Homework Turnin

Name:	Riley H Taylor
Email:	rileytaylor@email.arizona.edu
Student ID:	23183089
Section:	2J
Course:	CS 110 17sp
Assignment:	hw6
Receipt ID:	ea09fec7f51957da3cc66af69146726e

Warning: Your turnin is 2 days late. Assignment hw6 was due Tuesday, February 28, 2017, 7:00 PM.

Turnin Successful!

The following file(s) were received:

baby_names.py (4499 bytes)

```
# Author: Riley Taylor
# Course: CSC 110, Section 2J, Spring 2017
# Program: Baby Names
#
# Using census data, this program provides data about name ranking and
# meaning through a histogram display.

from drawingpanel import *


STARTING_YEAR = 1890
COLUMN_WIDTH = 60
LEGEND_HEIGHT = 30


def main():
    print("This program allows you to search through the\n"
          "data from the Social Security Administration\n"
          "to see how popular a particular name has been\n"
          "since 1890.\n")
    name = input("Name: ")
    gender = input("Gender: ")
    match = find_name("names.txt", name, gender)
    meaning = find_meaning("meanings.txt", name, gender)
    if (match == ""):
        match = '"' + name + '" not found.'
        print(match)
    else:
        print(match + meaning)
        p = DrawingPanel(780, 560, "white")
        draw_basics(p, meaning)
        draw_histogram(p, match)
```

```
# ------------------------------------------------------------------------
# find_name() retrieves the name data from the name file.
#
# PARAMETERS: fname -- a string. the file of names
#             name -- a string. The name to retreive.
#             gender -- a string. The gender to retrieve.
# ------------------------------------------------------------------------
def find_name(fname, name, gender):
    f = open(fname)
    names = f.readlines()
    match = ""
    for n in names:
        if (n.lower().startswith(name.lower() +
            " " +
            gender[0].lower())):
            match = n
    return match


# ------------------------------------------------------------------------
# find_meaning() retrieves the meaning from the meanings file.
#
# PARAMETERS: fname -- a string. the file of meanings
#             name -- a string. The name to retreive.
#             gender -- a string. The gender to retrieve.
# ------------------------------------------------------------------------
def find_meaning(fname, name, gender):
    m = open(fname)
    meanings = m.readlines()
    for m in meanings:
        if (m.startswith(name.upper() + " " + gender[0].lower())):
            return m
        elif (m.startswith(name.upper() + " mf")):
            return m


# ------------------------------------------------------------------------
# get_gender_color() returns the color for each gender.
#
# PARAMETERS: gender -- a string. The gender as a single, lowercase
#                       character.
# ------------------------------------------------------------------------
def get_gender_color(gender):
    if (gender == 'm'):
        return "green"
    else:
        return "yellow"


# ------------------------------------------------------------------------
# draw_basics() draws the legend and x axis.
#
# PARAMETERS: p -- an object. The DrawingPanel to draw in.
#             meaning_line -- a string. The meaning line for the
#                             header.
# ------------------------------------------------------------------------
def draw_basics(p, meaning_line):
    p.canvas.create_rectangle(0, 0, 780, LEGEND_HEIGHT, fill="gray")
    p.canvas.create_text(390, 16, text=meaning_line)
    p.canvas.create_rectangle(0, 560 - LEGEND_HEIGHT,
                              780, 560,
                              fill="gray")


# ------------------------------------------------------------------------
# draw_histogram() draws the bar graph of name ranks.
#
# PARAMETERS: p -- an object. The DrawingPanel to draw in.
#             name_line -- a string. The data for the columns.
# ------------------------------------------------------------------------
```

```
def draw_histogram(p, name_line):
    text_x = 15
    col_x = 0
    year = int(STARTING_YEAR)
    decades = name_line.split()
    color = get_gender_color(decades[1])
    for d in range(2, len(decades)):
        p.canvas.create_text(text_x,
                             552,
                             text=str(year))
        column_top = int(decades[d]) / 2 + LEGEND_HEIGHT
        if (int(decades[d]) == 0):
            column_top = 560 - LEGEND_HEIGHT
        p.canvas.create_rectangle(col_x,
                                  column_top,
                                  col_x + COLUMN_WIDTH / 2,
                                  560 - LEGEND_HEIGHT,
                                  fill=color, outline=color)
        p.canvas.create_text(col_x,
                             column_top,
                             text=str(decades[d]))
        year += 10
        text_x += COLUMN_WIDTH
        col_x += COLUMN_WIDTH

main()
```