100

PATTERN PROGRAMS IN C

SHAHRUKH KHAN

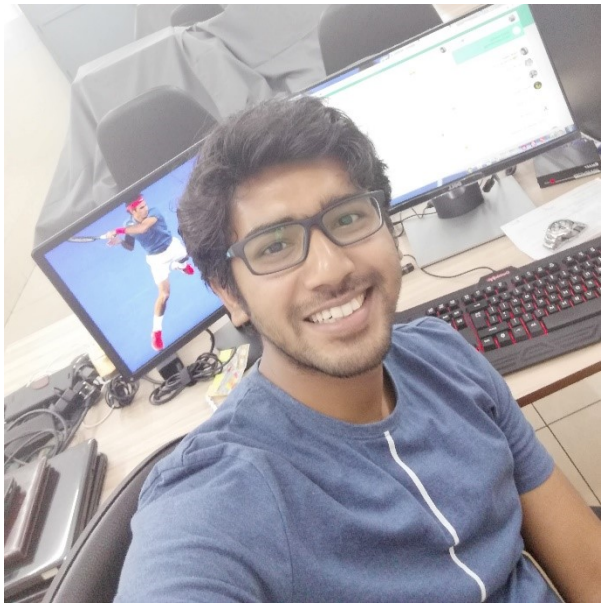
www.thesoftwareguy.in

Table of Contents

About the author

Hello Readers,

My name is Shahrukh Khan. I am an Entrepreneur, PHP Web Developer, and Blogger from Kolkata, India. I have a master's degree in Computer Application (MCA).



Building Software, Websites & Web Apps is my passion. I have around 10+ years of experience in the IT field working mostly on PHP and related environment.

I have worked for clients around the globe providing them customized web solutions. I write blogs at https://www.thesoftwareguy.in and over the years, I have enjoyed a lot interacting with the community, handling queries and solving their problems which indirectly helped me get a great experience.

Although I have written a lot of articles on my blog, but this is my first attempt to write a book. I would love to have your feedback either appreciation or constructive criticism, which will help me to write better next time.

CONNECT WITH ME

EMAIL	YOUTUBE	FACEBOOK	INSTAGRAM
thesoftwareguy7@gmail.com	/c/thesoftwareguy7	/thesoftwareguy7	/thesoftwareguy7

Important Announcement

Free projects & video tutorials will be available on my YouTube Channel soon.

Make sure you subscribe my channel & press the bell icon, to receive all notifications.

Click on the image below

Preface

When I was in college, I was hooked coding these pattern programs from square, pyramid, raindrops, kites, billboards, etc I tried to code them all. Sometimes I succeeded, on other occasions I failed. I have always wondered why these patterns programs are so common in all programming languages. What do you actually get practicing so many different variations of patterns over and over again? But I didn't have any clue back then.

But there was something about those pattern programs that kept us hooked. The time we invest in coding those programs, printing output, cursing ourselves after finding silly mistakes, patting on our back after solving a complex pattern. What actually did we get from it?

The answer is EXPERIENCE. Solving these complex problems, make us really think. Which is what programming is all about.

I am writing this book for students who are learning programming language or want to learn how to code pattern program. If you are in college, school or a newbie to programming, please remember that coding patterns (using loops/control structures) are the most essential part and you must master it.

I can guarantee you if you give your 100% to learn to code these patterns, you are surely going to become a very good problem-solving, which is a skill every great programmer needs to have in his armory.

Although the codes in this book are written in C, readers of other programming languages like Java, Python, etc can also gain knowledge from it, because the concept is same everywhere.

This book comes in PHP language version as well. To get a copy click here or visit this link. https://www.thesoftwareguy.in/100-pattern-programs-in-php-free-ebook

> **The only way to learn a new programming language is by writing programs in it. - Dennis Ritchie**

Resources

Thank you for downloading my book. I am sure you will make the best use of it.

I would like to let you know that this book comes bundled with source codes of 100 programs in separate text files and arranged in their respective folder.

In case, you got this book from any other source, download the bundle through the link given below.
https://www.thesoftwareguy.in/100-pattern-programs-in-c-free-ebook

Section	Programs Count
Warmup Exercises	05
Geometrical Pattern	30
Numbers Pattern	10
Alphabet Pattern	26
Mixed Sets	29
TOTAL	**100**

A. Warmup Exercises

In this section, you will get started with some of the basic programs to help you get acquainted with the working of loops.

BEFORE GETTING STARTED
I have tested codes and it works fine.
For learners/programmers who have coding experience in other programming languages like Java, Python etc. You guys can learn the code logic from here
To print star, alphabet or numbers, almost in every case I have printed the character, but sometimes to have a pretty output, I have used an extra space or %2d modifier.
If you can make any program code logic easier. Please feel free to let me know.

RECOMMENDED WAY OF READING
I will recommend you to install any C compiler software in your machine or use any online C tool so you can test the program while reading. I used Dev C++
JUST READING WONT WORK, YOU NEED TO PRACTICE, PRACTICE AND PRACTICE.

1. Program to print your name five times

PROGRAM LOGIC

- This is a very straight forward, start a loop and iterate it 5 times.

CODE - STYLE #1

```
int row;
for (row = 1; row <= 5; row++) {
   printf("Shahrukh Khan - Style 1\n");
}
```

CODE - STYLE #2

```
int row;
for (row = 100; row > 95; row--) {
   printf("Shahrukh Khan - Style 2\n");
}
```

OUTPUT

```
Shahrukh Khan
Shahrukh Khan
Shahrukh Khan
Shahrukh Khan
Shahrukh Khan
```

IMPORTANT NOTE
Our objective is to loop around 5 times, now it is our wish how we iterate it, i.e. increment or decrement wise. So NEVER think as if there's only a fixed pattern to do things, as directed in books or by your teacher. Use your creativity.

2. Program to print numbers from 1 to 10 but skipping numbers 4 and 6

PROGRAM LOGIC

- I am continuing with the last program
- Start a loop and iterate it 10 times
- Check if the current counter has value 4 or 6, if **yes** then **skip** else print the value.

CODE – STYLE #1

```
int row;
for (row = 1; row <= 10; row++) {
   if(row != 4 && row != 6)
      printf("%d\n",row);
}
```

CODE – STYLE #2

```
int row;
for (row = 1; row <= 10; row++) {
   if(row == 4 || row == 6)
      continue;
   printf("%d\n",row);
}
```

OUTPUT

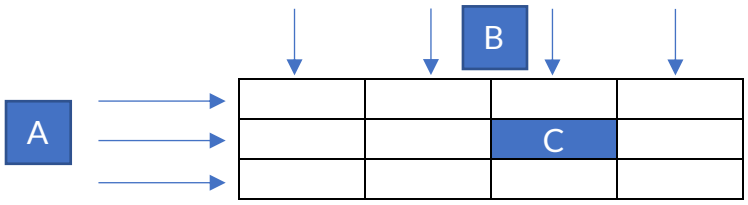
```
1
2
3
5
7
8
9
10
```

3. Program to print 4 X 3 grid with hello world

Before starting, it is important that you understand the concept of GRID/MATRIX. If you failed to understand this, I am Sorry to say, you won't be able keep up with the upcoming sophisticated patterns in the book. **SO PLEASE FOCUS.**

What is a GRID / Matrix? How it works

A matrix is a rectangular array arranged in rows and columns. Given below is a 4X3 (rows x columns) matrix or grid.



A – Represent rows
B – Represent Columns
C – Represent Cell

PROGRAM LOGIC

- Start an **outer loop (represent rows)** that iterate **n** number of times.
- Under the outer loop, start an **inner loop (represent columns)**, that iterate **m** number of times.
- In our case, the outer loop will iterate 4 times and inner loop 3 times,
- But the thing to notice here is that the inner loop gets iterated n x m times i.e. 4 x 3 = 12 times.

CODE

```
int row, column;
for (row = 1; row <= 4; row++) {
   printf("Hello %d ",row);
   for(column = 1; column <= 3; column++) {
      printf("World %d ",column);
   }
   printf("\n");
}
```

OUTPUT

```
Hello1 World1 World2 World3
Hello2 World1 World2 World3
Hello3 World1 World2 World3
Hello4 World1 World2 World3
```

4. Program to print 5 X 3 grid with cell numbers

PROGRAM LOGIC

- Start an **outer loop (columns)** that iterate 3 times.
- Under the outer loop start an **inner loop (column)**, that iterate 3 times.
- Inside the inner loop **print the current row and column value**.

CODE

```
int row, column;
for (row = 1; row <= 3; row++) {
   for(column = 1; column <= 4; column++) {
      printf("Row%dColumn%d ",row, column);
   }
   printf("\n");
}
```

OUTPUT

```
Row1Column1    Row1Column2    Row1Column3
Row2Column1    Row2Column2    Row2Column3
Row3Column1    Row3Column2    Row3Column2
Row4Column1    Row4Column2    Row4Column3
Row5Column1    Row5Column2    Row5Column3
```

5. Program to print table from 1 to 10

OUTPUT ON SCREEN

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

PROGRAM LOGIC

- Take 10 rows and 10 columns.
- Start an **outer loop (columns)** that iterate 10 times.
- Under the outer loop start an **inner loop (column)**, that iterate 10 times.
- Just **multiply** the **current row value** with the **current column value** inside the inner loop.

CODE

```
int row, column;
for (row = 1; row <= 10; row++) {
   for(column = 1; column <= 10; column++) {
      printf("%3d ", (row * column));
   }
   printf("\n");
}
```

B. Geometrical Pattern

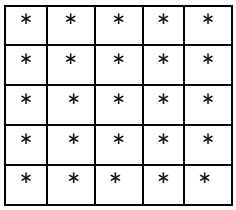
In this section, you will learn to print different geometrical shapes like a pyramid, right-angle triangle, reverse pyramid, inverted pyramid, diamond, rhombus, rectangle, hollow diamonds, etc and their different variations with stars, numbers and alphabet.

Please understand, that every shape is nothing but a pattern inside the grid marked with either space or character. So, always think the shapes in rows and columns rather as a whole.

There are 30 programs in this section, I have arranged them in such an order so that it is easy for you to understand. After about 10-12 programs if you still don't understand, I suggest you go back to program 1 and start again.

IMPORTANT NOTE
Before diving into this section, I will recommend you to practice Section A programs with sincerity. Try to play with them and shuffle variables and numbers. It will help boost your confidence.

1. Program to print square with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
* *	

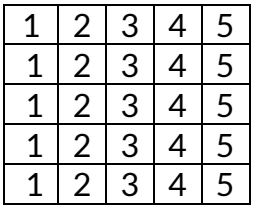
PROGRAM LOGIC

- Take 5 rows and 5 columns.
- Start an **outer loop (row)** that will iterate 5 times.
- Under the outer loop start an **inner loop (column)** that iterate the same number of times
- Inside the inner loop print **star**.

CODE

```
int row, column;
int row_length = 5; int column_length = 5;

for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= column_length; column++) {
      printf("*");
   }
   printf("\n");
}
```

2. Program to print square with number pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5	

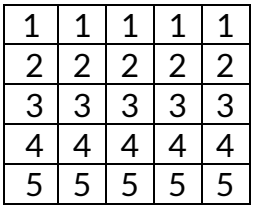
PROGRAM LOGIC

- The logic is same as program B1
- Print **current column** value instead of star.

CODE

```
int row, column;
int row_length = 5; int column_length = 5;

for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= column_length; column++) {
      printf("%d", column);
   }
   printf("\n");
}
```

3. Program to print square with number pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 1 1 1 1 2 2 2 2 2 3 3 3 3 3 4 4 4 4 4 5 5 5 5 5	

PROGRAM LOGIC

- The logic is same as program B1
- Print **current row** value instead of star.

CODE

```
int row, column;
int row_length = 5; int column_length = 5;

for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= column_length; column++) {
      printf("%d", row);
   }
   printf("\n");
}
```

4. Program to print square with alphabet pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
A A A A A B B B B B C C C C C D D D D D E E E E E	65 65 65 65 65 66 66 66 66 66 67 67 67 67 67 68 68 68 68 68 69 69 69 69 69

PROGRAM LOGIC

- The logic is same as program B1
- The ASCII code for capital A is 65 and small a is 97. Use whichever you prefer.
- Set a variable **ascii_code_start** to 65 and **increment by 1** after each row iteration.

CODE

```
int row, column;
int row_length = 5; int column_length = 5;
int ascii_code_start = 65;

for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= column_length; column++) {
      printf("%c", ascii_code_start);
   }
   ascii_code_start++;
   printf("\n");
}
```

5. Program to print square with alphabet pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
A B C D E B C D E F C D E F G D E F G H E F G H I	65 66 67 68 69 66 67 68 69 70 67 68 69 70 71 68 69 70 71 72 69 70 71 72 73

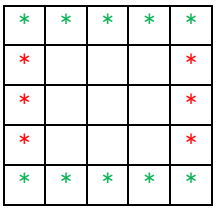
PROGRAM LOGIC

- The logic is same as program B1
- Inside the inner loop **add** ascii_code_start variable to **current row to current column**, use this condition (ascii_code_start + row + column)

CODE

```
int row, column;
int row_length = 5; int column_length = 5;
int ascii_code_start = 65;

for (row = 0; row < row_length; row++) {
   for (column = 0; column < column_length; column++) {
      printf("%c", (ascii_code_start + row + column));
   }
   printf("\n");
}
```

6. Program to print hollow square with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
* * * * * * * * * * * * * * * *	

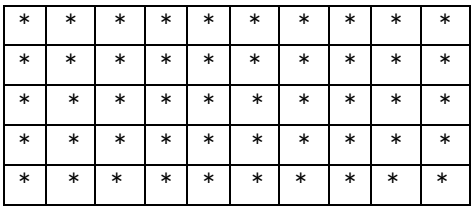
PROGRAM LOGIC

- The logic is same as program B1
- On the **first and last row** (marked in green), print star in **all columns**, use this condition (row == 1 || row == row_length)
- From **second till second-to-last row** (marked in red), print star in the **first and last column** only, use this condition (column == 1 || column == column_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column;
int row_length = 5; int column_length = 5;

for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_length ) {
      printf("*");
    } else if(column == 1 || column == column_length ) {
      printf("*");
    } else {
      printf(" ");
    }

  }
  printf("\n");
}
```

7. Program to print rectangle shape with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
* *	

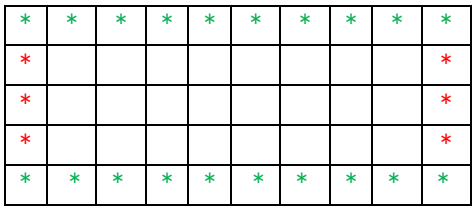
PROGRAM LOGIC

- Take 5 rows and 10 columns.
- Start an **outer loop (row)** that will iterate 5 times.
- Under the outer loop start an **inner loop (column)** that iterate the 10 times.
- Inside the inner loop print **star**.

CODE

```
int row, column;
int row_length = 5; int column_length = 10;

for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= column_length; column++) {
      printf("*");
   }
   printf("\n");
}
```

8. Program to print hollow rectangle with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
* *	

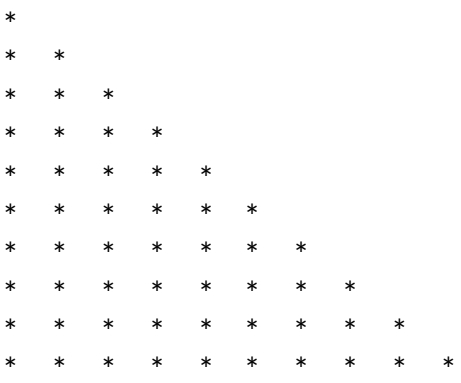
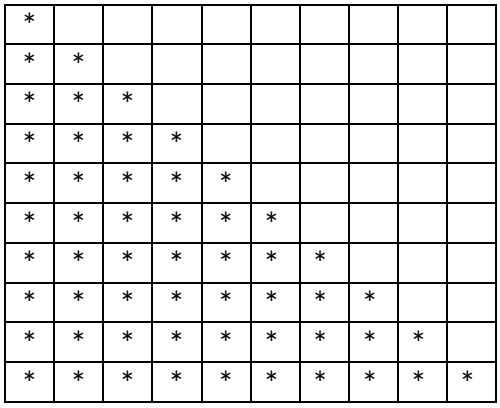
PROGRAM LOGIC

- The logic is same as program B7
- On the **first and last row** (marked in green), print star in **all columns**, use this condition (row == 1 || row == row_length)
- From **second till second-to-last row** (marked in red), print star in the **first and last column** only, use this condition (column == 1 || column == column_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column;
int row_length = 5; int column_length = 10;

for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_length ) {
      printf("*");
    } else if(column == 1 || column == column_length ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

9. Program to print right-angle triangle with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

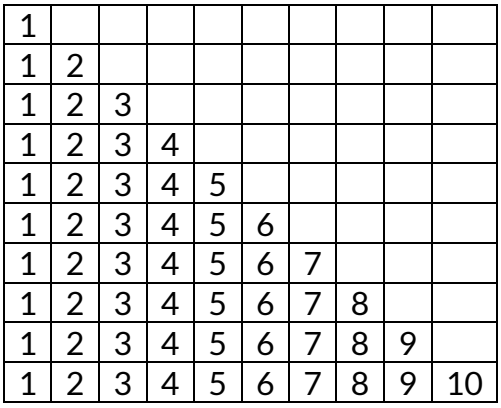
PROGRAM LOGIC

- Take 10 rows and 10 columns.
- Observe, as the **rows increase column value increases**. See Row 1 has 1 star, Row 2 has 2 stars.
- Start an **outer loop (row)** that will iterate 10 times.
- Under the outer loop start an **inner loop (column)** that iterate the same number of times
- Inside the inner loop print **star**.

CODE

