School of Computing and Information Systems

comp20005 Engineering Computation
Semester 1, 2019
Assignment 2

Learning Outcomes

In this project you will demonstrate your understanding of structures and arrays of structures, and will develop a computational solution for a non-trivial problem. You are expected to make extensive use of functions; and to demonstrate that you have adopted a clear and elegant programming style. You will find it difficult to create a working solution unless you plan your program carefully in advance, and develop it incrementally.

The Mission

You will again be working with numeric data, this time in connection with rainfall records. All of the data files used in this project have been sourced directly from the Bureau of Meteorology web site at `http://www.bom.gov.au/climate/data/`, and remain in the format in which they were downloaded. The program that you write will identify historical trends that may exist in the input data, and allow visualization of the data.

Stage 1 – Reading the Data (Marks up to 9/20)

In this stage you should read all of the data into internal structures suitable for use in the later stages, and create an output representation that provides an overview of the data that was read. Input will come from a comma-separated-values file, with each input line (after the first header line) looking like:

```
IDCJAC0001,086039,2000,01,28.2,Y
IDCJAC0001,086039,2000,02,34.5,Y
IDCJAC0001,086039,2000,03,22.5,Y
IDCJAC0001,086039,2000,05,96.3,Y
IDCJAC0001,086039,2000,06,42.4,Y
IDCJAC0001,086039,2000,07,45.1,Y
```

recording, for example, that the rainfall measured at site 086039 (Flemington Racecourse, in Melbourne) in January 2000 was 28.2 millimeters. The last value in each line is `Y` or `N` to indicate whether the data has been "validated". To read one line, you should use the format control string

```
"IDCJAC0001,%d,%d,%d,%lf,%c"
```

with suitable receiving variables, and then skip any remaining characters on that line until a newline character has been consumed. Note that `IDCJAC0001` is a fixed string that identifies the type of data, and will appear in all data files that your program will process.

You may assume that the lines in the input are always in ascending-year then ascending-month order, but also need to be aware that there may be missing data lines. For example, there is no data for April 2000 in the test file shown above. Missing values are completely typical of sensor-based data, and occur because of equipment malfunction, network errors, and so on.

A sample of the required output of this stage is:

```
mac: ass2-soln < rainfall-086039-2000-2009.csv
S1, site number 086039, 115 datalines in input
S1, 2000: Jan  Feb  Mar  ...  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
S1, 2001: Jan  Feb  Mar  ...  May  Jun  Jul  Aug  Sep* Oct  Nov* Dec
```

```
    <plus lines for the other seven years>
S1, 2009: Jan  Feb* Mar  Apr  May  Jun* Jul  Aug  Sep  Oct  Nov  Dec
```

where `rainfall-086039-2000-2009.csv` is one of the test files linked from the FAQ page; where `*` indicates a value that has not been validated; and where `...` indicates a completely missing value. Full output examples are linked from the FAQ page. Note that the input might start part way through one year and end part way through another, so be sure that you handle the first and last year correctly. There might also be whole missing years, in which case your program should report every month for that year as a `...` entry.

You should be able to get started on the required program for this stage quite quickly, based on your solution to the first project (or the sample solution to the first project). And an explicit permission – you may include this declaration as a global array if you wish to (and if you can see how it would be useful):

```
char *months[] = {"", "Jan", "Feb", "Mar", "Apr", "May", "Jun",
                      "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"};
```

Stage 2 – The Annual Rainfall Cycle (Marks up to 12/20)

Add further functionality to your program so that for each month of the year the average rainfall is computed and reported. All months for which at least one data item is available should be listed, and unverified items should also be counted and included in the average. A sample of output lines for the same test file is:

```
S2, Jan, 10 values, 2000-2009, mean of  23.5mm
S2, Feb, 10 values, 2000-2009, mean of  45.5mm
S2, Mar,  9 values, 2000-2009, mean of  33.8mm
    <plus lines for the other eight months>
S2, Dec, 10 values, 2000-2009, mean of  48.6mm
```

Full examples are linked from the FAQ page, including showing what should be written if there are no rainfall records in one or more of the months. Note that the monthly averages are required again in Stage 4, so they should be computed in this stage and then retained, rather than being computed again later.

