Looping

Victor Eijkhout, Susan Lindsey

Fall 2024
last formatted: August 28, 2024

For loops

1. 'For' statement

Sometimes you need to repeat a statement a number of times.
That's where the loop comes in. A loop has a counter, called
a loop variable, which (usually) ranges from a lower bound to an
upper bound.

Here is the syntax in the simplest case:

```cpp
// loop/sumsquares.cpp
int sum_of_squares{0};
for (int var=low; var<upper; ++var) {
  sum_of_squares += var*var;
}
cout << "The sum of squares from "
     << low << " to " << upper
     << " is " << sum_of_squares << '\n';
```

2. Loop syntax: variable

The loop variable is usually an integer:

```
for ( int index=0; index<max_index; index=index+1) {
  ...
}
```

But other types are allowed too:

```
for ( float x=0.0; x<10.0; x+=delta ) {
  ...
}
```

Beware the stopping test for non-integral variables!

Exercise 1

Write a loop that prints $x = 1/10, 2/10, \ldots, 1$. Do this

- with an integer loop variable
- with a `float` or `double` loop variable

How do you do the stopping test?
What do you observe?

3. Nested loops

Traversing a matrix
(we will discuss actual matrix data structures later):

```
for (int row=0; row<m; row++)
  for (int col=0; col<n; col++)
    ...
```

This is called 'loop nest', with
row: outer loop
col: inner loop.

4. Indefinite looping

Sometimes you want to iterate some statements not a predetermined number of times, but until a certain condition is met. There are two ways to do this.

First of all, you can use a 'for' loop and leave the upper bound unspecified:

```
for (int var=low; ; var=var+1) { ... }
```

5. Break out of a loop

This loop would run forever, so you need a different way to end it.
For this, use the break statement:

```
for (int var=low; ; var=var+1) {
  statement;
  if (some_test) break;
  statement;
}
```

Exercise 2

The integer sequence

$$u_{n+1} = \begin{cases} u_n/2 & \text{if } u_n \text{ is even} \\ 3u_n + 1 & \text{if } u_n \text{ is odd} \end{cases}$$

leads to the Collatz conjecture: no matter the starting guess u_1, the sequence $n \mapsto u_n$ will always terminate at 1.

$$5 \to 16 \to 8 \to 4 \to 2 \to 1$$
$$7 \to 22 \to 11 \to 34 \to 17 \to 52 \to 26 \to 13 \to 40 \to 20 \to 10 \to 5 \cdots$$

(What happens if you keep iterating after reaching 1?)

Try all starting values $u_1 = 1, \ldots, 1000$ to find the values that lead to the longest sequence: every time you find a sequence that is longer than the previous maximum, print out the starting number.

Breaking out of a loop

6. Where did the break happen?

Suppose you want to know what the loop variable was when the break happened. You need the loop variable to be global:

```
int var;
... code that sets var ...
for ( ; var<upper; var++) {
  ... statements ...
  if (some condition) break
  ... more statements ...
}
... code that uses the breaking value of var ...
```

In other cases: define the loop variable in the header!

7. Test in the loop header

If the test comes at the start or end of an iteration, you can move it to the loop header:

```cpp
bool need_to_stop{false};
for (int var=low; !need_to_stop ; var++) {
  ... some code ...
  if ( some condition )
    need_to_stop = true;
}
```

Exercise 3

Write an i, j loop nest that prints out all pairs with

$$1 \leq i, j \leq 10, \quad j \leq i.$$

Output one line for each `i` value.

Now write an i, j loop that prints all pairs with

$$1 \leq i, j \leq 10, \quad |i - j| < 2,$$

again printing one line per `i` value. Food for thought: this exercise is definitely easiest with a conditional in the inner loop, but can you do it without?

Optional exercise 4

Find all triples of integers u, v, w under 100 such that $u^2 + v^2 = w^2$.
Make sure you omit duplicates of solutions you have already found.

Exercise 5

Write a double loop over $0 \leq i, j < 10$ that prints the first pair where the product of indices satisfies $i \cdot j > N$, where N is a number your read in. A good test case is $N = 40$.

