2. Lab Exam

This is the lab test for Week 6 of COMSM1201. It is to be attempted `14:00 - 15:15`. (Longer if the Faculty has approved extra-time for you in examinations).

- You will attempt two questions. Each question is marked as pass or fail. No partial marks are available. Possible marks for this exam are therefore 0%, 50% or 100%.
- This examination is "open book". You may refer to textbooks, lecture notes, manual pages and source code previously written by yourself.
- Do not discuss this examination with others, either in person or online. If you have reason to believe someone has been colluding or cheating, contact one of the unit TAs or the Unit Director.
- All successful submissions will be checked for plagiarism.
- TAs will be available in their normal channels, but can only offer advice on logging in, submitting etc. They can't help with programming, debugging, or letting you know whether your submission is working or not.
- Any major announcements during the exam will be made via the Teams "General" Forum, so keep an eye on it. If I've made a mistake, or a clarification is needed, it'll be posted there.

Included in the `.zip` file are two string questions a1 and a2, and two integer array questions b1 and b2. For every question, there is a `main` file and a `func` file. The `main` files are not to be altered - they should be considered to be "read-only". You write your code in the corresponding `func` file. If, for instance, you are asked to attempt question a1, you will submit a `func_a1.c` file, which we will test by compiling it against the `main_a1.c` file:

```
gcc main_a1.c func_a1.c -o a1 -Wall -Wextra -Wfloat-equal
-pedantic -std=c90 -fsanitize=address -fsanitize=undefined
-g3 -lm
```

This should compile without any compiler warnings, and on execution show no output if the assertion tests run correctly.

A file `common.h`, which includes some standard `typedefs` and `#includes`, is also given.

We will also use other test files and assertions to check your code; this is mainly to prevent people "hard-wiring" a function in response to the assertions in `main()` - the changes we make to any assertions will keep the testing similar to what you have been given.

Which Questions are Mine?

Everyone attempts two questions: an "a" question (`a1` or `a2`) and a "b" question (`b1` or `b2`)
Your University username looks something like: `kb20345` or `nc17568`.
If the first letter of your username is in the range `a-m` (i.e. **k**`b20345`) attempt question `a1`, else attempt question `a2`.
If the final number of your username is even $0, 2, 4, 6, 8$ (i.e. `nc1756`**8**) then attempt question `b1`, else attempt question `b2`.

How to Submit

Make sure when you submit your files that they have the correct name i.e. either `func_a1.c` or `func_a2.c`, and either `func_b1.c` or `func_b2.c`. These files will contain only one function, have no `main()` etc. You will submit either one or two `func*.c` files.
Due to a quirk in Blackboard, every time you submit, all your old submissions are archived somewhere that we can't easily get to. Therefore, you can submit as many times as you like, but we will only mark the files that form part of your **final** submission. So, every time you submit, submit everything you'd like assessed.

Submission must be completed before the end of the examination. Submissions after the deadline will not be considered. [1].

---

[1] Allison reckons my use of "a" and "b" remind her of Dr. Seuss, which puts me in mind of his most famous poem : *This is no good. This is not right. Warnings pop out of my code all night.* Maybe you read it as a child ?