Review – Counted Loops

Our first loop was a “counted loop”, where we did *something* a fixed number of times.

```
for count = 1 to 5          // this is pseudocode
```

We also learned to take larger steps, and count upwards or downwards.

```
for count = 10 to 100 by steps of 10

for count = 100 to 10 by steps of 10
```

Review – Counted Loops

The “For” loop is actually a special case of a more general loop.

```
count := 1
loop
  count := count + 1
  exit when (count > 10)
end loop
```

Our exit from the loop depends on (count > 10), so this is called the “exit condition”.

Exit Conditions

An exit condition is just like the conditions we use in the *if-then-else* statements. For example,

if (age >= 16)

if (name = “Fred”)

if (age >= 16) and (name = “Fred”)

Just like the *if* statements, the exit conditions can be based on numbers or strings.

Conditional Loops

As we allow more conditions (rather than just counting up or down), the looping in our programs becomes much more powerful.

On the other hand, it also has the potential to become much more complicated. It is even more important to understand what you expect your program to do before you start writing code.

Don't underestimate the value of designing a program on paper first!

Conditional Loops
Exit Condition Placement

For consistency and readability, the exit condition will only appear at the very beginning or very end of the loop.

```
loop
    exit when (condition)
    do something
end loop
```

```
loop
    do something
    exit when (condition)
end loop
```