```
int row, column;
int row_length = 10;

for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= row; column++) {
      printf("*");
   }
   printf("\n");
}
```

10. Program to print right-angle triangle with number pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 1 2 1 2 3 1 2 3 4 1 2 3 4 5 1 2 3 4 5 6 1 2 3 4 5 6 7 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 9 1 2 3 4 5 6 7 8 9 10	

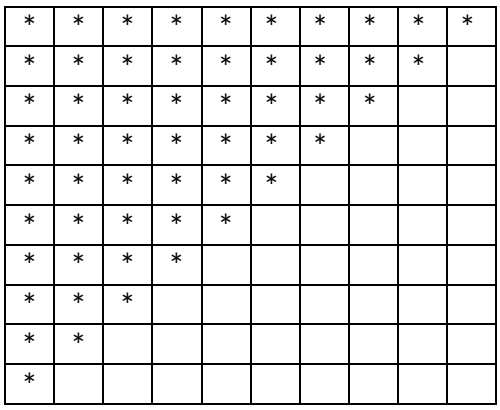
PROGRAM LOGIC

- The logic is same as program B9
- Print the **current column value** instead of star.

CODE

```
int row, column;
int row_length = 10;

for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= row; column++) {
      printf("%d", column);
   }
   printf("\n");
}
```

11. Program to print inverted right-angle triangle with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
* *	

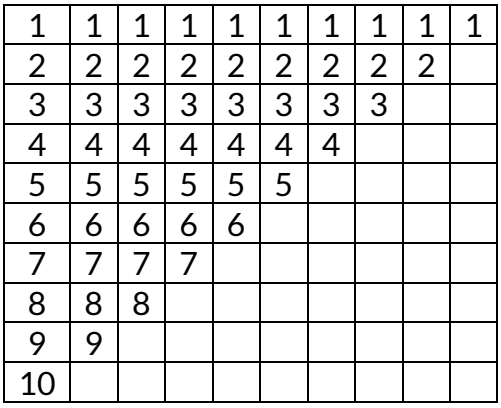
PROGRAM LOGIC

- Take 10 rows and 10 columns.
- Observe, as the **rows increase column value decreases**. See Row 1 has 10 stars, Row 2 has 9 stars.
- Start an **outer loop (row)** that will iterate 10 times.
- Under the outer loop start an **inner loop (column)** that iterate till **row length minus current row value**.
- Inside the inner loop print **star**.

CODE

```
int row, column;
int row_length = 10;

for (row = 0; row < row_length; row++) {
   for (column = 0; column < row_length - row; column++) {
      printf("*");
   }
   printf("\n");
}
```

12. Program to print inverted right-angle triangle with number pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3 4 4 4 4 4 4 4 5 5 5 5 5 5 6 6 6 6 6 7 7 7 7 8 8 8 9 9 10	

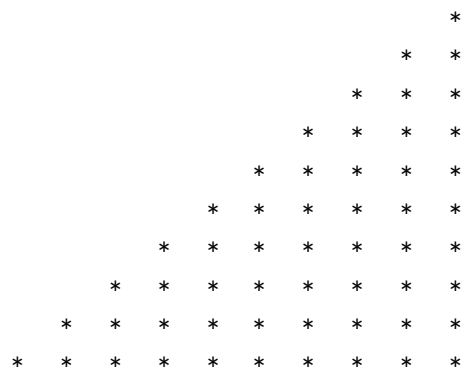
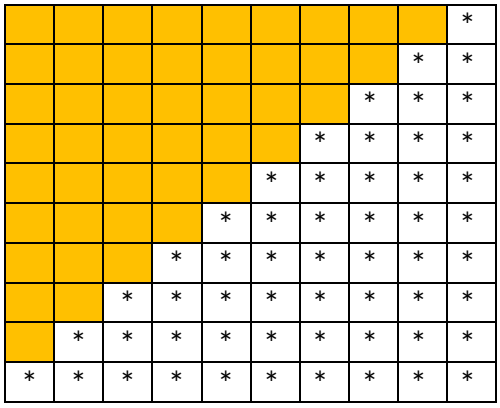
PROGRAM LOGIC

- The logic is same as program B11
- Initialize a counter variable with 1.
- Print the **counter value inside inner loop**.
- After each row iteration, **increment counter by 1.**

CODE

```
int row, column;
int row_length = 10;
int counter = 1;

for (row = 0; row < row_length; row++) {
   for (column = 0; column < row_length - row ; column++) {
      printf("%d", counter);
   }
   counter++;
   printf("\n");
}
```

13. Program to print mirror right-angle triangle with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

PROGRAM LOGIC

- Take 10 rows and 10 columns.
- The blank spaces in the grid (marked in yellow) will come in play here.
- Observe, as **rows increase, space decreases**. Row 1 has 9 spaces; Row 2 has 8 spaces.
- Again, as **rows increases column (star) value increases**. Row 1 has 10 stars; Row 2 has 9 stars.
- Start an **outer loop (row)** that will iterate 10 times.
- Under the outer loop, start an **inner loop (for spaces)** that iterate till **total row length minus current row value**.
- After the inner loop for spaces, start **loop again (for star)** that iterate till **current row value**. Print **star** inside this loop.

CODE

```
int row, column, spaces;
int row_length = 10;

for (row = 1; row <= row_length; row++) {
   for (spaces = 0; spaces < row_length - row ; spaces++) {
      printf(" ");
   }
   for (column = 1; column <= row ; column++) {
      printf("*");
   }
   printf("\n");
}
```

14. Program to print mirror right-angle triangle with number pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID

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

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

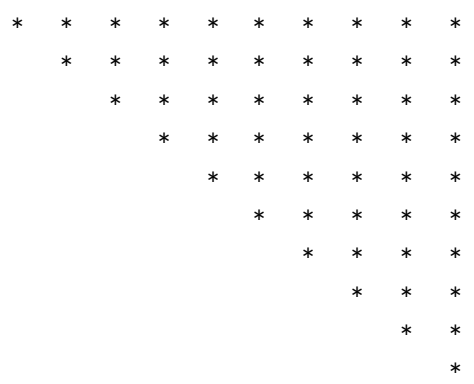
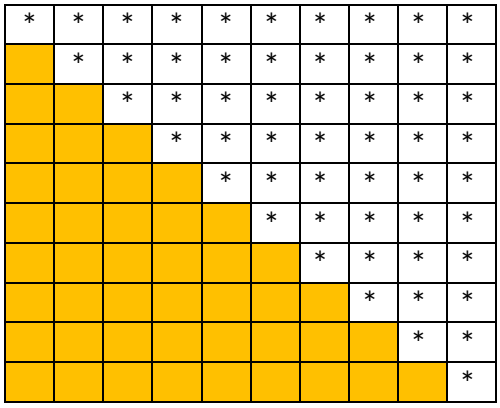
PROGRAM LOGIC

- The logic is same as program B13
- Print the **current column value** instead of star.

CODE

```
int row, column, spaces;
int row_length = 10;

for (row = 1; row <= row_length; row++) {
   for (spaces = 0; spaces < row_length - row ; spaces++) {
      printf(" ");
   }
   for (column = 1; column <= row ; column++) {
      printf("%d", column);
   }
   printf("\n");
}
```

15. Program to print inverted mirror right-angle triangle with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

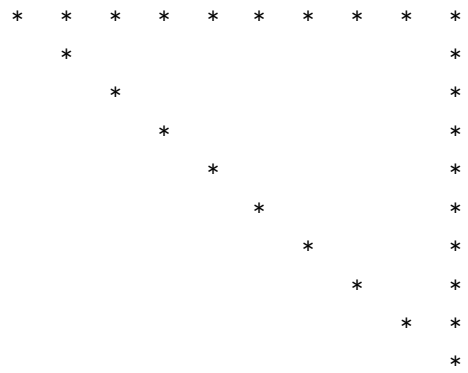
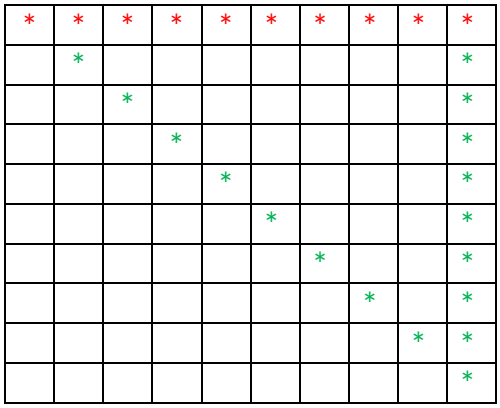
PROGRAM LOGIC

- Take 10 rows and 10 columns.
- The blank spaces in the grid (marked in yellow) will come in play here.
- Observe, as **rows increase, space increases**. Row 1 has 0 space; Row 2 has 1 space.
- Again, as **rows increases column (star) value decreases**. Row 1 has 10 stars; Row 2 has 9 stars.
- Start an **outer loop (row)** that will iterate 10 times.
- Under the outer loop, start an **inner loop (for spaces)** that iterate till **current row value**.
- After the inner loop for spaces, start **loop again (for star)** that iterate till **total row length minus current row value**. Print **star** inside this loop.

CODE

```
int row, column, spaces;  int row_length = 10;
for (row = 0; row < row_length; row++) {
   for (spaces = 0; spaces <  row ; spaces++) {
      printf(" ");
   }
   for (column = 0; column < row_length - row ; column++) {
      printf("*");
   }
   printf("\n");
}
```

16. Program to print inverted hollow mirror right-angle triangle with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

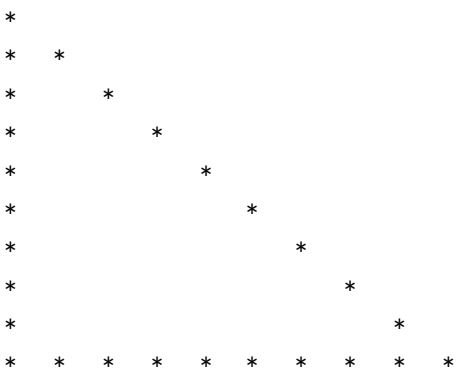
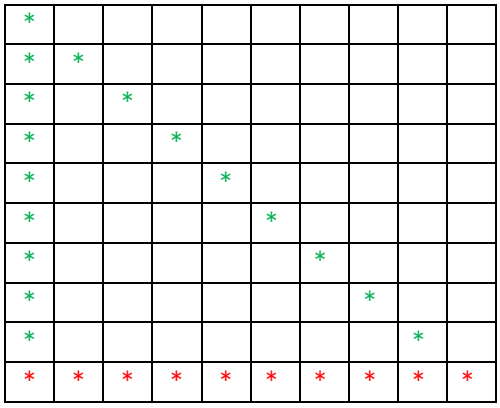
PROGRAM LOGIC

- The logic is same as program B15
- On the **first and last row** (marked in red), print star in **all columns**, use this condition (row ==0)
- From **second till last row** (marked in green), print star in the **current column that matches current row and last column**, use this condition (column == 0 || column == (row_length - row)-1)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column, spaces;  int row_length = 10;
for (row = 0; row < row_length; row++) {
  for (spaces = 0; spaces <  row ; spaces++) {
    printf(" ");
  }
  for (column = 0; column < row_length - row ; column++) {
    if(row == 0 ) {
      printf("*");
    } else if(column == 0 || column == (row_length - row)-1 ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

17. Program to print hollow right-angle triangle with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

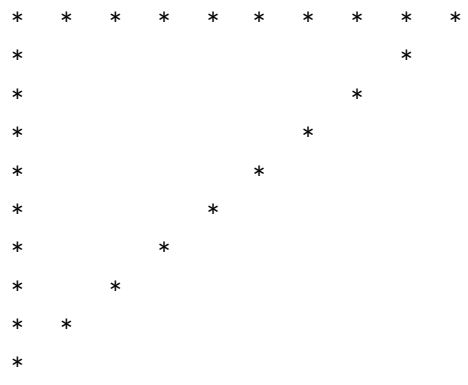
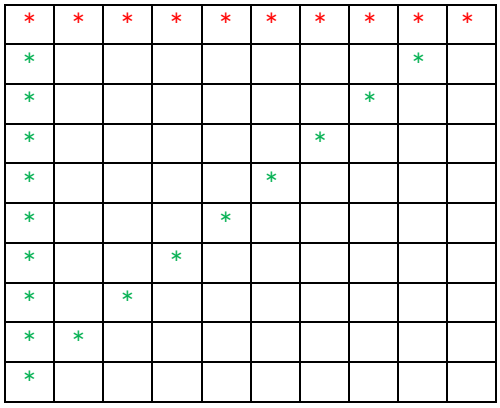
PROGRAM LOGIC

- The logic is same as program B9
- From **first till second-to-last row** (marked in green), print star in the **first column and current column that matches current row**, use this condition (column == 1 || column == row)
- On the **last row** (marked in red), print star in **all columns**, use this condition (row ==row_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column;  int row_length = 10;
for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= row; column++) {
      if(column == 1 || column == row ) {
         printf("*");
      } else if(row == row_length ) {
         printf("*");
      } else {
         printf(" ");
      }
   }
   printf("\n");
}
```

18. Program to print hollow inverted right-angle triangle with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

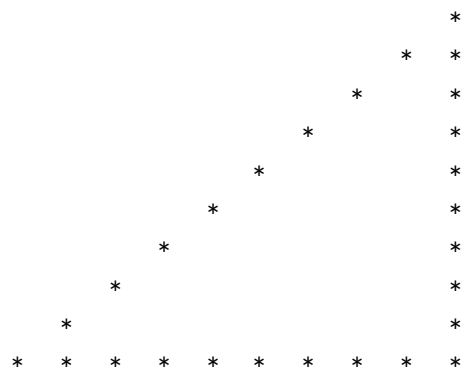
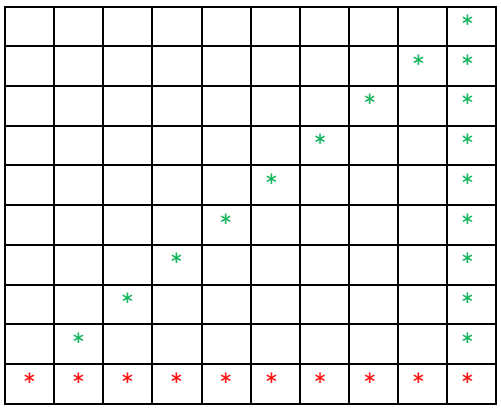
PROGRAM LOGIC

- The logic is same as program B11
- On the **first row** (marked in red), print star in **all columns**, use this condition (row ==0)
- From **second till last row** (marked in green), print star in the **first column and current column that matches current row**, use this condition (column == 0 || column == (row_length - row)-1)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 10;

for (row = 0; row < row_length; row++) {
  for (column = 0; column < row_length - row; column++) {
    if(row == 0 ) {
      printf("*");
    } else if(column == 0 || column == (row_length - row)-1 ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

19. Program to print hollow mirror right angle triangle with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

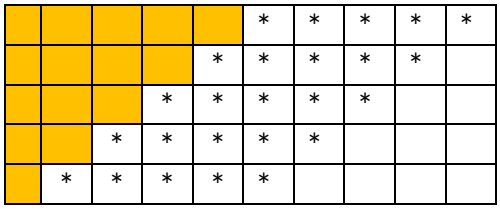
PROGRAM LOGIC

- The logic is same as program B13
- From **first till second-to-last row** (marked in green), print star in the **current column that matches current row and last row**, use this condition (column == 1 || column == row)
- On the **last row** (marked in red), print star in **all columns**, use this condition (row ==row_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column, spaces; int row_length = 10;
for (row = 1; row <= row_length; row++) {
   for (spaces = 0; spaces < row_length - row ; spaces++) {
      printf(" ");
   }
   for (column = 1; column <= row ; column++) {
      if(column == 1 || column == row ) {
         printf("*");
      } else if(row == row_length ) {
         printf("*");
      } else {
         printf(" ");
      }
   }
   printf("\n");
}
```

20. Program to print rhombus with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

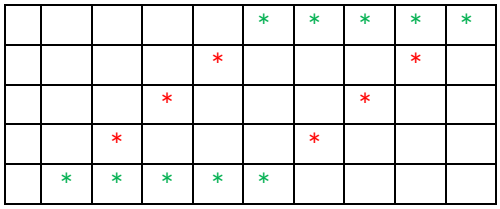
PROGRAM LOGIC

- Take 5 rows and 10 columns.
- The blank spaces in the grid (marked in yellow) will come in play here.
- Observe, as **rows increase, space decreases**. Row 1 has 5 spaces; Row 2 has 4 spaces.
- Start an **outer loop (row)** that will iterate 5 times.
- Under the outer loop, start an **inner loop (for spaces)** that iterate till **total row length minus current row value**.
- After the inner loop for spaces, start **loop again (for star)** that iterate 5 times. Print **star** inside this loop.

CODE

