THE WHILE AND DO-WHILE LOOP

The **while** statement is a looping structure that executes a set of statements while a *boolean expression* is true. If the expression is false, the loop is exited. The *boolean expression* is checked before the statements in the loop execute. The **while** statement takes the following form:

```
while (boolean expression)
{
      Statements you want to loop
}
```

The **do-while** statement is a almost identical to the while statement, except that the *boolean expression* is checked *after* the statements in the loop execute. Also, notice that a semi-colon is included after the while statement in a **do-while** loop. The **do-while** statement takes the following form:

```
do
{
      Statements you want to loop
}
while (boolean expression);
```

Example 1: In this case, 10 alerts in all will be displayed showing the numbers 1 to 10.

```
var counter = 1;

while(counter <= 10)
{
      alert(counter);
      counter++;
}
```

Notice that the counter variable needs to be declared BEFORE the loop is created. This is because any variable you create INSIDE of the loop cannot be checked in the while condition. The following code would NOT work because counter was declared INSIDE of the loop.

```
while(counter <= 10) //does not work!
{
      var counter = 1;
      alert(counter);
      counter++;
}
```

Example 2: Here is an example of a do-while loop that will accomplish the same task as Example 1.

```
var counter = 1;

do
{
      alert(counter);
      counter++;
}
while(counter <= 10);
```

INFINITE LOOPS

Infinite loops can also be created in while loops as well. The following loops would be infinite. Can you see why in each case?

```
var counter = 1;

while(counter <= 10)
{
      alert(counter);
}
```

```
var counter = 11;

while(counter >= 10)
{
      alert(counter);
      counter++;
}
```

SENTINEL PATTERN

The sentinel pattern in computer science is used to determine when a loop will terminate (stop iterating). The sentinel value is a variable that is used to accomplish this. A certain value (*true/false*) will indicate the loop should continue to run. When the sentinel value changes, the loop will terminate. The sentinel value changes when the loop has accomplished the particular task it is being used for.

```
var finished = false; //this variable will store the sentinel value
var sum = 0;
alert("This program will calculate the sum of the given numbers.  Enter 0 to terminate.");

while (!finished) //this line literally reads as "while NOT finished"
{
      input = Number(prompt("Please enter a number"));
      if (input == 0)
        finished = true; //this changes the sentinel value in order to exit the loop
      else
        sum = sum + input;
}

alert("Your sum is " + sum);
```

TRACING A LOOP

When using loops, a useful skill is to be able to trace a loop as it runs. This means keep track of values relevant to the loop as they change to help understand how the loop is working. Here is an example:

Loop	**Trace Steps to show each Calculation**	

var num1, num2, answer;
num1 = 11;
num2 = 5;
while (num1 > num2)
{
answer = num1 – num2;
alert(answer);
num1 = num1 - 3;
}
	1. num1 = **11**
	2. num2 = **5**
	3. num1 > num2 = 11 > 5 is true
	4. answer = 11 – 5 = **6**
	5. num1 = 11 – 3 = **8**
	6. num1 > num2 = 8 > 5 is true
	7. answer = 8 – 5 = **3**
	8. num1 = 8 – 3 = **5**
	9. num1 > num2 = 5 > 5 is false
	10. loop ends
		Notice that the tracing the code involves recording the execution of each line and keeping track of the variable values.
		
		This skill is useful because it allows you to manually check your code to understand how it is working, or to find a bug if one exists.

ADDITIONAL INFORMATION

For additional information about for loops, feel free to check out https://www.w3schools.com/js/js_loop_while.asp.

CHECK FOR KNOWLEDGE

1. Why would you use a for loop over a while loop? Give an example.

2. Trace the following pieces of code and describe:
   a) What is the value of num1, num2 and answer when the loop begins?
   b) What are the values of answer on the first, second and last pass of the loop?
   c) What is the value of num1 when the loop stops – if it does stop?

var num1, num2, answer;
num1 = 1;
num2 = 10;
while (num1 < num2)
{
answer = num2 – num1;
alert(answer);
num1 = num1 + 2;
}	

var num1, num2, answer;
num1 = 0;
num2 = 20;
do
{
answer = num2 – num1;
alert (answer);
num1 = num1 + num1;
}
while (num1 < num2);	

var num1, num2, answer;
num1 = 10;
num2 = -10;
while (num1 < num2)
{
answer = num2 – num1;
alert (answer);
num1 = num1 + num1;
}	

var num1, num2, answer;
num1 = -10;
num2 = 10;
while (num1 < num2)
{
answer = num2 – num1;
alert (answer);
num1 = num1 + num1;
}	

PRACTICE EXERCISES

1. **PASSWORD CHECKER:** Create a function that creates a secret password and repeatedly asks the user to enter the correct password until they are successful.
2. **LETTERS:** Use a loop to count the number of letters in a sentence. (letters only, not #,special char or spaces).
3. **HACKER SPEAK:** Turn a sentence into h4ck3r sp34k! A lot of people on the internet like to replace certain letters with numbers that look like letters. For example, 4 looks like an 'A', 3 looks like an 'E', 1 looks like an 'I' and 0 looks like an 'O'. Create a function that takes in a sentence from the user and outputs the converted sentence as **h4ck3r sp34k!**
4. **PIG LATIN** is a 'language' used by many small children to encrypt their conversations. It works like this:

   Words that begin with a vowel have 'ay' concatenated at the end.

   eg. over -> overay, under -> underay

   Words that begin with a consonant have the consonant moved to the end, and 'ay' tacked on.

   eg. hello -> ellohay

   Write a program that will accept a word, and convert it to Pig Latin. For a 4++ have it convert sentences as well.