Stage 3 – Climate Change? (Marks up to 16/20)

Ok, now for some trend analysis. Suppose that a sequence of n values $\langle r_0 \ldots r_{n-1} \rangle$ is given (in this case, r_i is the rainfall amount for some specific month across a sequence of years), and we wish to know if there is an overall upward or downward trend over the n values. One simple *correlation coefficient* that can be used is *Kendall's* τ (pronounced "tau"), which for data already ordered on one aspect (in our case, ordered by time) is computed as:

$$\tau = \frac{1}{n(n-1)/2} \cdot \left(\sum_{i=0}^{n-2} \sum_{j=i+1}^{n-1} \delta(r_i, r_j) \right)$$

where $\delta(r_i, r_j)$ is $+1$ if $r_i < r_j$; is -1 if $r_i > r_j$; and is 0 if $r_i = r_j$. A τ value of $+1.0$ indicates perfect monotonic increase across the sequence of values; a τ value of -1.0 indicates perfect monotonic decrease across the sequence of values; and a τ value of 0.0 indicates neither consistent growth nor consistent decrease.

In this stage you are to compute twelve τ values from the input data, one τ for each of the months. Each τ value will be computed from the sequence $\langle r_i \rangle$ that represents the rainfall for one specific month across the span of years available in the data. No τ value should be reported if there are less than two instances of that particular month in the input file, see the FAQ for examples.

```
S3, Jan, 10 values, 2000-2009, tau of -0.11
S3, Feb, 10 values, 2000-2009, tau of -0.11
S3, Mar,  9 values, 2000-2009, tau of -0.11
    <plus lines for the other eight months>
S3, Dec, 10 values, 2000-2009, tau of  0.36
```

Stage 4 – Let It Rain! (Marks up to 20/20)

And now for the fun. Add further functionality to your program so that, for each year number specified as an argument on the command-line, a bar chart is plotted showing the rainfall for that year broken down by months, compared to the average for that month across the years listed in the input data file. One complete graph is to be produced for each year that is specified as a program argument, see the examples linked from the FAQ page. If no arguments are supplied, no graphs are to be generated. To control the height of the graph, a vertical scaling factor should be identified by finding the smallest integer such that the height of the maximum point plotted in the graph above the axis is at most 24 rows.

The example below shows the required output for the year 2003 when using the same sample data file. To make this graph, the largest point to be plotted is identified as being 71.6 mm of rain, and hence a scale factor of $\lceil 71.6/24 \rceil = 3$ is required. Then to get the first bar, January 2003 had a rainfall (in this data file) of 9.8 mm and the January average (in this data file) is 23.5 mm, and so with a scale factor of three, the January average is plotted in the 8 th cell high ($\lceil 23.5/3 \rceil = 8$) and the January actual is plotted from the first to the 4 th cells high ($\lceil 9.8/3 \rceil = 4$). Note that the graph cells in the row to the right of each numbered label v (for example, look at $v = 24$ in the example) get used for values of r (rainfall amount) in the range $v - \mathit{scale} < r \leq v$. The January monthly average in this data file is 23.5, and hence is marked in the row labeled $v = 24$. If the value had been exactly 24.0 it would also be plotted in that cell, but if it was 24.1 it would appear in the row labeled 27. Finally, note that the two digits used in each bar are the last two digits of the year number that is being plotted.

```
mac: ass2-soln 2003 < rainfall-086039-2000-2009.csv
    <output for stages 1, 2, 3>
S4, 2003 max is  71.6, scale is 3
  72 |                                 03             03
  69 |                 03              03             03
  66 |                 03              03             03
  63 |                 03              03             03
  60 |                 03              03   03        03
  57 |                 03              03   03        03   ****  03
  54 |                 03              03   03        03         03
  51 |                 03              03   03  **** *03*       *03*
  48 |      ****       03              03  *03*        03         03
  45 |                 03              03   03        03         03
  42 |                 03             *03*  03        03         03
  39 |                *03* ****  03    03   03   03   03         03
  36 |           ****  03       *03*  03   03   03   03         03
  33 |                 03        03   03   03   03   03         03
  30 |                 03        03   03   03   03   03         03
  27 |            03   03   03   03   03   03   03   03         03
  24 | ****       03   03   03   03   03   03   03   03         03
  21 |       03   03   03   03   03   03   03   03   03   03   03
  18 |       03   03   03   03   03   03   03   03   03   03   03
  15 |       03   03   03   03   03   03   03   03   03   03   03
  12 |  03   03   03   03   03   03   03   03   03   03   03   03
   9 |  03   03   03   03   03   03   03   03   03   03   03   03
   6 |  03   03   03   03   03   03   03   03   03   03   03   03
   3 |  03   03   03   03   03   03   03   03   03   03   03   03
   0 +-----+----+----+----+----+----+----+----+----+----+----+----+
      Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
```

