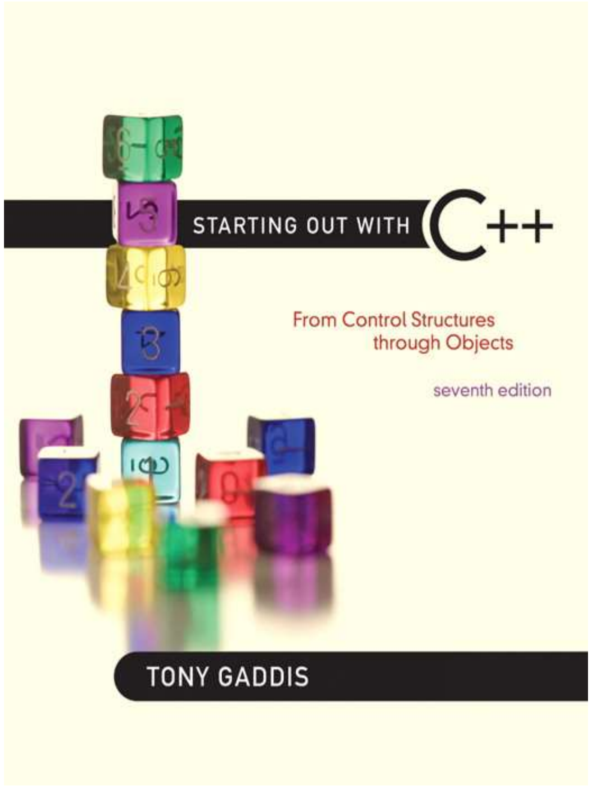
20 - 10 - 2022



The `do-while` Loop, Break and continue statement, Nesting of loops

Dry Run/Tracing of loops

```
int main()
{
    int n = 6, x = 2, i = 0;
    while (i <= n)
    {
        if (i % 2 == 1)
                x = x + pow(2, i) * i;
        i++;
        cout << x << "-";
    }
}
```

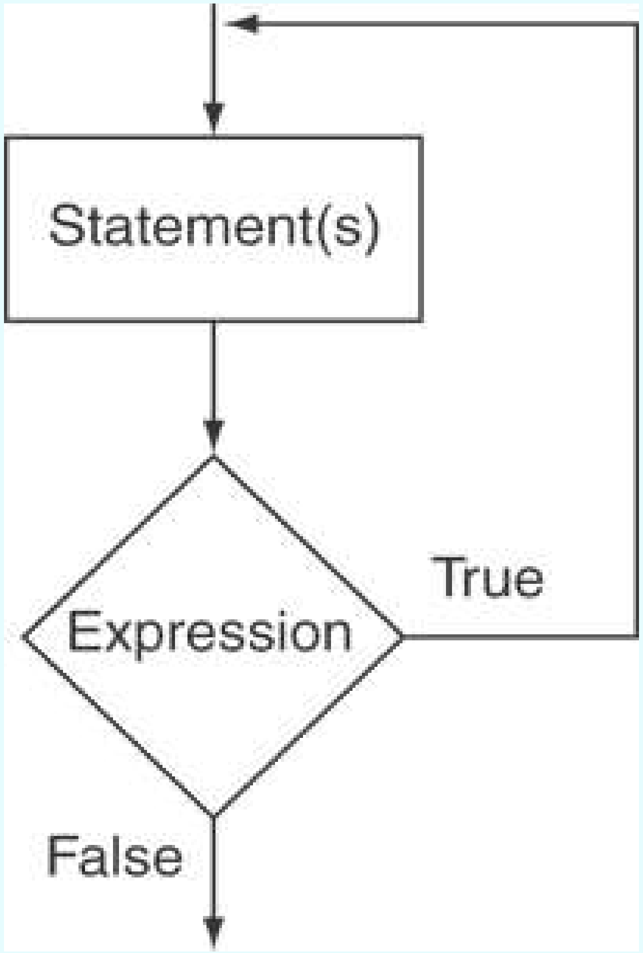
The `do-while` Loop

- `do-while`: a posttest loop – execute the loop, then test the `expression`
- General Format:

```
   do
      statement;   // or block in { }
  while (expression);
```