```
int row, column, spaces; int row_length = 10;
for (row = 0; row < row_length; row++) {
   for (spaces = 1; spaces <= row_length - row ; spaces++) {
      printf(" ");
   }
   for (column = 1; column <= row_length ; column++) {
      printf("*");
   }
   printf("\n");
}
```

21. Program to print hollow rhombus with star

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

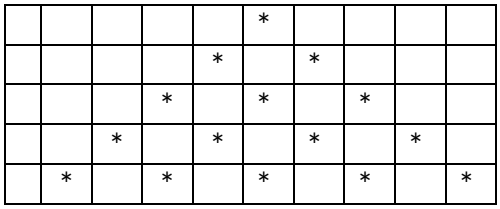
PROGRAM LOGIC

- The logic is same as program B20
- On the **first and last row** (marked in green), print star in **all columns**, use this condition (row == 0 || row == row_length-1)
- From **second till second-to-last row** (marked in red), print star in the **first and last column** only, use this condition (column == 1 || column == row_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column, spaces;
int row_length = 10;

for (row = 0; row < row_length; row++) {
  for (spaces = 1; spaces <= row_length - row ; spaces++) {
    printf(" ");
  }
  for (column = 1; column <= row_length ; column++) {
    if(column == 1 || column == row_length ) {
      printf("*");
    } else if(row == 0 || row == (row_length-1) ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

22. Program to print pyramid with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

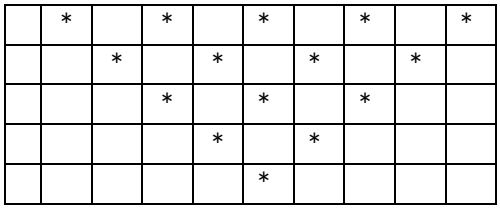
PROGRAM LOGIC

- The logic is same as program B13. The trick here is **playing with spaces**.
- Observe, as **rows increases spaces decreases** (marked in yellow). Row 1 has 10 stars; Row 2 has 9 stars.
- Again, as the **rows increase column (stars) increases**. See Row 1 has 1 star, Row 2 has 2 stars.
- Start an **outer loop (row)** that will iterate 5 times.
- Under the outer loop, start an **inner loop (for spaces)** that iterate till **total row length minus current row value**.
- After the inner loop for spaces, start **loop again (for star)** that iterate till **current row value**. Print **star** inside this loop.

CODE

```
int row, column, spaces;
int row_length = 10;

for (row = 1; row <= row_length; row++) {
   for (spaces = 0; spaces < row_length - row ; spaces++) {
      printf(" ");
   }
   for (column = 1; column <= row ; column++) {
      // one extra space after star
      printf("* ");
   }
   printf("\n");
}
```

23. Program to print inverted pyramid with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

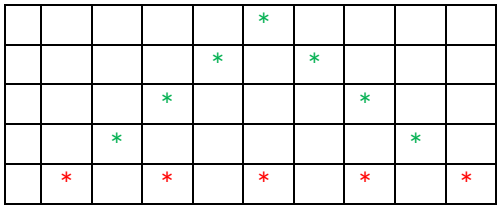
PROGRAM LOGIC

- The logic is same program B22, switch the numbers to make it upside-down.
- Start an **outer loop (row)** that will iterate 5 times.
- Under the outer loop, start an **inner loop (for spaces)** that iterate till **current row value.**
- After the inner loop for spaces, start **loop again (for star)** that iterate till **total row length minus current row value**. Print **star** inside this loop.

CODE

```
int row, column, spaces;
int row_length = 5;

for (row = 0; row < row_length; row++) {
   for (spaces = 0; spaces < row ; spaces++) {
      printf(" ");
   }
   for (column = 0; column < row_length - row ; column++) {
      printf("* ");
   }
   printf("\n");
}
```

24. Program to print hollow pyramid with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

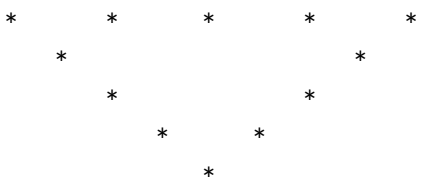
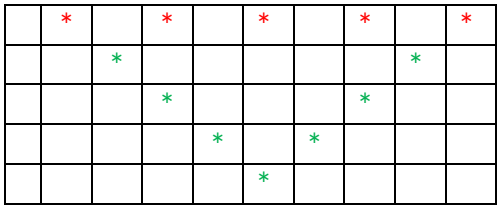
PROGRAM LOGIC

- The logic is same as program B22
- From **first till second-to-last row** (marked in green), print star in the first row and **current column that matches current row**, use this condition (column == 1 || column == row)
- On the **last row** (marked in red), print star in **all columns**, use this condition (row == row_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column, spaces; int row_length = 10;
for (row = 1; row <= row_length; row++) {
   for (spaces = 1; spaces <= row_length - row ; spaces++) {
      printf(" ");
   }
   for (column = 1; column <= row ; column++) {
      if(column == 1 || column == row ) {
         printf("* ");
      } else if(row == row_length ) {
         printf("* ");
      } else {
         printf("  ");
      }
   }
   printf("\n");
}
```

25. Program to print inverted hollow pyramid with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

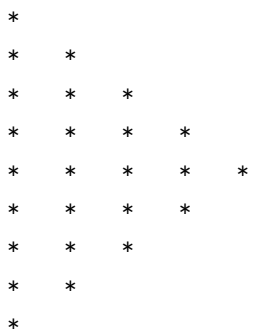
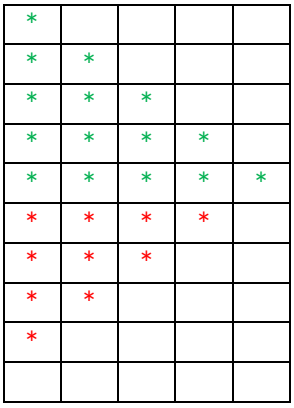
PROGRAM LOGIC

- The logic is same as program B23, switch the numbers to make it upside-down.
- On the **first row** (marked in red), print star in **all columns**, use this condition (row ==0)
- From **second till last row** (marked in green), print star in the first row and **current column that matches current row**, use this condition (column == 0 || column == (row_length - row)-1)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column, spaces;  int row_length = 10;
for (row = 0; row < row_length; row++) {
   for (spaces = 0; spaces < row ; spaces++) {
      printf(" ");
   }
   for (column = 0; column < row_length - row ; column++) {
      if(row == 0 ) {
         printf("* ");
      } else if(column == 0 || column == (row_length - row)-1 ) {
         printf("* ");
      } else {
         printf("  ");
      }
   }
   printf("\n");
}
```

26. Program to print acute triangle with star pattern – style 1

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

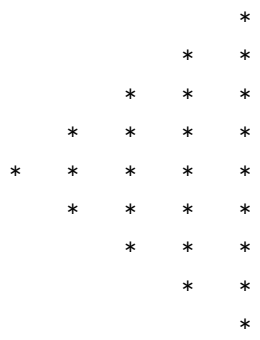
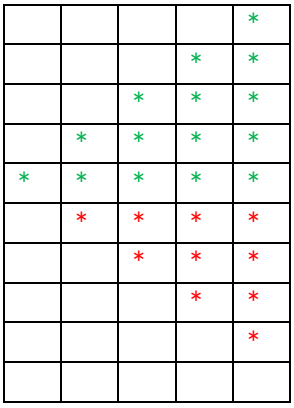
PROGRAM LOGIC

- In the upper section (marked in green) logic is same as the program B9
- In lower section (marked in red) logic is same as the program B11

CODE

```
int row, column;
int row_length = 5;
// upper section
for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= row; column++) {
      printf("*");
   }
   printf("\n");
}
// lower section
for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= row_length - row; column++) {
      printf("*");
   }
   printf("\n");
}
```

27. Program to print acute triangle with star pattern – style 2

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

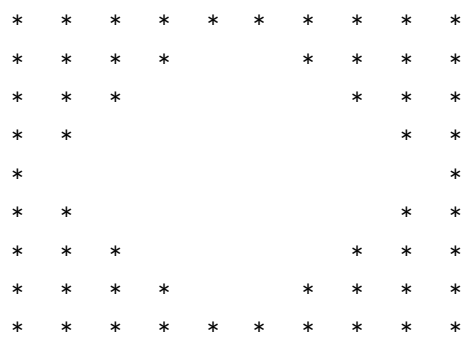
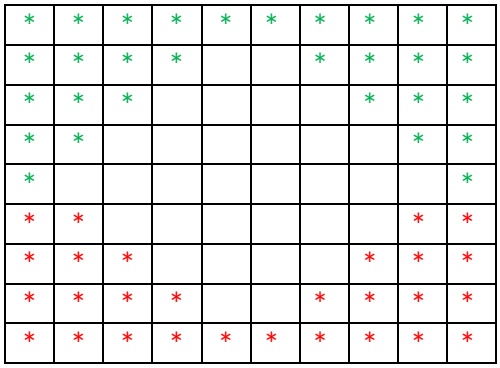
PROGRAM LOGIC

- In the upper section (marked in green) logic is same as the program B13
- In the lower section (marked in red) logic is same as the program B15

CODE

```
int row, column, spaces; int row_length = 5;

// upper section
for (row = 1; row <= row_length; row++) {
   for (spaces = 0; spaces < row_length - row ; spaces++) {
      printf(" ");
   }
   for (column = 1; column <= row ; column++) {
      printf("*");
   }
   printf("\n");
}
// lower section
for (row = 1; row <= row_length; row++) {
   for (spaces = 0; spaces < row ; spaces++) {
      printf(" ");
   }
   for (column = 1; column <= row_length - row  ; column++) {
      printf("*");
   }
   printf("\n");
}
```

28. Program to print hollow diamond with star

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

PROGRAM LOGIC

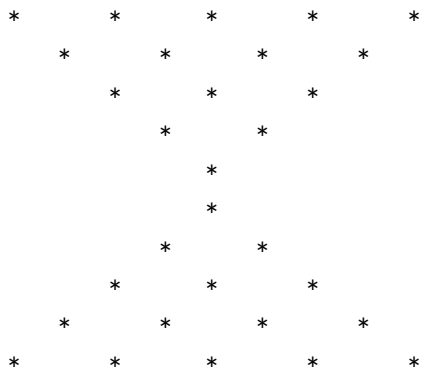
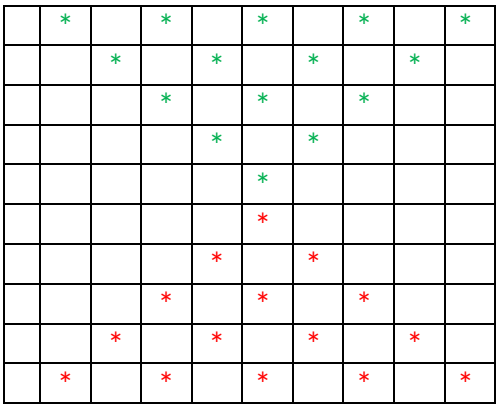
- This program is a combination of multiple sub program covered so far.
- In the upper section (marked in green) is a combination of program B11 & B15
- In the lower section (marked in red) is a combination of program B9 & B13

CODE

```
int row, column, spaces;
int row_length = 5;

// upper section
for (row = 0; row < row_length; row++) {
   //left section
   for (column = 1; column <= row_length - row; column++) {
      printf("*");
   }
   for (spaces = 0; spaces < row ; spaces++) {
      printf(" ");
   }

   // right section
   for (spaces = 0; spaces < row ; spaces++) {
      printf(" ");
   }
   for (column = row_length-row; column >= 1; column--) {
      printf("*");
   }
   printf("\n");
}
```

```
// lower section
for (row = 2; row <= row_length; row++) {
   //left section
   for (column = 1; column <= row; column++) {
      printf("*");
   }
   for (spaces = 0; spaces < row_length-row ; spaces++) {
      printf(" ");
   }

   // right section
   for (spaces = 1; spaces <= row_length-row ; spaces++) {
      printf(" ");
   }
   for (column = 1; column <= row; column++) {
      printf("*");
   }
   printf("\n");
}
```

29. Program to print hourglass with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

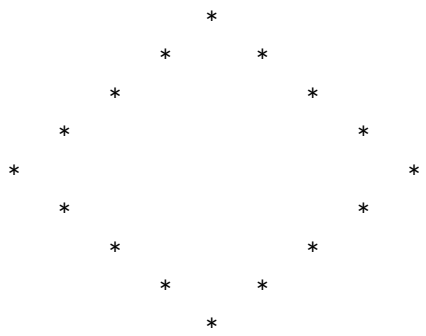
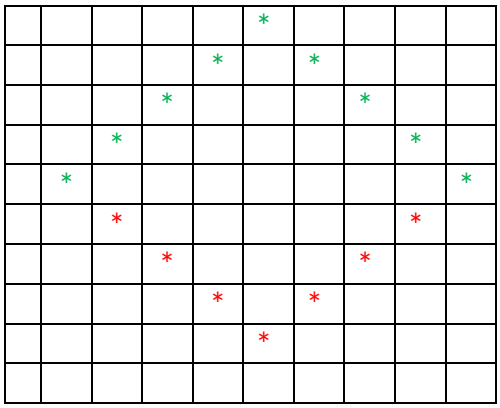
PROGRAM LOGIC

- In the upper section (marked in green) logic is same as the program B23
- In the lower section (marked in red) logic is same as the program B22

CODE

```
int row, column, spaces; int row_length = 5;
// upper section
for (row = 0; row < row_length; row++) {
   for (spaces = 0; spaces < row ; spaces++) {
      printf(" ");
   }
   for (column = 0; column < row_length - row; column++) {
      printf("* ");
   }
   printf("\n");
}

// lower section
for (row = 1; row <= row_length; row++) {
   for (spaces = 1; spaces <= row_length-row ; spaces++) {
      printf(" ");
   }
   for (column = 1; column <= row; column++) {
      printf("* ");
   }
   printf("\n");
}
```

30. Program to print hollow diamond with star

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

PROGRAM LOGIC

- This program is a combination of two programs.
- In the upper section (marked in green) logic is same as the program B24
- In the lower section (marked in red) logic is same as the program B25

CODE

```
int row, column, spaces;
int row_length = 5;

// upper section
for (row = 1; row <= row_length; row++) {
   for (spaces = 1; spaces <= row_length-row ; spaces++) {
      printf(" ");
   }
   for (column = 1; column <= row; column++) {
      if (column == 1 || column == row) {
         printf("* ");
      } else {
         printf("  ");
      }
   }
   printf("\n");
}
```

```
// lower section
for (row = 1; row < row_length; row++) {
   for (spaces = 0; spaces < row ; spaces++) {
      printf(" ");
   }
   for (column = 0; column < row_length-row; column++) {
      if (row == (row_length-1) ) {
         printf("* ");
      } else if (column == 0 || column == (row_length-row)-1 ) {
         printf("* ");
      } else {
         printf("  ");
      }
   }
   printf("\n");
}
```

C. Numbers Pattern

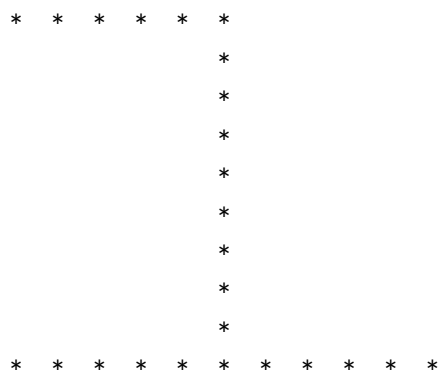
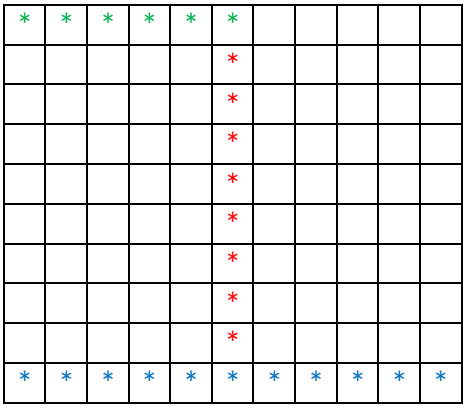
In this section, you will learn to print all the number from 0 to 9 with star pattern.

I am going to print those number with star pattern as closely as possible. If you think you can have a better version of the code, please do notify me.

There are 10 programs in this section. **I recommend not to skip this section**, because this will help you learn the basics of printing alphabet with star pattern.

IMPORTANT NOTE
Before diving into this section. I am assuming you have finished learning Section B. If not please go back to Section B.

1. Program to print number one with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

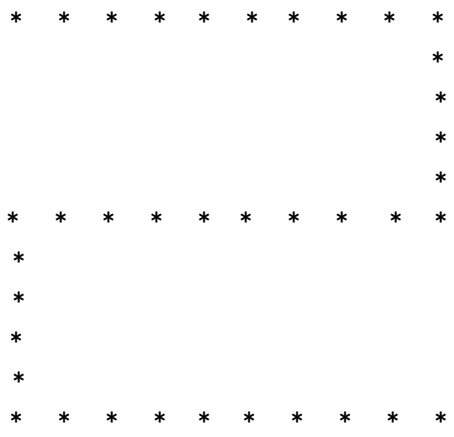
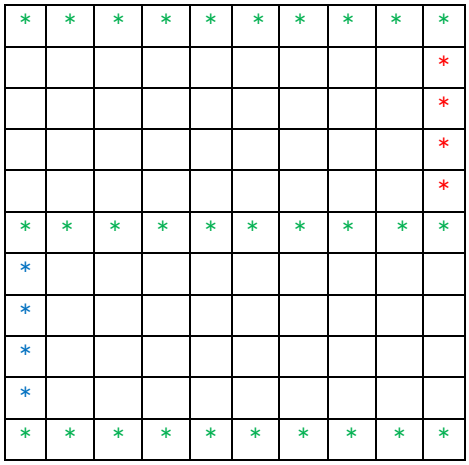
PROGRAM LOGIC

- Take 10 rows and 11 columns, for the 6th column will be the exact middle.
- On the **first row** (marked with green), print star from **first till middle column**, use this condition (row == 1 && column < column_middle)
- From **second till second-to-last row** (marked with red), print star at the middle column, use this condition (column == column_middle)
- Finally, on the **last row** (marked with blue) print star in **all columns**, use this condition (row == row_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 10; int column_length = 11; int column_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 && column < column_middle ) {
      printf("*");
    } else if(column == column_middle ) {
      printf("*");
    } else if(row == row_length ) {
      printf("*");
    } else {
      printf(" ");
    }

  }
  printf("\n");
}
```

