The purpose of this lab is to gain experience with command prompt I/O with python, and to become familiar with the use of the IDE and python interpreter. The completed lab must be submitted via Blackboard NLT (no later than) the beginning of the following week's lab session (i.e. one week after assignment). If you have questions, use the Blackboard discussion forums (and instructor/TA office hours) to obtain assistance.

## Lab Requirements:

- The source code for this lab must be submitted in a file named lab3.py
- The source code file must contain a file header formatted as in previous labs.
- The source code file should use self-documenting code and additional comments (as required) to improve code readability.
- The solution for the lab must take input from the user on the command line.
- The solution must then present a menu with options to show the day of the week for that date, show what happened on that date, or quit.
- The solution must correctly display the day of the week
- The solution must correctly open the webbrowser with the appropriate link


## Lab Procedure:

You are going to use the power of Python to create a program that can tell people the day of the week they were born, and what happened on that day! ©

## Computing the day of the week you were born.

To do this you will ask the user on the command line for three pieces of information:
D: day of the month they were born
M: month of the year they were born
YYYY:Year they were born
You will store that information in variables.
Algorithm:
Step 1: First you must convert the four digit year to a two digit century C and two digit year Y , so 1986 would be: $\mathrm{C}=19, \mathrm{Y}=86$

How? YYYY / 100 gives the two digit century. Using simple math you can then find the two-digit year as well.

Step 2: Next you must convert the month so that the year begins in March. In the formula the year must start in March and go through February. So, March=3, December=12, January=13, February=14. This also means that Jan 10, 1981 is really the $13^{\text {th }}$ month in 1980 (according to the algorithm). So, when the user enters Jan 10, 1981 the values should end be modified to: $\mathrm{D}=10, \mathrm{M}=13, \mathrm{Y}=80$.

Hint: To do this, just check if the user entered a month equal to 1 or 2 and then modify the values of $Y$ and $M$ if they did.

Step 3: Then, you will compute the day they were born by computing the sums below:
$\mathrm{S} 1=\mathrm{C} / 4-2 * \mathrm{C}-1$
$\mathrm{S} 2=5^{*} \mathrm{Y} / 4$
$\mathrm{S} 3=26^{*}(\mathrm{M}+1) / 10$
Finally, then the day of the week they were born is: $(\mathrm{S} 1+\mathrm{S} 2+\mathrm{S} 3+\mathrm{D}) \% 7$
where $0=$ Sunday, $1=$ Monday, etc...
Step 4: Use an if/elif statement to tell the user the day of the week they were born using text. A helpful example:
if someVar $=0$ :
print "The variable is zero"
elif someVar $==1$ :
print "The variable is one"
elif someVar $==2$ :
print "The variable is two"
This algorithm is taken from: http://www.bbc.co.uk/dna/h2g2/A22548314. More examples and information can be found there.

- The sample program executions are provided below. Examine this output carefully. Your application should reproduce this output exactly.


## What happened on that day?

To figure this out, we go to our friend Google! This is really the easy part, but fun!

## Sample code:

import webbrowser
...
\# Open the URL with the appropriate news in it
\# Make sure to use the ORIGINAL Month and Year!!
\# Create a date string from the date the user entered that looks like this: 4/20/1980
\# Use str(aNumberVariable) to convert a number to a string
\# Then use "+" to concatenate two strings. For example:
author $=$ "Dan" $+\operatorname{str}(10)+$ "Fleck" \# Creates string: Dan10Fleck
theDate= FIGURE THIS OUT \# but it MUST be formatted like this: 4/20/1980
\# Create a URL string with the correct date in it.
url="http://news.google.com/archivesearch?as_q=the\&num=10\&btnG=Search+Archives\&as_epq=\&as _oq=\&as_eq=\&as_user_ldate="+theDate+"\&as_user_hdate="+theDate+"\&lr=\&as_src=\&as_price=p1 \&as_scoring="
webbrowser.open(url) \# Open the browser!

## Creating the menu

Use the information provided during the lectures and the textbook to create a menu as seen in the examples below. The menu should do the appropriate action if a valid letter is pressed, and do nothing if an invalid letter is pressed (the program should not crash in this case).

