CSI 333 – Fall 2011

Programming Assignment I

Administrative Information

- Individual assignment.
- **Deadline:** 11 PM, Friday, Sep. 16, 2011.

  Cutoff Point: 11 PM, Sunday, Sep. 18, 2011.
- The C source file for the program must be named `p1.c`.
- The source file must be submitted using the `turnin-csi333` command.
- README file (on `itsunix.albany.edu`) by 10 PM on Saturday, September 10, 2011.

```
~csi333/public/prog1/prog1.README
```

Administrative Information (continued)

Lateness Policy:

- No penalty if the program is submitted by 11 PM on **Friday, Sep. 16, 2011**.
- Lateness penalty: 10 points per day.
- Program **won't** be accepted **after 11 PM on Sunday, Sep. 18, 2011**.
- If you submit both a regular version and a late version, only the late version will be graded.

Important Notes:

- Programs that don't compile or don't generate the executable won't receive any credit.
- Your program must compile and work correctly on `itsunix.albany.edu`.

Project Description

Total Points: 100 (90 for correctness; 10 for documentation).

Weightage: 5%

Goal: Compute and print the number of days that are strictly between two specified dates in the year 2011.

Note: Each date is specified by two integers, the first representing the month and the second representing the day.

Examples:

- The integers 3 and 17 represent March 17, 2011.
- The integers 10 and 5 represent October 5, 2011.

Examples of Program Execution

- Assume that the executable version of the program is in the file a.out.

Example 1:

```
unix2>  a.out
Enter start date:  2  10
Enter end date  :  3  7
Answer =  24
unix2>
```

Example 2:

```
unix2>  a.out
Enter start date:  2  29
Error: Invalid date.
unix2>
```

Examples of Program Execution (continued)

Example 3:

```
unix2>  a.out
Enter start date:  5  15
Enter end date   :  4  9
Error: End date precedes start date.
unix2>
```

Program Outline

1. Prompt the user for the first date. Read the date and check if it is valid. If it is not valid, print a suitable error message and **stop**.
2. Prompt the user for the second date. Read the date and check if it is valid. If it is not valid, print a suitable error message and **stop**.
3. If the second date precedes the first date, print a suitable error message and **stop**.
4. Compute the number of days which are strictly between the first and the second dates.
5. Print the answer computed in Step 4 and **stop**.

Note: The handout for the program gives the conditions for the validity of a specified date.

Additional Notes

- For this assignment, your program may consist of just one function, namely `main`.
- Your program should read all its input from `stdin`; that is, you must use the `scanf` function for reading the input.
- Your program must write all its output to `stdout`; that is, you must use the `printf` function for producing prompts, error messages and the final result.
- You may assume that for each prompt, the user will type two integers separated by one or more spaces.

Additional Notes (continued)

- Other than the validity checks mentioned in Steps 1 through 3 in the program outline above, no other error checks are needed.
- In the following two cases, the answer produced by your program should be 0.
  1. The start date and the end date are the same.
  2. The end date is the day that is immediately after the start date.

Additional Notes (continued)

- In your program, after each call to the function `printf`, include the following C statement:

```
fflush(stdout);
```

Example 1:

```
printf("Enter start date: "); fflush(stdout);
```

Example 2:

```
printf("Answer = %d\n", num_days); fflush(stdout);
```

Program Grading

- Programs will be graded using a script written by the TAs.
- The script will compile your source program, generate the executable version and run the executable on new test data.
- The TAs will grade the version that you submit; once the submission is closed, you **won't** be allowed to make any changes to your program.