2. Program to print number two with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

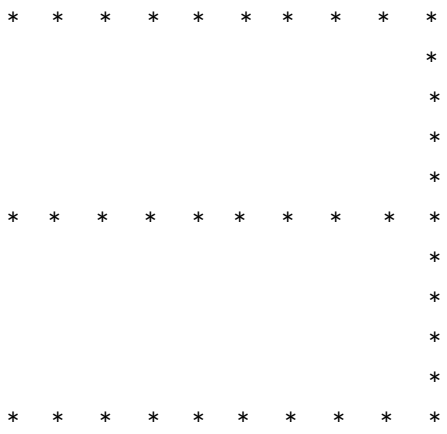
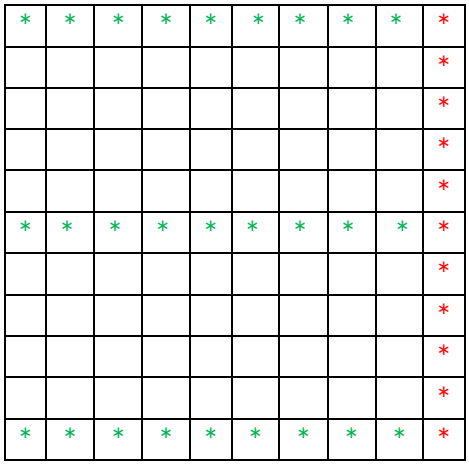
PROGRAM LOGIC

- Take 11 rows and 10 columns, for the 6th row will be the exact middle.
- On the **first, middle and last row** (marked with green), print star in **all columns,** use this condition (row == 1 || row == row_middle || row == row_length)
- From **second till middle row** (marked with red), print star at the **last column,** use this condition (column == column_length && row < row_middle)
- Also, from **middle to second-to-last row** (marked with blue) print star at the **first column**, use this condition (column == 1 && row > row_middle)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 11; int column_length = 10; int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_middle || row == row_length ) {
      printf("*");
    } else if(column == column_length && row <= row_middle ) {
      printf("*");
    } else if(column == 1 && row > row_middle) {
      printf("*");
    } else {
      printf(" ");
    }

  }
  printf("\n");
}
```

3. Program to print number three with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

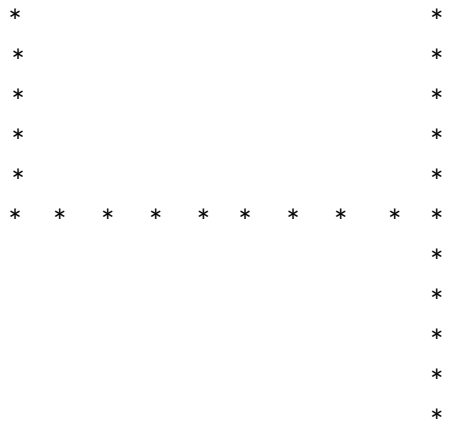
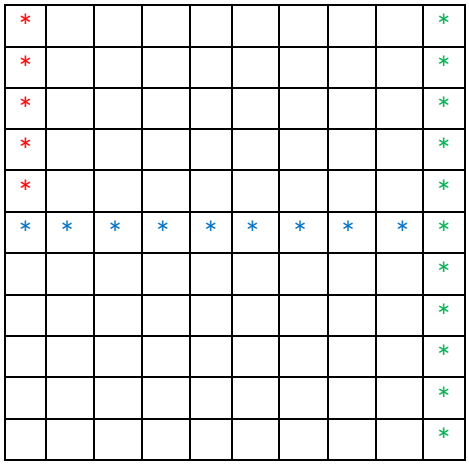
PROGRAM LOGIC

- Take 11 rows and 10 columns, for the 6th row will be the exact middle.
- On the **first, middle and second-to-last row** (marked with green), print star in **all columns,** use this condition (row == 1 || row == row_middle || row == row_length-1)
- From **second till second-to-last row** (marked with red), print star at the **last column,** use this condition (column == column_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 11;
int column_length = 10; int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_middle || row == row_length ) {
      printf("*");
    } else if(column == column_length ) {
      printf("*");
    } else {
      printf(" ");
    }

  }
  printf("\n");
}
```

4. Program to print number four with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

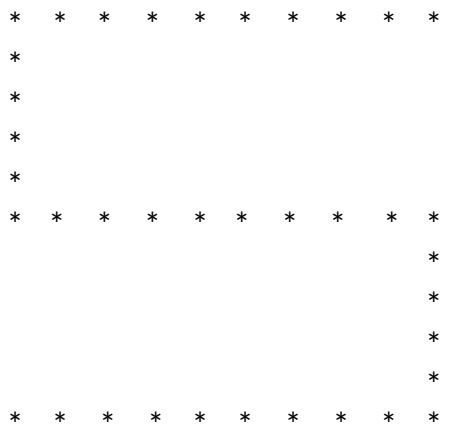
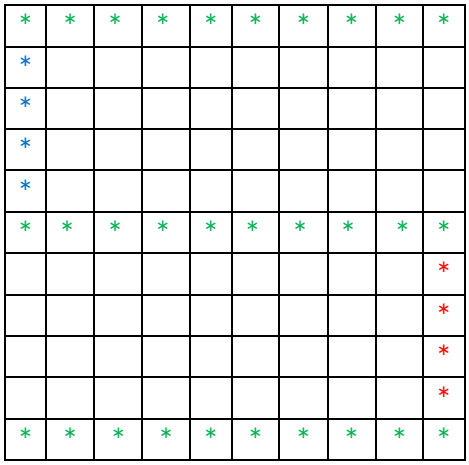
PROGRAM LOGIC

- Take 11 rows and 10 columns, for the 6th row will be the exact middle.
- On the **first column** (marked with red), print star from **first till middle row**, use this condition (column == 1 && row <= row_middle)
- At the **mid row** (marked with blue), print star in **all columns**, use this condition (row == row_middle)
- Finally, on the **last column** (marked with green) print star in **all row**, use this condition (column == column_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column;  int row_length = 11;
int column_length = 10; int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(column == 1 && row < row_middle ) {
      printf("*");
    } else if(row == row_middle ) {
      printf("*");
    } else if(column == column_length ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

5. Program to print number five with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

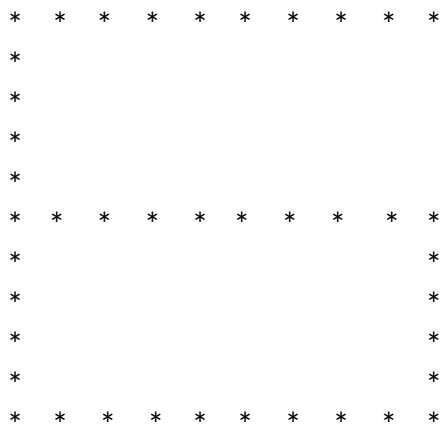
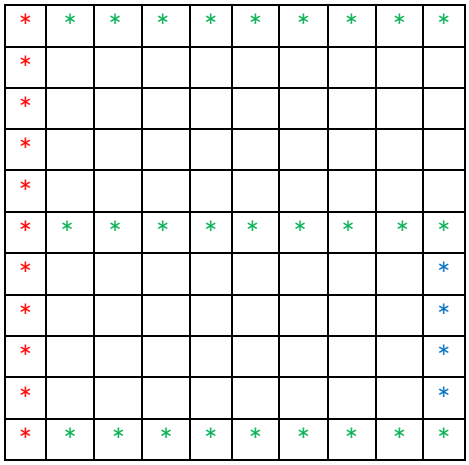
PROGRAM LOGIC

- The logic is same as program C2, just play with column arrangements.
- Take 11 rows and 10 columns, for the 6th row will be the exact middle.
- On the **first, middle and last row** (marked with green), print star in **all columns,** use this condition (row == 1 || row == row_middle || row == row_length)
- From **second till middle row** (marked with blue), print star at the **first column,** use this condition (column == 1 && row < row_middle)
- Also, from **middle to second-to-last row** (marked with red) print star at the **first column**, use this condition (column == column_length && row > row_middle)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 11; int column_length = 10; int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_middle || row == row_length ) {
      printf("*");
    } else if(column == 1 && row <= row_middle ) {
      printf("*");
    } else if(column == column_length && row > row_middle) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

6. Program to print number six with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

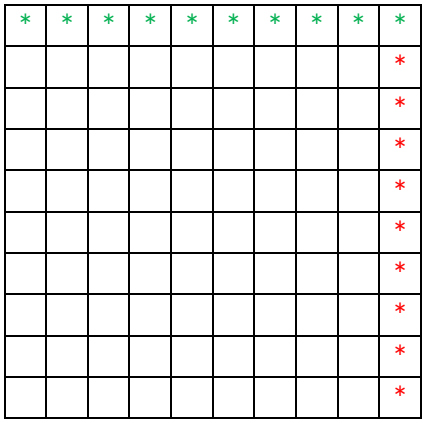
PROGRAM LOGIC

- Take 11 rows and 10 columns, for the 6th row will be the exact middle.
- On the **first, middle and last row** (marked with green), print star in **all columns**, use this condition (row == 1 || row == row_middle || row == row_length)
- For **first column** (marked with red), print star in **each row**, use this condition (column == 1)
- Finally, from **middle till second-to-last row** (marked with blue), print star at the **last column,** use this condition (column == column_length && row > row_middle)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 11;int column_length = 10; int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_middle || row == row_length ) {
      printf("*");
    } else if(column == 1 ) {
      printf("*");
    } else if(column == column_length && row > row_middle) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

7. Program to print number seven with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
* * * * * * * * * * * * * * * * * * *	

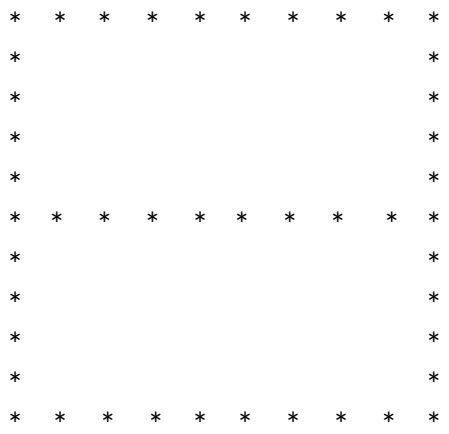
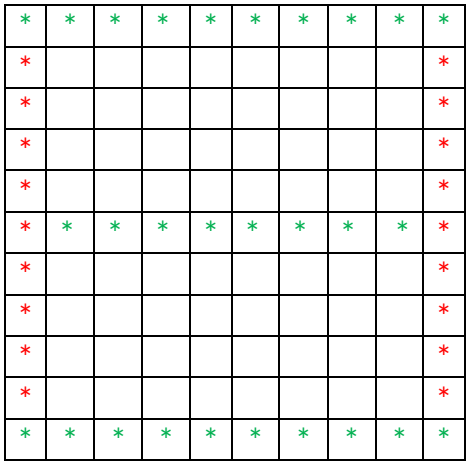
PROGRAM LOGIC

- Take 10 rows and 10 columns.
- On the **first row** (marked with green), print star in **all columns**, use this condition (row == 1)
- For **last column** (marked with red), print star in **each row**, use this condition (column == column_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 10;int column_length = 10;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 ) {
      printf("*");
    } else if(column == column_length) {
      printf("*");
    } else {
      printf(" ");
    }

  }
  printf("\n");
}
```

8. Program to print number eight with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

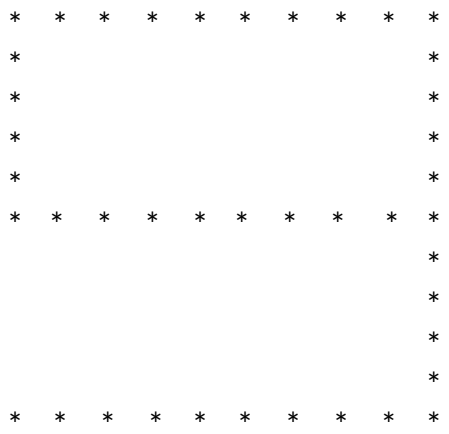
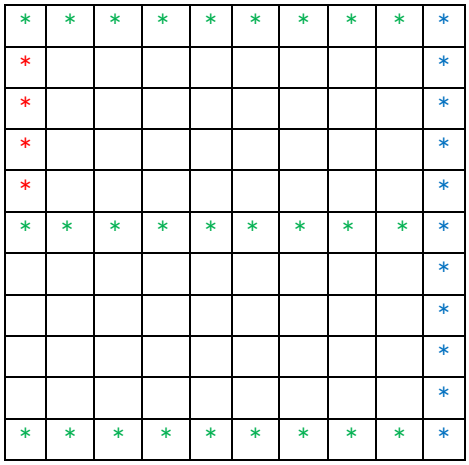
PROGRAM LOGIC

- Take 11 rows and 10 columns, for the 6th row will be the exact middle.
- On the **first, middle and second-to-last row** (marked with green), print star in **all columns,** use this condition (row == 1 || row == row_middle || row == row_length-1)
- From **second till second-to-last row** (marked with red), print star at the **first and last column,** use this condition (column == 1 || column == column_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 11; int column_length = 10;  int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_middle || row == row_length ) {
      printf("*");
    } else if(column == 1 || column == column_length ) {
      printf("*");
    } else {
      printf(" ");
    }

  }
  printf("\n");
}
```

9. Program to print number nine with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

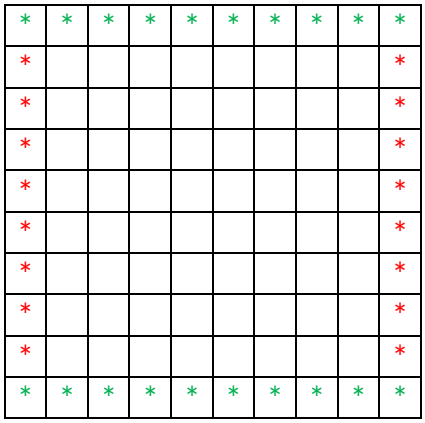
PROGRAM LOGIC

- Take 11 rows and 10 columns, for the 6th row will be the exact middle.
- On the **first, middle and last row** (marked with green), print star in **all columns**, use this condition (row == 1 || row == row_middle || row == row_length)
- From **first till middle column** (marked with red), print star in **first column,** use this condition (column == 1 && row < row_middle)
- Finally, on the **last column** (marked with blue), print star at **each row,** use this condition (column == column_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 11; int column_length = 10; int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_middle || row == row_length ) {
      printf("*");
    } else if(column == 1 && row < row_middle ) {
      printf("*");
    } else if(column == column_length ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

10. Program to print number zero with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
* *	

PROGRAM LOGIC

- This pattern is similar to hollow square, check the logic from the program B6

CODE

```
int row, column; int row_length = 10; int column_length = 10;
for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= column_length; column++) {
      if(row == 1 || row == row_length ) {
         printf("*");
      } else if(column == 1 || column == column_length ) {
         printf("*");
      } else {
         printf(" ");
      }
   }
   printf("\n");
}
```

D. Alphabet Pattern

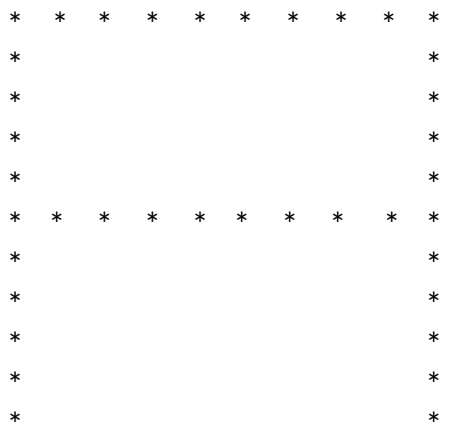
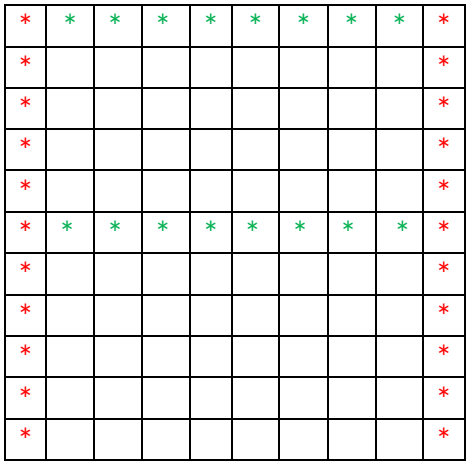
In this section, you will learn to print letters from A to Z

I hope you had a great time solving patterns on numbers and geometrical shape.

There are 26 programs in this section, if you have solved the numbers pattern then I am sure you won't face trouble solving these problems.

IMPORTANT NOTE
Before diving into this section. I am assuming you have finished learning Section B & Section C. If not please go back and read.

