Employing vector arrays

A vector is an alternative to a regular array, and has the advantage that its size can be changed as the program requires. Like regular arrays, vectors can be created for any data type, and their elements are also numbered starting at zero.

In order to use vectors in a program, the C++ vector library must be added with an `#include <vector>` preprocessor directive at the start of the program. This library contains the predefined functions in the table below, which are used to work with vectors:

<table>
<thead>
<tr>
<th>Function:</th>
<th>Description:</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>at( number )</code></td>
<td>Gets the value contained in the specified element number</td>
</tr>
<tr>
<td><code>back()</code></td>
<td>Gets the value in the final element</td>
</tr>
<tr>
<td><code>clear()</code></td>
<td>Removes all vector elements</td>
</tr>
<tr>
<td><code>empty()</code></td>
<td>Returns true (1) if the vector is empty, or returns false (0) otherwise</td>
</tr>
<tr>
<td><code>front()</code></td>
<td>Gets the value in the first element</td>
</tr>
<tr>
<td><code>pop_back()</code></td>
<td>Removes the final element</td>
</tr>
<tr>
<td><code>push_back( value )</code></td>
<td>Adds a final element to the end of the vector, containing the specified value</td>
</tr>
<tr>
<td><code>size()</code></td>
<td>Gets the number of elements</td>
</tr>
</tbody>
</table>

A declaration to create a vector looks like this:

```cpp
vector < data-type > vector-name ( size );
```

An `int` vector will, by default have each element automatically initialized with a zero value. Optionally, a different initial value can be specified after the size in the declaration, with this syntax:

```cpp
vector < data-type > vector-name ( size, initial-value );
```

The functions to work with vectors are simply appended to the chosen vector name by the dot operator. For example, to get the size of a vector named “vec” you would use `vec.size()`.
Start a new program by specifying the C++ library classes to include, and a namespace prefix to use
```cpp
#include <vector>    // Include vector support.
#include <iostream>
using namespace std;
```

Add a main function containing a final `return` statement
```cpp
int main()
{
    // Program code goes here.
    return 0;
}
```

In the main function, insert a statement to declare and initialize a vector array of three elements of the value 100
```cpp
vector<int> vec(3, 100);
```

Now, insert statements to manipulate the vector elements
```cpp
cout << "Vector size: " << vec.size() << endl;
cout << "Is empty?: " << vec.empty() << endl;
cout << "First element: " << vec.at(0) << endl;
vec.pop_back();   // Remove final element.
vec.clear();      // Remove all elements.
vec.push_back(200);  // Add an element.
```

Save, compile, and run the program to see the output
```bash
C:\MyPrograms>gcc vector.cpp -o vector.exe
C:\MyPrograms>vector
Vector size: 3
Is empty?: 0
First element: 100
Vector size: 2
Final element: 100
Vector size: 0
Vector size: 1
First element: 200
C:\MyPrograms>
```