MTH6150

Numerical Computing in C and C++

Exercise Sheet 2

1. Either open the project you created in the first session; or follow the steps again to create a project, and then add a code file to the project.

2. A minimal example of declaring and initializing a variable and outputting it to the screen is as follows:

```
#include <iostream>
using namespace std;

int main() {
        int k = 3;
        cout << k << endl;
}
```

   Type in this code, or your own similar example, into the code file in your project, then compile and run it.

   Change the above code so that `k` is initialized on one line and then given a value later. Also, try it with type `double` instead of `int`.

3. Try making use of a variable which has not been given a value, for example as follows:

```
double x;
double y = x*2;
cout << y << endl;
```

   Is there anything in the error window when compiling the program? Remember, if you can't see the error window, click View -> Error List.

   If the program compiles, what happens when it runs?

4. Run the program with each of the expressions on slide 12, outputting the result each time, and check that you understand the difference between them.

5. Type in the code on the slide "Prefix and postfix operators", then add code to output `m` and `n`. Run this code.

   Repeat the above, but modifying the code by replacing `++` with `--`.

6. Write code that uses the operators on the slide "Changing a single variable" to do the following, with each part as a separate command:

   - Declare a variable of type `double` and initialize it with a value of 2.0
   - Add 4 to it.
   - Multiply it by 3.
   - Subtract 6.

- Divide it by 4.
- Output the final value of the variable, and check that this is what you expect.

7. Type in the code on the slide "Testing for equality" and output the value of `b` in both ways shown there.

   Once we have used `cout << boolalpha << ...` once in the program, subsequent commands which output `bool` variables will also output as true/false rather than 0/1.

8. Which of `a,b,c,d,e,f,g,h` below evaluates to true and which to false? Add each line of code one at a time, and note down what you think each answer will be before adding code to output the relevant variable and running the code.

```
int m = 6;
int n = 4;
bool a = false;
bool b = m==3;
bool c = n>m && m<7;
bool d = m != n;
bool e = (m!=3) == (n<m);
bool f = !e || (3<2);
bool g = a || b && c || d;
bool h = (a || b) && (c || d);
```

   Experiment with variations on these expressions to make sure you understand how they work.

9. Include the following code in your program and then run it:

```
double x = 2000;
cout << x*x << endl;
```

   The notation that you see in the output can also be used for input. Either 2E6 or 2e6 can be used to mean 2×10^6 (E stands for exponent). Check that you can use this and that the result is what you expect by including a line such as the following:

```
double x = 2E6;
```

10. Run the code on slide 6 to see the amount of memory that one `int` uses, and the minimum and maximum values that can be stored. From the minimum and maximum values, calculate how many different values can be stored. Is this consistent with the amount of memory that one `int` uses? Remember, one byte is 8 bits, where each bit can be 0 or 1.

    A different integer variable type that uses more memory is `long long`. Repeat the above with this data type.

11. Try running the following code:

```
int n = numeric_limits<int>::max();
cout << n << endl;
n = n+1;
cout << n << endl;
```

    Can you explain what is happening?

12. As we have seen in session 1, when outputting to the screen, we need to explicitly tell the program if we want a space or a new line. To output two numerical variables `x` and `y` with a space between them we can do this:

```
cout << x << " " << y << endl;
```

Alternatively, we can use a tab character, which will be useful later when outputting to a file:

```
cout << x << "\t" << y << endl;
```

And an alternative to `endl` is:

```
cout << x << "\t" << y << "\n";
```

`\t` and `\n` can also be used within longer pieces of text:

```
cout << "Hello\tworld\n";
```

Experiment with these both with numbers and text. Also try mixing numbers and text as in:

```
cout << "The value of x is " << x << endl;
```

13. Run the following code:

```
double m = 5;
double n = 4;
cout << m/n << endl;
```

Now change the variable type of `m` and `n` to `int` and run the code again.

If `m` and `n` are `int` variables then `m/n` is also interpreted as an `int` variable, which can cause unexpected problems. We will see other examples of this later, and how to guard against errors.