4.4 For loops

Students:
Section 4.5 is a part of 2 assignments: CSC108 CH04.1-4.6 CAA
Please browse to this assignment through Blackboard,Learn so zyBooks knows where to send your activity. Learn more
Includes: CA
Due: 09/11/2025, 11:59 PM EDT

4.5 More for loop examples

Example: Finding the max

Analyzing data is a common programming task. A common data analysis task is to find the maximum value in a list of values. A loop can achieve that task by updating a max seen so far variable on each iteration.

Figure 4.5.1: Finding the max in a list.

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

// Outputs max of list of integers
// Prior value initializes list size
// Array size 5 0 2  prior 7

int main() {
  int maxVal;
  int currValue;
  int numValues;
  int i;

  cin >> numValues;

  for (i = 0; i < numValues; ++i) {
    cin >> currValue;

    if (i == 0) {  // First iteration
      maxVal = currValue;
    }
    else if (currValue > maxVal) {
      maxVal = currValue;
    }

    if (numValues > 0) {
      cout << "Max: " << maxVal << endl;
    }
  }

  return 0;
}
```

```
4 10 0 2 5
Max: 10

Max: 10
5 -140 -90 -10 -80 -30
Max: -10
```

Feedback?

PARTICIPATION ACTIVITY 4.5.1: Finding the max.

Consider the example above.

1) Before entering the loop, what is the maximum value seen so far from the list of integers?
- 0
- 7
- No such value

2) The loop's first iteration gets the list's first integer into variable currValue. Is that the maximum value seen so far?
- Yes
- No

3) For each iteration after the first iteration, the comparison _____ is checked.
- maxValue > currValue
- currValue > maxValue
- currValue == maxValue

Feedback?

Beyond iterating N times

The three parts of a for loop may be adjusted to do more than just iterate N times. For example, a for loop can output various sequences. The following outputs multiples of 5 from 10 to 50.

Figure 4.5.2: Outputting multiples of 5 from 10 to 50.

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

// Outputs 10 15 20 25 30 35 40 45 50

int main() {
  int i;

  for (i = 10; i <= 50; i = i + 5) {
    cout << i << " ";
  }

  cout << endl;

  return 0;
}
```

```
10 15 20 25 30 35 40 45 50
```

Feedback?

PARTICIPATION ACTIVITY 4.5.2: For loops beyond iterating N times.

Type the output of the for loop. Whitespace matters, including after the last item.

Example:

```
for (k = 1; k <= 5; ++k) {
  (for i to output, followed by a space)
}
```

Outputs:
1 2 3 4 5

1) `for (i = 0; i < 5; ++i) {`
`  (for i to output, followed by a space)`
`}`

Check Show answer

2) `for (i = 0; i <= 5; ++i) {`
`  (for i to output, followed by a space)`
`}`

Check Show answer

3) `for (i = 0; i <= 10; i = i + 2) {`
`  (for i to output, followed by a space)`
`}`

Check Show answer

4) `for (i = 10; i >= 0; i = i - 2) {`
`  (for i to output, followed by a space)`
`}`

Check Show answer

5) `for (i = 5; i >= 5; ++i) {`
`  (for i to output, followed by a space)`
`}`

Check Show answer

6) `for (i = 0; i < 5; ++i) {`
`  (for 2 * i to output, followed by a space)`
`}`

Check Show answer

Feedback?

Example: Outputting a table of temperatures

Programs are sometimes used to auto-generate data tables. The following program generates a table of Celsius and Fahrenheit temperature values, in increments of 5.C. The for loop counts from -10 to 40, in increments of 5, and names the loop variable curr; rather than i for more descriptive.

Figure 4.5.3: Auto-generate a data table: Celsius to Fahrenheit.

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

int main() {
  int curr;
  double degreesCentF;

  for (curr = -10; curr <= 40; curr += 5) {
    degreesCentF = (curr * 9.0 / 5.0) + 32.0;

    cout << curr << " ";
    cout << degreesCentF << "\n";
    cout << endl;

