

## LAB 07 TASKS

**Question#01)** Write a C++ program to input an upper bound by user and print all numbers up to that limit?

### Source Code

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     int upperLimit;
5     cout << "Enter upper limit: ";
6     cin >> upperLimit;
7     for (int i = 1; i <= upperLimit; i++) {
8         cout << i << " ";
9     }
10    return 0;
11 }
```

### Output

```
Enter upper limit: 25
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
-----
Process exited after 10.23 seconds with return value 0
Press any key to continue . . .
```

**Question#02)** Write a C++ program to input an upper bound and separate even & odd from that limit?

### Source Code

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     int upperLimit;
5     cout << "Enter upper limit: ";
6     cin >> upperLimit;
7     cout << "Even numbers: ";
8     for (int i = 2; i <= upperLimit; i += 2) cout << i << " ";
9     cout << "\nOdd numbers: ";
10    for (int i = 1; i <= upperLimit; i += 2) cout << i << " ";
11    return 0;
12 }
```

### Output

```
Enter upper limit: 12
Even numbers: 2 4 6 8 10 12
Odd numbers: 1 3 5 7 9 11
-----
Process exited after 10.8 seconds with return value 0
Press any key to continue . . .
```

**Question #03)** Write a C++ program that prints complete alphabets using for loop? (a-z)

### Source Code

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     for (char c = 'a'; c <= 'z'; c++) {
5         cout << c << " ";
6     }
7     return 0;
8 }
```

### Output

```
a b c d e f g h i j k l m n o p q r s t u v w x y z
-----
Process exited after 0.1244 seconds with return value 0
Press any key to continue . . .
```

**Question #04)** Write a C++ program that ask the user a number and prints the table of that number up to 10?

#### Source Code

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     int number;
5     cout << "Enter a number: ";
6     cin >> number;
7     for (int i = 1; i <= 10; i++) {
8         cout << number << " x " << i << " = " << number * i << "\n";
9     }
10    return 0;
11 }
```

#### Output

```
Enter a number: 10
10 x 1 = 10
10 x 2 = 20
10 x 3 = 30
10 x 4 = 40
10 x 5 = 50
10 x 6 = 60
10 x 7 = 70
10 x 8 = 80
10 x 9 = 90
10 x 10 = 100

-----
Process exited after 2.478 seconds with return value 0
Press any key to continue . . .
```

**Question #05)** Write a C++ program that displays the product of all odd numbers from 1 to 10 using for loop?

#### Source Code

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     int product = 1;
5     for (int i = 1; i <= 10; i += 2) {
6         product *= i;
7     }
8     cout << "Product of odd numbers from 1 to 10 is: " << product;
9     return 0;
10 }
```

#### Output

```
Product of odd numbers from 1 to 10 is: 945
-----
Process exited after 0.1496 seconds with return value 0
Press any key to continue . . .
```

**Question #06)** Write a C++ program that finds the sum of squares of integer from 1 to n. Where n is a positive value entered by user (i.e.  $1^2 + 2^2 + 3^2 \dots \dots \dots + n^2$ ).

#### Source Code

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     int n, sum = 0;
5     cout << "Enter a positive number: ";
6     cin >> n;
7     for (int i = 1; i <= n; i++) {
8         sum += i * i;
9     }
10    cout << "Sum of squares from 1 to " << n << " is: " << sum;
11    return 0;
12 }
```

#### Output

```
Enter a positive number: 36
Sum of squares from 1 to 36 is: 16206
-----
Process exited after 5.327 seconds with return value 0
Press any key to continue . . .
```

**Question #07)** Write a C++ program counts the number of digits in the number entered by user?

#### Source Code

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     int number, count = 0;
5     cout << "Enter a number: ";
6     cin >> number;
7     while (number != 0) {
8         count++;
9         number /= 10;
10    }
11    cout << "Number of digits: " << count;
12    return 0;
13 }
```

#### Output

```
Enter a number: 20
Number of digits: 2
-----
Process exited after 6.11 seconds with return value 0
Press any key to continue . . .
```

**Question # 08)** Write a C++ program to reverse a number entered by user using loop.

#### Source Code

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     int number, reverse = 0;
5     cout << "Enter a number: ";
6     cin >> number;
7     while (number != 0) {
8         reverse = reverse * 10 + number % 10;
9         number /= 10;
10    }
11    cout << "Reversed number: " << reverse;
12    return 0;
13 }
```