1. Program to print letter A with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

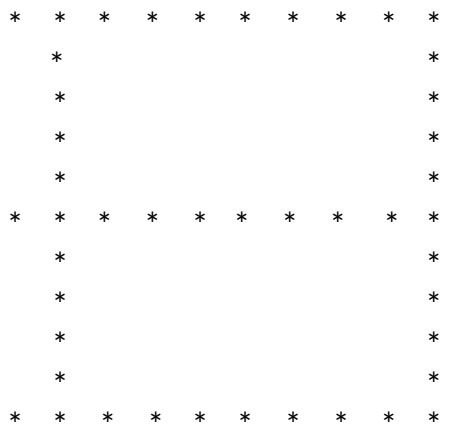
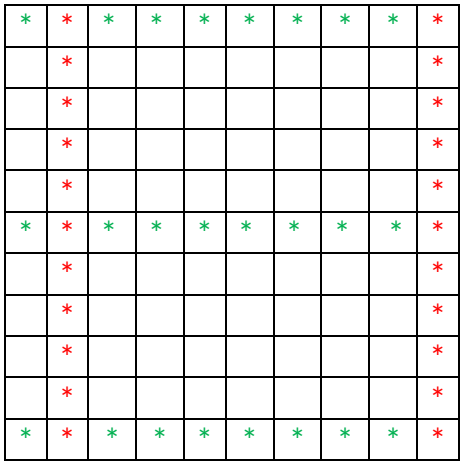
PROGRAM LOGIC

- Take 11 rows and 10 columns, for the 6th row will be the exact middle.
- On the **first and middle row** (marked with green), print star in **all columns,** use this condition (row == 1 || $row == row_middle)
- Print star at the **first and last column of each row** (marked with red)**,** use this condition (column == 1 || column == column_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 11;
int column_length = 10; int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_middle ) {
      printf("*");
    } else if(column == 1 || column == column_length ) {
      printf("*");
    } else {
      printf(" ");
    }

  }
  printf("\n");
}
```

2. Program to print letter B with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

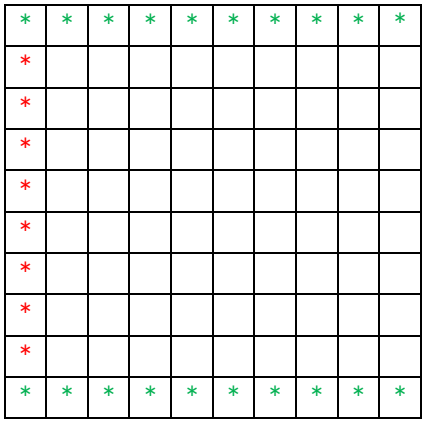
PROGRAM LOGIC

- Take 11 rows and 10 columns, for the 6th row will be the exact middle.
- On the **first, middle and last row** (marked with green), print star in **all columns,** use this condition (row == 1 || row == row_middle || row == row_middle)
- Print star at the **second and last column of each row** (marked with red)**,** use this condition (column == 2 || column == column_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 11;
int column_length = 10; int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_middle || row == row_length ) {
      printf("*");
    } else if(column == 2 || column == column_length ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

3. Program to print letter C with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
* *	

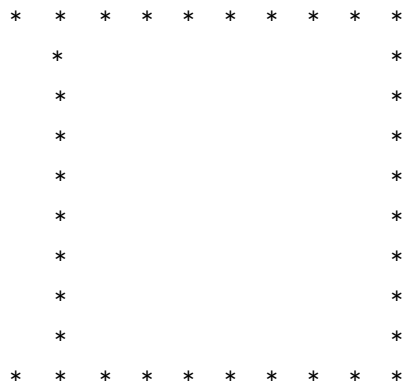
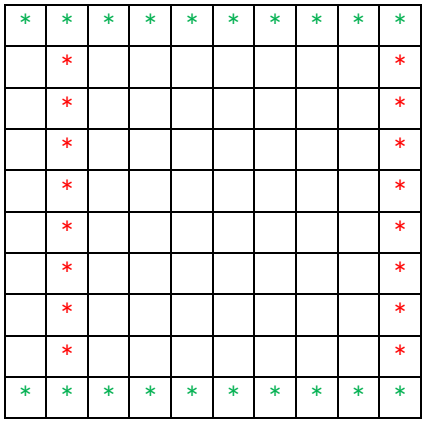
PROGRAM LOGIC

- Take 10 rows and 10 columns.
- On the **first and last row** (marked with green), print star in **all columns,** use this condition (row == 1 || row == row_middle)
- Print star at the **first column of each row** (marked with red), use this condition (column == 1)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 10; int column_length = 10;
for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= column_length; column++) {
      if(row == 1 || row == row_length ) {
         printf("*");
      } else if(column == 1 ) {
         printf("*");
      } else {
         printf(" ");
      }
   }
   printf("\n");
}
```

4. Program to print letter D with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

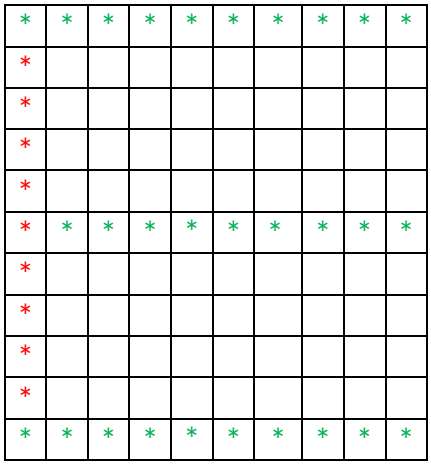
PROGRAM LOGIC

- Take 10 rows and 10 columns.
- On the **first and last row** (marked with green), print star in **all columns,** use this condition (row == 1 || row == row_middle)
- Print star at the **second and last column of each row** (marked with red), use this condition (column == 2 || column == column_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 10; int column_length = 10; int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_length ) {
      printf("*");
    } else if(column == 2 || column == column_length ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

5. Program to print letter E with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
* *	

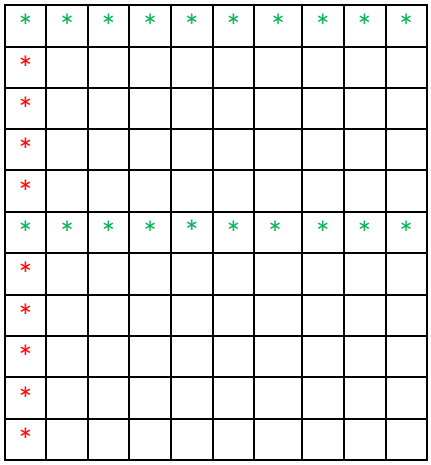
PROGRAM LOGIC

- Take 11 rows and 10 columns, for the 6th row will be the exact middle.
- On the **first, middle and last row** (marked with green), print star in **all columns,** use this condition (row == 1 || row == row_middle || row == row_middle)
- Print star at the **first column of each row** (marked with red), use this condition (column == 1)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 11;
int column_length = 10; int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_middle || row == row_length ) {
      printf("*");
    } else if(column == 1 ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

6. Program to print letter F with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
* *	

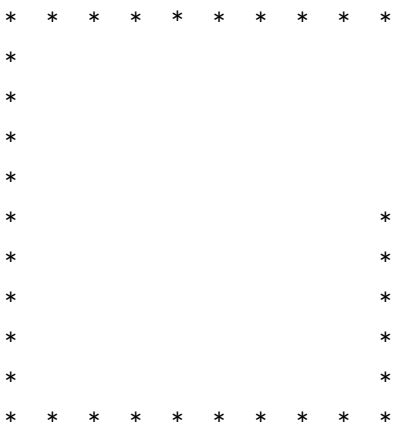
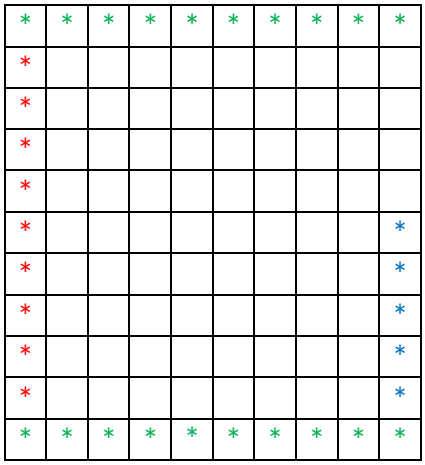
PROGRAM LOGIC

- Take 11 rows and 10 columns, for the 6th row will be the exact middle.
- On the **first and middle row** (marked with green), print star in **all columns,** use this condition (row == 1 || row == row_middle)
- Print star at the **first column of each row** (marked with red), use this condition (column == 1)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 11;
int column_length = 10; int row_middle = 6;

for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_middle ) {
      printf("*");
    } else if(column == 1 ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

7. Program to print letter G with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

PROGRAM LOGIC

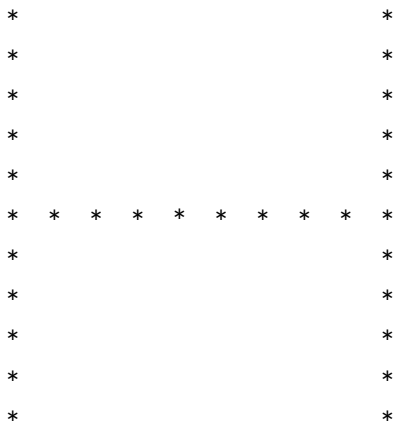
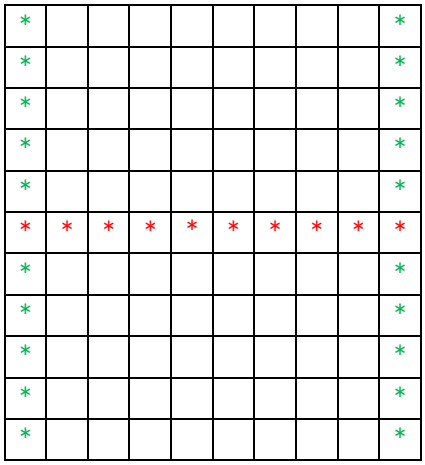
- Take 11 rows and 10 columns, for the 6th row will be the exact middle.
- On the **first and last row** (marked with green), print star in **all columns**, use this condition (row == 1 || row == row_length)
- Print star on the **first column of each row** (marked with red), use this condition (column == 1)
- Finally, from the **middle column till second-to-last column** (marked with blue), print star in **last column,** use this condition (column == column_length && row >= row_middle)
- Finally, on the **last column**, print star at **each row,** use this condition (column == column_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 11; int column_length = 10; int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_length ) {
      printf("*");
    } else if(column == 1 ) {
      printf("*");
    } else if(column == column_length && row > row_middle ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
```

```
}
```

8. Program to print letter H with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

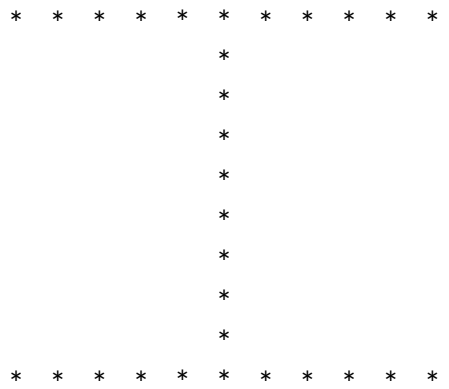
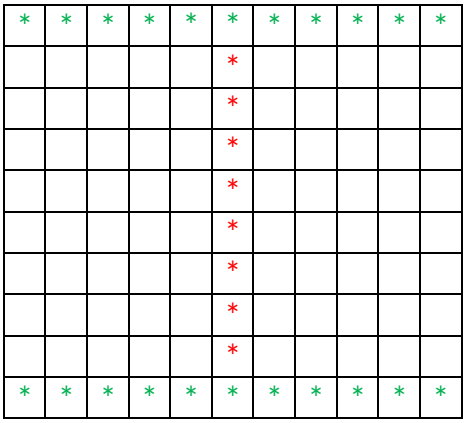
PROGRAM LOGIC

- Take 11 rows and 10 columns, for 6th row will be the exact middle.
- Print star at the **first and last column of each row** (marked with green), use this condition (column == 1 || column == column_length)
- On the **middle row** (marked with red), print star in **all columns**, use this condition (row == row_middle)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 11;
int column_length = 10; int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == row_middle) {
      printf("*");
    } else if(column == 1 || column == column_length) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

9. Program to print letter I with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

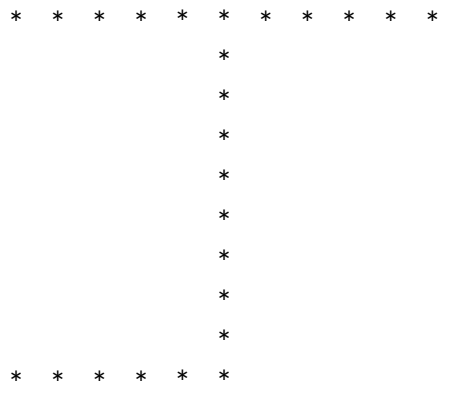
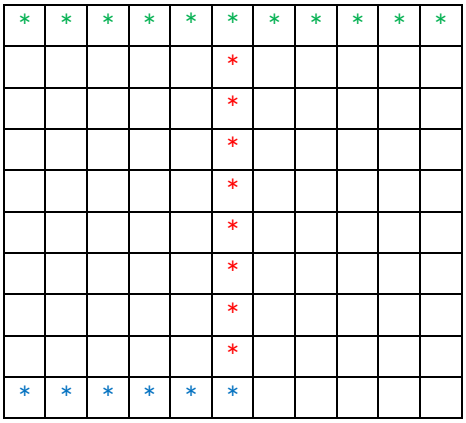
PROGRAM LOGIC

- Take 10 rows and 11 columns, for the 6th column will be the exact middle.
- On the **first row** (marked with green), print star in **all columns**, use this condition (row == 1)
- Print star at the **middle column of each row** (marked with red), use this condition (column == column_middle)
- On the **last row**, print star from **first till middle column** (marked with blue), use this condition (column < column_middle && row == row_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 10;
int column_length = 11; int column_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_length) {
      printf("*");
    } else if(column == column_middle ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

10. Program to print letter J with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

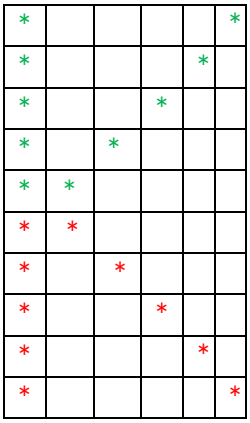
PROGRAM LOGIC

- Take 10 rows and 11 columns, for 6th column will be the exact middle.
- On first row (marked with green) print star on each column, use this condition (row == 1)
- From the second till last second row (marked with red), print star on middle column only, use this condition (column == column_middle)
- Finally, on last row (marked with blue) print star from first to middle column only, use this condition (column < column_middle && row == row_length)
- If the condition does not satisfy, print spaces.

CODE

```
int row, column; int row_length = 10; int column_length = 11; int column_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 ) {
      printf("*");
    } else if(column == column_middle ) {
      printf("*");
    } else if(column < column_middle && row == row_length ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

11. Program to print letter K with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

PROGRAM LOGIC

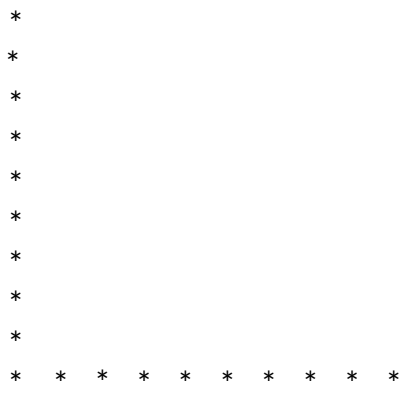
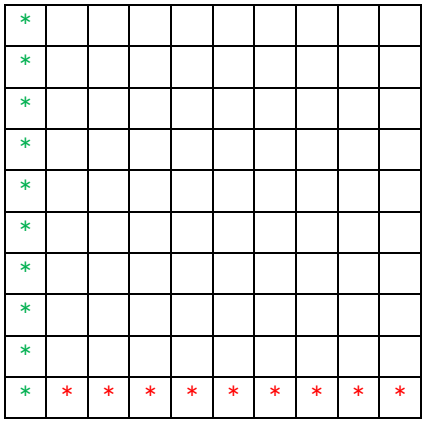
- Take 10 rows and 10 columns, program is divided into two parts
- For first section (marked in green), have star on first column, then have spaces and then again star. Please observe as row increases, spaces between stars decreases. Use this condition (column <= (row_length - row))
- For second section (marked in red), have star on first column, then have spaces and then again star. Please observe as row increases, spaces between stars also increases. Use this condition (column == row)
-

CODE

```
int row, column;
int row_length = 5;
int column_length = 5;

// upper section
for (row = 1; row <= row_length; row++) {
   printf("*");
   for (column = 1; column <= column_length; column++) {
      if(column <= (row_length-row) ) {
         printf(" ");
      } else {
         printf("*");
         break;
      }
   }
   printf("\n");
}
```

```
// lower section
for (row = 1; row <= row_length; row++) {
   printf("*");
   for (column = 1; column <= column_length; column++) {
      if(column == row ) {
         printf("*");
      } else {
         printf(" ");
      }
   }
   printf("\n");
}
```

12. Program to print letter L with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

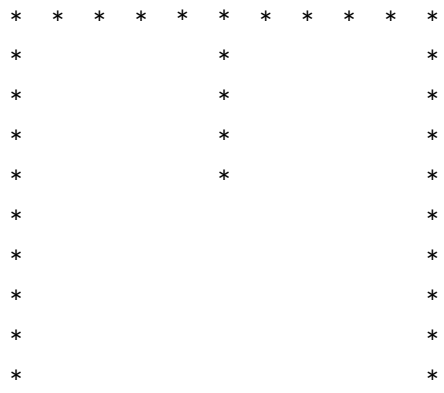
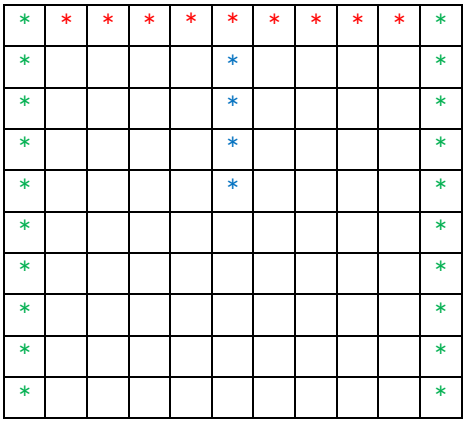
PROGRAM LOGIC

