CSCI 1200 The Art of Computational Thinking
Instructor: Fleming
Homework 5—Due Sunday October 8th at 11:55pm

Objectives

1. To understand the programming pattern simple decision and its implementation using a Python if statement.
2. To understand the programming pattern two-way decision and its implementation using a Python if-else statement.
3. To understand the programming pattern multi-way decision and its implementation using a Python if-elif-else statement.
    a. To understand the concept of Boolean expressions and the bool data type.
4. To be able to read, write, and implement algorithms that employ decision structures, including those that employ sequences of decisions and nested decision structures.

Madame Maxine's Fortune Palace

Your job is to write a fortune-telling program. You will be writing one single program that when run will give the user information about their future by reading their palm. Your program should be nicely broken down into functions as follows. You will turn in one file called `fortune_teller.py`. This program must contain the following information in comments at the top of the file:
Your Name
Your TA's Name
Your section number
The homework's name
A brief description of the program and what it does.

The fortune telling program works as following: after printing out a greeting, it prompts the user for today's date and the day of the month that the user was born on.

Using this information, it determines whether what line on the user's hand to read. If today's date is between the 1st and the 9th, it reads the user's life line. If today's date is between the 10th and the 19th, it reads the user's heartline. If today's date is between the 20th and the 29th, it prompts the user for their birth month and reads the user's head line. If today's date is the 30th or the 31st, the program declares today to be a bad day for fortune telling. Once the program has told the user's fortune, it says goodbye and exits.

Your program *must* include the following functions:

- **`get_today`**: this function should prompt the user for what date it is today and **return** it as an int.
- **`get_birthday`**: this function should prompt the user for the day of the month that they were born on and **return** it as an int.

- **`get_birthmonth`**: this function should prompt the user for the the month that they were born in and **return** it as an int.
- **`likes_spicy_food`**: this function should prompt the user about whether or not they like spicy food and **return** a boolean value. `True` if the user likes spicy food, `False` otherwise.
- **`read_lifeline`**: this function should take three arguments, today's date, the user's birthday, and the user's spicy food preference. Using these, it should read the user's life line in the following manner: if the user likes spicy food and their birthday is divisible by 3, they have a long, deep life line. Otherwise, if the user likes spicy food and today's date is divisible by 4 or 5, the user has a short, deep line. If the user likes spicy food and neither of the prior two conditions are met, the fortune teller should report the presence of a dark cloud preventing them from reading the life line. If the user doesn't like spicy food, they have a forked and branching lifeline.
    - Output:
    - Long, deep line: `"You have a long, deep line. Your life will be filled with health and balance."`
    - Short, deep line: `"You have a short, deep line. You possess remarkable fortitude."`
    - Dark cloud: `"A dark cloud passes over my inner eye."`
    - Forked and branching: `"Your line is forked and branching. Interesting events await you."`
- **`read_heartline`**: this function should take two arguments, today's date and the user's birthday. Using these, the fortune teller will read the user's heartline based on the difference of today and the user's birthday. If there is no difference, the user's heartline is the perfect example of long. If the difference is less than -10 or greater than 10, the user's line is faint. Otherwise, the user has a very deep heartline.
    - Output:
    - Perfect example of long: `"Your heartline is the perfect example of long. You are filled with warmth and love."`
    - Faint: `"You have a faint line. You will need to work to stay present emotionally."`
    - Very deep: `"The deepness of your heartline will cause trouble if you are not aware."`
- **`read_headline`**: this function should take two arguments, the user's birthday and the user's birth month. If the user was born on the ides of March, report that the crosses in their headline don't bode well. Otherwise, if the user was born in the month with the same number as their birthday, their head line is deep and wavy. If the previous conditions weren't met and the user's birthmonth and birthday have the same parity (both even or both odd), their line is branched and upwards. In all other cases, the headline should be reported as branched.
    - Output:
    - Crosses: `"I'm afraid the crosses that I see here cannot be good."`
    - Deep and wavy: `"The deep and wavy line here shows excellent memory but not without conflict."`

- Branched and upwards: "`The branched and upwards line here shows big dreams and self awareness.`"
- Branched: "`The branched line here shows many events unknown to even the spirits.`"

Your program *may additionally include* any other functions that are appropriate. Remember, the main function of your program should read like a table of contents and should not include the main computational work of the program.

You may assume that the user will only enter valid input in the accepted ranges. You may assume that the user will not enter days that do not exist (e.g. the 67th day of the month or the -1st month of the year).