#### Output

```
Enter a number: 356
Reversed number: 653
-----
Process exited after 3.604 seconds with return value 0
Press any key to continue . . .
```

**Question # 09)** Write a program to find first and last digit of a number.

#### Source Code

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     int number, firstDigit, lastDigit;
5     cout << "Enter a number: ";
6     cin >> number;
7     lastDigit = number % 10;
8     while (number >= 10) {
9         number /= 10;
10    }
11    firstDigit = number;
12    cout << "First digit: " << firstDigit << "\nLast digit: " << lastDigit;
13    return 0;
14 }
```

#### Output

```
Enter a number: 2048
First digit: 2
Last digit: 8
-----
Process exited after 6.346 seconds with return value 0
Press any key to continue . . .
```

**Question # 10)** Write a program to calculate product of digits of a number.

#### Source Code

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     int number, product = 1;
5     cout << "Enter a number: ";
6     cin >> number;
7     while (number != 0) {
8         product *= number % 10;
9         number /= 10;
10    }
11    cout << "Product of digits: " << product;
12    return 0;
13 }
```

#### Output

```
Enter a number: 47
Product of digits: 28
-----
Process exited after 3.127 seconds with return value 0
Press any key to continue . . .
```

**Question # 11)** Write a program to find frequency of each digit in a given integer.

#### Source Code

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     int number, frequency[10] = {0};
5     cout << "Enter a number: ";
6     cin >> number;
7     while (number != 0) {
8         frequency[number % 10]++;
9         number /= 10;
10    }
11    for (int i = 0; i < 10; i++) {
12        if (frequency[i] > 0) cout << "Digit " << i << " occurs " << frequency[i] << " times\n";
13    }
14    return 0;
15 }
```

#### Output

```
Enter a number: 272747
Digit 2 occurs 2 times
Digit 4 occurs 1 times
Digit 7 occurs 3 times
-----
Process exited after 9.861 seconds with return value 0
Press any key to continue . . .
```

**Question # 12)** Write a program to check whether a number is Armstrong number or not.

#### Source Code

```
1 #include <iostream>
2 #include <cmath>
3 using namespace std;
4 int main() {
5     int number, original, remainder, result = 0, n = 0;
6     cout << "Enter a number: ";
7     cin >> number;
8     original = number;
9     while (original != 0) {
10        original /= 10;
11        n++;
12    }
13    original = number;
14    while (original != 0) {
15        remainder = original % 10;
16        result += pow(remainder, n);
17        original /= 10;
18    }
19    if (result == number)
20        cout << number << " is an Armstrong number.";
21    else
22        cout << number << " is not an Armstrong number.";
23    return 0;
24 }
```

#### Output

```
Enter a number: 36
36 is not an Armstrong number.
-----
Process exited after 3.922 seconds with return value 0
Press any key to continue . . .
```



**Question # 13)** Write a program to convert Binary to Decimal number system.

### Source Code

```
1 #include <iostream>
2 #include <cmath>
3 using namespace std;
4 int main() {
5     int binary, decimal = 0, base = 1, remainder;
6     cout << "Enter a binary number: ";
7     cin >> binary;
8     while (binary > 0) {
9         remainder = binary % 10;
10        decimal += remainder * base;
11        binary /= 10;
12        base *= 2;
13    }
14    cout << "Decimal equivalent: " << decimal;
15    return 0;
16 }
```

### Output

```
Enter a binary number: 12
Decimal equivalent: 4
-----
Process exited after 14.11 seconds with return value 0
Press any key to continue . . .
```

**Question # 14)** Write a program to print Pascal triangle upto n rows.

### Pascal's Triangle

```

      1   1
     1 2 1
    1 3 3 1
   1 4 6 4 1
  1 5 10 10 5 1
 1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
1 8 28 56 70 56 28 8 1
```

### Source Code