If there is no record of rainfall for a particular month within the year being plotted (or if there is no data for that whole year at all, including cases when the specified year is outside the span of years

included in the data file), plot that month as zero (which means, no marking in the graph), but still plot the average. If there is no average, because there were no readings at all for that month in the input file, don't plot the average either.

Modifications to the Specification

There are bound to be areas where this specification needs clarification or correction. Refer to the FAQ page at `http://people.eng.unimelb.edu.au/ammoffat/teaching/20005/ass2/` regularly for updates to these instructions. There is already a range of information provided there that you need to be aware of, with more likely to follow.

The Boring Stuff...

This project is worth 20% of your final mark. A rubric explaining the marking expectations is linked from the FAQ page, and you should read it carefully.

You need to submit your program for assessment; detailed instructions on how to do that are linked from the FAQ page. You can (and should) use `submit` **both early and often** – to get used to the way it works, and also to check that your program compiles correctly on the test server, which has some different characteristics to the lab machines. *Failure to follow this simple advice is likely to result in tears.* Only the last submission that you make before the deadline will be marked.

You may discuss your work during your workshop, and with others in the class, but what gets typed into your program must be individual work, not copied from anyone else. So, do **not** give hard copy or soft copy of your work to anyone else; do **not** "lend" your "Uni backup" memory stick to others for any reason at all; and do **not** ask others to give you their programs "just so that I can take a look and get some ideas, I won't copy, honest". The best way to help your friends in this regard is to say a very firm "**no**" when they ask for a copy of, or to see, your program, pointing out that your "**no**", and their acceptance of that decision, is the only thing that will preserve your friendship. *A sophisticated program that undertakes deep structural analysis of C code identifying regions of similarity will be run over all submissions. Students whose programs are identified as containing significant overlaps will be evaluated for mark penalties of for referral to the Student Center for possible disciplinary action without further warning. This message **is** the warning.* See `https://academicintegrity.unimelb.edu.au` for more information. Note also that solicitation of solutions via posts to online forums, whether or not there is payment involved, is also taken very seriously. In the past students have had their enrolment terminated for such behavior.

Very Important: *The FAQ page contains a link to a program skeleton that you must start with, including an Authorship Declaration that you must "sign" and include at the top of your submitted program. Marks will be deducted if you do not include the declaration, or do not sign it, or do not comply with its expectations.*

Deadline: Programs not submitted by **10:00am on Monday 27 May** will lose penalty marks at the rate of two marks per day or part day late. Students seeking extensions for medical or other "outside my control" reasons should email `ammoffat@unimelb.edu.au` as soon as possible after those circumstances arise. If you attend a GP or other health care professional as a result of illness, be sure to take a Health Professional Report form with you (get it from the Special Consideration section of the Student Portal), you will need this form to be filled out if your illness develops in to something that later requires a Special Consideration application to be lodged. You should scan the HPR form and send it in connection with any non-Special Consideration assignment extension requests.

A sample solution will be linked from the FAQ page by June 4, and it is hoped that marks will be available on the LMS by the evening of Tuesday 11 June.