

Looping

Victor Eijkhout and Carrie Arnold and Charlie Dey

Fall 2017

Looping

Repeat 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:

```
for (int var=low; var<upper; var++) {  
    // statements involving var  
    cout << "The square of " << var << " is " << var*var << endl;  
}
```

C difference: Use compiler flag -std=c99.

Exercise 1

Read an integer value, and print 'Hello world' that many times.

Loop syntax

- Loop variable is usually an integer.
- The stopping test be any test; can even be empty.
- The increment can be a decrement or something like `var*=10`
- Any and all of initialization, test, increment can be empty:

```
for(;;) ...
```

- (The loop variable can be defined outside the loop:

```
int var;
```

```
for (var=low; var<upper; var++) {
```

```
but it's cleaner to make it local.)
```

Nested loops

Traversing a matrix:

```
for (int i=0; i<m; i++)  
    for (int j=0; j<n; j++)  
        ...
```

Exercise 2

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

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

Same, but

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

Exercise 3

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.

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 upperbound unspecified:

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

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 4

Write a double loop over $0 \leq i, j < 10$ that prints the first pair where the product of indices satisfies $i \cdot j > 80$.

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;  
}
```

While loop

The other possibility is a *while* loop, which repeats until a condition is met.

Syntax:

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

or

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

The while loop does not have a counter or an update statement; if you need those, you have to create them yourself.

While syntax 1

```
cout << "Enter a positive number: " ;  
cin >> invar;  
while (invar>0) {  
    cout << "Enter a positive number: " ;  
    cin >> invar;  
}  
cout << "Sorry, " << invar << " is negative" << endl;
```

Problem: code duplication.

While syntax 2

```
do {  
    cout << "Enter a positive number: " ;  
    cin >> invar;  
} while (invar>0);  
cout << "Sorry, " << invar << " is negative" << endl;
```

More elegant.

Exercise 5

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.

For $u_1 < 1000$ 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.

Exercise 6

One bank account has 100 dollars and earns a 5 percent per year interest rate. Another account has 200 dollars but earns only 2 percent per year. In both cases the interest is deposited into the account.

After how many years will the amount of money in both accounts be the same?

Project Exercise 7

Read an integer and determine whether it is prime by testing for the smaller numbers whether they are a divisor of that number.

Print a final message

Your number is prime

or

Your number is not prime: it is divisible by

where you report just one found factor.

Project Exercise 8

Rewrite the previous exercise with a boolean variable to represent the primeness of the input number.