```
1 #include <iostream>
2 using namespace std;
3 int main() {
4     int rows, coef = 1;
5     cout << "Enter number of rows: ";
6     cin >> rows;
7     for (int i = 0; i < rows; i++) {
8         for (int space = 1; space <= rows - i; space++)
9             cout << " ";
10        for (int j = 0; j <= i; j++) {
11            if (j == 0 || i == 0)
12                coef = 1;
13            else
14                coef = coef * (i - j + 1) / j;
15            cout << coef << " ";
16        }
17        cout << endl;
18    }
19    return 0;
20 }
```

### Output

```
Enter number of rows: 5
      1   1
     1 2 1
    1 3 3 1
   1 4 6 4 1
  1 5 10 10 5 1
-----
Process exited after 3.875 seconds with return value 0
Press any key to continue . . .
```

**Question # 15)** Write a program to print following patterns by using nested loops.

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

Rhombus Star Pattern

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

Hollow Rhombus Star Pattern

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

Mirrored Rhombus Star Pattern

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

Hollow Mirrored Rhombus Star Pattern

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

Right Triangle Star Pattern

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

Hollow Right Triangle Star Pattern

```

      *
     **
    ***
   ****
  *****

```

Mirrored Right Triangle Star Pattern

```

      *
     **
    *  *
   *    *
  *      *
 *****

```

Hollow Mirrored Right Triangle Star Pattern

```

*****
****
***
**
*

```

Inverted Right Triangle Star Pattern

```

*****
*   *
*  *
**
*

```

Hollow Inverted Right Triangle Star Pattern

```

*****
 ****
  ***
   **
    *

```

Inverted Mirrored Right Triangle Star Pattern

```

*****
*   *
 *  *
  **
   *

```

Hollow Inverted Mirrored Right Triangle Star Pattern

```

      *
     ***
    *****
   ********
  *********
 **********

```

Pyramid Star Pattern

```

      *
     * *
    *   *
   *     *
  *       *
 *         *
*         *

```

Hollow Pyramid Star Pattern

```

*****
*****
*****
***
*

```

Inverted Pyramid Star Pattern

```

*****
 *       *
  *     *
   *   *
    * *
     *

```

Hollow Inverted Pyramid Star Pattern

```

*
**
***
****
*****
*****
****
***
**
*

```

Half Diamond Star Pattern



```

      *
     **
    ***
   ****
  *****
 *****
  *****
   ***
    **
     *

```

Mirrored Half Diamond Star Pattern

```

      *
     ***
    *****
   *****
  *****
 *****
  *****
   *****
    *****
     *

```

Diamond Star Pattern

```

*****
****  ****
***    ***
**      **
*        *
*        *
**      **
***    ***
****  ****
*****

```

Hollow Diamond Star Pattern

```

*****
  ****
    ***
      **
        *
      **
    ***
  ****
*****

```

Right Arrow Star Pattern

```

      *****
     *****
    *****
   *****
  *****
 *****
*****

```

Left Arrow Star Pattern

```

      +
      +
      +
      +
+++++++
      +
      +
      +
      +

```

Plus Star Pattern

```

 *       *
 *     *
 *   *
 * *
 *
 * *
 *   *
 *     *
 *       *
 *     *
 *       *

```

X Star Pattern

