UNIVERSITY OF SOUTHAMPTON — ELECTRONICS AND COMPUTER SCIENCE

COMP1204 Data Management: Coursework One

> *"A baby has brains, but it doesn't know much. Experience is the only thing that brings knowledge, and the longer you are on earth the more experience you are sure to get."* — The Wonderful Wizard of Oz.

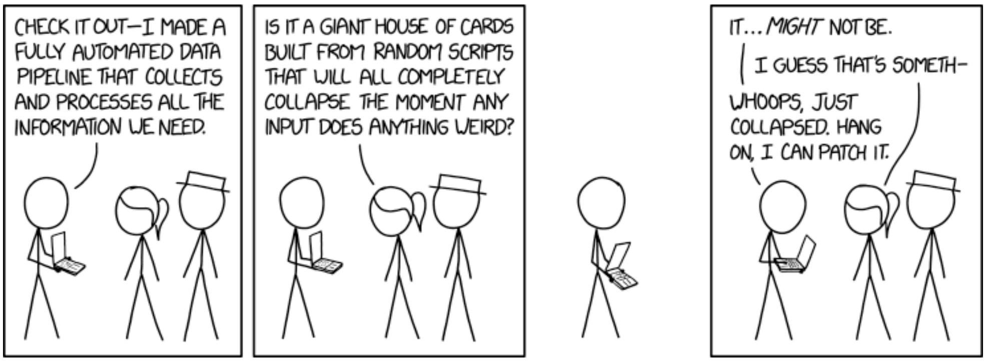

CHECK IT OUT—I MADE A FULLY AUTOMATED DATA PIPELINE THAT COLLECTS AND PROCESSES ALL THE INFORMATION WE NEED.
IS IT A GIANT HOUSE OF CARDS BUILT FROM RANDOM SCRIPTS THAT WILL ALL COMPLETELY COLLAPSE THE MOMENT ANY INPUT DOES ANYTHING WEIRD?
IT... MIGHT NOT BE.
I GUESS THAT'S SOMETH-
WHOOPS, JUST COLLAPSED. HANG ON, I CAN PATCH IT.

Figure 1: Data is only as useful as the pipeline that provides it: `https://xkcd.com/2054/`

Data is often all over the place. It comes in many different forms, and has many weird quirks — often it cannot be directly used for any real purpose. **Enter data cleaning:** the process of making data behave and manifest in a usable format. **This is your task** — take some (rather horrible) looking data and change it into an easy to use, accessible format that is actually useful. **Our data of choice? Hurricane monitoring!** (*"Toto, I've a feeling we're not in Kansas anymore"*[1]). In this task, you are given `kml` files[2], and we ask you to turn them into (rather more usable) `csv` files containing only the information that we really care about. However, we require that you show your working (via `git`), and also provide us with a summary of your work (via a written report). **Fear not!** We have provided some pieces of helper code to produce some (lovely looking) plots, and all of your code will be automatically tested upon submission, such that you will receive near instant feedback[3].

[1] In the original plot of the 1939 film, the storm was actually a tornado (`https://en.wikipedia.org/wiki/The_Wizard_of_Oz_(1939_film)`). However, we are computer scientists not meteorologists, therefore we will not discuss this difference any further.

[2] What even is a `kml` file you ask? Try opening it in any normal text editor.

[3] Provided you check your email!

1 OVERVIEW

This coursework will cover three key topics that will have been introduced in the first three weeks of the module: Unix, Latex, and Git. The coursework is divided into three parts: (i) Unix scripting for file processing, (ii) report writing using Latex, (iii) use of Git for version control. Each part carries a percentage of the marks for this coursework as detailed in Table 1. You should be able to complete various parts of this coursework as we go through the lectures, and parts of this assignment will be covered during lab sessions. Help will also be available throughout the labs. The key points are:

- This coursework is worth 10% of this module.
- The deadline for submission of your scripts and report is **15th March 2022 by 4pm**[4].
- Feedback will be given within 2 weeks after the deadline.
- You are only allowed to use Bash (Unix) commands and Latex (Overleaf and other editors are acceptable). Use of other scripting languages or document formats (e.g., Microsoft Word) will get you zero marks for the relevant sections.
- For FAQs, please see: `https://secure.ecs.soton.ac.uk/noteswiki/w/COMP1204/2022/FAQ`

For submission instructions, please see Section 4 at the end of this document.

Unix scripting for file processing	80%
Report Writing in Latex	10%
Use of Git for version control	10%

Table 1: The weighting given to the different parts of this coursework.

2 LEARNING OUTCOMES

This problem sheet aims to achieve the following learning outcomes:

- Knowledge of Unix commands including but not limited to sed, awk, grep and pipes.
- Knowledge of Latex commands and document preparation.
- Data cleaning techniques using pattern matching and filtering.
- Use of Git during code development for version control and management.