- Note that a semicolon is required after `(expression)`

The Logic of a `do-while` Loop

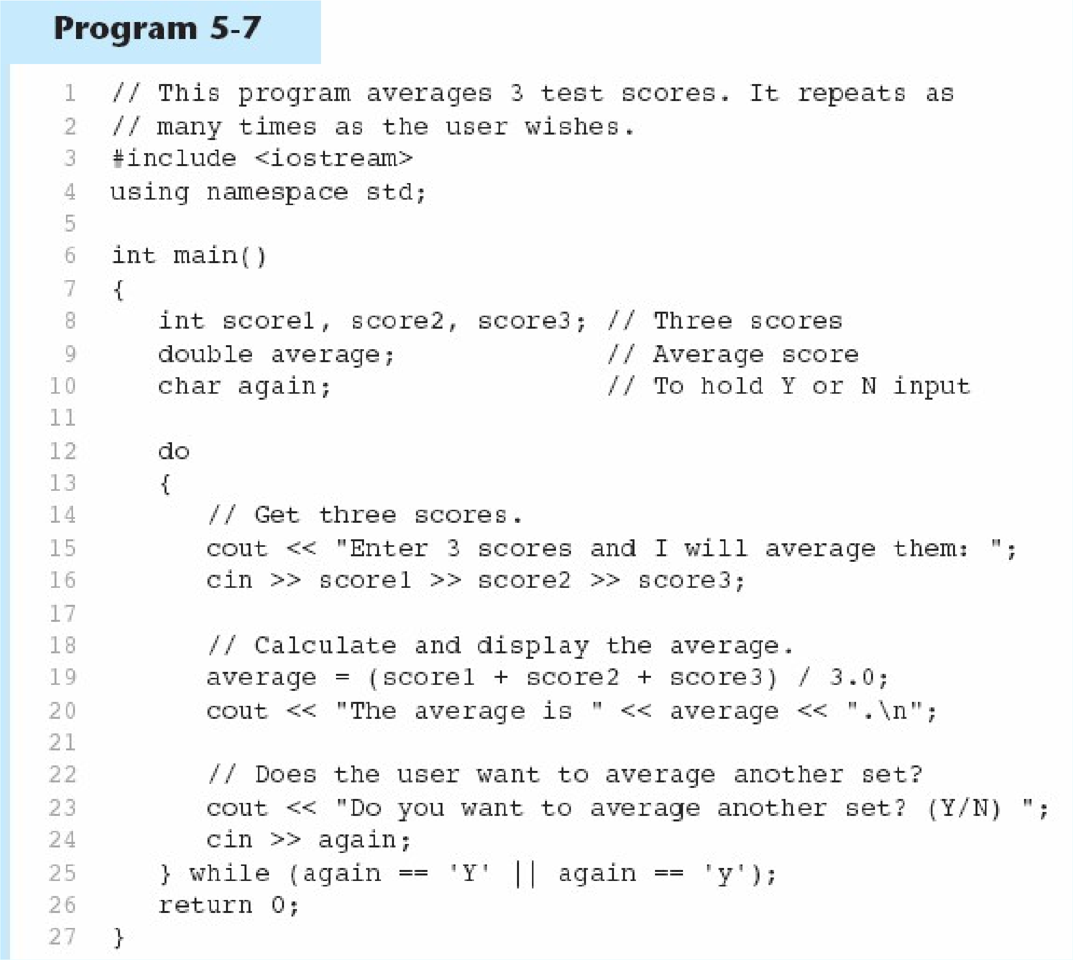


An Example `do-while` Loop

```
int x = 1;
do
{
    cout << x << endl;
} while(x < 0);
```

Although the test expression is false, this loop will execute one time because `do-while` is a posttest loop.