```

 ***
*   *
*   *
*   *
 ***
*   *
*   *
*   *
 ***

```

Eight Star Pattern

## Heart Star Pattern

```

1 #include <iostream>
2 using namespace std;
3 // Function to print Rhombus Star Pattern
4 void printRhombus(int n) {
5     for (int i = 0; i < n; i++) {
6         for (int j = 0; j < n - i; j++)
7             cout << " ";
8         for (int j = 0; j < n; j++)
9             cout << "*";
10        cout << endl;
11    }
12 }
13 // Function to print Hollow Rhombus Star Pattern
14 void printHollowRhombus(int n) {
15     for (int i = 0; i < n; i++) {
16         for (int j = 0; j < n - i; j++)
17             cout << " ";
18         for (int j = 0; j < n; j++) {
19             if (j == 0 || j == n - 1 || i == 0 || i == n - 1)
20                 cout << "*";
21             else
22                 cout << " ";
23         }
24         cout << endl;
25     }
26 }
27 // Function to print Mirrored Rhombus Star Pattern
28 void printMirroredRhombus(int n) {
29     for (int i = 0; i < n; i++) {
30         for (int j = 0; j < i; j++)
31             cout << " ";
32         for (int j = 0; j < n; j++)
33             cout << "*";
34         cout << endl;
35     }
36 }
37 // Function to print Hollow Mirrored Rhombus Star Pattern
38 void printHollowMirroredRhombus(int n) {
39     for (int i = 0; i < n; i++) {
40         for (int j = 0; j < i; j++)
41             cout << " ";
42         for (int j = 0; j < n; j++) {
43             if (j == 0 || j == n - 1 || i == 0 || i == n - 1)
44                 cout << "*";
45             else
46                 cout << " ";
47         }
48         cout << endl;
49     }
50 }
51 // Function to print Right Triangle Star Pattern
52 void printRightTriangle(int n) {
53     for (int i = 1; i <= n; i++) {
54         for (int j = 1; j <= i; j++)
55             cout << "*";
56         cout << endl;
57     }
58 }

```

```

59 // Function to print Hollow Right Triangle Star Pattern
60 void printHollowRightTriangle(int n) {
61     for (int i = 1; i <= n; i++) {
62         for (int j = 1; j <= i; j++) {
63             if (j == 1 || j == i || i == n)
64                 cout << "*";
65             else
66                 cout << " ";
67         }
68         cout << endl;
69     }
70 }
71 // Function to print Mirrored Right Triangle Star Pattern
72 void printMirroredRightTriangle(int n) {
73     for (int i = 1; i <= n; i++) {
74         for (int j = 1; j <= n - i; j++)
75             cout << " ";
76         for (int j = 1; j <= i; j++)
77             cout << "*";
78         cout << endl;
79     }
80 }
81 // Function to print Hollow Mirrored Right Triangle Star Pattern
82 void printHollowMirroredRightTriangle(int n) {
83     for (int i = 1; i <= n; i++) {
84         for (int j = 1; j <= n - i; j++)
85             cout << " ";
86         for (int j = 1; j <= i; j++) {
87             if (j == 1 || j == i || i == n)
88                 cout << "*";
89             else
90                 cout << " ";
91         }
92         cout << endl;
93     }
94 }
95 // Function to print Inverted Right Triangle Star Pattern
96 void printInvertedRightTriangle(int n) {
97     for (int i = n; i >= 1; i--) {
98         for (int j = 1; j <= i; j++)
99             cout << "*";
100         cout << endl;
101     }
102 }
103 // Function to print Hollow Inverted Right Triangle Star Pattern
104 void printHollowInvertedRightTriangle(int n) {
105     for (int i = n; i >= 1; i--) {
106         for (int j = 1; j <= i; j++) {
107             if (j == 1 || j == i || i == n)
108                 cout << "*";
109             else
110                 cout << " ";
111         }
112         cout << endl;
113     }
114 }
115 // Function to print Inverted Mirrored Right Triangle Star Pattern
116 void printInvertedMirroredRightTriangle(int n) {
117     for (int i = n; i >= 1; i--) {
118         for (int j = 1; j <= n - i; j++)
119             cout << " ";
120         for (int j = 1; j <= i; j++)
121             cout << "*";
122         cout << endl;
123     }
124 }

```