3 THE ASSIGNMENT

You work as a data scientist for the National Oceanographic and Atmospheric Administration Centre[5]and you've recently been assigned to the tropical cyclone tracking team. You have been tasked with extracting storm data from tropical cyclone reports and producing maps of where the storm is.

[4] Standard ECS late submission penalities will be applied.

[5] `https://www.nhc.noaa.gov/index.shtml`

3.1 Project Setup

Before you start, you need some way of tracking your work and managing your code. Create a new project on `https://git.soton.ac.uk`. The project should be named comp1204-<userid>-cwk1. So if your userid is je5g15, the project should be named comp1204-je5g15-cwk1. **Make sure the visibility level of the project is set to private. Now invite user je5g15 as Maintainer on the project.** As your work on this project, make sure to regularly commit your work to your `git` project.

You can find the files you need for this coursework here: `https://secure.ecs.soton.ac.uk/notes/comp1204/coursework/cw1_material.tar.gz`. This contains the data files, a Latex report template, and a plotting script to create storm maps. Download this file and extract it using standard UNIX file decompression commands.

3.2 Unix file processing

We need you to extract the useful data contained in a set of hurricane reports. These reports are quite verbose and contain lots of extra information — we only want the important parts. The information we're after is:

- Storm Latitude: The North/South position of the storm.
- Storm Longitude: The East/West position of the storm.
- Minimum Sea Level Pressure: The minimum air pressure of the storm, measured at sea level. A drop in pressure usually indicates an oncoming storm, and the greater the drop in pressure, the bigger the storm[6]. This is usually measured using millibars (mb) or inches of mercury (in Hg) — **we'd like you to report the pressure in mb.**
- Maximum Intensity: The maximum velocity of the winds in the storm. This can be measured in knots, mph, or kph — **we'd like you to report the intensity in knots.**

This information is contained in the provided reports at regular intervals, i.e., there are several UTC timestamps (time and date) with associated positions, pressures, and intensities. We would like you to gather the requested information at each timestep, and output it in a `csv` file. The `csv` file should be formatted like this:

```
Timestamp,Latitude,Longitude,MinSeaLevelPressure,MaxIntensity
0000 UTC SEP 07,16.9 N,-41.3 W,1006 mb,30 knots
0600 UTC SEP 07,17.0 N,-41.8 W,1006 mb,30 knots
1200 UTC SEP 07,17.1 N,-42.1 W,1005 mb,35 knots
1800 UTC SEP 07,17.3 N,-42.3 W,1005 mb,35 knots
0000 UTC SEP 08,17.6 N,-42.4 W,1004 mb,40 knots
0600 UTC SEP 08,17.9 N,-42.6 W,1000 mb,45 knots
1200 UTC SEP 08,18.3 N,-43.0 W,997 mb,50 knots
1800 UTC SEP 08,18.6 N,-43.7 W,997 mb,50 knots
```

From this output, it is easy to see a few things: we have regular reports every six hours, the storm has been moving north-west, and it is building (decreasing sea pressure and increasing wind speed). This is much easier to see in the `csv` file than in the original report. Please note this is only example data and is not taken from the reports you've been provided with!

[6] `https://secoora.org/education-outreach/hurricanes/hurricane-glossary/`

Unix script breakdown:

1. Your script should be named `create_csv.sh` and should take two arguments: an input path (to a `kml` file) and an output path (where the `csv` file will be created), like so:

   ```
   % ./create_csv.sh report.kml storm_info.csv
   Converting report.kml -> storm_info.csv...
   Done!
   ```

   Note your script doesn't actually need to have any command line output — all we care about is the `csv` file. However, it can be helpful to print messages to the command line for debugging purposes.

2. You can assume that our test harness will always provide your script with exactly two arguments, and that they will always be valid and correct paths (i.e., you don't need to sanitise the inputs, but this is something that you would normally have to do!).

3. The `csv` file created by your script should be in the exact same format as the example above. It should contain a single header line detailing the five columns of data (Timestamp, Latitude, Longitude, MinSeaLevelPressure, and MaxIntensity), followed by rows of data extracted from the `kml` file.

4. The data in your `csv` file should appear in the same order as it was in the `kml` file, i.e., it is not necessary to reorder the data in any way[7]. We will check the contents of your `csv` files against that of our solution — they should match exactly!

Note that we ask you not to submit any `csv` files. Instead, you submit your Unix script, and our test harness will run your script against several `kml` files. You are provided with three `kml` files for developing your script, but we also have additional, unseen `kml` files that your script will be tested against (so you can't hard code anything!). See Section 4 for more details on the submission format.

