while-loop

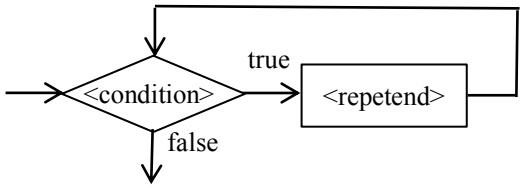
The syntax of the while-statement, or while-loop,is:

while (<condition>) <repetend>

where

1. The <condition> is a boolean expression.
2. The <repetend[1]> is any statement — either a single statement or a <block>

The following flow chart shows how the while-loop is executed.



As an example, the for-loop on the left can be rewritten as the while-loop on the right.

(1)
```
int k;
for (k= 0; k < 10; k= k+1) {
    System.out.println(k);
}
```

```
int k= 0;
while (k < 10) {
    System.out.println(k);
    k= k+1;
}
```

The following two loops are equivalent. They do not terminate unless the repetend causes termination.

```
for ( ; ; ) some-statement;
```

```
while (true) some-statement
```

1. Use a for-loop when there is an obvious counter variable (as in for-loop (1) above). If there isn't, a while-loop is usually a better choice.
2. It is possible to use the break statement in the repetend. Its execution immediately terminates execution of the for-loop. We advise against this. Changing control using a break statement makes it harder to reason about the loop. If possible, restructure to avoid using it.
3. Execution of a continue statement within the repetend terminates execution of the repetend, so that the condition is evaluated next.
4. Loops are best understood (and developed) using loop invariants. See the tutorials on program correctness and loop invariants that are associated with this list of definitions and concepts.

[1] *Repetend* means *the thing to be repeated.* In the 1980's, a 13-year old who was studying Gries's book "The Science of Programming" used the term in an email. From then on, we have used that word.