- Take 10 rows and 10 columns.
- Print star at the **first column of each row** (marked with green, use this condition (column == 1)
- For **last column** (marked with red), print star in **all columns**, use this condition (row == row_length)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 10; int column_length = 10;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(column == 1 ) {
      printf("*");
    } else if(row == row_length ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

13. Program to print letter M with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

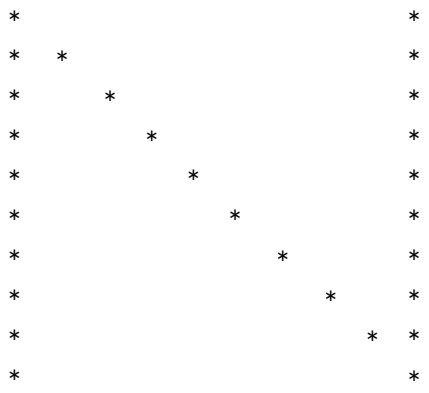
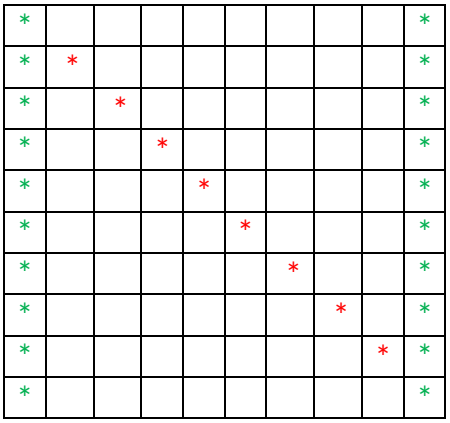
PROGRAM LOGIC

- Take 10 rows and 11 columns, for 6th column will be the exact middle.
- Print star at the **first & last column of each row** (marked with green), use this condition (column == 1 || column == column_length)
- On the **last row** (marked with red), print star in **all columns**, use this condition (row == 1)
- From **first till middle row** (marked with blue), print star in **middle column,** use this condition (column == column_middle && row < row_length / 2)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 10; int column_length = 11; int column_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1) {
      printf("*");
    } else if(column == 1 || column == column_length ) {
      printf("*");
    } else if(column == column_middle && row <= row_length/2 ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

14. Program to print letter N with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

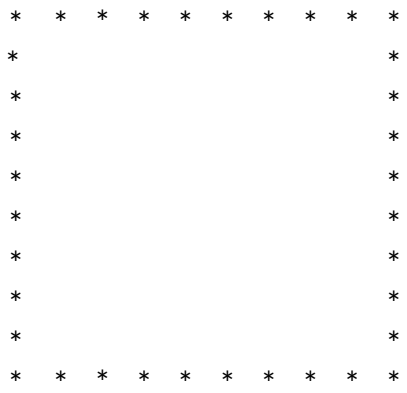
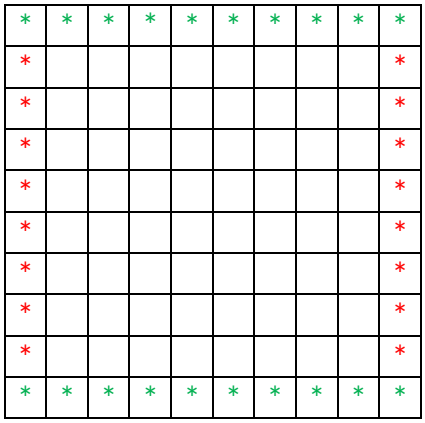
PROGRAM LOGIC

- Take 10 rows and 10 columns.
- Print star at the **first & last column of each row** (marked with green), use this condition (column == 1 || column == column_length)
- From **first till second-to-last column** (marked in red), print star where **current column that matches current row,** use this condition (column == row)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column;
int row_length = 10;
int column_length = 10;

for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(column == 1 || column == column_length ) {
      printf("*");
    } else if(column == row ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

15. Program to print letter O with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

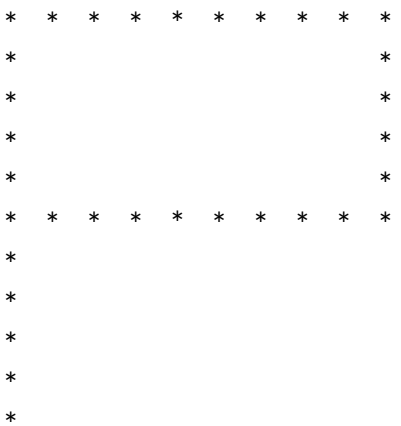
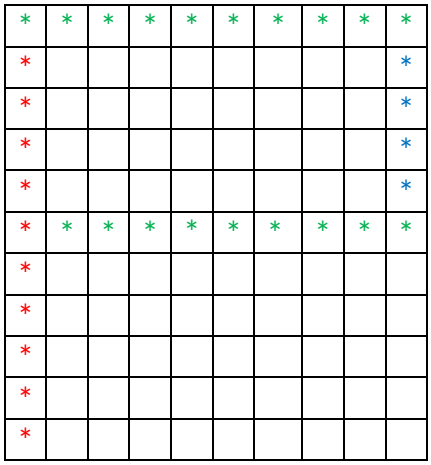
PROGRAM LOGIC

- This pattern is similar to hollow square, check the logic from the program B6

CODE

```
int row, column; int row_length = 10; int column_length = 10;
for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= column_length; column++) {
      if(column == 1 || column == column_length ) {
         printf("*");
      } else if(row == 1 || row == row_length ) {
         printf("*");
      } else {
         printf(" ");
      }
   }
   printf("\n");
}
```

16. Program to print letter P with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

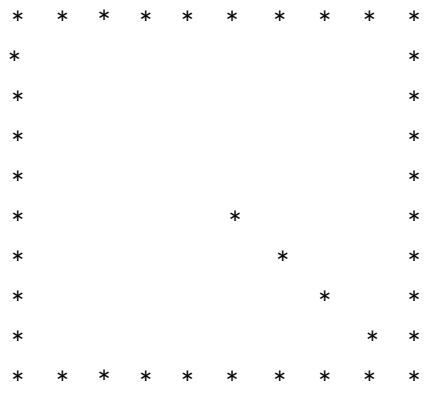
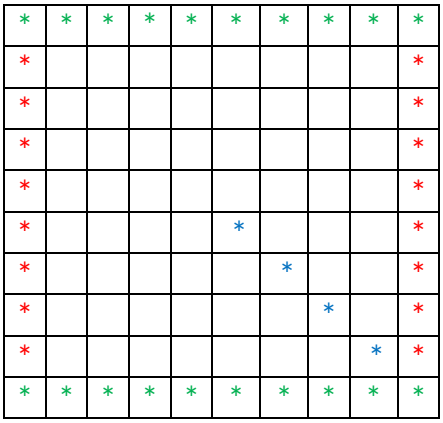
PROGRAM LOGIC

- Take 11 rows and 10 columns, for the 6th row will be the exact middle.
- On the **first and last row** (marked with green), print star in **all columns**, use this condition (row == 1 && row == row_middle)
- From **second till middle row** (marked with blue), print star at the **last column**, use this condition (column == column_length && row < row_middle)
- Print star at the **first column of each row** (marked with red), use this condition (column == 1)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 11; int column_length = 10; int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_middle) {
      printf("*");
    } else if(column == 1) {
      printf("*");
    } else if(column == column_length && row < row_middle ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

17. Program to print letter Q with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

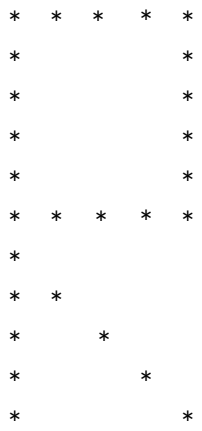
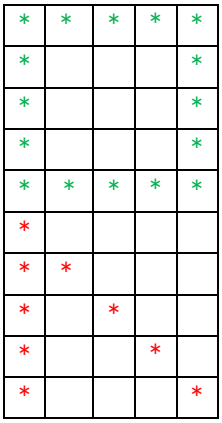
PROGRAM LOGIC

- Take 10 rows and 10 columns, and 5th row is the mid-section here.
- On the **first and last row** (marked with green), print star in **all columns**, use this condition (row == 1 && row == row_middle)
- Print star at the **first and last column of each row** (marked with red), use this condition (column == 1 && column == column_length)
- Finally, for the inner part, print star from **mid row till second-to-last row where current column is equals to current row** (marked with blue), use this condition (row >= row_middle && column == row)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 11; int column_length = 10; int row_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_length) {
      printf("*");
    } else if(column == 1 || column == column_length) {
      printf("*");
    } else if(row >= row_middle && column == row ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

18. Program to print letter R with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

PROGRAM LOGIC

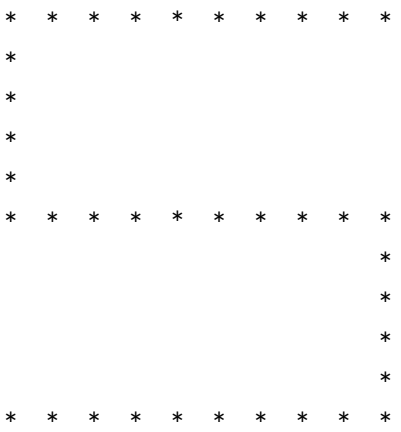
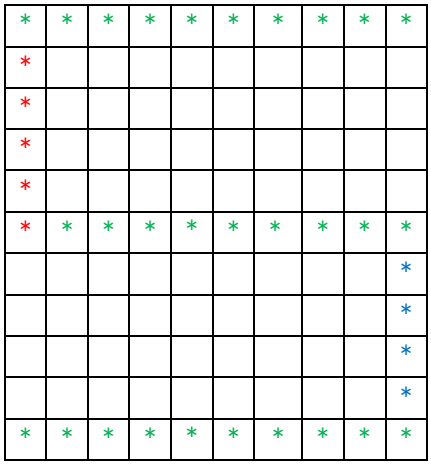
- The program is divided into two parts.
- The upper section (marked in green) is like letter O, logic is same as program B6
- The lower section (marked in red) is like the lower section of letter K, logic is same as program B16

CODE

```
int row, column;
int row_length = 5;
int column_length = 5;

// upper section
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_length) {
      printf("*");
    } else if(column == 1 || column == column_length) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

```
// lower section
for (row = 1; row <= row_length; row++) {
   printf("*");
   for (column = 1; column <= column_length; column++) {
      if(column == row) {
         printf("*");
      } else {
         printf(" ");
      }
   }
   printf("\n");
}
```

19. Program to print letter S with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

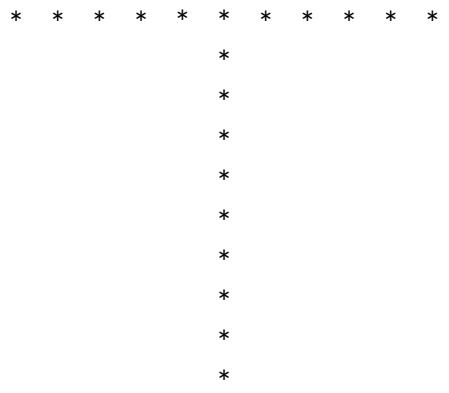
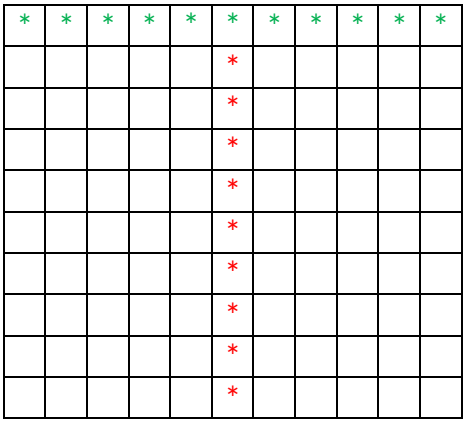
PROGRAM LOGIC

- The logic is same as the program C5

CODE

```
int row, column;
int row_length = 11;
int column_length = 10;
int row_middle = 6;

for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_middle || row == row_length ) {
      printf("*");
    } else if(column == 1 && row < row_middle ) {
      printf("*");
    } else if(column == column_length && row > row_middle ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

20. Program to print letter T with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

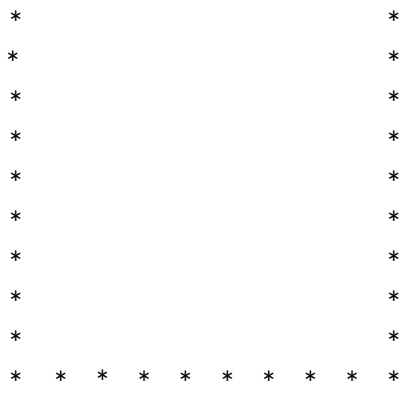
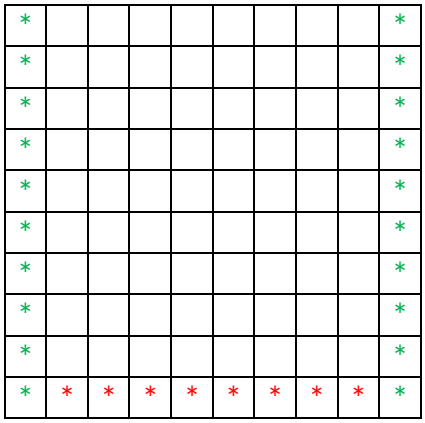
PROGRAM LOGIC

- Take 10 rows and 11 columns, for 6th column will be the exact middle.
- On the **first row** (marked in green), print star in **all columns**, use this condition (row == 1)
- From **second till last row** (marked in red), print star in the **middle column** only, use this condition (column == column_middle)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column;
int row_length = 10;
int column_length = 11;
int column_middle = 6;

for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1) {
      printf("*");
    } else if(column == column_middle ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

21. Program to print letter U with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

PROGRAM LOGIC

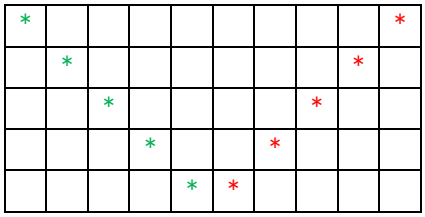
- Take 10 rows and 10 columns.
- Print star at the **first and last column of each row** (marked with green), use this condition (column == 1 || column == column_length)
- On the **last row** (marked with red), print star in **all columns,** use this condition (row == row_middle)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column;
int row_length = 10;
int column_length = 10;

for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(column == 1 || column == column_length ) {
      printf("*");
    } else if(row == row_length ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

22. Program to print letter V with star pattern

VERSION 1

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

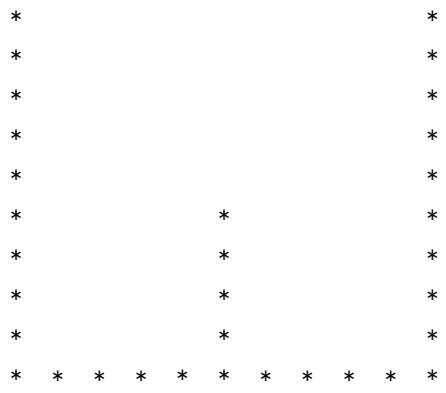
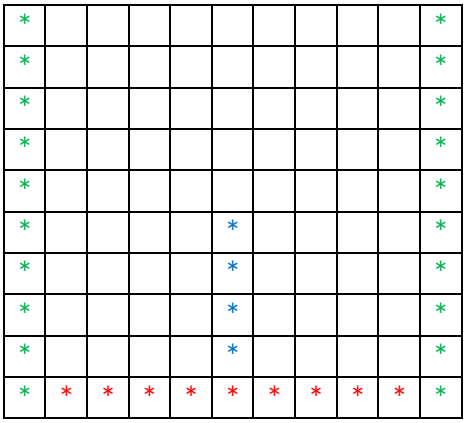
PROGRAM LOGIC

- Take 5 rows and 5 columns (loop twice, side by side).
- Start an **outer loop (row)** that will iterate 5 times.
- For first section (marked in green), it's a combination of spaces and star.
- As **row increases spaces increase**, after that print star when **column number is equals to current row,** use this condition (column == row).
- For second section (marked in red), again a combination of spaces and star.
- As row increases spaces decrease, after that print star when **current column is equals to current row**, use this condition (column == column_length+1).
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 5; int column_length = 5;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(column == row) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  for (column = 1; column <= column_length; column++) {
    // remove +1 if you want a sharp angle
    if(column == (column_length-row)+1) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

23. Program to print letter W with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

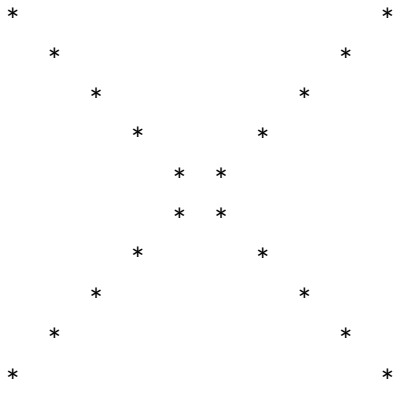
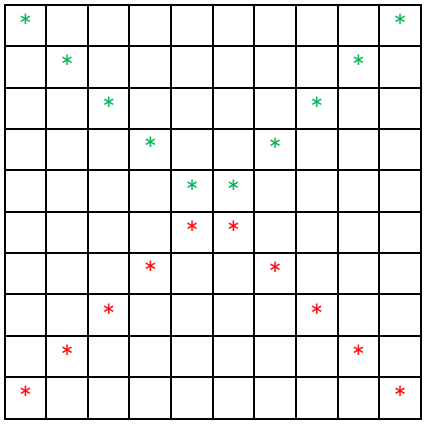
PROGRAM LOGIC

- It's like a reversed letter M
- Take 10 rows and 11 columns, for 6th column will be the exact middle.
- Print star at the **first & last column of each row** (marked with green), use this condition (column == 1 || column == column_length)
- On the **last row** (marked with red), print star in **all columns**, use this condition (row == row_length)
- From **middle till last row** (marked with blue), print star in **middle column,** use this condition (column == column_middle && row > row_length / 2)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column; int row_length = 10; int column_length = 11; int column_middle = 6;
for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == row_length) {
      printf("*");
    } else if(column == 1 || column == column_length ) {
      printf("*");
    } else if(column == column_middle && row >= row_length/2 ) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

24. Program to print letter X with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

PROGRAM LOGIC

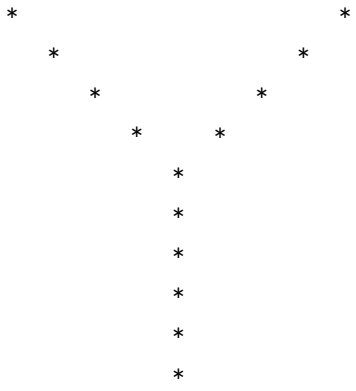
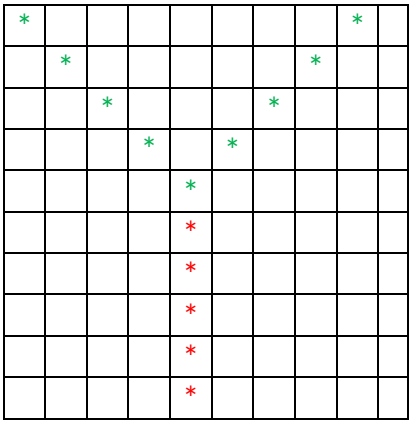
- The program is divided into two parts.
- The upper section (marked in green), logic is same as letter V, program D22
- The lower section (marked in red), it's like inverted letter V.

