COMP 206 Winter 2018 – Assignment 1

This is the final version (we hope), uploaded on Jan 30, 2018.

Objectives:

Gain our first hands-on experience with coding software system specifications in C. Specific focus on working with C strings, reading text files and structured output to standard output.

Instructions:

Write the two C programs specified in the following questions. Test them very carefully and pay careful attention to detail, as marking will ensure that you program meets the specifications very precisely. Every program you submit must compile with "gcc program_name" on mimi.cs.mcgill.ca without errors or warnings. When run with the examples shown here, as well as additional correct inputs, the corresponding correct output must be generated. If run with incorrect inputs, the program must never crash (segfault, bus error, etc),but rather print informative errors and exit cleanly.

Automated Testing:

The eval.py script contained in the assignment folder can be used to ensure your program works well with our automated grading. It will compile run several of the basic tests and report the results. To run it, place your C files in a folder, and run "$ python <folder_name>". You will see a fairly long report generated. When everything works, you should see the string "Your program's output is the same as expected output: True" for both questions. This is similar to the process we'll use for real grading, but that will include a wider range of test cases, so do run your program directly yourself and ensure it always works!

Handing In:

Upon completion, create an archive file with both your C programs using the command:

$ tar –czf A1_submission.tar.gz q1_julia_explorer.c q2_calendar.c

Submit the resulting archive on My Courses.

Due Date:

1. For full marks: Friday Feb 9th
2. Final deadline (with 10% penalty): Tuesday Feb 13th

Question #1 – Explore the Julia Sets (40 marks)

A Julia Set is a construction from complex math that can be used to generate a wide range of fractal patterns, with the chaotic property: very small changes in the initial conditions lead to dramatic changes in the generated shape. A standard Linux tool, gnuplot, is capable of plotting a Julia Set from a simple text script and we have provided you with the basis of this script as the attached file "shape_template.txt".

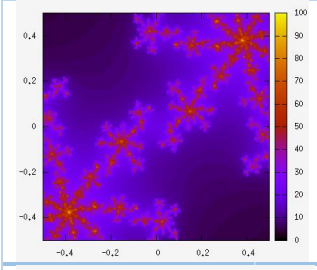
You will create a C program to automate the process of inserting the initial conditions, two floating point numbers, into the correct place in the script, a simple token replacement. In case you try this on your own computer, the command "sudo apt-get install gnuplot-x11" is likely to be helpful.

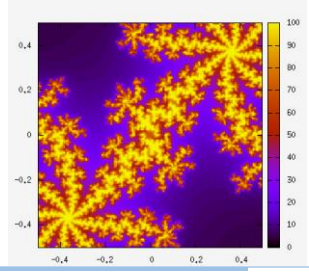
Write a program **q1_julia_explorer.c** that allows the user to produce Julia Set plots interactively:

1. Accept exactly three command-line arguments <file_path> <a> <b>:
    a. <file_path>: A path to the provided "shape_template.txt" gnuplot script
        i. Identify a correct shape_template.txt file as one your program can open for reading, and which contains two "tags": #A# and #B#. For any incorrect file, output "Error: bad file" and return with code –1.
    b. <a>: A floating point value **a** (which has meaning in the construction of the Juliaset)
    c. <b>: A floating point value **b** (which has meaning in the construction of the Juliaset)
    d. For any problem with the a or b arguments, output "Error: bad float arg" and return –1
2. Produce a modified gnuplot script on standard out, where the modifications are exactly:
    a. The #A# tag text is replaced with **a**, printed with 6 decimals (use %.6f)
    b. The #B# tag text is replaced with **b**, printed with 6 decimals (use %.6f)
    c. Note that the provided shape_template.txt is only an example. Your code should definitely handle the tags in this file, but hardcode as little about it as possible, such that any other file with the same tags can also be used for testing.

Examples test runs:

```
$
$ gcc q1_julia_explorer.c -o q1_julia_explorer
$ ./q1_julia_explorer shape_template.txt 39.111 67 | gnuplot -p
$ ./q1_julia_explorer shape_template.txt 35.999 63.9090 | gnuplot -p
$
```





Question #2 – Formatting the year (60 marks)

Write a C program named **q2_calendar.c** that prints one year's full calendar to standard output.

