**1. Right-Angled Triangle**

**Pattern:**

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

**Code:**

#include <iostream>

using namespace std;

int main() {

int n;

cout << "Enter number of rows: ";

cin >> n;

for(int i = 1; i <= n; i++) {

for(int j = 1; j <= i; j++) {

cout << "\*";

}

cout << endl;

}

return 0;

}

**2. Inverted Right-Angled Triangle**

**Pattern:**

\*\*\*\*\*

\*\*\*\*

\*\*\*

\*\*

\*

**Code:**

#include <iostream>

using namespace std;

int main() {

int n;

cout << "Enter number of rows: ";

cin >> n;

for(int i = n; i >= 1; i--) {

for(int j = 1; j <= i; j++) {

cout << "\*";

}

cout << endl;

}

return 0;

}

**3. Pyramid Pattern**

**Pattern:**

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

**Code:**

#include <iostream>

using namespace std;

int main() {

int n;

cout << "Enter number of rows: ";

cin >> n;

for(int i = 1; i <= n; i++) {

// Print spaces

for(int j = 1; j <= n - i; j++)

cout << " ";

// Print stars

for(int j = 1; j <= 2\*i - 1; j++)

cout << "\*";

cout << endl;

}

return 0;

}

**4. Diamond Pattern**

**Pattern:**

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*

\*

**Code:**

#include <iostream>

using namespace std;

int main() {

int n;

cout << "Enter number of rows (for half diamond): ";

cin >> n;

// Upper half

for(int i = 1; i <= n; i++) {

for(int j = 1; j <= n - i; j++)

cout << " ";

for(int j = 1; j <= 2\*i - 1; j++)

cout << "\*";

cout << endl;

}

// Lower half

for(int i = n-1; i >= 1; i--) {

for(int j = 1; j <= n - i; j++)

cout << " ";

for(int j = 1; j <= 2\*i - 1; j++)

cout << "\*";

cout << endl;

}

return 0;

}

**5. Hollow Square Pattern**

**Pattern:**

\*\*\*\*\*

\* \*

\* \*

\* \*

\*\*\*\*\*

**Code:**

#include <iostream>

using namespace std;

int main() {

int n;

cout << "Enter size of square: ";

cin >> n;

for(int i = 1; i <= n; i++) {

for(int j = 1; j <= n; j++) {

if(i == 1 || i == n || j == 1 || j == n)

cout << "\*";

else

cout << " ";

}

cout << endl;

}

return 0;

}