Small Basic

Conditional Loops
In a previous exercise, you used a ‘for’ loop – a loop which depends on a counter. This kind of loop is used where the number of iterations is known before you enter the loop (for example, calculating interest for a given number of years, displaying the 7 times table between 1 and 12, etc).

Many other times, an instruction must be repeated until a condition changes and the number of iterations is NOT known before the loop starts. An example may be entering a password (we don’t know how many attempts the user will need), or reaching a target value.

Conditional loops can (and often do) include counters, but are controlled by checking a variable against a value.

The code sample below asks the user for a password, and allows an infinite number of re-tries:

```smallbasic
TextWindow.WriteLine("Enter the password")
password=TextWindow.Read()
While password<"abc"
    TextWindow.WriteLine("Incorrect. Try again")
    password=TextWindow.Read()
Endwhile
TextWindow.WriteLine("Good work")
```

Notes
• The syntax for a conditional loop is
  ```smallbasic
  While <condition>
      'remember to change the condition
  Endwhile
  ```
• The condition is the same as for an ‘if’ statement – usually a variable compared to a value
• The comparison operators are the same: >, <, >=, <=, <> , =
• It is OK to compare strings for equality (as in the example above)
• It is usually best to use inequalities (<, <=, >, >=) when comparing numbers (why?)
• ALWAYS provide a way to change the loop control variable INSIDE the loop (why?)

Conditional Loop with a Counter?
Quite often we need a condition AND a counter, for example – you may want to limit the number of password attempts that someone makes.

A counter is created simply by incrementing a variable inside the loop:

```smallbasic
n = n + 1
```

This apparently impossible statement works from right to left – the computer adds 1 to the value of ‘n’, then stores it back in the same location.
In this way, a counter can be added to the previous loop to become:

```smallbasic
TextWindow.WriteLine("Enter the password")
password=TextWindow.Read()
attempts=0
While password<>"abc" and attempts <=3
    TextWindow.WriteLine("Incorrect. Try again")
    password=TextWindow.Read()
    attempts = attempts +1
EndWhile
If attempts<=3 then
    TextWindow.WriteLine("Correct password")
Endif
```

**Pick the errors:**

**Example 3**

```smallbasic
n=1
While n=10
    n=n+1
    TextWindow.WriteLine(n)
EndWhile
```

**Example 4**

```smallbasic
savings=12
While savings<100
    totalsavings  = savings + totalsavings
EndWhile
TextWindow.WriteLine(totalsavings)
```

**Answers:**

**Example 3** – this loop will never start.

**Explanation:** the condition n=10 is false the very first time, so the loop is never run. The line n = n+1 is never executed.

**Example 4** – this loop will never end.

**Explanation:** the loop control variable (savings) is never changed.

The condition should have been

```smallbasic
While totalsavings<100
```
**Exercise**

Chapter 5 in the “Microsoft Small Basic” text (which comes with Small Basic) is about loops. For revision, re-read this chapter.

1. Re-create the password example (above) and test it.
2. Copy the two examples (3 and 4) above and correct the mistakes.
   - the first one should show the numbers between 1 and 10,
   - the second should show how many weeks it would take to save up 100 dollars, if 12 dollars was saved each week
3. Make a guessing game. Generate a random number (see code below). Ask the user to guess a number between 1 and 10. Use a loop to count the number of attempts (the condition will be
   
   ```
   while guess<>target
   ```
   
4. Imagine you had $1000 and invested it at compound interest at 7%. How many years would it take for the original amount to reach 1 million dollars?
5. Create a program that adds a series of numbers (the user will enter 0 to signal the end of the number series). You should display the total only.
   - Also calculate the average of all the numbers