COSC-111 Spring 2014

Project-1

February 13, 2014

Your assignment is to write a Python program that helps the user calculate how many days he or she will need to recover from jetlag. Your program is due **Mon, Feb-24, at 11:59 pm**. To turn it in, use the Submit server at `app.cs.amherst.edu/Submit` and submit two files:

1. Your file of Python code. Name it **buley.py**.
2. A *trace file* showing how your code works on a particular set of inputs. See the end of this handout for more information.

Buley's Formula The International Civil Aviation Organization (ICAO) uses Buley's formula to calculate how many days of rest a person (such as a pilot) needs to recover from jet lag. The formula takes four inputs:

- **T** is the number of hours in transit
- **z** is the number of time zones crossed.
- **dep** is the local departure time.
- **arr** is the local arrival time.

Departure and Arrival times are used in this formula according to the 24-hour military clock. So 1200 is noon, 1430 is 2:30 pm, and so forth.

The times are used to find the departure coefficient d and the arrival coefficient a according to this table:

Time of day	d = dept-coeff	a = arr-coef
0800-1159	0	4
1200-1759	1	2
1800-2159	3	0
2200-0059	4	1
0100-0759	3	3

Where 0=good, 1,2 = fair, 3 = poor, and 4 = bad. If $z \geq 4$ the formula is

$$days = \frac{\frac{T}{2} + (z-4) + d + a}{10}$$

If $z < 4$ then the $(z-4)$ part should be set to zero in this formula.

For full credit your program should:

1. Prompt the user for the necessary information.

2. Check for bad inputs and refused to do a wrong calculation if found (more about this below).

   This includes checking for invalid time, and checking that T and/or z are in reasonable ranges. if a bad input is detected, the program should print an error message explaining what the problem is, and invite the user to try again.

   Your program does not have to check that the user typed in numbers (as opposed to strings) – in that case just let it crash. Your program does not have to check for sensible *combinations* of values (perhaps the pilot is flying around in circles or traveling at the speed of light), just for invalid *ranges* of individual values. But you are welcome to add these checks if you can figure out how to do it.

3. Calculate how many days of rest are need to recover, according to the above formula.

4. Tell the user how many days of rest are needed. The program should round this number *up* to the nearest half-day.

Example: The user leaves Montreal at 1800 hours local time ($d = 3$), spends nine hours travelling, and arrives in Paris at 0800 hours ($a = 4$), having crossed 5time zones. The number of days of rest needed is 1.25. Rounding up to the nearest half-day, the program prints the answer as 1.5 days.

Checking your work. A chart of world time zones is attached for your amusement. Here are some example inputs and the answers. Use these to check your work.

- The user travels 5 hours through 5 time zones, departing at 5 minutes after midnight and arriving 5 minutes after midnight. The answer should be 0.85, rounded up to 1 day.

- The user travels 12 hours through 6 time zones, departing at 10pm and arriving at 4am. The answer should be 1.5, rounded up to 1.5 days.

- The user travels 3 hours through 2 time zones, departing at 10 am and arriving at 8pm. The answer is 0.15, rounded up to 0.5 days.

- The user travels 1 hour through 0 time zones (i.e. staying in the same time zone), departing at 11:30 pm and arriving at 12:30 am. The answer is 0.55 days, rounded up to 1 day.

What to turn in. As usual point your browser to `www.cs.amherst.edu/Submit`, find the Program 1 submit page, and turn in the following two files.