3.3 Report

To summarise your approach, you need to write a report. We have provided a template `tex` file that you should use to write your report. Your report should detail the following:

1. The first page of the report must have your name and student ID written on it as well as the title. This page does not need to be a title page, i.e., it can also contain actual report content.

2. The report should be divided into three sections:
   - **Introduction:** A short section introducing the aims of the work.
   - **Create CSV Script:** Details of your `create_csv.sh` script. Your script should included **in full** via Latex environments such as `verbatim`, `listings`, or `algorithmic`. Along with your script, we ask you to clearly explain what it is doing. You can provide details of how your script works with code comments as well as an overall textual description, for example:

[7] We're assuming that all of the data in the `kml` files is ordered chronologically, but we haven't actually verified this for every report we have.

```
# Try to leave this place
echo "There's no place like home."
```

- **Storm Plots:** The three output plots generated by the `create_map_plot.sh` script applied to `csv` output files produced by your `create_csv.sh` script:

  ```
  % ./create_map_plot.sh storm_info.csv storm_plot.png
  ```

  Note this requires the files `plot-locations-on-map.gpi` and `world-50m.txt` to be in the **same** location as the `create_map_plot.sh` script. In your report, each plot should come with a caption stating which `kml` file it represents.

3.4 Assessment Criteria

Your submission will be evaluated as follows:

UNIX: For the `create_csv.sh` script, we will use an automated script checker that will pass `kml` files to your code and check the `csv` outputs. Your scripts will be tested on previously unseen `kml` files. In particular, we will check that:

1. The code produces the expected `csv` output.
2. The code is efficient (slow code will be penalised). Note that most solutions will run in a reasonable time; this penalty is included as a safeguard for code that hangs.

Indicative feedback may include: code does not work, code works partially (i.e., only some of the output is correct), code is inefficient (i.e., it times out).

REPORT: For the report we will check the following in particular:

1. Your report is written in Latex and uses the template provided.
2. Your script must be included **in full** using Latex environments such as `verbatim`, `listings`, or `algorithmic`.
3. The first page of the report should contain your student name and ID.
4. The report should be appropriately structured into sections (see above).
5. The script code environment should be appropriately captioned.
6. The report should contain three output figures showing the storm locations (using the provided `create_map_plot.sh` script). These figures should also be captioned.

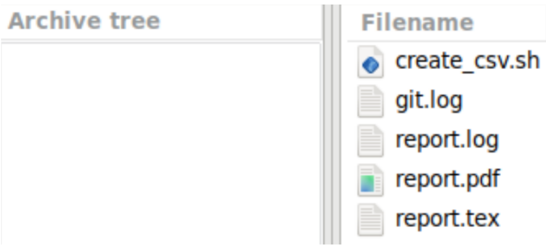
Indicative feedback may include: report not produced in Latex, poorly structured document, script missing, figures missing, poor explanation of script, missing captions.

GIT: We expect to see appropriate use of `git` for version control when writing your scripts and report. Note this is not limited to just developing your script: we expect to see commits concerning the completion of your report and the creation of the final submission file. We will assess if sufficient commits and meaningful commit messages have been provided.

4 Submission Instructions

Submit your work using the ECS electronic hand-in system. The submission is to be made by **4pm** on the due date listed above. Please submit a single file to the ECS electronic hand-in system as detailed below:

- Your submission must be a compressed archive named `comp1204_cw1.tar.gz`.
- Your archive should contain the following files, named exactly as shown:
  - `create_csv.sh`
    Your script to create `csv` files from `kml` files. This is the script that will be assessed in our test handler.
  - `report.pdf`
    Your report exported as a `pdf` file.
  - `report.tex`
    The raw Latex file for your report.
  - `report.log`
    The Latex log file generated when you compile your Latex file (on Overleaf you need to navigate the options to download the log file). The report `log` and `tex` files allow us to verify that you've used Latex to create your report.
  - `git.log`
    A log of your use of git throughout the project, which can be automatically generated by git. We use this to check if you've made a reasonable number of commits with good commit messages.
- Your files should be in the root directory of the archive, otherwise our test harness will be unable to locate them. Please ensure your files are not contained in **any** folders within the archive, like so:



Archive tree

Filename

create_csv.sh
git.log
report.log
report.pdf
report.tex

- Failure to follow these instructions will incur a penalty. In particular, you will lose (possibly all) marks if:
  1. You use Word to create your document.
  2. You compress to a ZIP archive and change the extension of the file.
  3. Your archive does not follow the correct file structure.

Please note that issues such as missing files, incorrect submission structure, etc. will be caught by our automated test harness. You will be informed of these issues via email, and can resubmit your archive any number of times. This means you can submit your `create_csv.sh` script in an archive on its own to test your code prior to writing the report — you will be warned of the missing report files, but your code will still be tested.