Students:
Section 4.10 is a part of 1 assignment: **CSC108 CH04.7-4.13 P4B**
Includes: PA
Due: 03/13/2025, 11:59 PM EDT

Please browse to this assignment through BlackboardLearn so zyBooks knows where to send your activity. Learn more

4.10 Variable name scope

Scope of names

A declared name is only valid within a region of code known as the name's **scope**. Ex: A variable userNum declared in main() is only valid within main(), from the declaration to main()'s end.

Most of this material declares variables at the top of main() (and if the reader has studied functions, at the top of other functions). However, a variable may be declared within other blocks too. A **block** is a brace-enclosed { ... } sequence of statements, such as found with an if-else, for loop, or while loop. A variable name's scope extends from the declaration to the closing brace }.

PARTICIPATION ACTIVITY 4.10.1: Variable name scope extends to the end of the declaration's block.

Start 2x speed

```
#include <iostream>
using namespace std;

int main() {

   // int val1 = userNum;    // ERROR
   int userNum = 2;          // Name valid to main's "}"
   int newNum = userNum + 1;
   int i;

   for (i = 0; i < newNum; ++i) {
      int valSquared;        // Name valid to for's "}"
      valSquared = userNum * userNum;
      cout << i << "^2 squared: " << valSquared << endl;
   }

   // cout << "Last value: " << valSquared << endl; // ERROR

   return 0;
}
```

Captions

Feedback?

PARTICIPATION ACTIVITY 4.10.2: Variable name scope.

Refer to the animation above.

1) userNum can be used in newNum's declaration.
   - True
   - False

2) If uncommented, userNum can be used in val1's declaration.
   - True
   - False

3) userNum can be used within the for loop's block of statements.
   - True
   - False

4) valSquared can be used within the for loop's block.
   - True
   - False

5) valSquared can be used in the for loop's loop variable update, such as replacing ++i by i = i + valSquared.
   - True
   - False

6) valSquared can be used just before main's return statement
   - True
   - False

Feedback?

For loop index

Programmers commonly declare a for loop's index variable in the for loop's initialization statement. That index variable's scope covers the other parts of the for loop, up to the for loop's closing brace. The reason is clear from the for loop's equivalent while loop code shown below, noting the braces around the equivalent code.

Table 4.10.1: Index variable declared in a for loop's initialization statement.

for loop	Equivalent while loop

`for (int i = 0; i < 5; ++i) {`
`  x = x + i;`
`}`
 	
`x = x + i;  // ERROR`
	`{`
	`  int i = 0;`
	`  while (i < 5) {`
	`  x = x + i;`
	`  ++i;`
	`  }`
	
	`x = x + i;  // ERROR`

Feedback?

The approach of declaring a for loop's index variable in the for loop's initialization statement makes clear that the variable's sole purpose is to serve as that loop's index.

This material avoids declaring index variables in for loops

This material's authors have found that declaring all variables first, then using those variables in the rest of the code, can simplify learning for students. Thus, this material avoids late declarations of variables, including declaring index variables in for loops. With that said, declaring index variables in for loops is extremely common and considered good practice by many programmers, and thus is something to consider, if one can do so without confusion.

PARTICIPATION ACTIVITY 4.10.3: For loop index declared in loop's initialization statement.

Given the following for loop, determine whether index i's scope includes the indicated region.

```
(a)
for (int i = 0; (b); (c) ) {
   (d)
}
(e)
```

1) (a)
   - Yes
   - No

2) (b)
   - Yes
   - No

3) (c)
   - Yes
   - No

4) (d)
   - Yes
   - No

5) (e)
   - Yes
   - No

6) Suppose the above for loop is followed by a second for loop also with int i = 0 in the initialization statement. Will the compiler generate an error due to two declarations of i?
   - Yes
   - No

Feedback?

Common error

A common error is to declare a variable inside a loop whose value should persist across iterations. Below, the programmer expects the output to be 0, 1 (0+1), 3 (0+1+2), 6 (0+1+2+3), and 10 (0+1+2+3+4), but instead the output is just 0, 1, 2, 3, 4.

Figure 4.10.1: Common error: A variable declared within a loop block is (unexpectedly) re-initialized every iteration.

```
#include <iostream>
using namespace std;

int main() {
   int i = 0;

   while (i < 5) {
      int tmpSum = 0;
      tmpSum = tmpSum + i; // Logic error: Sum is always just i
      cout << "tmpSum: " << tmpSum << endl;
      i = i + 1;
   }

   return 0;
}
```

```
tmpSum: 0
tmpSum: 1
tmpSum: 2
tmpSum: 3
tmpSum: 4
```

Feedback?

PARTICIPATION ACTIVITY 4.10.4: Common error of a variable declared within a loop block being reinitialized every iteration.

Given the following code, indicate j's value at the specified point.

```
for (int i = 0; i < 5; ++i) {
   int j = 0;

   j = j * i;
}
```

1) At the end of iteration i = 0.

   Check Show answer

2) At the end of iteration i = 1.

   Check Show answer

3) At the end of iteration i = 2.

   Check Show answer

4) After the loop terminates, can j be output? Type yes or no.

   Check Show answer

Feedback?

How was this section? Provide section feedback

Activity summary for assignment: CSC108 CH04.7-4.13 P4B
Due: 03/13/2025, 11:59 PM EDT

0 / 36 points

Please browse to this assignment through BlackboardLearn so zyBooks knows where to send your activity. Learn more

Completion details