Project 1

What's in a Name?

Due November 19, 2021 at 5pm

(5% bonus for submission before 5pm Nov 18)

Project 1 is the first assignment where we won't be telling you exactly what to do! You will get the chance to make your own decisions about how you want your program to handle things, what its functions should be called, and how you want to go about designing it. Project 1 will also be ***substantially longer*** than any of the single homework assignments you've completed so far, so make sure to take the time to plan ahead.

In order to encourage you to start planning (and implementing) during the first week of the assignment, you have a **mandatory design review due before 5pm Friday November 12**. This design review consists of demonstrating your progress to date to a member of the course staff. More details about expectations and course staff office hours are available below.

Learning Goals

1. Design and implement a larger scale program than previously.
2. Make good choices about which functions to design and their inputs and outputs.
3. Do more complicated things with strings, lists, and loops.
4. Read and write files.
5. Use a complex Python module (graphics.py).
6. Create graphical output.
7. Identify and fix errors.

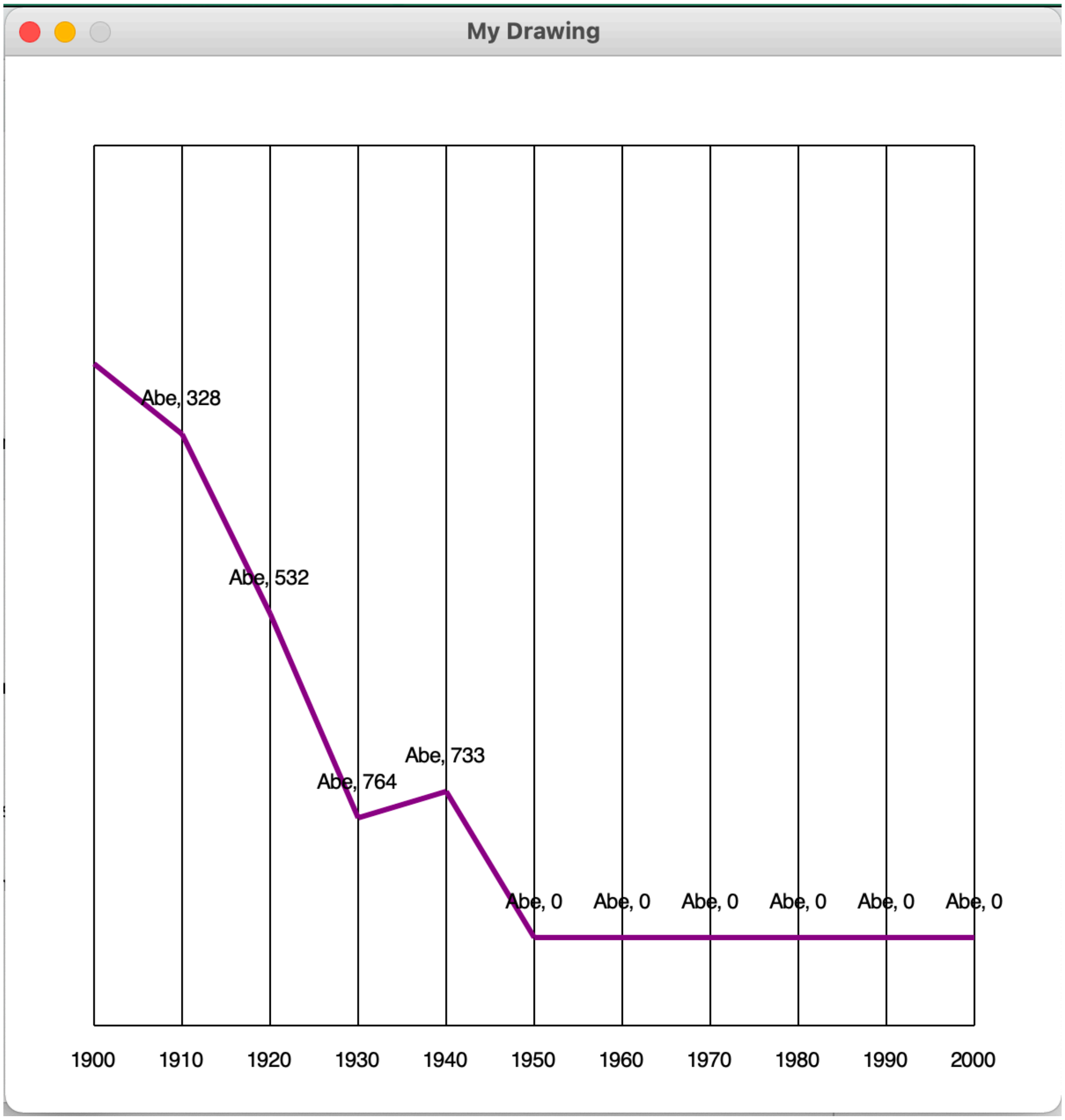
The Assignment

This project will give you practice with file processing in conjunction with other constructs we have learned this semester (loops, conditionals, and functions), as well as learning some graphics. We will be processing a file with data obtained from the Social Security Administration. They provide a web site showing the distribution of names chosen for children over the last 100 years in the US (http://www.ssa.gov/OACT/babynames/). In this project, you will search for a requested name in the file and graph the popularity of that name in the period from 1900 to 2000.

Your program should give an introduction and then prompt the user for a name to display. Then it will search the data file for that name. If it finds it, it should graph the data for that name. If not, it should generate a short message indicating that the name is not found. If the name is found, graph the data in the format shown below.