## Bonus Points

A $20 \%$ bonus will be awarded if you add in code to check that the date is valid (valid month, valid day of that month). Other features can also receive bonus points, but check with you GTA before coding extras to ensure you will get points for the extra work. Be sure to note any bonus options you created in the file header of your program.

## Sample Program \#1: (Bolded and italicized is user-typed input)

...spring09/non_web/lab_solutions/lab3 > python lab3.py
What day were you born on [1-31]?04
What month were you born in [1-12]?12
What year were you born in [1999]? 1940
What would you like to do?
a) Show day of the week for date
b) Show what happened that day
q) quit

Choice ->a
12/4/1940 was a Wednesday
What would you like to do?
a) Show day of the week for date
b) Show what happened that day
q) quit

Choice ->b
What would you like to do?
a) Show day of the week for date
b) Show what happened that day
q) quit

Choice ->a
12 / 4 / 1940 was a Wednesday
What would you like to do?
a) Show day of the week for date
b) Show what happened that day
q) quit

Choice $->\boldsymbol{q}$
...spring09/non_web/lab_solutions/lab3 >

## Sample Program \#2: (Bolded and italicized is user-typed input)

...spring09/non_web/lab_solutions/lab3 > python lab3.py
What day were you born on [1-31]?22
What month were you born in [1-12]?1
What year were you born in [1999]?1988
What would you like to do?
a) Show day of the week for date
b) Show what happened that day
q) quit

Choice ->b
What would you like to do?
a) Show day of the week for date
b) Show what happened that day
q) quit

Choice ->a
1/22/1988 was a Friday
What would you like to do?
a) Show day of the week for date
b) Show what happened that day q) quit

Choice ->qwewqewqeqwrqwrw
What would you like to do?
a) Show day of the week for date
b) Show what happened that day q) quit

Choice ->q

Lab Assignment \#3

|  | Excellent (85\% or higher) | Average (60\% or higher) | Needs Improving (Less then 60\%) | Points Possible |
| :---: | :---: | :---: | :---: | :---: |
| Core Concepts <br> (Topics of Focus) | - Read input from user. <br> - Printed output to user. <br> - Used comments to document code <br> - Used appropriate if statements, remainder operator, webbrowser package <br> - Used appropriate functions to structure the program. | - Calculation is implemented, but returns the wrong result. <br> - Header and/or in-line comments are missing <br> - Attempted to use if statements, remainder operator, webbrowser package <br> - Used functions, but in a poorly organized way. | - Failed to perform calculation. <br> - Failed to read input from user. <br> - Failed to print output to user. <br> - No use of functions. | 3 |
| Program Flow / Code Organization | - Program begins with input statements <br> - Math is done correctly and succinctly <br> - Code is organized in a logical manner with thoughtful reasoning behind it. <br> - Minimal use of code (e.g. no irrelevant statements or complexity) | - Math in unclear or done in an inefficient manner. <br> - Program contains unused or redundant lines of code. | - Program is unreadable | 2 |
| User Interface / Input/Output | - Code correctly prompts the user for the appropriate inputs. <br> - Menu is displayed and accepts only the correct inputs and calls an appropriate function. <br> - Output is descriptive to the user and is well formatted. | - Menu only works once or has a limit that prevents infinite choices. <br> - Input prompt does not tell the user what to enter <br> - Output does not match the specification | - No menu is present or most menu options do not work. <br> - Program does not prompt for input <br> - Program does not return required output (inches, feet, yards) | 3 |
| Syntax / <br> Overall Coding Guidelines | - Code runs without any errors. <br> - Comments are meaningful and professional. <br> - Code is clean and easy to read. Uses meaningful variable names <br> - Complex algorithms (three lines or more) are commented with information on purpose and start/stop states. | - Comments are not helpful for understanding the code <br> - variable names are not meaningful <br> - code generates errors in edge cases | - Code generates errors in common cases <br> - no comments were used | 2 |
| Bonus Points | Code correctly rejects any date that has an invalid month or day for that particular month. | None given. | None given. | (+2) |
| Final Score |  |  |  | $10(+2)$ |

## Additional Comments:

