

How to Run

To run this program from the terminal, enter `python Lab10.py`.

Inputs


The program does not require inputs to run, but it is interactive.

Interactive Elements

- Clicking will exit the welcome screen
- Pressing `m` draws the planet Mars
- Pressing `return` animates the orbital motion of Mars
- Pressing `escape` will close the window and quit the program

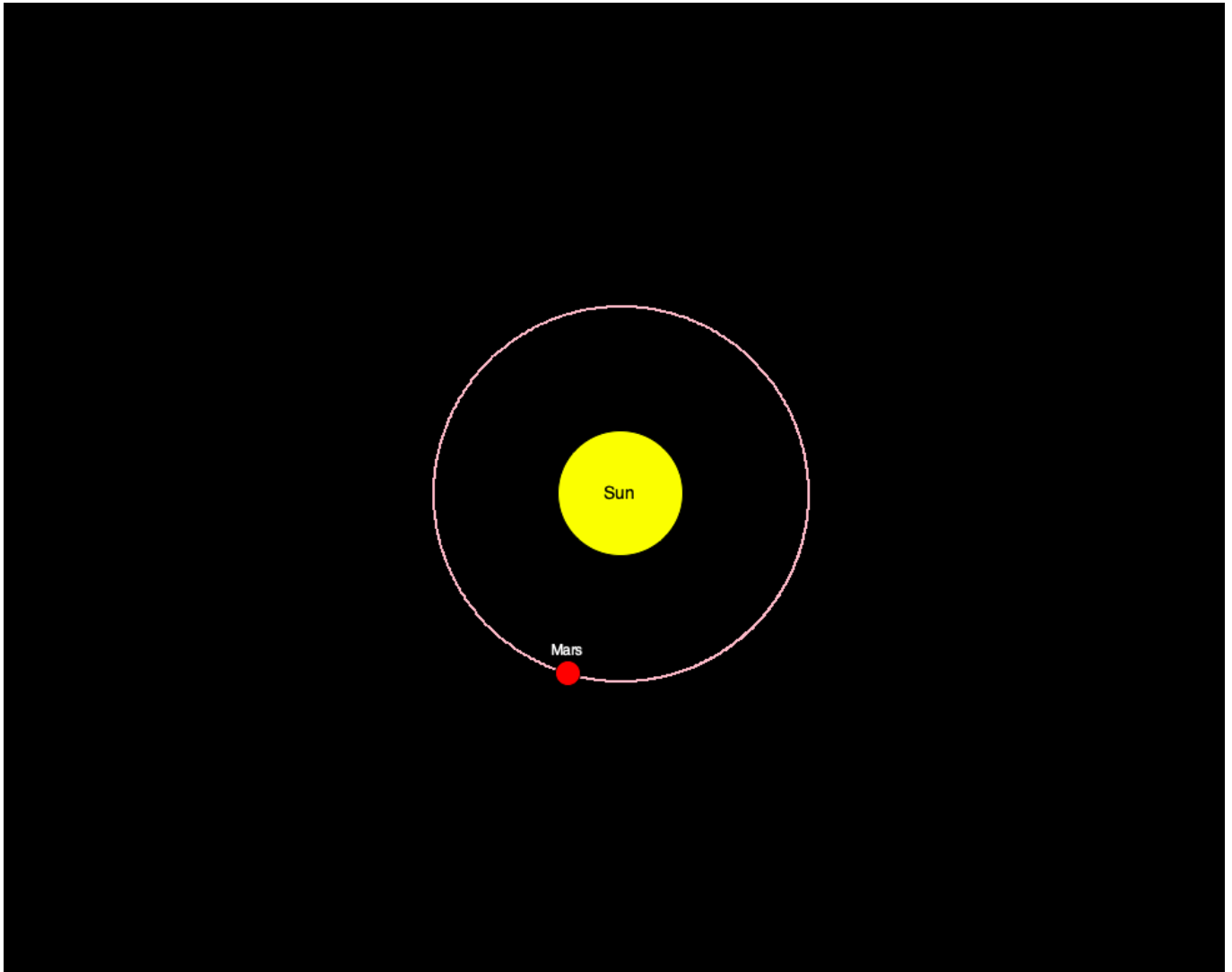
Outputs

The program creates an animated simulation of a simplified version of Mars' orbit around the Sun. The simulation is *not* to scale.



Click to start
To add planets:
Press 'm' for Mars
Press 'return' to orbit
Press 'esc' to quit

The welcome screen with instructions for how to interact with the simulation.



A still shot of Mars orbiting the Sun.

Findings

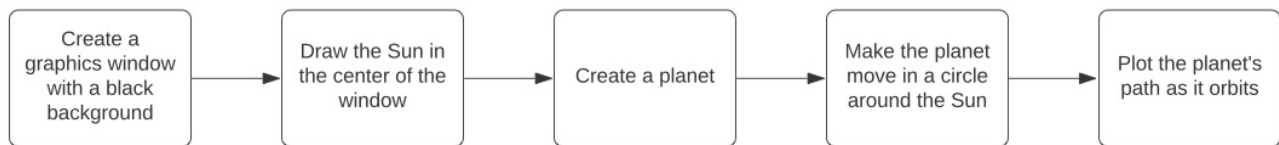
- This is the most exciting thing I've done in Python to date.
- Animating circular motion using `graphics.py` is pretty tricky.
- Having a `Planet` class will be extremely useful as I continue to develop this simulation by adding more planets!
- I may create an `orbit` function within the `Planet` class in future implementations.
- I hope NASA's programmers earn generous salaries.

Concept Map

To prepare for this lab, I created a simple flowchart.

Lab 10 Prep: Interactive Animation

Cori Hatley | November 10, 2020



Orbital motion was the most challenging element of this implementation. Furthermore, the orbital motion in this animation is an *extremely simplified version* of Mars' *actual* orbit. Planets don't orbit in perfect circles. My next hurdle as I continue to develop this program will be to simulate more realistic orbital motion using Newton's Universal Law of Gravitation.

Collaborators

Devon Gardner helped me create my Planet class. Devon also helped me debug my orbital motion, because Mars initially decided to orbit its own starting point and then crash into the Sun.

References

Downey, A.B. (2002.) Labs. In Miller, B., & Ranum, D. (Interactive Ed.), *How to Think Like A Computer Scientist: Astronomy Animation*. Runestone Interactive Project, Luther College.

<https://runestone.academy/runestone/books/published/thinkcspy/Labs/astronomylab.html>

This lab in our interactive textbook served as the inspiration for this program.

Eaton, C. (2020.) *Rain Interactive*. New College of Florida.

I borrowed Professor Eaton's `welcome_screen` function in my implementation.

Stack Overflow. (2017, Jan 9.) *Get Circle to Orbit with Given Equations Using Python Graphics Module*.

<https://stackoverflow.com/questions/22695590/get-circle-to-orbit-with-given-equations-using-python-graphics-py-module>

For some reason, I had a mental lapse when it came to converting circular motion into a linear equation. This source helped me to realize that I simply needed to calculate points along the circular orbit and move the planet to those new points. The motion equations I implemented in my program are borrowed from this solution, but they still required significant debugging.

Zelle, J. *Graphics Reference*.

I referenced this source when creating `graphics.py` objects and using the `graphics.py` package's mutator methods.