A `do-while` Loop in Program 5-7

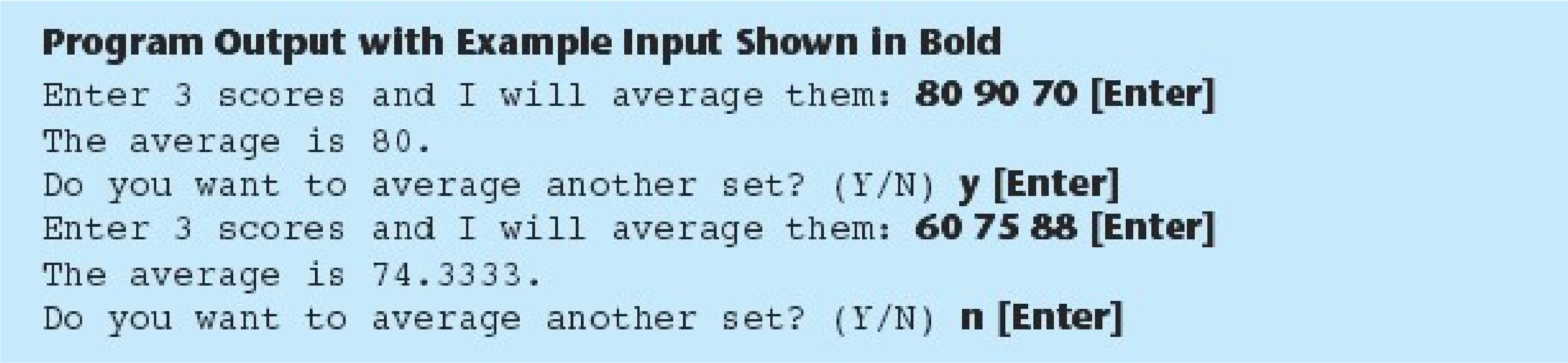


Program 5-7

```
// This program averages 3 test scores. It repeats as
// many times as the user wishes.
#include <iostream>
using namespace std;

int main()
{
   int score1, score2, score3; // Three scores
   double average;             // Average score
   char again;                 // To hold Y or N input

   do
   {
      // Get three scores.
      cout << "Enter 3 scores and I will average them: ";
      cin >> score1 >> score2 >> score3;

      // Calculate and display the average.
      average = (score1 + score2 + score3) / 3.0;
      cout << "The average is " << average << ".\n";

      // Does the user want to average another set?
      cout << "Do you want to average another set? (Y/N) ";
      cin >> again;
   } while (again == 'Y' || again == 'y');
   return 0;
}
```

Continued…

A `do-while` Loop in Program 5-7



Program Output with Example Input Shown in Bold

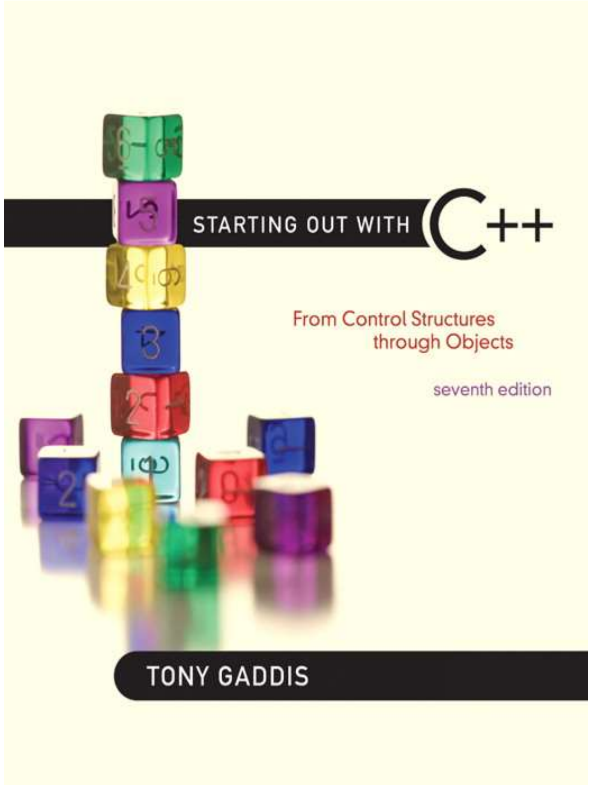
```
Enter 3 scores and I will average them: 80 90 70 [Enter]
The average is 80.
Do you want to average another set? (Y/N) y [Enter]
Enter 3 scores and I will average them: 60 75 88 [Enter]
The average is 74.3333.
Do you want to average another set? (Y/N) n [Enter]
```

`do-while` Loop Notes

- Loop always executes at least once
- Execution continues as long as *`expression`* is `true`, stops repetition when *`expression`* becomes `false`
- Useful in menu-driven programs to bring user back to menu to make another choice (*see* Program 5-8 *on pages* 245-246)

Dry Run/Tracing of loops

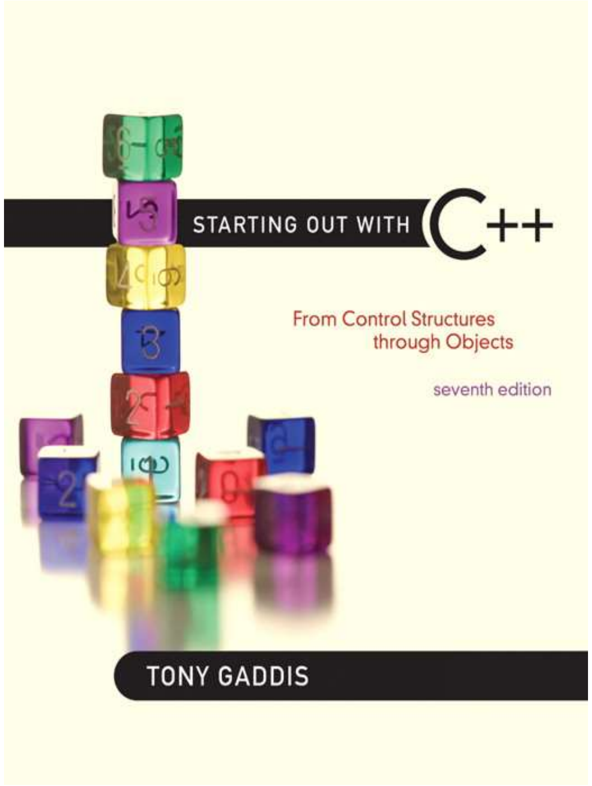
```
int main()
{
    int x, y = 6;
    for (x = 4; x < y; x++)
    {
        y %= x;
        cout << y << "-";
    }

    do{
        cout << --x << "-";
        x *= 2;
    } while (x <= 10);
}
```



Deciding Which Loop to Use

Deciding Which Loop to Use

- The `while` loop is a conditional pretest loop
  - Iterates as long as a certain condition exits
  - Validating input
  - Reading lists of data terminated by a sentinel
- The `do-while` loop is a conditional posttest loop
  - Always iterates at least once
  - Repeating a menu
- The `for` loop is a pretest loop
  - Built-in expressions for initializing, testing, and updating
  - Situations where the exact number of iterations is known



Breaking and Continuing a Loop

Breaking Out of a Loop

- Can use `break` to terminate execution of a loop
- Use sparingly if at all – makes code harder to understand and debug
- When used in an inner loop, terminates that loop only and goes back to outer loop