```

125 // Function to print Pyramid Star Pattern
126 void printPyramid(int n) {
127     for (int i = 1; i <= n; i++) {
128         for (int j = 1; j <= n - i; j++)
129             cout << " ";
130         for (int j = 1; j <= (2 * i - 1); j++)
131             cout << "*";
132         cout << endl;
133     }
134 }
135 // Function to print Hollow Pyramid Star Pattern
136 void printHollowPyramid(int n) {
137     for (int i = 1; i <= n; i++) {
138         for (int j = 1; j <= n - i; j++)
139             cout << " ";
140         for (int j = 1; j <= (2 * i - 1); j++) {
141             if (j == 1 || j == (2 * i - 1) || i == n)
142                 cout << "*";
143             else
144                 cout << " ";
145         }
146         cout << endl;
147     }
148 }
149 // Function to print Inverted Pyramid Star Pattern
150 void printInvertedPyramid(int n) {
151     for (int i = n; i >= 1; i--) {
152         for (int j = 1; j <= n - i; j++)
153             cout << " ";
154         for (int j = 1; j <= (2 * i - 1); j++)
155             cout << "*";
156         cout << endl;
157     }
158 }
159 // Function to print Hollow Inverted Pyramid Star Pattern
160 void printHollowInvertedPyramid(int n) {
161     for (int i = n; i >= 1; i--) {
162         for (int j = 1; j <= n - i; j++)
163             cout << " ";
164         for (int j = 1; j <= (2 * i - 1); j++) {
165             if (j == 1 || j == (2 * i - 1) || i == n)
166                 cout << "*";
167             else
168                 cout << " ";
169         }
170         cout << endl;
171     }
172 }
173 // Function to print Half Diamond Star Pattern
174 void printHalfDiamond(int n) {
175     for (int i = 1; i <= n; i++) {
176         for (int j = 1; j <= i; j++)
177             cout << "*";
178         cout << endl;
179     }
180     for (int i = n - 1; i >= 1; i--) {
181         for (int j = 1; j <= i; j++)
182             cout << "*";
183         cout << endl;
184     }
185 }

```



```

186 // Function to print Mirrored Half Diamond Star Pattern
187 void printMirroredHalfDiamond(int n) {
188     for (int i = 1; i <= n; i++) {
189         for (int j = 1; j <= n - i; j++)
190             cout << " ";
191         for (int j = 1; j <= i; j++)
192             cout << "*";
193         cout << endl;
194     }
195     for (int i = n - 1; i >= 1; i--) {
196         for (int j = 1; j <= n - i; j++)
197             cout << " ";
198         for (int j = 1; j <= i; j++)
199             cout << "*";
200         cout << endl;
201     }
202 }
203 // Function to print Diamond Star Pattern
204 void printDiamond(int n) {
205     printPyramid(n);
206     printInvertedPyramid(n);
207 }
208 // Function to print Hollow Diamond Star Pattern
209 void printHollowDiamond(int n) {
210     printHollowPyramid(n);
211     printHollowInvertedPyramid(n);
212 }
213 // Function to print Right Arrow Star Pattern
214 void printRightArrow(int n) {
215     printPyramid(n);
216     printInvertedRightTriangle(n);
217 }
218 // Function to print Left Arrow Star Pattern
219 void printLeftArrow(int n) {
220     printInvertedRightTriangle(n);
221     printPyramid(n);
222 }
223 // Function to print Plus Star Pattern
224 void printPlus(int n) {
225     for (int i = 1; i <= n; i++) {
226         for (int j = 1; j <= n; j++) {
227             if (j == n / 2 + 1 || i == n / 2 + 1)
228                 cout << "*";
229             else
230                 cout << " ";
231         }
232         cout << endl;
233     }
234 }
235 // Function to print X Star Pattern
236 void printX(int n) {
237     for (int i = 1; i <= n; i++) {
238         for (int j = 1; j <= n; j++) {
239             if (j == i || j == n - i + 1)
240                 cout << "*";
241             else
242                 cout << " ";
243         }
244         cout << endl;
245     }
246 }

```



