Using Vectors in C++

Vectors are very useful for storing and referencing data in C++. Let's say we want to take used input on a program for *N* apples, from 1-5, and output the number (word) and numeral for what they asked for. This might look like

```
Please enter the number of apples you would like: 4
You would like four (4) apples.
```

This can be done using if-else statements, as before. A neater solution is to use a vector where we have stored the words for the numbers, and just reference the needed location.

Start by including the vector library in the header:

```
#include <vector>
```

In the main body of the code, we can now create a new vector in a similar way we have created new variables in the past:

```
vector <string> numbers;
```

What this line is doing is this: we have created a vector that will contain strings, and we will call it numbers. *Notice that we have said nothing so far about how long that vector is!* We can now start storing values into the vector, using the `push_back()` function. This will take whatever argument we pass to it at "push it to the back" of our vector (that is, append it to the end). Say we have a vector `a = [1,2,3]`. If we call `a.push_back(4)`, the vector will become `a = [1,2,3,4]`. In our case, we will use this to add in the words for our numbers. **Remember that vector indexing starts at 0!**

```
numbers.push_back("zero");
numbers.push_back("one");
...
numbers.push_back("five");
```

We now have a vector of our strings, which we can reference and use the same way we would use any other variable! E.g., `numbers[1] = one`. What's important to note here is that `push_back()` is creating a new spot in the computer's memory to store data. We cannot right now just store variables to new locations, i.e. `numbers[7] = "seven"`. This location does not exist in the memory, and the computer will not know what to do!

The downside to this method is that the elements must be added in order, and the memory is extended every time. What if we already know how long the vector should be? For our example, we know that we want a total of six elements. We can then declare our vector as:

```
vector <string> numbers(6);
```

We have now told the computer that we are creating a vector of strings called numbers that will have 6 elements. Those elements are now blocked off for us in memory, and we can use them as we choose! Let's store what we would like into the vector:

```
numbers[0] = "zero";
numbers[1] = "one";
...
numbers[5] = "five";
```

We now have the same vector we had before! Now say we want to pass this vector to a function. It can be done similarly to how other variables are passed:

```
void function(vector <string> numbers){
        We can now use the numbers vector in this function
}
```

For the complete examples from class, see vector.cpp and vector2.cpp. These are attached here, and can also be accessed in my public CRC directory, ~zmiksis/Public/SciComp.