Module IN3013/INM173 – Object-Oriented Programming in C++

Solutions to Exercise Sheet 1

1.

2.

3.

4. Counting the number of words in the input:

```
#include <iostream>
#include <string>

using namespace std;

int main() {
        int count;
        string s;
        count = 0;
        while (cin >> s)
                count++;
        cout << count << '\n';
        return 0;
}
```

   Note that the `main` function returns an `int`: this is not output, but gives the status of the program: 0 indicates success and anything else indicates failure. Usually this doesn't matter, but sometimes the status is used.

5. The idea with these exercises was that you were not to modify the `getline` function, just use it. In fact `getline` is provided by the `iostream` library.

   Counting the number of lines in the input is similar to the last exercise, except that we read lines using `getline`:

```
#include <iostream>
#include <string>

using namespace std;
```

```
int main() {
        int count;
        string s;
        count = 0;
        while (getline(cin, s))
                count++;
        cout << count << '\n';
        return 0;
}
```

6. To print the longest line of the input, we need only store the longest line seen so far. The `string` type has a member function `size()` that returns the length of the string.

```
#include <iostream>
#include <string>

using namespace std;

int main() {
        string longest;
        string s;
        while (getline(cin, s))
                if (s.size() > longest.size())
                        longest = s;
        cout << longest << '\n';
        return 0;
}
```

   Note that the assignment `longest = s` involves a copy of strings, so we are free to overwrite `s` with the next line.

7. To print the input lines in reverse order, we need to store all the lines in a `vector` of `string`s. We start with an empty `vector`, and add the `string` containing each line to the end using `push_back`.

```
#include <iostream>
#include <string>
#include <vector>

using namespace std;
```

```
int main() {
        vector<string> v;
        string s;
        while (getline(cin, s))
                v.push_back(s);
        for (int i = v.size()-1; i >= 0; i--)
                cout << v[i] << '\n';
        return 0;
}
```

Note that `push_back` stores a copy of the string in the vector, so we can safely overwrite `s` later.

Members of the vector are referenced with the usual array notation `v[i]`.

As with `string`, the `vector` type has a member function `size()` that returns the number of elements in the vector.