```

247 // Function to print Eight Star Pattern
248 void printEight(int n) {
249     for (int i = 1; i <= n; i++) {
250         for (int j = 1; j <= n; j++) {
251             if (i == 1 || i == n || j == 1 || j == n || (i == j && i <= n / 2) || (i + j == n + 1 && i <= n / 2))
252                 cout << "***";
253             else
254                 cout << " ";
255         }
256         cout << endl;
257     }
258 }
259 // Function to print Heart Star Pattern
260 void printHeart(int n) {
261     for (int i = n / 2; i <= n; i += 2) {
262         for (int j = 1; j <= n - i; j += 2)
263             cout << " ";
264         for (int j = 1; j <= i; j++)
265             cout << "***";
266         for (int j = 1; j <= n - i; j++)
267             cout << " ";
268         for (int j = 1; j <= i; j++)
269             cout << "***";
270         cout << endl;
271     }
272     for (int i = n; i >= 1; i--) {
273         for (int j = i; j < n; j++)
274             cout << " ";
275         for (int j = 1; j <= (i * 2) - 1; j++)
276             cout << "***";
277         cout << endl;
278     }
279 }
280 // Main function to execute the star patterns
281 int main() {
282     int n = 5;
283     cout << "Rhombus Star Pattern:\n";
284     printRhombus(n);
285     cout << "\nHollow Rhombus Star Pattern:\n";
286     printHollowRhombus(n);
287     cout << "\nMirrored Rhombus Star Pattern:\n";
288     printMirroredRhombus(n);
289     cout << "\nHollow Mirrored Rhombus Star Pattern:\n";
290     printHollowMirroredRhombus(n);
291     cout << "\nRight Triangle Star Pattern:\n";
292     printRightTriangle(n);
293     cout << "\nHollow Right Triangle Star Pattern:\n";
294     printHollowRightTriangle(n);
295     cout << "\nMirrored Right Triangle Star Pattern:\n";
296     printMirroredRightTriangle(n);
297     cout << "\nHollow Mirrored Right Triangle Star Pattern:\n";
298     printHollowMirroredRightTriangle(n);
299     cout << "\nInverted Right Triangle Star Pattern:\n";
300     printInvertedRightTriangle(n);
301     cout << "\nHollow Inverted Right Triangle Star Pattern:\n";
302     printHollowInvertedRightTriangle(n);
303     cout << "\nInverted Mirrored Right Triangle Star Pattern:\n";
304     printInvertedMirroredRightTriangle(n);
305     cout << "\nPyramid Star Pattern:\n";
306     printPyramid(n);
307     cout << "\nHollow Pyramid Star Pattern:\n";
308     printHollowPyramid(n);
309     cout << "\nInverted Pyramid Star Pattern:\n";
310     printInvertedPyramid(n);
311     cout << "\nHollow Inverted Pyramid Star Pattern:\n";
312     printHollowInvertedPyramid(n);
313     cout << "\nHalf Diamond Star Pattern:\n";
314     printHalfDiamond(n);
315     cout << "\nMirrored Half Diamond Star Pattern:\n";
316     printMirroredHalfDiamond(n);
317     cout << "\nDiamond Star Pattern:\n";
318     printDiamond(n);
319     cout << "\nHollow Diamond Star Pattern:\n";
320     printHollowDiamond(n);
321     cout << "\nRight Arrow Star Pattern:\n";
322     printRightArrow(n);
323     cout << "\nLeft Arrow Star Pattern:\n";
324     printLeftArrow(n);
325     cout << "\nPlus Star Pattern:\n";
326     printPlus(n);
327     cout << "\nX Star Pattern:\n";
328     printX(n);
329     cout << "\nEight Star Pattern:\n";
330     printEight(n);
331     cout << "\nHeart Star Pattern:\n";
332     printHeart(n);
333     return 0;
334 }

```

## Output

Rhombus Star Pattern:

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

Hollow Rhombus Star Pattern:

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

Mirrored Rhombus Star Pattern:

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

Hollow Mirrored Rhombus Star Pattern:

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

Right Triangle Star Pattern:

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

Hollow Right Triangle Star Pattern:

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

Mirrored Right Triangle Star Pattern:

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

Hollow Mirrored Right Triangle Star Pattern:

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

Inverted Right Triangle Star Pattern:

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

Hollow Inverted Right Triangle Star Pattern:

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

Inverted Mirrored Right Triangle Star Pattern:

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

Pyramid Star Pattern:

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

Hollow Pyramid Star Pattern:

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

Inverted Pyramid Star Pattern:

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

Hollow Inverted Pyramid Star Pattern:

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

Half Diamond Star Pattern:

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

Mirrored Half Diamond Star Pattern:

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

Diamond Star Pattern:

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

Hollow Diamond Star Pattern:

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

Right Arrow Star Pattern:

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

Left Arrow Star Pattern:

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

Plus Star Pattern:

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

X Star Pattern:

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

Eight Star Pattern:

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

Heart Star Pattern:

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

---

Process exited after 0.2887 seconds with return value 0  
Press any key to continue . . .