Fall 2020
Computer Science I — Maíra Marques Samary PhD

TA's Office Hours – online

James Monahan - monahajm@bc.edu - https://bccte.zoom.us/j/2822792652
Tuesdays 7:00 PM – 8:00 PM
Wednesdays 4:00 PM – 5:00 PM

Jennifer Joseph - - josephjz@bc.edu - https://bccte.zoom.us/j/5882755193
Wednesdays 11:00 AM – 12:00 PM
Thursdays 3:00 PM - 4:00 PM

Liam Murphy- - murpaue@bc.edu - https://bccte.zoom.us/j/3085424208
Tuesdays 2PM-4PM

Homework 1

Due date – 09/13/20 11:59 PM (Boston)

General Instructions

Create a folder named ***LASTNAME_FIRSTNAME***. You will populate the folder with **ALL** of the .py files you write for this homework. To submit the homework, verify the folder includes all your .py files, compress (zip) the folder then upload to Canvas. Remember to include the following comments at the **top of each** of your .py files:

```
# author:
# assignment:
# description:
```

What to submit in Canvas?

Make sure all your files are saved in the folder LASTNAME_FIRSTNAME, then compress (zip) the folder and upload to Canvas.

If you encounter any problems in completing the assignment or in the submission process, please don't hesitate to ask for help. The sooner, the better!

Problem

In this homework, you will ask the user on the command line for three pieces of information:

D: day of the month the user was born
M: month of the year the user was born
YYYY: Year the user was born

You will store that information in variables

Like in the example below:

What day you were born [1-31]? ***18***
What month you were born [1-12]? ***01***
What year you were born [1999]? ***1978***

After asking the user the questions, your software will present to the user the menu below:

What would you like to do?

a) Show day of the week for date

b) Show the hit songs of the week

q) quit

If the user types a:

Your software will have to compute the day of the week that the user was born with the following algorithm. You must implement the algorithm, you are not allowed to use more fancy functions to give you the exact day of the week.

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: C=19, 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 13th month in 1980 (according to

the algorithm). So, when the user enters Jan 10, 1981 the values should end be modified to: D=10, M=13, 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: (Be careful, the lines below are not exactly the code that you have to do! , look at the origin of the algorithm to do see what you have to do exactly!!!)

```
S1 = C/4 – 2*C -1
S2 = 5*Y/4
S3 = 26*(M+1)/10
```

Finally, then the day of the week they were born is: (S1 + S2 + S3 + 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.

If the user types b:

Sample code:

```
import webbrowser

...

# Open the URL with the songs were on the hit list in the week

# Create a date string from the date the user entered in the format:  YEAR/MONTH/DAY

 # An Example:  1978-01-18
```

#TIP – the month is January, normally just represented as number 1, but here you need to represent days and months in 2-digit format. So, January needs to be represented as 01

Use str(aNumberVariable) to convert a number to a string

Then use "+" to concatenate two strings. For example: author = "Maira" + str(10) + "Marques" # Creates string: Maira10Marques

theDate= FIGURE THIS OUT # but it MUST be formatted like this: **1978-01-18**

 # Create a URL string with the correct date in it.

url="https://www.billboard.com/charts/hot-100/" + theDate

webbrowser.open(url) # Open the browser!

If the user types c:

You end the program.

Some examples of the program running:

EXAMPLE 1

What day you were born [1-31]? ***18***
What month you were born [1-12]? ***01***
What year you were born [1999]? ***1978***

What would you like to do?
a) Show day of the week for date
b) Show the hit songs of the week
q) quit

a

OUTPUT
1/18/1978 was a Wednesday

EXAMPLE 2

What day you were born [1-31]? ***18***
What month you were born [1-12]? ***01***
What year you were born [1999]? ***1978***

What would you like to do?
a) Show day of the week for date
b) Show the hit songs of the week
q) quit

b

OUTPUT
None
Your default web browser opens with the billboard list of the week of your birthday

---

RUBRIC

	Excellent (85% or higher)	Good (60% or higher)	Fair (40% or higher)	
Core Concepts (Topics of Focus)	• Read input from user. • Printed output to user. • Used comments to document code • Used appropriate if statements, remainder operator, web browser package • Used appropriate functions to structure the program.	• Calculation is implemented, but returns the wrong result. • Header and/or in-line comments are missing • Attempted to use if statements, remainder operator, web browser package • Used functions, but in a poorly organized way.	• Failed to perform calculation. • Failed to read input from user. • Failed to print output to user. • No use of functions.	3
Program Flow / Code Organization	• Program begins with input statements • Math is done correctly and succinctly • Code is organized in a logical manner with thoughtful reasoning behind it. • Minimal use of code (e.g. no irrelevant statements or complexity)	• Math in unclear or done in an inefficient manner. • 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. • Menu is displayed and accepts only the correct	• Menu only works once or has a limit that prevents infinite choices.	• No menu is present or most menu options do not work. • Program does not prompt for input	3

	inputs and calls an appropriate function. • Output is descriptive to the user and is well formatted.	• Input prompt does not tell the user what to enter • Output does not match the specification	• Program does not return required output (inches, feet, yards)	
Syntax / Overall Coding Guidelines	• Code runs without any errors. • Comments are meaningful and professional. • Code is clean and easy to read. • Uses meaningful variable names • Complex algorithms (three lines or more) are commented with information on purpose and start/stop states.	• Comments are not helpful for understanding the code • variable names are not meaningful • code generates errors in edge cases	• Code generates errors in common cases • no comments were used	2
FINAL SCORE				**10**