  return 0;
}
```

```
-10 14.0
-5 23.0

0 32.0

5 41.0
10 50.0
15 59.0
20 68.0
25 77.0
30 86.0
35 95.0
40 104.0
```

Feedback?

PARTICIPATION ACTIVITY 4.5.3: For loop generating a table of temperature values.

Consider the example above.

1) What is the loop variable's name?

Check Show answer

2) What are the values of curr? for the first four iterations?
Type as 1 2 3 4

Check Show answer

3) What is the loop expression? (The expression checked for whether to enter the loop body).

Check Show answer

Feedback?

Loop style issues

Starting with 0

Programmers in C, C++, Java, and other languages have generally standardized on looping N times by starting with i = 0 and checking for i < N, rather than by using i = 1 and i <= N. One reason is due to other constructs (arrays / vectors), often used with loops, start with 0. Another is simply that a choice was made.

The ++ operators

The ++ operator can appear as ++i (**prefix form**) or as i++ (**postfix form**). ++i increments i first and then evaluates the result, while i++ evaluates the result first and then increments i. The distinction is relevant in a statement like x = ++i;, x = i++; if i is 5, the first yields x = 6, the second x = 5.

Some consider ++i safer for beginners in case they type i = ++i, which typically works as expected (whereas i = i++ does not), so this material uses ++i throughout. The - operator also has prefix and postfix versions. Incidentally, the C++ programming language gets its name from the ++ operator, suggesting C++ as an increment or improvement over its C language predecessor.

In-loop declaration of i

Variables can be declared throughout code, so many programmers use `for (int i = 0; i < N; ++i)`. But, the teaching experience of this material's authors suggests such declarations may confuse learners who may declare variables within loops, repeatedly re-declaring variables, etc. This material avoids the in-loop declaration approach. The authors hope to make the learning less error-prone, and have confidence that programmers can easily pick up on the common in-loop declaration approach later.

PARTICIPATION ACTIVITY 4.5.4: Miscellaneous for loop and ++ topics.

1) Do these loops iterate the same number of times?

```
for (i = 0; i < 5; ++i)
  (...)

for (k = 1; k <= 5; ++k)
  (...)
```

- Yes
- No

2) Does this for loop iterate 5 times?

```
for (k = 0; k < 5; k = k + 1)
  (...)
```

- Yes
- No

3) Is the following valid code?

```
for (int k = 0; k < 5; ++k)
  (...)
```

- Yes
- No

Feedback?

Common errors / good practice

A common error is to also have a ++i statement in the loop body, causing the loop variable to be updated twice per iteration.

Figure 4.5.4: Common error: loop variable updated twice.

```
// Loop variable updated twice per iteration
for (i = 0; i < 5; ++i) {
  cout << i << " ";
  ++i;  // Oops
}
```

Feedback?

While the initialization and update parts of a for loop can include multiple statements separated by a comma, good practice is to use a single statement for each part. Good practice also is to use a for loop's parts to count the necessary loop iterations, with nothing added or omitted. The following loop examples should be avoided, if possible.

Figure 4.5.5: Avoid these for loop variations.

```
// initializationExpression not related to counting iterations: mone y = rand() %mod%mod loop
for (i = 0; a < mone; ++i) {  a = rand() %mod
  (...) // loop body
}

// updateExpression not related to counting iterations: mone = * 2 * 2 into loop body
for (i = 0; i < 5; ++i, a = a * 2) {
  (...) // loop body
}

// conditionExpression not related to counting iterations: mone = * 2 * 2 into loop body
for (i = 0; a < 5; ++i) {  a = a * 2;
  (...) // loop body
}
```

Feedback?

PARTICIPATION ACTIVITY 4.5.5: For loop: Common errors / good practice.

1) Putting ++i at the end of a for loop body, in addition to in the updateExpression part, yields a syntax error.
- True
- False

2) The above two for loop variations each yield a syntax error.
- True
- False

Feedback?

CHALLENGE ACTIVITY 4.5.1: For loops.

zyDE 4.5.1: For loops

This activity failed to load. Please try refreshing the page. If that fails, you might also try

clearing your browser's cache.

If an issue persists, send feedback to zyBooks support

CHALLENGE ACTIVITY 4.5.2: Find extreme value.

zyDE 4.5.2: Find extreme value.

Start

Write a for loop that iterates five times to find the minimum value in a list of five integer values. Within the loop:

- Read an integer value from input into variable inputVal.
- The first value assigned to inputVal is the smallest integer value read by default, since no other values have been read yet. Thus, if the current iteration is the first iteration, assign minimumVal with inputVal.
- Otherwise, if inputVal is less than minimumVal, assign minimumVal with the value of inputVal.

Click here for example

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

int main() {
  int inputVal;
  int minimumVal;
  int i;

  /* Your code goes here */

  cout << minimumVal << endl;

  return 0;
}
```

1 2

Check Next level

Feedback?

CHALLENGE ACTIVITY 4.5.3: All values fit a category?

zyDE 4.5.3: All values fit a category?

Start

Integer inputVal is read from input representing the number of integers to be read next. Use a for loop to read in the remaining integers from input into val. If all integers read from input are negative, assign allNegative with true. Otherwise, assign allNegative with false.

Code at the end of main() outputs "All match!" if allNegative is true, or "Not all match!" if allNegative is false.

Ex: If the input is:

```
3
-100 -92 -55
```

then the output is:

```
All match
```

Note: Negative integers are less than zero.

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

int main() {
  int numIn;
  bool allNegative;
  int i;
  int val;

  allNegative = true;
  cin >> numIn;

  /* Your code goes here */

  if (allNegative) {
    cout << "All match!" << endl;
  }
  else {

  }
```

1 2

Check Next level

Feedback?

How was this section? Provide section feedback

Activity summary for assignment: CSC108 CH04.1-4.6 CAA
Due: 09/11/2025, 11:59 PM EDT

0 / 36 points

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

Completion details