CODE

```
int row, column;
int row_length = 5;
int column_length = 5;

// upper section
for (row = 1; row <= row_length; row++) {

  for (column = 1; column <= column_length; column++) {
    if(column == row) {
      printf("*");
    } else {
      printf(" ");
    }
  }

  for (column = 1; column <= column_length; column++) {

    // remove +1 if you want a sharp angle
    if(column == (column_length-row)+1) {
      printf("*");
    } else {
      printf(" ");
    }
```

```
    }
    printf("\n");
}

// lower section
for (row = 1; row <= row_length; row++) {

    for (column = 1; column <= column_length; column++) {
        // remove +1 if you want a sharp angle
        if(column == (column_length-row)+1) {
            printf("*");
        } else {
            printf(" ");
        }
    }

    for (column = 1; column <= column_length; column++) {

        if(column == row) {
            printf("*");
        } else {
            printf(" ");
        }
    }
    printf("\n");
}
```

25. Program to print letter Y with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

PROGRAM LOGIC

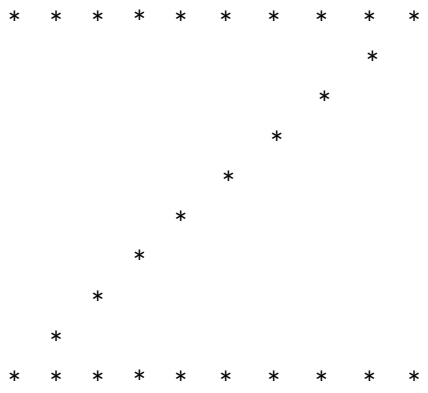
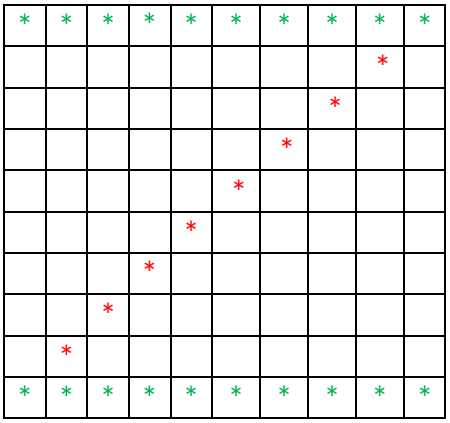
- The program is divided into two parts.
- For upper section (marked in green), base logic is same as letter V, program D22
- For lower section (marked in red), print stars at the **last column**, use this condition ($column == $column_length)

CODE

```
int row, column; int row_length = 5; int column_length = 5;
// upper section
for (row = 1; row <= row_length; row++) {

  for (column = 1; column <= column_length; column++) {
    if(column == row) {
      printf("*");
    } else {
      printf(" ");
    }
  }

  for (column = 1; column <= column_length; column++) {

    // remove +1 if you want a sharp angle
    if(column == (column_length-row)) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
```

```
}

// lower section
for (row = 1; row <= row_length; row++) {

   for (column = 1; column <= column_length; column++) {
      if(column == column_length) {
         printf("*");
      } else {
         printf(" ");
      }
   }
   printf("\n");
}
```

26. Program to print letter Z with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

PROGRAM LOGIC

- Take 10 rows and 10 columns.
- On the **first and last row** (marked with green), print star in **all columns,** use this condition (row == 1 || row == $row_length)
- From **second till second-to-last column** (marked in red), print star where **star in a descending order,** use this condition (column <= $row_length-$row)
- If the condition does not satisfy, print **spaces**.

CODE

```
int row, column;
int row_length = 10;
int column_length = 10;

for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= column_length; column++) {
    if(row == 1 || row == row_length ) {
      printf("*");
    } else if(column == (row_length-row)+1) {
      printf("*");
    } else {
      printf(" ");
    }
  }
  printf("\n");
}
```

E. Mixed Sets

In this section, you will learn to code different patterns based on what you have learnt so far. I am sure that after a couple of programs in this section, you will surely able to grasp the concept and easily understand the program.

There are 29 programs in this section. The programs are in random order.

IMPORTANT NOTE
I am assuming that you have covered all the previous sections and now you are ready to Rock'N'Roll with the upcoming programs.

1. Program to the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 5 10 16 23 2 7 13 20 28 3 9 16 24 33 4 11 19 28 38 5 13 22 32 43	1 +4 +5 +6 +7 +1 +5 +6 +7 +8 +1 +6 +7 +8 +9 +1 +7 +8 +9 +10 +1 +8 +9 +10 +11

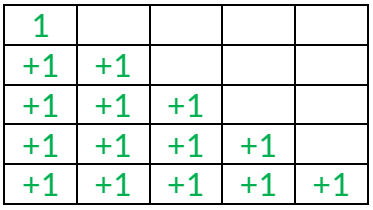
PROGRAM LOGIC

- The logic is same as program B9
- Observe, first column of each row increments by 1.
- When iterating column wise, add current row value to current column value plus 2, use this code (row+column+2)

CODE

```
int row, column;
int row_length = 5;
int print_value;

for (row = 1; row <= row_length; row++) {
   print_value = row;
   for (column = 1; column <= row_length; column++) {
      printf("%d ", print_value );
      print_value += row+column+2;
   }
   printf("\n");
}
```

2. Program to print Floyd triangle

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	

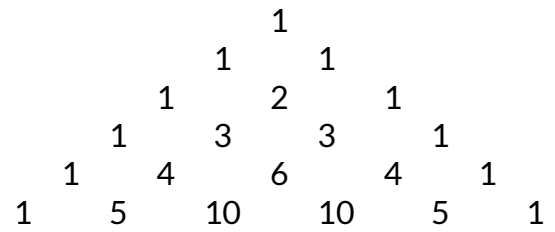
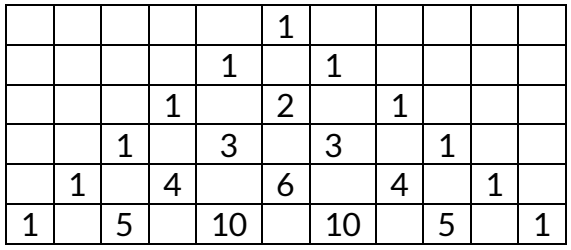
PROGRAM LOGIC

- The logic is same as program B9
- Observe, at each iteration value **increments by 1**.
- Use a counter that increments by 1 inside **inner loop**.

CODE

```
int row, column;
int row_length = 5;
int counter = 1;

for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= row; column++) {
    printf("%d ", counter);
    counter++;
  }
  printf("\n");
}
```

3. Program to print pascal triangle

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

PROGRAM LOGIC

- The logic is same as program B22
- In Pascal's triangle, each number is the sum of the two numbers directly above it. Read more here
- For first row and first column of each row, initialize the value as 1.
- For the rest calculate by `v = v * (row - column + 1) / column;`

CODE

```
int row, column, spaces;
int row_length = 6; int v;

for (row = 0; row < row_length; row++) {
  for (spaces = 1; spaces < row_length - row ; spaces++) {
    printf(" ");
  }
  for (column = 0; column <= row ; column++) {
    if(column == 0 || row == 0) {
      v = 1;
    } else {
      v = v * (row-column + 1)/column;
    }
    printf("%d ",v);
  }
  printf("\n");
}
```

4. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 7 12 16 19 2 8 15 23 3 9 14 4 10 5	1 +6 +5 +4 +3 +1 +6 +7 +8 +1 +6 +5 +1 +6 +1

PROGRAM LOGIC

- It has an inverted right-angle triangle shape; logic is given in program B11
- Calculate the **current row number** is **odd or even**.
- Print the **current row value** in the **first column**.
- Iterating column wise, take a variable print_value **add 6 and current row value** to it. and print it.
- if row is odd, **decrement** print_value by 1, else row is even, **increment** by 1.

CODE

```
int row, column; int row_length = 5; int counter_value, print_value;

for (row = 1; row <= row_length; row++) {
   counter_value = 6;
   print_value = row;
   printf("%d ", print_value);
   for (column = 1; column <= row_length-row; column++) {
      print_value += counter_value;
      printf("%d ", print_value);
      counter_value = (row%2 == 0) ? counter_value+1 : counter_value-1 ;
   }
   printf("\n");
}
```

5. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
5 6 7 8 9 4 5 6 7 3 4 5 2 3 1	5 +1 +1 +1 +1 -1 +1 +1 +1 -1 +1 +1 -1 +1 -1

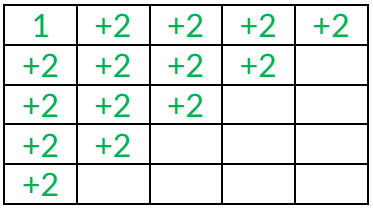
PROGRAM LOGIC

- It has an inverted right-angle triangle shape; logic is given in program B11
- Observe, as **row starts in descending order**. First value of row is 5, then 4.
- Iterating column wise, take **current row value** and **decrement by 1**.

CODE

```
int row, column;
int row_length = 5;
int value;

for (row = 0; row < row_length; row++) {
   for (column = row_length-row, value = column; column >= 1; column--, value++) {
      printf("%d ", value);
   }
   printf("\n");
}
```

6. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 3 5 7 9 3 5 7 9 5 7 9 7 9 9	

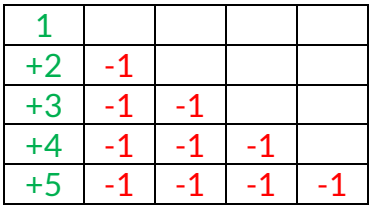
PROGRAM LOGIC

- It has an inverted right-angle triangle shape; logic is given in program B11
- Observe, **row starts with 1**, iterating row or column wise **increment value by 2**.

CODE

```
int row, column;
int row_length = 5;
int start_value, inner_value;
start_value = inner_value = 1;

for (row = 0; row < row_length; row++, start_value+=2) {
   inner_value = start_value;
   for (column = 1; column <= row_length-row; column++) {
      printf("%d ", inner_value);
      inner_value += 2;
   }
   printf("\n");
}
```

7. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 3 2 6 5 4 10 9 8 7 15 14 13 12 11	

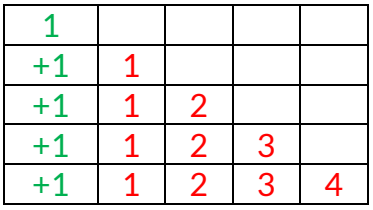
PROGRAM LOGIC

- It is a right-angle triangle shape; logic is given in program B9
- As row increases, the first column (marked in green), gets **incremented by the same number as current row value**.
- Iterating column wise, (marked in red) **decrement value by 1**.

CODE

```
int row, column;
int row_length = 5;
int row_counter = 1;
int column_counter = 1;
int step_row_value = 1;

for (row = 1; row <= row_length; row++) {
   column_counter = step_row_value;
   for (column = 1; column <= row; column++) {
      printf("%d ", column_counter);
      column_counter--;
   }
   step_row_value += 2;
   printf("\n");
}
```

8. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 2 1 3 1 2 4 1 2 3 5 1 2 3 4	

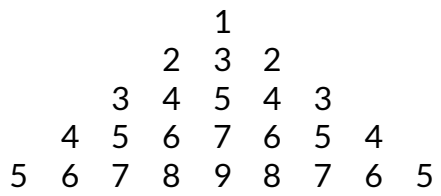
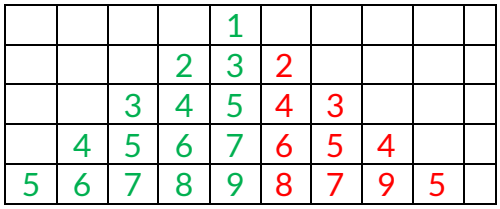
PROGRAM LOGIC

- It is a right-angle triangle shape; logic is given in program B9
- For the **first column of each row** (marked in green) **print current row value**.
- For the **second section** (marked in red) inside the inner loop, print the **current column value**.

CODE

```
int row, column;
int row_length = 5;

for (row = 1; row <= row_length; row++) {
   printf("%d ", row);
   for (column = 1; column < row; column++) {
      printf("%d ", column);
   }
   printf("\n");
}
```

9. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

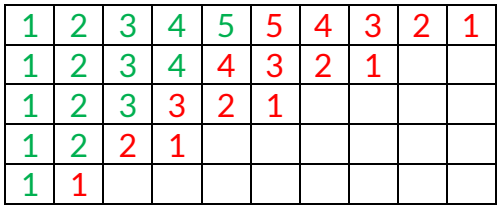
PROGRAM LOGIC

- It is a pyramid shape; logic is given in program B22
- Break the program into two section.
- The **left section** (marked in green) has spaces and value. Here, number of rows is equals to number of columns. After printing the spaces use value_print to print the output and **increment it by 1**.
- Adjust the value_print by decrementing it by 2.
- The **right section** (marked in red) number of columns is one less than number of rows. Inside the **inner loop** print value_print variable and **decrement it by 1**.