Below is example output of the program. User input is shown **in bold**. Your program should **exactly reproduce** the output shown below.

```
Welcome to Madame Maxine's Fortune Palace.
Here, we gaze deeply into your soul and find the
secrets that only destiny has heretofore known!

The power of my inner eye clouds my ability to keep
track of mundane things like the date.
Tell me, what day of the month is it today: 3
Numerology is vitally important to fortune telling.
Tell me, what day of the month were you born on: 1

One more question before we begin.
Do you like spicy food? (1 = yes, 2 = no) 1
I will now read your lifeline.
A dark cloud passes over my inner eye.

These insights into your future are not to be enjoyed or dreaded,
they simply come to pass.
Good day.
```

```
Welcome to Madame Maxine's Fortune Palace.
Here, we gaze deeply into your soul and find the
secrets that only destiny has heretofore known!

The power of my inner eye clouds my ability to keep
track of mundane things like the date.
Tell me, what day of the month is it today: 18
Numerology is vitally important to fortune telling.
Tell me, what day of the month were you born on: 1

I will now read your heartline.
You have a faint line. You will need to work to stay present emotionally.

These insights into your future are not to be enjoyed or dreaded,
they simply come to pass.
Good day.
```

```
Welcome to Madame Maxine's Fortune Palace.
Here, we gaze deeply into your soul and find the
secrets that only destiny has heretofore known!
```

```
The power of my inner eye clouds my ability to keep
track of mundane things like the date.
Tell me, what day of the month is it today: 28
Numerology is vitally important to fortune telling.
Tell me, what day of the month were you born on: 4

I need one last piece of information.
Tell me, what month were you born in: 4
I will now read your headline.
The deep and wavy line here shows excellent memory but not without conflict.

These insights into your future are not to be enjoyed or dreaded,
they simply come to pass.
Good day.
```

```
Welcome to Madame Maxine's Fortune Palace.
Here, we gaze deeply into your soul and find the
secrets that only destiny has heretofore known!

The power of my inner eye clouds my ability to keep
track of mundane things like the date.
Tell me, what day of the month is it today: 30
Numerology is vitally important to fortune telling.
Tell me, what day of the month were you born on: 12

Today is a bad day for fortune telling.

These insights into your future are not to be enjoyed or dreaded,
they simply come to pass.
Good day.
```

Testing

It is your responsibility to test your program. We will provide files containing example output. You can use an online difference checker tool (like https://text-compare.com/) to compare and contrast the contents of these files with the output of your program. If your program output does not exactly match the files we provide, you will not receive full credit.

Style and Comments (15 points)

Comments (5 points):

- Your code should be well commented. Use comments to explain what you are doing, especially if you have a complex section of code. These comments are intended to help other developers understand how your code works. These comments should begin with pound signs (#).
- Each function definition should have a docstring. This comment does two things: it describes the purpose of the function and it describes how to use it. You should not describe the mechanics of how the function works, but rather what it does. E.g. “This function counts the number of even numbers in a list” (good) versus “this function uses a for loop to loop through a list and increment a counter each time it finds an even number” (bad). Docstrings always go directly after the definition line.
    - Example (notice the usage of `"""` at the beginning and end of the comment):
    - `def convertCF(celsius):`

```
    """Form a complex number.

    Args:
    celsius (int) -- the temperature in celsius

    Returns:
    int - the temperature in farenheit
    """
    (your code goes here)
```

Naming (5 points)

- Your variable names should be meaningful and concise
- Your variable names should be formatted like: `user_birthday` (words start with lowercase letters and are separated with underscores)
- Function names should be formatted the same way as your variables: `get_today()` (words start with lowercase letters and are separated with underscores)

Proper Variable Usage (2 points)

- You should not use global variables

Main Function (3 points)

- You should have a `main` function. The only code in your program that is not inside a function should be a single call to `main()`. Your main function may contain: function calls, control statements (if/elif/else), and print statements. It may only contain a **maximum of 3 print statements**.
    - (Optional) It is generally good practice when writing python programs to run your main code from inside an `if __name__ == '__main__':` statement, as shown in the last section of chapter 7.7 in your textbook: https://runestone.stage.csel.io/thinkcspy1200/Functions/mainfunction.html . Doing this will be required for the homework after this one, but is optional here.
      Example:

```
def main():
    print('This is the main function')

main()
```

      Becomes:

```
if __name__ == '__main__':
    print('This is the main function')
```

      Notice that you no longer have an explicit function call to main.

Extra Credit (+10 Points)

You may earn up to 10 points in extra credit by adjusting your program to read the user's fortune in the case that today's date is the 31st. You can choose what happens in this case. Your adjustments should not change the main function of the program in a significant way and should not change the behaviour of the program for days that are not the 31st. You may prompt the user for more input, you may use loops, you can do want you want. To receive the extra credit, you must accurately comment what your added functionality does and how it acts in each case.