CSCI251/CSCI851 Autumn-2020
Advanced Programming (**LT5**)

Lecture Tutorial 5

From the Lab

- Read the output of what you are producing:
- Debug-A:
  - A negative value for the number of fangs (teeth) doesn't really make sense.
  - Mitten should have been kitten, a young cat.
    - This isn't the easiest for non-native English speakers.

- Debug-B:
  - A fair few people missed picking up that the following doesn't test the number of arguments, it changes them…

```
if (argc = 4)
```

  - Changing this to == isn't enough anyway because that isn't what you actually want as a value.
  - The code said …

```
cout << "The next three values should be the same!" << endl;
```

  - For a fair few people they weren't the same.

cout vs cerr: buffered vs non-buffered.

- Using Cerr we see one element printed at a time, with a clear time gap between them, and some backspacing to correct ourselves.
- With Cout there is just a long pause then a statement is printed.
- Cout is buffered, this means values are stored and only output when they are flushed.
  - So those backspacing changes are taking place in the buffer.
- Cerr doesn't buffer, every symbol goes straight to the output.

Union …

- The task illustrates the idea that a union allocates enough space to store any of the data elements, but it uses the same space to store whichever we want.
- You can read and write as if the content is of any of the different types, but you are likely to run into trouble if you write as one type and read (interpret) it as another.

Namespaces

- Name clashes is the problem we want to avoid.
- If you use

```
using namespace X;
```

- … then everything in your current scope needs to avoid clashing with elements already in X.