the value

```
for ( i=1; i<=5; i++) {
    for ( j=1; j<=i ; j++) {
        printf("%d ", j);
    }
    printf("\n");
}
```

		Columns == Inner Loop				
		1	2	3	4	5
Rows == Outer Loop	1					1
	2				1	2
	3			1	2	3
	4		1	2	3	4
	5	1	2	3	4	5
		START				END

OUTPUT

```
for ( i=1; i<=5; i++)
{
    for ( j=1; j<=i ; j++)
    {
        printf(“%d ”, j);
    }
    printf(“\n”);
}
```

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

Now, Let’s see how to **implement an inner loop** for printing **spaces**…

Analysis the given pattern & Make a Code

4. **Implement inner loop for space**
   a. Imagine a range from 1 to total no. of rows, here **1 to 5**
   b. Now just **initialize** a loop control variable with either **1** or **5** and make a single iteration to finalize

```
for ( i=1; i<=5; i++) {
        for ( k=5; k>i ; k--)
                printf(“ ”);
        for ( j=1; j<=i ; j++)
                printf(“%d ”, j);
        printf(“\n”);
}
```

		Columns == Inner Loop				
		1	2	3	4	5
Rows == Outer Loop	1					1
	2				1	2
	3			1	2	3
	4		1	2	3	4
	5	1	2	3	4	5
		START				END

OUTPUT

```
for ( i=1; i<=5; i++)
{
     for ( k=5; k>i ; k--)
          printf(“ ”);
     for ( j=1; j<=i ; j++)
          printf(“%d ”, j);
     printf(“\n”);
}
```

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

Let’s see **Custom pattern** in detail with an example…

01

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

Break down into row & column

		Columns				
		1	2	3	4	5
Rows	1	*	*	*	*	*
	2	*				*
	3	*	*	*	*	*
	4	*				*
	5	*				*

Analysis the given pattern & Make a Code

Here, our typical analysis for row & column is not applicable...

1. Solve it with manually using control structure and looping

```
for ( i=1; i<=5; i++ )
{
        if ( i==1 || i==3 )
                printf(“ * * * * * ”);
        else
                printf(“ *       * ”);
        printf(“\n”);
}
```

		Columns == Inner Loop				
		1	2	3	4	5
Rows == Outer Loop	1	*	*	*	*	*
	2	*				*
	3	*	*	*	*	*
	4	*				*
	5	*				*

C Language

Let’s start now…

▶