1. Accept exactly 2 command-line arguments that control the calendar details:
    a. Maximum number of characters **DAYSIZE** to print for the day-of-week labels, which must be 2 or greater
    b. The day-of-week that starts the year **FIRSTDAY** as an integer from 1 to 7 (1 is Sunday in English)
2. Your output must precisely follow the **format specification** given by the following examples, so you should make sure you can replicate them precisely. Note that the examples were cut off at April only because we ran out of space on the page. The output must continue until Dec 30th. The full text specification of this format is on the next page.

```
$
$ gcc q2_calendar.c -o q2_calendar
$ ./q2_calendar 2 1
|----------------------------------|
|             January              |
|----------------------------------|
| Su | Mo | Tu | We | Th | Fr | Sa |
|----------------------------------|
| 1  | 2  | 3  | 4  | 5  | 6  | 7  |
| 8  | 9  | 10 | 11 | 12 | 13 | 14 |
| 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| 29 | 30 |    |    |    |    |    |
|----------------------------------|
|             February             |
|----------------------------------|
| Su | Mo | Tu | We | Th | Fr | Sa |
|----------------------------------|
|    |    | 1  | 2  | 3  | 4  | 5  |
| 6  | 7  | 8  | 9  | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| 20 | 21 | 22 | 23 | 24 | 25 | 26 |
| 27 | 28 | 29 | 30 |    |    |    |
|----------------------------------|
|              March               |
|----------------------------------|
| Su | Mo | Tu | We | Th | Fr | Sa |
|----------------------------------|
|    |    |    |    | 1  | 2  | 3  |
| 4  | 5  | 6  | 7  | 8  | 9  | 10 |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 |
| 18 | 19 | 20 | 21 | 22 | 23 | 24 |
| 25 | 26 | 27 | 28 | 29 | 30 |    |
|----------------------------------|
|              April               |
|----------------------------------|
| Su | Mo | Tu | We | Th | Fr | Sa |
|----------------------------------|
|    |    |    |    |    |    | 1  |
| 2  | 3  | 4  | 5  | 6  | 7  | 8  |
| 9  | 10 | 11 | 12 | 13 | 14 | 15 |
| 16 | 17 | 18 | 19 | 20 | 21 | 22 |
```

```
$
$ gcc q2_calendar.c -o q2_calendar
$ ./q2_calendar 1 3
ERROR: Cannot print days smaller than size 2.
$ ./q2_calendar 5 8
ERROR: The first day of the week must be between 1 and 7.
$ ./q2_calendar 10 7
|-------------------------------------------------------------------------------------------|
|                                          January                                          |
|-------------------------------------------------------------------------------------------|
| Sunday     | Monday     | Tuesday    | Wednesday  | Thursday   | Friday     | Saturday   |
|-------------------------------------------------------------------------------------------|
|            |            |            |            |            |            | 1          |
| 2          | 3          | 4          | 5          | 6          | 7          | 8          |
| 9          | 10         | 11         | 12         | 13         | 14         | 15         |
| 16         | 17         | 18         | 19         | 20         | 21         | 22         |
| 23         | 24         | 25         | 26         | 27         | 28         | 29         |
| 30         |            |            |            |            |            |            |
|-------------------------------------------------------------------------------------------|
|                                         February                                          |
|-------------------------------------------------------------------------------------------|
| Sunday     | Monday     | Tuesday    | Wednesday  | Thursday   | Friday     | Saturday   |
|-------------------------------------------------------------------------------------------|
|            | 1          | 2          | 3          | 4          | 5          | 6          |
| 7          | 8          | 9          | 10         | 11         | 12         | 13         |
| 14         | 15         | 16         | 17         | 18         | 19         | 20         |
| 21         | 22         | 23         | 24         | 25         | 26         | 27         |
| 28         | 29         | 30         |            |            |            |            |
|-------------------------------------------------------------------------------------------|
|                                          March                                            |
|-------------------------------------------------------------------------------------------|
| Sunday     | Monday     | Tuesday    | Wednesday  | Thursday   | Friday     | Saturday   |
|-------------------------------------------------------------------------------------------|
|            |            |            | 1          | 2          | 3          | 4          |
| 5          | 6          | 7          | 8          | 9          | 10         | 11         |
| 12         | 13         | 14         | 15         | 16         | 17         | 18         |
| 19         | 20         | 21         | 22         | 23         | 24         | 25         |
| 26         | 27         | 28         | 29         | 30         |            |            |
|-------------------------------------------------------------------------------------------|
|                                          April                                            |
|-------------------------------------------------------------------------------------------|
| Sunday     | Monday     | Tuesday    | Wednesday  | Thursday   | Friday     | Saturday   |
|-------------------------------------------------------------------------------------------|
|            |            |            |            |            | 1          | 2          |
| 3          | 4          | 5          | 6          | 7          | 8          | 9          |
| 10         | 11         | 12         | 13         | 14         | 15         | 16         |
| 17         | 18         | 19         | 20         | 21         | 22         | 23         |
| 24         | 25         | 26         | 27         | 28         | 29         | 30         |
```

Detailed calendar specification:

1. Accept exactly 2 command-line arguments:
    a. Maximum number of characters **DAYSIZE** to print for the day-of-week labels, which must be 2 or greater
    b. The day-of-week that starts the year **FIRSTDAY** as an integer from 1 to 7 (1 is Sunday in English)
2. Your output must precisely follow this specification (and shown in the examples below) for printing a year's calendar.
    a. Every row must begin and end with the one pipe symbol "|"
    b. Each row contains (**DAYSIZE+3) x 7 + 1** visible symbols, and a newline
    c. Each month is composed of:
        i. A full-wdith **separator line** of dashes "-"
        ii. A line composed of the month's full name, centered using spaces, with one less space before than after if needed
        iii. A **separator line**
    d. A line with 7 columns of day labels, with pipes "|" in between. Each day label contains:
        i. A single space
        ii. The day-of-week name cut down to **DAYSIZE** characters if needed
        iii. Additional spaces to pad up to **DAYSIZE** if needed
        iv. A single space
    e. A **separator line**
    f. 5 or 6 lines that contain the days of that month:
        i. Print exactly 30 days always (avoids leap-years etc).
        ii. Separate day sections with pipes "|" in between
        iii. Lay out the date numbers with the usual logic:
            1. The first day of Junuary is specified with the argument **FIRSTDAY**
            2. The first day of each other month is on the weekday following the last weekday of the previous month
            3. When a week is not "full" with dates, blank days are printed to maintain spacing, both at the start and end of the month
    g. Each non-empty day section must contain, in order:
        i. A single space
        ii. The date as a one or two digit integer, following correct ordering
        iii. Additional spaces to pad up to **DAYSIZE** if the number was too short
        iv. A single space
    h. The final line of the year must be a **separator line**