Assignment 1: the 4C process

1. **C**`REATE`: Create a directory, say `ass1`, download all related files into `ass1`, then create `ass1/ass1.c` that satisfies the requirements ☺
2. **C**`OPY`: Copy the whole directory ass1 to your university's drive `H:`. Note: if you work in lab computers, you don't need to do this step.
3. **C**`HECK`: login into the server `dimefox.eng.unimelb.edu.au`, then on that server, check/test to make sure that your program is correct.
4. **C**`OMMIT`: while in `dimefox`, commit/submit your `ass1.c`, and verify.

Assignment 1: the 4C process

1. **C**REATE: Create a directory, say `ass1`, download all related files into `ass1`, then create `ass1/ass1.c` that satisfies the requirements ☺

In `minGW` or `Terminal` window, when you are in your `COMP20005` (or similar) directory, do:

```
mkdir ass1
```

```
cd ass1
```

then, download all needed files from `LMS` → `Assignment1` to this directory. That includes 12 files listed in point 5 of `LMS`→`Assignment1`.

Now, it's time to build `ass1.c` (you can choose other name).

Assignment 1: the 4C process

2. COPY: Copy the whole directory ass1 to your university's drive `H:`. Note: if you work in lab computers, you don't need to do this step.

Skip this step if you are working on a lab's PC. Otherwise, supposing that `ass1` is your current directory. You need:

`cd ..`	make parent of ass1 the current directory
`scp –r ass1 XXX@dimefox.eng.unimelb.edu.au:`	copy whole directory to your H:

Notes:

- replace `XXX` by your uni's login name
- there is a colon `:` at the end of `scp`
- if you do that outside uni, you need to install and run `VPN` first (instructions available in LMS)

Assignment 1: the 4C process

3. CHECK:

login into `dimefox:` from your `minGW/Terminal,` type:

`ssh XXX@dimefox.eng.unimelb.edu.au`

and answer as needed. You will see your prompt changed to

`bash $`

now, use ls and cd to navigate into your **`ass1`** directory and compile:

`bash $gcc –Wall –o ass1 ass1.c`

Then, test your program against, say, pedestrians-melbcent-p100.tsv:

`bash $./ass1 < pedestrians-melbcent-p100.tsv > o1.txt`

that will write output to o1.txt. You can compare that with Alistair's one by

`bash $diff o1.txt pedestrians-melbcent-p100-out.txt`

which lists the differences between 2 files. No output means the 2 files are exactly the same, bravo.

Assignment 1: the 4C process

4. COMMIT: while in `dimefox`, submit your `ass1.c`, and verify.

To submit you must be on `dimefox`, and `ass1` must be your current directory. Use the command:

```
bash $ submit comp20005 ass1 ass1.c
```

After that you can verify your submission using:

```
bash $ verify comp20005 ass1 > receipt.txt
```

It's a good idea to open `receipt.txt` with `jEdit` to see its content. To be able to use `jEdit`, on `receipt.txt`, you first need to copy that file to your local computer if it's not a Lab's one:

```
bash $ exit
$ scp XXX@dimefox.eng.unimelb.edu.au:ass1/receipt.txt .
```

Assignment 1: the 4C process

Today Work: minimal requirement
Create a simple .c file (perhaps empty, perhaps just reading the data), then try all other 3 steps. Make sure that you can submit, at least from a lab PC.
Then, incrementally **CREATE** your ass1.c, do **COPY**-**CHECK**-**COMMIT** after every major development.

Assignments: advices

- *Be active in the subject's Discussion Forum!*
- *Visit* **LMS→Assignment 1** *frequently!*
- *Make as many submissions as you want, only the last one (before deadline) counts.*
- *To simplify, do commit at uni. If you want to commit from home, beware that it might be complicated!*
- *Read the specifications carefully.*
- *Check your program carefully, at least with all supplied data. Do the testing on* `dimefox`*.*
- *Read the marking rubric carefully and try to maximize your marks!*
- *Skim the sample solution to 2015 (in LMS.Assignment1, point 6), focusing on* `main()`*. You can learn something from there.*
- *START EARLY, START RIGHT NOW!*