You should ignore case when comparing the name typed by the user with the names in the input file. The strings that you display on the graph should use the name from the input file, even if the user types the name in a different case. For example, if the user asks you to search for "ABE", you should find it even though the input file has it as "Abe" and the drawing panel should use the input file's "Abe" rather than what the user typed when it displays the name/rank information in the graph.

Here is a sample graph for the name “Abe”:



Input data

Every ten years, the data gives the 1000 most popular boy and girl names for kids born in the US. You will use a simplified version of the original SSA data. On each line we have the name, followed by the rank of that name in 1900, 1910, 1920, ..., 2000 (11 numbers). A rank of 1 was the most popular name that year, while a rank of 997 was not very popular. A 0 means the name did not appear in the top 1000 that year at all. The lines are in alphabetical order, although we will not depend on that.

```
...
Sam 58 69 99 131 168 236 278 380 467 408 466
Samantha 0 0 0 0 0 0 272 107 26 5 7
Samara 0 0 0 0 0 0 0 0 0 0 886
Samir 0 0 0 0 0 0 0 0 920 0 798
Sammie 537 545 351 325 333 396 565 772 930 0 0
Sammy 0 887 544 299 202 262 321 395 575 639 755
Samson 0 0 0 0 0 0 0 0 0 0 915
Samuel 31 41 46 60 61 71 83 61 52 35 28
Sandi 0 0 0 0 704 864 621 695 0 0 0
Sandra 0 942 606 50 6 12 11 39 94 168 257
...
```

We see that “Sam” was #58 in 1900 and is slowly moving down. “Samantha” popped on the scene in 1960 and is moving up strong to #7. “Samir” barely appears in 1980, but by 2000 is up to #798. The database is for children born in the US, so ethnic trends show up when immigrants have kids.

Drawing module

Use the module graphics.py (created by John Zelle) to create your display. The most recent version of the library can obtained at http://mcsp.wartburg.edu/zelle/python. Go to that site and download the file “graphics.py”. The site also contains documentation in html and pdf format. A simple program to draw a single line might look like this:

```
from graphics import *

def main():
     win = GraphWin("My Drawing", 600, 600)
     aLine =Line(Point(150,150), Point (325, 440))
     aLine.setOutline(“purple”)
     aLine.draw(win)
     win.getMouse() # pause for click in window
     win.close()
main()
```

Basic Functionality

Your graph should have the following components:

1. A reference grid with a series of black vertical lines (called decade lines), one for each decade of data, labeled below with the year of that decade. The grid should also have horizonal lines at the positions of name rank 1 (near the top) and 1000 (near the bottom).
2. A line graph showing the changing rank of the name. The most popular name (rank = 1) should cross the decade line at top horizontal line. A name with no rank (indicated by the value 0 in the data) should cross the decade line at the bottom horizontal line. A straight line segment should join the intersection points of adjacent decade lines.
3. A label showing the name and its rank at each decade, near the point where the name line crosses the decade line.

Advice and Suggestions
Do not try to program the entire project up at once. You need to break this program down into simple pieces that you code and test one at a time.

Here is one possible way you could start this solution:

0. Use a "temporary" **main()** to interact and test each function as you create and write them.
1. Start with a smaller version of the data file to reduce your processing time.
2. Read and print your smaller test data file to verify that you can get the data.
3. Get a name from the user and print the line for that name.
4. After that works, start to add the graphics by drawing a single line.
5. And so on….

Some comments on coding standards
For this assignment, you'll need to follow the class coding standards, a set of rules designed to make your code clear and readable.

- You should be commenting your code, and using constants in your code (not magic numbers or strings).
- Any numbers other than 0 or 1 are magic numbers!
- Adhere to the coding standards by including function header comments for each of the functions (other than main).
- Including an explicit collaboration statement describing any people you may have talked with about this project or any sources you might have consulted.

Extra credit
Each of these additional capabilities are worth a possible extra five percent:

- Instead of drawing a line graph, create a solid graph by filling the area beneath the line. Adjust label positioning to be visible above the solid area.
- Create a display that makes it easy to compare the trends in two names using solid graphs that do not overlap with one another. You may change the layout and annotations of the graph to create an effective design.

Preliminary design review
The preliminary design review is worth 10% of this project. For the preliminary design review, you should write a description of the project design structure as you did in Lab 8. Specifically, you should sketch the contents of the program, breaking it down into **data** and **functions** and then order the functions into a program structure. Your sketch should contain the following elements:

1. Data: what data is required by the project
   Variable names w/ description of its purpose
   Data type of each variable (string, int, list, etc.)

2. Algorithms:
   High-level processes the program needs to have.

3. Full Program pseudocode:

● Arrange the various functions in order that they need to happen.
● Include loops and some notion of when they terminate.
● Place the necessary functions in the loops.
● The pseudocode should contain a mix of high and low level items. Some aspects you might not solve until you start coding, so it's fine to give a high-level description.

Explain your project design structure to a member of the COS 125 staff during office hours before 5pm on Friday November 12. Here is the schedule of staff office hours:

Penny: Mon 3:15-5pm, Wed 10:30-12pm, BD 348A

Zach: Tues 10:30am-12:30pm; Thurs 10:30-11am and 1:30-2:30; BD 138

Allison: Fri 12-2pm; BD 138

Sam: Mon 12:30-1:30pm, Fri 1:30-2:30, BD 138

Dharani: Wed 9-11am; BD 138

Hannah: Fri 12:30-3:30pm, BD 138

How to turn in your homework

Turn in each program in its own file. When turning in your own assignment make sure to add your last name to the file name (for example: Rheingans_p1.py). You should not turn in the training file. If you do the extra credit, submit two files: YourName_p1.py with the base assignment and YourName_p1-xc.py with the extra-credit version.