## Looping Structures

### while

while (*condition*)
{
    *code block to be executed*
}

```
int i;
while (i < 10)
{
    cout << "The number is " << i << endl;
    ++i;
}
```

### do

do
{
    *code block to be executed*
}
while (*condition*);

```
int i;
do
{
    cout << "The number is " << i << endl;
    ++i;
}
while (i < 10);
```

### for

for (*statement 1; statement 2; statement 3*)
{
    *code block to be executed*
}

```
for (int i = 0; i < 10; ++i)
{
    cout << "The number is " << i << endl;
}
```

## **break** and **continue**

- `break` and `continue` can be used to manipulate flow in loops and other structures.
- `break` will break out of the loop, e.g.

```
for (int i = 0; i < 10; i++)
{
```

```
        if (i == 7)
        {
            break;
        }
        cout << "The number is " << i << endl;
    }
```

- `continue` will continue the next iteration of the loop, e.g.

```
    for (int i = 0; i < 10; i++)
    {
        if (i == 7)
        {
            continue;
        }
        cout << "The number is " << i << endl;
    }
```

- previously you've seen `break` used in a `switch` to break out of the switch structure.

### Counting in C, or Zero-based Numbering

- When counting programmers start at 0, not 1.
- This is due to "zero-based numbering" where the initial element of a sequence is assigned the index 0 rather than 1. The initial element is sometimes termed the *zeroth* element.
- The mathematics of programming is generally cleaner and more efficient when starting at 0.
- As with the above `for` example, `for (int i = 0; i < 10; ++i)`, yes, you could start at 1, but then the loop begins to get messy as,
  `for (int i = 1; i <= 10; ++i)` or
  `for (int i = 1; i < 11; ++i)`
- When we look at arrays in particular, you'll notice that zero-based numbering makes programming flow much smoother.
- As 0 is often a valid indicator of position, some functions will return -1 to indicate no position.

## Fibonacci Example

- In mathematics, the Fibonacci numbers are the numbers in the following integer sequence, called the Fibonacci sequence, and characterized by the fact that, every number after the first two is the sum of the two preceding ones:
  0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, …
- The sequence is named after Italian mathematician Leonardo of Pisa, known as Fibonacci, who was considered "the most talented Western mathematician of the Middle Ages." He is best known for popularizing Hindu–Arabic numerals in the Western World.
- In his book, "Liber Abaci" (1202) Fibonacci considers the growth of a rabbit population, assuming that: a newly born pair of rabbits, one male, one female, are put in a field; rabbits are able to mate at the age of one month so that at the end of its second month a female can produce another pair of rabbits; rabbits never die and a mating pair always produces one new pair (one male, one female) every month from the second month on. The puzzle that Fibonacci posed was: how many pairs will there be in one year?
- A good application of a loop in C++ is a program to calculate Fibonacci's problem,

```
int nTerms;
cout << "Enter number of Fibonacci terms: ";
cin >> nTerms;

cout << endl;
cout << "First " << nTerms << " terms are-" << endl;
int first = 0, second = 1, next;
for (int i = 0; i < nTerms; ++i)
{
      if (i <= 1)
      {
            next = i;
      }
      else
      {
            next = first + second;
            first = second;
            second = next;
      }
      cout << next << endl;
}
```

- Fibonacci numbers appear unexpectedly often in mathematics, so much so that there is an entire journal dedicated to their study.
- Fibonacci numbers are often seen in biological settings, such as branching in trees, the arrangement of leaves on a stem, the fruit sprouts of a pineapple, the flowering of an artichoke, an uncurling fern and the arrangement of a pine cone's bracts.