```
n = 0;
sum = 0;
while (n < 10) {
  scanf("%d", &i);
  if (i == 0)
    continue;
  sum += i;
  n++;
  /* continue jumps to here */
}
```

If `continue` were not available, we could have written the example as follows:

```
n = 0;
sum = 0;
while (n < 10) {
  scanf("%d", &i);
  if (i != 0) {
    sum += i;
    n++;
  }
}
```

The `goto` Statement

`break` and `continue` are jump statements that transfer control from one point in the program to another. Both are restricted: the target of a `break` is a point just *beyond* the end of the enclosing loop, while the target of a `continue` is a point just *before* the end of the loop. The `goto` statement, on the other hand, is capable of jumping to *any* statement in a function, provided that the statement has a ***label***. (C99 places an additional restriction on the `goto` statement: it can't be used to bypass the declaration of a variable-length array.)

C99

variable-length arrays ➤8.3

A label is just an identifier placed at the beginning of a statement:

labeled statement

identifier : *statement*

A statement may have more than one label. The `goto` statement itself has the form

goto statement

`goto` *identifier* ;

Executing the statement `goto` *L*; transfers control to the statement that follows the label *L*, which must be in the same function as the `goto` statement itself.

If C didn't have a `break` statement, here's how we might use a `goto` statement to exit prematurely from a loop:

```
for (d = 2; d < n; d++)
  if (n % d == 0)
    goto done;
```