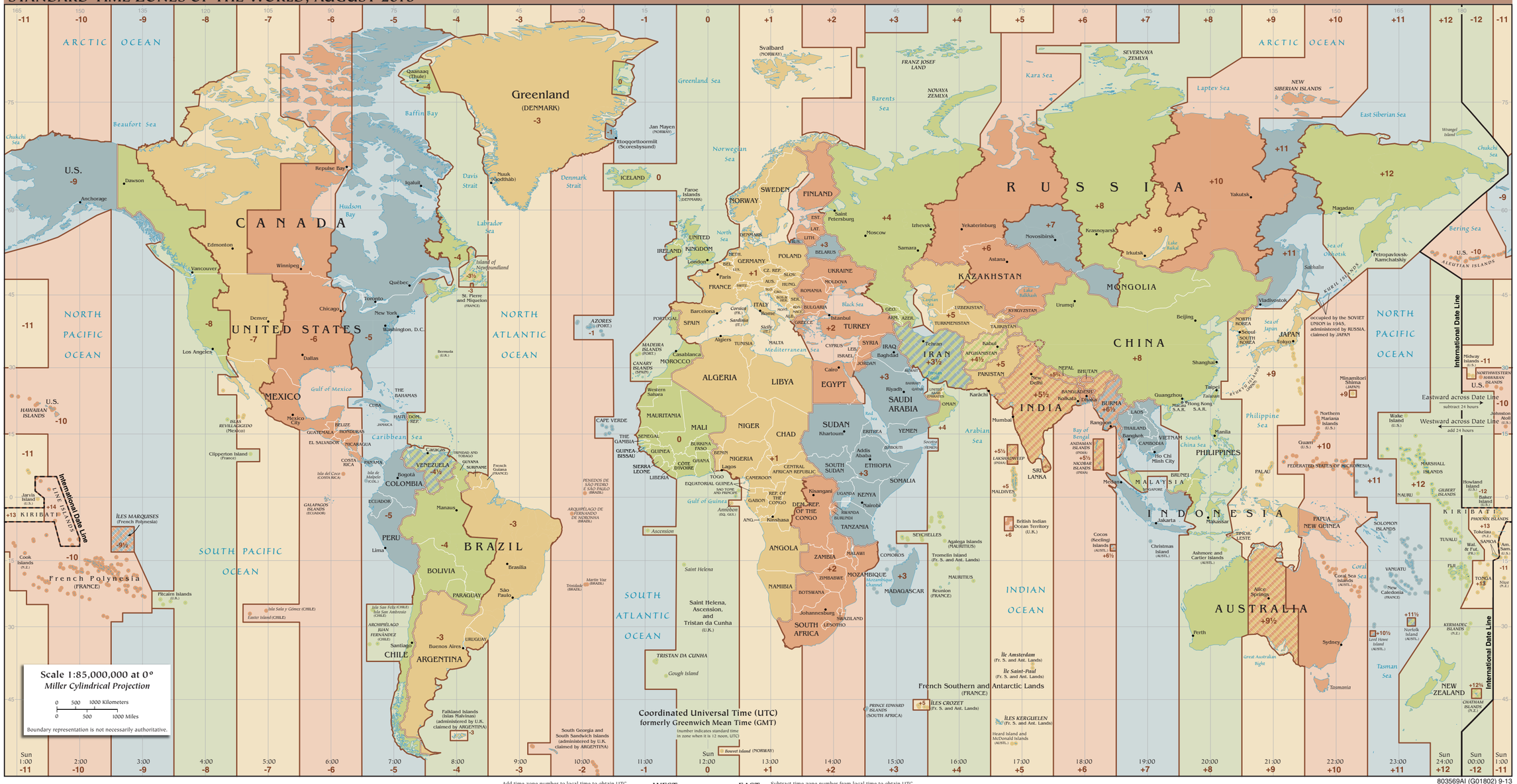
1. The program **buley.py**, of course.
2. A file named **proj1script.txt** that is a transcript record of you (as user) interacting with your program. You can make this script by running your program the usual way inside the IDLE shell (the interactive part), and then selecting **File:Save As...** in the menu bar.

   (a) Your transcript should show your program running with the inputs from the "Checking your work" section above.

   (b) Then, as user, you should type in some errors to show off how your program copes with bad inputs. Use as many bad inputs as you need to demonstrate your work.

Note: Do not try to edit the **proj1script.txt** file, for example to clean up typos and errors that you might have created as user. You will not be penalized for being a clumsy user. But, if you edit the file you may create mis-matches between what the script says your program does and what your program actually does. A professor who notices such a mismatch will wonder if you're trying to fake the output to get a better grade. You of course do not want to alarm the professor and waste his or her time in that way.

STANDARD TIME ZONES OF THE WORLD, AUGUST 2013



Add time zone number to local time to obtain UTC.
Subtract time zone number from UTC to obtain local time.

WEST — EAST

Subtract time zone number from local time to obtain UTC.
Add time zone number to UTC to obtain local time.

803569AI (G01802) 9-13