# Learning Objectives: Nested Loops

- **Explain nested loop syntax**
- **Identify the relationship between the variables in each loop with the output produced**

# Nested Loops

---

A **nested loop** is a loop that exists inside of another loop. An advantage of using nested loops is that the loops can work together to create unique and complex outputs. However, due to their complexity potential, it is rare to see the implementation of more than two nested loops. If possible, it is recommended that you re-factor your code to reduce this complexity.

### Syntax

The code below will draw a rectangle of 100 # in a 10 x 10 grid. The first loop controls the row of output, while the second loop prints 10 # to the screen.

```
for (int row = 0; row < 10; row++) { //outer loop
  for (int col = 0; col < 10; col++) { //inner loop
    cout << "#";
  }
  cout << "" << endl; //adds new line
}
```

Code Visualizer

challenge

## What happens if you:

- Replace `row < 10` with `row < 5` in the code above?
- Replace `col < 10` with `col < 20` in the code above?

Code Visualizer

### Nested Loop Coding Challenge 1

Using nested loops, write some code that outputs the following:

```
##########
##########
##########
##########
##########
##########
##########
```

Code Visualizer

▼ **Hint**
The output is the same character (#). Make sure that your nested loops have the right numbers in the boolean expressions to get the appropriate number of rows and columns.

## Nested Loop Coding Challenge 2

Using nested loops, write some code that outputs the following:

```
<<<<<<<<<<
>>>>>>>>>>
<<<<<<<<<<
>>>>>>>>>>
<<<<<<<<<<
```

Code Visualizer

▼ **Hint**
The output is a < when the outer loop variable is even (0, 2, 4) and a > when the outer loop variable is odd (1, 3).

## Nested Loop Coding Challenge 3

Using nested loops, write some code that outputs the following:

```
1
22
333
4444
55555
```

Code Visualizer

▼ **Hint**
Note how the pattern goes from 1 to 5 starting on line 1 (through line 5) and prints the line number equal to the amount of times as that numbered line.

First, the outer loop should start at `1`. Second, the inner loop should run the same amount of times as the row number up to the row number's limit.

---

▼ **Sample Solutions**

There are *multiple* ways to solve the challenges above but here are some sample solutions using various combinations of `for` and `while` loops:

```
int row = 0;
while (row < 7) {
  int col = 0;
  while (col < 11) {
    cout << "#";
    col++;
  }
  cout << "" << endl;
  row++;
}
```

```
for (int row = 0; row < 5; row++) {
  if (row % 2 == 0) {
    int col = 0;
    while (col < 10) {
      cout << "<";
      col++;
    }
    cout << "" << endl;
  }
  else {
    int col = 0;
    while (col < 10) {
      cout << ">";
      col++;
    }
    cout << "" << endl;
  }
}
```

```
for (int row = 1; row <= 5; row++) {
  for (int col = 1; col <= row; col++) {
    cout << row;
  }
  cout << "" << endl;
}
```

---