CODE

```
int row, column1, column2, spaces; int row_length = 5; int value_print = 1;
for (row = 1; row <= row_length; row++) {
  for (spaces = 1; spaces <= row_length-row; spaces++) {
    printf(" ");
  }
  value_print = row;
  for (column1 = 1; column1 <= row; column1++) {
    printf("%d", value_print);
    value_print++;
  }
  value_print -= 2;
  for (column2 = 1; column2 < row; column2++) {
    printf("%d", value_print);
    value_print--;
  }
  printf("\n");
}
```

10. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
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	

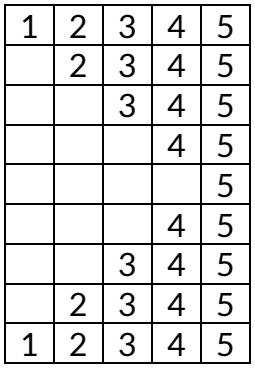
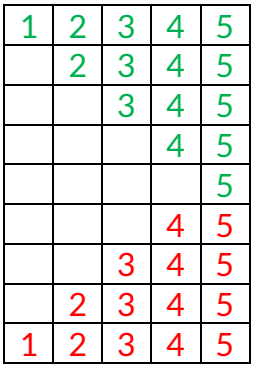
PROGRAM LOGIC

- It has an inverted right-angle triangle shape; logic is given in program B11
- Break the program into two section.
- Start an **outer loop (row)** that will iterate 5 times.
- Under the outer loop start an **inner loop (column)** (marked in green), print the **current column value** inside the loop.
- After the first inner loop, start **another inner loop** (marked in red) that will be executed same number of times of the other inner loop. To print the pattern, iterate the **loop value descending wise.**

CODE

```
int row, column1, column2;
int row_length = 5;

for (row = 1; row <= row_length; row++) {
  for (column1 = 1; column1 <= row_length-row; column1++) {
    printf("%d ", column1);
  }
  for (column2 = row_length-row; column2 >= 1; column2--) {
    printf("%d ", column2);
  }
  printf("\n");
}
```

11. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

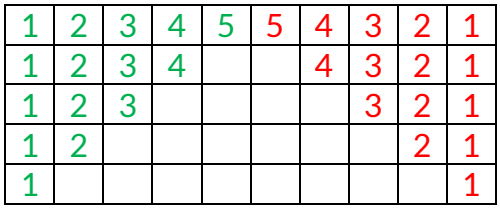
PROGRAM LOGIC

- The upper section (marked in green) logic is given in the program B15
- The lower section (marked in red) logic is given the program B13

CODE

```
int row, column, spaces; int row_length = 5; int value;
// upper section
for (row = 0; row < row_length; row++) {
   for (spaces = 0; spaces < row; spaces++) {
      printf(" ");
   }
   for (column = 0, value = row+1; column < row_length-row; column++, value++) {
      printf("%d", value);
   }
   printf("\n");
}
// lower section
for (row = 1; row < row_length; row++) {
   for (spaces = 1; spaces < row_length-row; spaces++) {
      printf(" ");
   }
   for (column = 0, value = row_length-row; column <= row; column++, value++) {
      printf("%d", value);
   }
   printf("\n");
}
```

12. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
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	

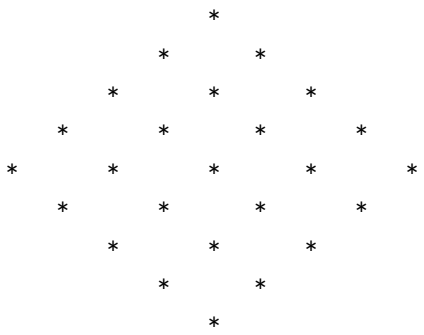
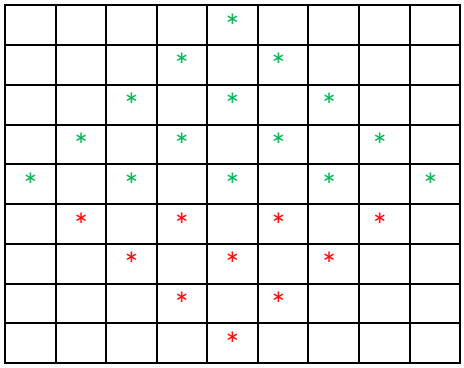
PROGRAM LOGIC

- The left section (marked in green) logic is given in the program B11
- The right section (marked in red) logic is given the program B15

CODE

```
int row, column, spaces;
int row_length = 5;

for (row = 0; row < row_length; row++) {
   // left section
   for (column = 1; column <= row_length-row; column++) {
      printf("%d", column);
   }
   for (spaces = 0; spaces < row; spaces++) {
      printf(" ");
   }

   // right section
   for (spaces = 0; spaces < row; spaces++) {
      printf(" ");
   }
   for (column = row_length-row; column >= 1; column--) {
      printf("%d", column);
   }

   printf("\n");
}
```

13. Program to print the full diamond with star pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

PROGRAM LOGIC

- The upper section (marked in green) logic is given in the program B22
- The lower section (marked in red) logic is given the program B23

CODE

```
int row, column, spaces; int row_length = 5;
// upper section
for (row = 1; row <= row_length; row++) {
   for (spaces = 1; spaces <= row_length-row; spaces++) {
      printf(" ");
   }
   for (column = 1; column <= row; column++) {
      printf("* ");
   }
   printf("\n");
}
// lower section
row_length -= 1;
for (row = 0; row < row_length; row++) {
   for (spaces = 0; spaces <= row; spaces++) {
      printf(" ");
   }
   for (column = 0; column < row_length-row; column++) {
      printf("* ");
   }
   printf("\n");
}
```

14. Program to print the alphabet pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
A B C D E F G H I J K L M N O	65 66 67 68 69 70 70 71 72 73 74 75 76 77 78

PROGRAM LOGIC

- It is a right-angle triangle shape; logic is given in program B9
- Start with the ASCII value of A (65), then use a counter and **increment by 1**.

CODE

```
int row, column;
int row_length = 5;
int ascii_code_start = 65;

for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= row; column++) {
      printf("%c ", ascii_code_start);
      ascii_code_start++;
   }
   printf("\n");
}
```

15. Program to print the alphabet pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
A B A C B A D C B A E D C B A	65 66 65 67 66 65 68 67 66 65 69 68 67 66 65

PROGRAM LOGIC

- It is a right-angle triangle shape; logic is given in program B9
- Observe, ASCII value **increases row wise** and **decreases column wise** by 1

CODE

```
int row, column;
int row_length = 5;
int ascii_code_start = 65;

for (row = 1; row <= row_length; row++) {
   for (column = row; column >= 1; column--) {
      printf("%c ", (ascii_code_start + column)-1 );
   }
   printf("\n");
}
```

16. Program to print the alphabet pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
Z Y X V V Y X V V X W V W V V	90 89 88 87 86 89 88 87 86 88 87 86 87 86 86

PROGRAM LOGIC

- It's an inverted right-angle triangle shape; logic is given in program B11
- Start with the ASCII value of Z (90)
- For **first column of each row**, set the start value by **subtracting ASCII value of Z** by **current row value**.
- Observe, ASCII value **decrement by 1** both row wise and column wise.

CODE

```
int row, column;
int row_length = 5;
int ascii_code_start = 90;
int print_value;

for (row = 0; row < row_length; row++) {
   print_value = ascii_code_start - row;
   for (column = 0; column < row_length-row; column++) {
      printf("%c ", print_value);
      print_value--;
   }
   printf("\n");
}
```

17. Program to print the alphabet pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
A A B A B A A B A B A B A B A	65 65 66 65 66 65 65 66 65 66 65 66 65 66 65

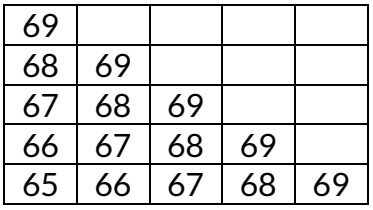
PROGRAM LOGIC

- It is a right-angle triangle shape; logic is given in program B9
- Start with the ASCII value of A (65) and B (66)
- If current column is an odd number print A else B, use this condition ((column % 2 != 0) ? ascii_code_start : ascii_code_start +1)

CODE

```
int row, column;
int row_length = 5;
int ascii_code_start = 65;
int print_value;

for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= row; column++) {
      print_value = (column % 2 != 0 ) ? ascii_code_start : ascii_code_start +1;
      printf("%c ", print_value);
   }
   printf("\n");
}
```

18. Program to print the alphabet pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
E D E C D E B C D E A B C D E	

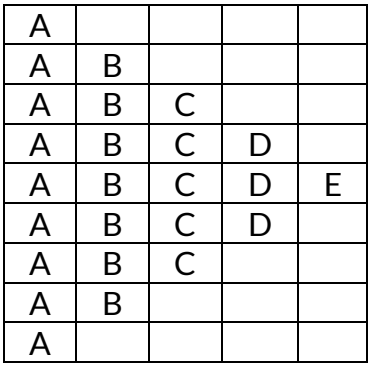
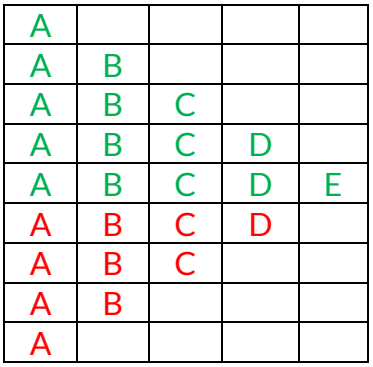
PROGRAM LOGIC

- It is a right-angle triangle shape; logic is given in program B9
- The ASCII value of A is 65 and E is 69. Start from E.
- Observe, ASCII value **decreases row wise** and **increases column wise** by 1

CODE

```
int row, column;
int row_length = 5;
int ascii_code_start = 65;
int print_value;

for (row = row_length; row >= 1; row--) {
   for (column = row; column <= row_length; column++) {
      print_value = (ascii_code_start + column )-1;
      printf("%c ", print_value);
   }
   printf("\n");
}
```

19. Program to print the half diamond with alphabet pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

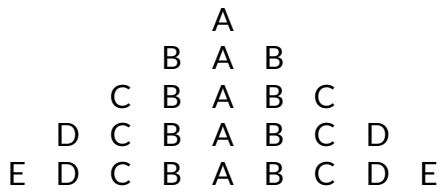
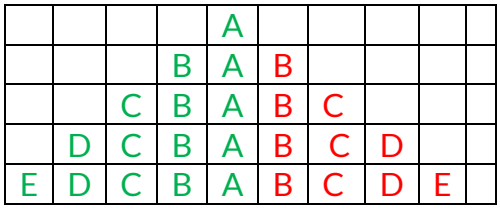
PROGRAM LOGIC

- The logic is same as program B26
- Just use character instead of number.

CODE

```
int row, column;
int row_length = 5;
int ascii_code_start = 65;
int print_value;

for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= row; column++) {
      print_value = (ascii_code_start + column )-1;
      printf("%c ", print_value);
   }
   printf("\n");
}

for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= row_length-row; column++) {
      print_value = (ascii_code_start + column )-1;
      printf("%c ", print_value);
   }
   printf("\n");
}
```

20. Program to print the pyramid with alphabet pattern

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
	

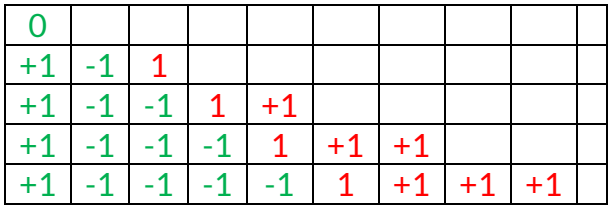
PROGRAM LOGIC

- The logic is same as program B22
- Just use character instead of number.

CODE

```
int row, column1, column2, spaces;
int row_length = 5;
int ascii_code_start = 65;
int print_value;

for (row = 1; row <= row_length; row++) {
  for (spaces = 1; spaces <= row_length-row; spaces++) {
    printf("  ");
  }
  for (column1 = row; column1 >= 1; column1--) {
    print_value = (ascii_code_start + column1 )-1;
    printf("%c ", print_value);
  }
  for (column2 = 1; column2 < row; column2++) {
    print_value++;
    printf("%c ", print_value);
  }
  printf("\n");
}
```

21. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
0 1 0 1 2 1 0 1 2 3 2 1 0 1 2 3 4 3 2 1 0 1 2 3 4	

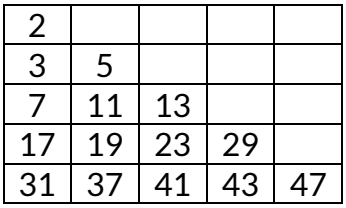
PROGRAM LOGIC

- Program is divided in two sections. Logic is given in program B9
- For the left section (marked in green) value **increases row wise** but **decreases column wise by 1**.
- For the right section (marked in red) value of **first column is fixed at 1**, but increment by 1 when iterated.

CODE

```
int row, column;
int row_length = 5;
int print_value = 0;

for (row = 0; row < row_length; row++) {
   print_value += row;
   for (column = row; column >= 0; column--) {
      printf("%d ", column);
   }
   for (column = 1; column <= row; column++) {
      printf("%d ", column);
   }
   printf("\n");
}
```

22. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47	

PROGRAM LOGIC

- It is a right-angle triangle shape; logic is given in program B9
- Check for **prime number**, if found print it.
- Inside the inner loop, set the last found prime number in a variable, and break the loop. So, when you iterate again, it starts from the last found prime number.

CODE

```
int row, column; int row_length = 5; int flag,check; int counter = 0; int last_found = 1;
for (row = 1; row <= row_length; row++) {
  counter = 0;
  for (column = last_found + 1; column <= 1000; column++) {
  // check for prime number
  flag = 1;
  for (check = 2; check <= column / 2; check++) {
    if (column % check == 0) {
      flag = 0;
      break;
    }
  }
  if (flag == 1) {
    printf("%d ", column);
    counter++;
    if (row == counter) {
      last_found = column;
      break;
    }
  }
  }
  printf("\n");
}
```

23. Program to print the alphanumeric pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
A1 B2 C3 D4 E5 B2 C3 D4 E5 C3 D4 E5 D4 E5 E5	651 662 673 684 695 662 673 684 695 673 684 695 684 695 695

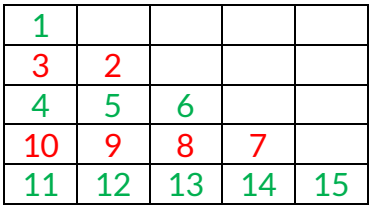
PROGRAM LOGIC

- The logic is same as program B11
- Observe, starting with ASCII value of A (marked in green) combining with current row number (marked in red).
- While iterating column wise, character and number both increments by 1.
- Print character and digit separately.

CODE

```
int row, column; int row_length = 5;
int ascii_code_start = 65; char print_value;

for (row = 1; row <= row_length; row++) {
   for (column = 0; column <= row_length-row; column++) {
      //print_value = (ascii_code_start + column )-1;
      printf("%c", (ascii_code_start+row+column)-1);
      printf("%d", (row+column));
      printf(" ");
   }
   printf("\n");
}
```

24. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 3 2 4 5 6 10 9 8 7 11 12 13 14 15	

PROGRAM LOGIC

- It is a right-angle triangle shape; logic is given in program B9
- For **odd rows** (marked in green), increment the value by 1 and print it,
- For **even rows** (marked in red), decrement the value by 1 and print it,

CODE

```
int row, column;
int row_length = 5;
int odd = 1;
int even = 0;

for (row = 1; row <= row_length; row++) {
  even = odd + row - 1;
  for (column = 1; column <= row; column++) {
    if(row % 2 == 0){
      printf("%d ", even);
    } else {
      printf("%d ", odd);
    }
    odd++;
    even--;
  }
  printf("\n");
}
```

25. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID

1				
2	4			
7	11	16		
22	29	37	46	
56	67	79	92	106

1				
+1	+2			
+3	+4	+5		
+6	+7	+8	+9	
+10	+11	+12	+13	+14

PROGRAM LOGIC

- It is a right-angle triangle shape; logic is given in program B9
- Initialize a print variable to 1.
- Also, inside the inner loop, **increment a counter by 1** and add **counter to print variable**

CODE

```
int row, column;
int row_length = 5;
int step_counter = 0;
int print_value = 1;

for (row = 1; row <= row_length; row++) {
   for (column = 1; column <= row; column++) {
      printf("%d ", print_value);
      step_counter++;
      print_value += step_counter;

   }
   printf("\n");
}
```

26. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 2 3 4 5 2 3 4 5 1 3 4 5 2 1 4 5 3 2 1 5 4 3 2 1	1 2 3 4 5 2 3 4 5 1 3 4 5 2 1 4 5 3 2 1 5 4 3 2 1

PROGRAM LOGIC

- The upper section (marked in green) logic is given in the program B11
- The lower section (marked in red) logic is given the program B13

CODE

```
int row, column1, column2;
int row_length = 5;

for (row = 1; row <= row_length; row++) {
  for (column1 = row; column1 <= row_length; column1++) {
    printf("%d", column1);
  }
  for (column2 = row-1; column2 >= 1; column2--) {
    printf("%d", column2);
  }
  printf("\n");
}
```

27. Program to print the pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 1 0 1 0 1 1 0 1 0 1 0 1 0 1	1 1 0 1 0 1 1 0 1 0 1 0 1 0 1

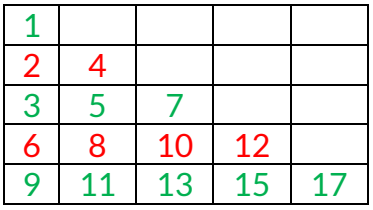
PROGRAM LOGIC

- It is a right-angle triangle shape; logic is given in program B9
- The trick here is to print the **modulus value** of the **current value of inner column**, use this condition ($column%2)

CODE

```
int row, column;
int row_length = 5;

for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= row; column++) {
    printf("%d", (column % 2));
  }
  printf("\n");
}
```

28. Program to print the number pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 2 4 3 5 7 6 8 10 12 9 11 13 15 17	

PROGRAM LOGIC

- It is a right-angle triangle shape; logic is given in program B9
- For **odd rows** (marked in green), increment the value by 2, print only odd number,
- For **even rows** (marked in red), increment the value by 2, print only even number,

v

CODE

```
int row, column;
int row_length = 5;
int odd = 1;
int even = 2;

for (row = 1; row <= row_length; row++) {
  for (column = 1; column <= row; column++) {
    if(row % 2 == 0){
      printf("%d ", even);
      even += 2;
    } else {
      printf("%d ", odd);
      odd += 2;
    }

  }
  printf("\n");
}
```

29. Program to print the number pattern given below

OUTPUT ON SCREEN	UNDERSTANDING THE GRID
1 2 3 4 5 2 1 3 4 5 3 2 1 4 5 4 3 2 1 5 5 4 3 2 1	1 2 3 4 5 2 1 3 4 5 3 2 1 4 5 4 3 2 1 5 5 4 3 2 1

PROGRAM LOGIC

- The lower section (marked in green) logic is given in the program B9
- The upper section (marked in red) logic is given the program B15

CODE

```
int row, column1, column2;
int row_length = 5;

for (row = 1; row <= row_length; row++) {
   for (column1 = row; column1 >= 1; column1--) {
      printf("%d", column1);
   }
   for (column2 = row+1; column2 <= row_length; column2++) {
      printf("%d", column2);
   }
   printf("\n");
}
```

Feedback

That's all for now readers.

This is my first attempt to write a book. I would love to have your feedback either appreciation or constructive criticism, which will help me to write better next time.

Share your feedback: https://www.thesoftwareguy.in/100-pattern-programs-in-c-free-ebook

In case have any other query please connect me via email or through social media.

CONNECT WITH ME




EMAIL	YOUTUBE	FACEBOOK	INSTAGRAM
thesoftwareguy7@gmail.com	/c/thesoftwareguy7	/thesoftwareguy7	/thesoftwareguy7

Free projects, & video tutorials will be available on my *YOUTUBE Channel*
Make sure you SUBSCRIBE and press the BELL icon, to receive all notifications.

Remember your only competition is YOU.
Be a better version of what you were YESTERDAY.