QUEEN MARY, UNIVERSITY OF LONDON

MTH6150

Numerical Computing in C and C++

Exercise Sheet 7

1. Run the code on slide 7 of lecture 7.

   Then change the program in the following ways, one at a time, to practise the syntax for a `vector`, and each time output the elements of the `vector` to the screen:

   - Change number of elements in the `vector` from 5 to 3.
   - Change the number of elements to 0.
   - Create a `vector` of length 5 with each element equal to -1.
   - Use the syntax at the bottom of slide 10 to create a `vector` with elements $1, 2, 3, 4$.
   - Do the same as the previous part, but by initially creating a `vector` of size 0 and then adding each element one at a time using `.push_back(...)`, which is illustrated on slide 11.
   - Resize the `vector` to size 8.
   - Remove the last element.
   - Clear the `vector` (i.e. give it size 0).

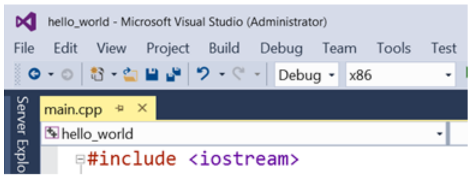
2. Write code to generate the first n Fibonacci numbers, as defined in exercise sheet 4, but this time storing them in a `vector<int>`. Storing the values in a vector should make the code more straightforward compared to the version that did not do this.

   Output the contents of the vector to the screen.

3. Write a program to output the first n prime numbers, where n is an input integer. Store the prime numbers in a `vector<int>`, and make use of the stored values to make checking whether the next integer is prime or not more efficient.

4. Run the code on slide 13 of lecture 7 to see what happens when you run the program. Check that the message you when running the program makes sense.

   So far in Visual Studio we have been running projects in Debug mode. Change the project settings to Release mode. This is done by selecting Release in the white drop-down box that in the image below shows as Debug (next to the box saying x86).


hello_world - Microsoft Visual Studio (Administrator)
File Edit View Project Build Debug Team Tools Test
Debug
x86
main.cpp
hello_world
Server Explo

```
#include <iostream>
```

   Compile (i.e. build) the code again after this change, and run it.

Also, try creating a `vector` of size 0, and then using the syntax `pop_back()` to remove the last element, as on slide 12. Do this both in Debug and Release mode.

In exercise sheet 2 question 3, you ran code in which a variable was used without being given a value. Add code to output `y` to the screen, and run the program after compiling it both in Debug and Release mode.

Also run the code on slide 7 (again). After you have compiled it, look in the Error List, which you can see by clicking View -> Error List. Is there a warning there? If it warns about a signed/unsigned mismatch, this is because in `i<v.size()`, `i` is an `int`, which is signed, i.e. it can hold negative as well as non-negative values, whereas `v.size()` is an unsigned integer type, i.e. it can only hold non-negative values. This warning will go away if we change the project type to 64-bit (from 32-bit), which can be done using the drop-down menu to the right of the Debug/Release menu. Change x86 to x64.