Secondly, find a pair with $i \cdot j > N$, but with the smallest value for $i + j$. (If there is more than one pair, report the one with lower i value.) Can you traverse the i, j indices such that they first enumerate all pairs $i + j = 1$, then $i + j = 2$, then $i + j = 3$ et cetera? Hint: write a loop over the sum value $1, 2, 3, \ldots$, then find i, j.

You program should print out both pairs, each on a separate line, with the numbers separated with a comma, for instance $8, 5$.

Suggestive picture 1
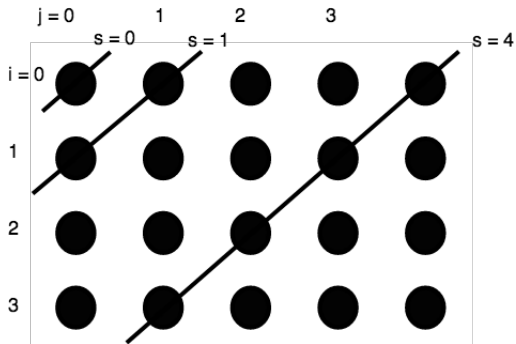
Suggestive picture 2
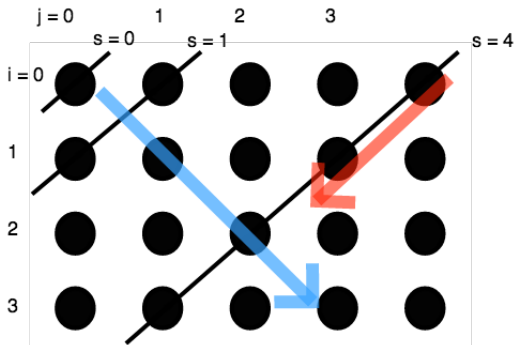
8. Skip iteration

```
for (int var=low; var<N; var++) {
  statement;
  if (some_test) {
    statement;
    statement;
  }
}
```

Alternative:

```
for (int var=low; var<N; var++) {
  statement;
  if (!some_test) continue;
  statement;
  statement;
}
```

The only difference is in layout.

While loops

9. While loop

Syntax:
```
while ( condition ) {
  statements;
}
```

or
```
do {
  statements;
} while ( condition );
```

10. **Pre-test while loop**

```
float money = inheritance();
while ( money < 1.e+6 )
  money += on_year_savings();
```

11. While syntax 1

```
Code:

  // basic/whiledo.cpp
  cout << "Enter a positive
      number: " ;
  cin >> invar; cout << '\n';
  cout << "You said: " << invar
      << '\n';
  while (invar<=0) {
    cout << "Enter a positive
      number: " ;
    cin >> invar; cout << '\n';
    cout << "You said: " <<
      invar << '\n';
  }
  cout << "Your positive number
      was "
      << invar << '\n';
```

```
Output:

Enter a positive
    ↪number:
You said: -3
Enter a positive
    ↪number:
You said: 0
Enter a positive
    ↪number:
You said: 2
Your positive number
    ↪was 2
```

Problem: code duplication.

12. While syntax 2

```
Code:

// basic/dowhile.cpp
int invar;
do {
  cout << "Enter a positive
    number: " ;
  cin >> invar; cout << '\n';
  cout << "You said: " <<
    invar << '\n';
} while (invar<=0);
cout << "Your positive number
    was: "
    << invar << '\n';
```

```
Output:

Enter a positive
    ↪number:
You said: -3
Enter a positive
    ↪number:
You said: 0
Enter a positive
    ↪number:
You said: 2
Your positive number
    ↪was: 2
```

The post-test syntax leads to more elegant code.

Optional exercise 6

A horse is tied to a post with a 1 meter elastic band. A spider that was sitting on the post starts walking to the horse over the band, at 1cm/sec. This startles the horse, which runs away at 1m/sec. Assuming that the elastic band is infinitely stretchable, will the spider ever reach the horse?