The `continue` Statement

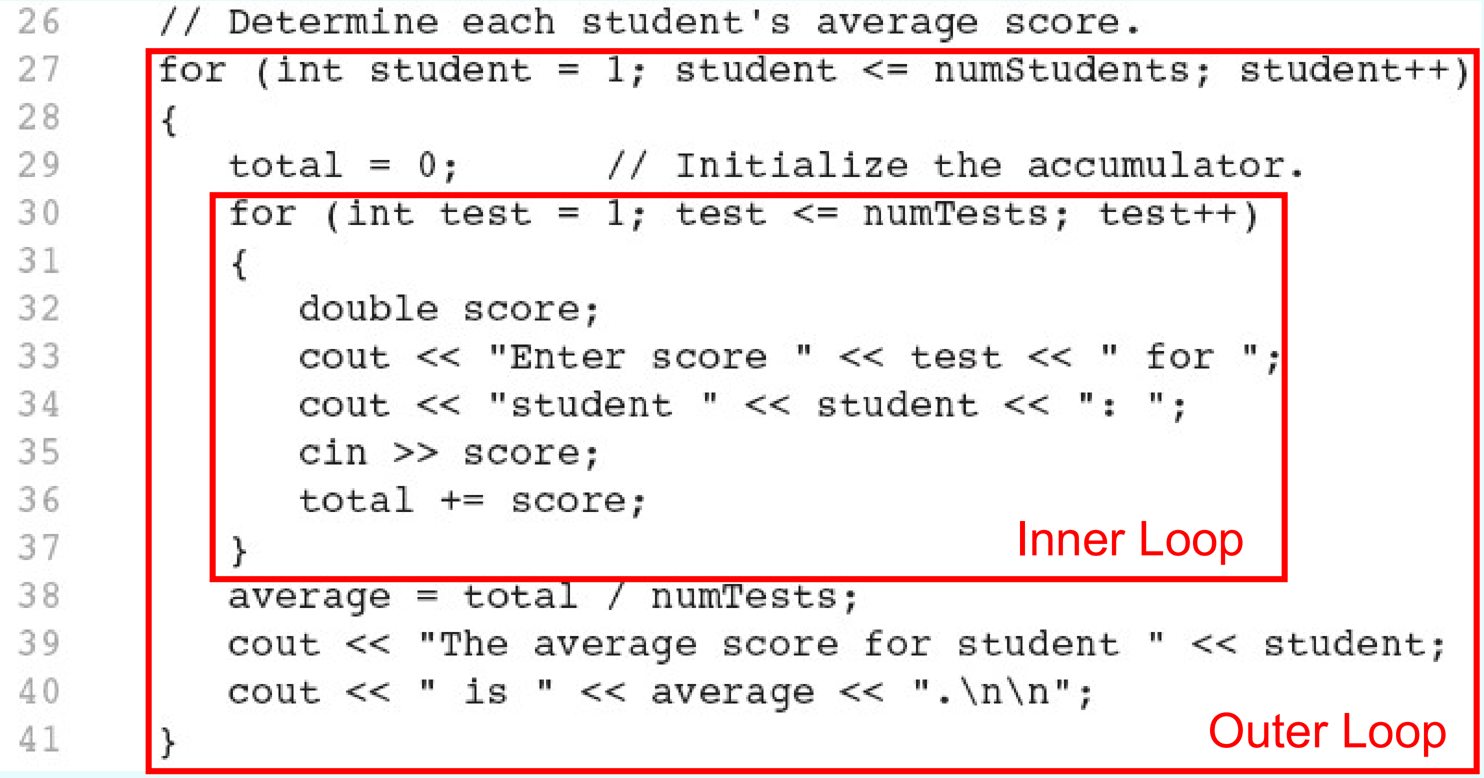
- Can use `continue` to go to end of loop and prepare for next repetition
  - `while`, `do-while` loops: go to test, repeat loop if test passes
  - `for` loop: perform update step, then test, then repeat loop if test passes
- Use sparingly – like `break`, can make program logic hard to follow

Nested Loops

- A nested loop is a loop inside the body of another loop
- Inner (inside), outer (outside) loops:

```
for (row=1; row<=3; row++)  //outer
 for (col=1; col<=3; col++)//inner
   cout << row * col << endl;
```

Nested `for` Loop in Program 5-14



```
// Determine each student's average score.
for (int student = 1; student <= numStudents; student++)
{
   total = 0;        // Initialize the accumulator.
   for (int test = 1; test <= numTests; test++)
   {
      double score;
      cout << "Enter score " << test << " for ";
      cout << "student " << student << ": ";
      cin >> score;
      total += score;
   }
   average = total / numTests;
   cout << "The average score for student " << student;
   cout << " is " << average << ".\n\n";
}
```

Nested Loops - Notes

- Inner loop goes through all repetitions for each repetition of outer loop
- Inner loop repetitions complete sooner than outer loop
- Total number of repetitions for inner loop is product of number of repetitions of the two loops.

THE END