CMPT120: Introduction to Computing Science and Programming I

Instructor: Hassan Khosravi

Summer 2012

Assignment 1

Due: Wednesday June 13th

This assignment is to be done individually.

The university policy on academic dishonesty (cheating) will be taken very seriously in this course. You may not discuss the specific questions nor their solutions with any other student. You are encouraged to discuss the general concepts involved in the questions. Please take advantage of office hours offered by the instructor and the TAs if you are having difficulties

Part I: Written

Create a text file named part1.txt that contains the answers to these questions.

1. Convert the following binary numbers into decimal, assuming they are *unsigned integers*. Show the calculation you did to get the decimal value.
    a. 1101
    b. 10011101
    c. 01111
    d. 00000001
    e. 10000000
2. Convert the following binary numbers into decimal, assuming they are *two's complement values*. Show the calculations you did to get the decimal value.
    a. 1101
    b. 10011101
    c. 01111
    d. 00000001
    e. 10000000
3. Using this number of bits, what are the highest and lowest numbers we can represent with unsigned integers and twos complement?
    a. 4
    b. 8
    c. 10
4. Do the following calculations assuming they are two's complement values using 8 digits.
    a. 00101001 + 10001001
    b. 11001100 + 10011110
    c. 11100101 - 01100111

5. Write pseudocode (as described in Section 1.4 of the course guide) describing the procedure of the following guessing game:
   At the beginning of the program, it chooses a random number between 1 and 100. The user will be asked to guess a number, and told whether the actual number chosen by the computer is higher or lower then their guess. When the user guesses the number, a message will tell them that they have guessed the correct number and the game should end.

   [Note this isn't the game described in lecture: it's the other way around. Here, the user is guessing, not the computer.]

Part II: Programming

For this assignment, you will be asking the user to enter a date and then doing some manipulation of the value.

The actual part of the program visible to the user will be quite simple. First, ask the user for the date in the form "**2011/05/18**" (in year/month/day format; this represents May 18, 2011).

Once that is done, calculate the day of the year (where January 1 is the first day of the year) and display it. When your program runs, it should look like this:

```
Enter a date in the form YYYY/MM/DD: 2000/01/01
2000/01/01 is day 1 of the year 2000.
```

Here is another example:

```
Enter a date in the form YYYY/MM/DD: 2007/09/05
2007/09/05 is day 248 of the year 2007.
```

The calculation of the day of year (DOY) is relatively straightforward. If you can determine the DOY of the first day of that month, the answer is a simple addition away.

You will need to do some checking of the input to make sure it's in the correct format, but checking every possible error is beyond the scope of this assignment. You will first need to check that the slashes are in the right places, and then that the specified date is a real day.

Leap Years

The difficulty comes in leap years (when there are 29 days in February). The rule for which years are leap years is more complicated than a lot of people realize:

> Years divisible by 4 are leap years,
> except, years divisible by 100 are not leap years,
> except, years divisible by 400 are leap years.

If the year is an integer in the variable year, this boolean expression is true for exactly the leap years:

```
year%400 == 0 or (year%100 != 0 and year%4 == 0)
```

In leap years, the 29th of February is a valid date, and every day after that is one day later in the year.

Suggested Plan

Write and test small parts of the program at a time. Test each part (by printing the value of variables and making sure they are as you expect) as you go and make sure it's working before you move on. Doing this will let you concentrate on smaller, more manageable problems. Here is a suggested plan of attack:

1. Get the user input working as required.
2. Check to make sure the fifth and eighth character are a slash. Print out an error message if not.
3. Extract the year, month, and day from the string and convert to integers.
4. Get the output working with the parts you already have: “2007/09/05 is day ??? of the year 2007.” You will fill in the ???later.
5. Create an if/elif block that sets two variables as appropriate to the month (ignoring leap years). You will need to know the number of days in the month, and the DOY of the *first* day of that month (e.g. March has 31 days, and the first of March is the 60th day of the year).
6. After the if/elif block, use those two variables to calculate the day of the year. Include this in the print statement so the output looks correct (and should be correct, except for leap years).
7. Insert code after the big if/elif block that fixes the value of the two variables as appropriate in leap years.
8. Add checking for invalid dates (month or day invalid).

Hints

To create this program, you will need the following aspects of Python that you might not have seen yet:

- String indexing. If you have a string variable s, you can get at the first character (at the left of the string) with s[0], the second with s[1], and so on.
- String slicing. To extract the day, month, and year from the user's input, you will need to not only extract individual characters, but larger parts of the string. We don't need to worry about the details; here are some relevant examples run in the Python interactive interpreter:

```
>>> s = "abcd/ef/gh"
>>> s[0:4]
```

```
'abcd'
>>> s[5:7]
'ef'
>>> s[8:10]
'gh'
```

- You will need to convert strings of digits to integers to do the arithmetic.

  ```
  year = int(????)
  ```

- Try to use good variable names and keep your code as readable as possible. There won't be any marks for style in this assignment, but there will be in the future, so you might as well get in the habit.

Sample Test Data

Date	DOY	Note
2011/02/15	46	non-leap year
2011/03/02	61	non-leap year
2012/02/15	46	leap year, before Feb 29
2012/03/02	62	leap year, after Feb 29
1900/03/02	61	divisible by 100 rule
2000/03/02	62	divisible by 400 rule
2000/01/02	2	
2000/06/30	182	
2016/07/12	194	
2009/12/25	359	
3421/04/26	116	
2010/12/88		invalid date
2010/06/31		invalid date
2011/02/29		invalid date
2011/13/01		invalid date
2012/01/00		invalid date
2012/00/01		invalid date

There are some Python modules that can do this kind of calculation for you. You are expected to do the calculation yourself for this assignment, without help from imported modules.

Submitting Your Assignment

You must create your assignment in electronic form and submit it online.

Put both files (part1.txt and your code for the second part) in a zip file. There are some instructions in the website if you have trouble with zipping your files. Call the zip file assignment1 (so it would be assignment1.rar or assignment1.zip).

Login to your account in https://courses.cs.sfu.ca/. You should be able to upload your file when you go into assignment1.