CS 151
Spring 2023
Final Project

Interactive Environmental Art
Preview Check in: Friday April 28th in lab (Make up for CLAS lab April 27th)
Project and Report Due: Wed-Fri May 3-5, 2023 11:59pm

This project is an assessment of your ability to meet 2 major course objectives:
LO 2. Students can convert a problem statement into a working solution that solves the problem.

LO 3. Students understand abstraction and can break down a program into appropriate procedural and object-oriented components.

Problem Statement:
You have been commissioned to design and implement a new interactive environmental artwork minimum of 3 scenes around a climate change theme of your choice. One of the scenes should be based on the graph/data you selected for the environmental art lab.
In addition, your client has specified that the scenes allow the viewer to interact with the artwork in some way. Your 3 scenes for the client should be as seamless as possible which means your main code should ideally run from one file, using file management methods to create a smooth interaction experience for the user.

The goals of this final project are to demonstrate:

- **understanding and mastery of course coding skills, CS concepts, and best practices in code organization and documentation.**
- **create complex, interactive program(s) that adapt modular elements from previous projects in novel and creative ways.**
- **apply best practices in top-down OO design methods in the program composition.**

The project must demonstrate:

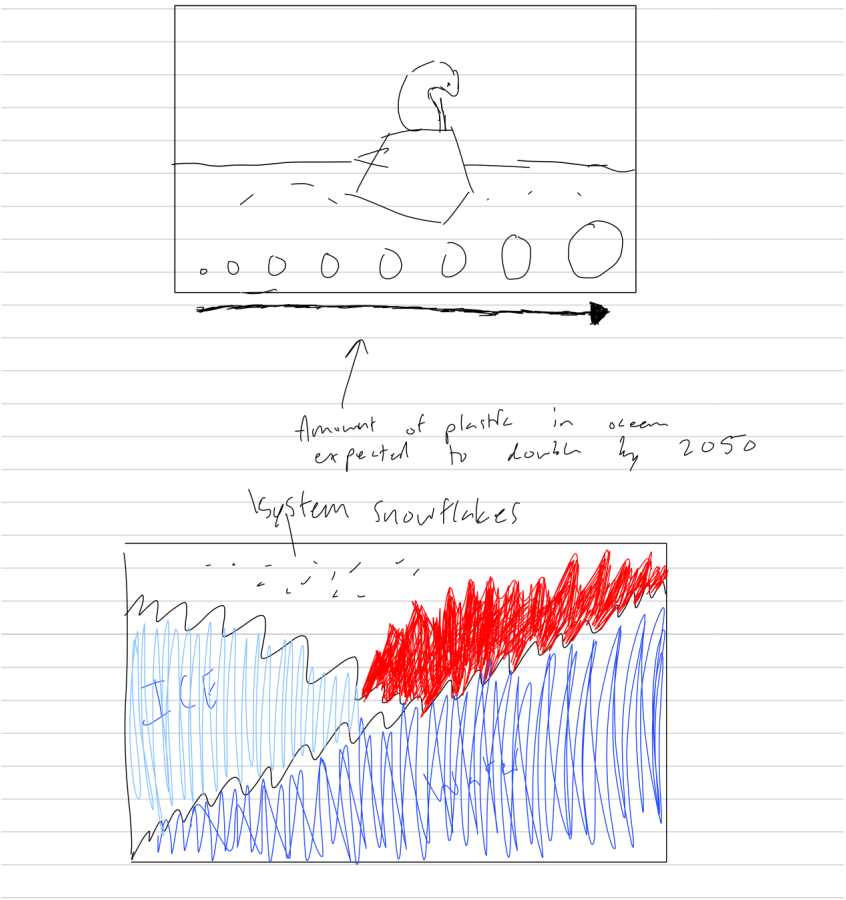
- **Modularity** making functions work as independent units that you can re-use at any time. This means making small, modular functions with keyword parameters that contribute to larger composite functions with test functions for composite shapes and scenes.
- **User Interaction:** Use conditionals, user input prompts, and command-line arguments to add nuance and interactive features to your environmental artwork.
- **Best practices for code organization:** File docstrings, function docstrings, file comments, instructions for user related to interactive features (command line or user input text), Effective use of main code to if __name__ == '__main__': in files.
- **Function complexity:** randomization, return values, and multiple assignments, use lists to effectively organize multiple complex objects, dictionaries, sets, tuples, recursion.
- **Encapsulation:** (using accessor and mutator methods) to separate how data is stored from how it is used in Classes and Objects.

Choose 3 out of 4 components/features:

- **Complex Objects and Scenes:** One scene is made with Zelle complex objects (e.g., collections of Zelle graphics primitives). Each complex object will have a function that initializes it (__init__) . Complex objects that change will have a function that controls it (e.g., changes it, animates it, etc.) and a function that tests it.
- **Recursion:** At least one use of a recursive function to perform some type of analysis in your solution. You can use recursive L-systems or another recursive image.
- **L-system and Interpreter Classes:** One scene that uses a turtle interpreter and L-system classes that converts strings into fractal shapes, trees, and other turtle graphics. The functions Your top-level program will include both the L-system and turtle_interpreter modules. The drawString method will demonstrate it can handle five additional characters in the strings.
- **Inheritance and Polymorphism:** One part of the project that uses parent and child classes with accessor and mutator methods to separate how data is stored from how it is used in Classes and Objects. Remember that Polymorphism is using the same operations on objects of different types but getting unique behavior appropriate for each object. You do not need to use the L-system classes or inheritance to demonstrate this -- using turtle shapes would be a potentially simpler implementation of this task.

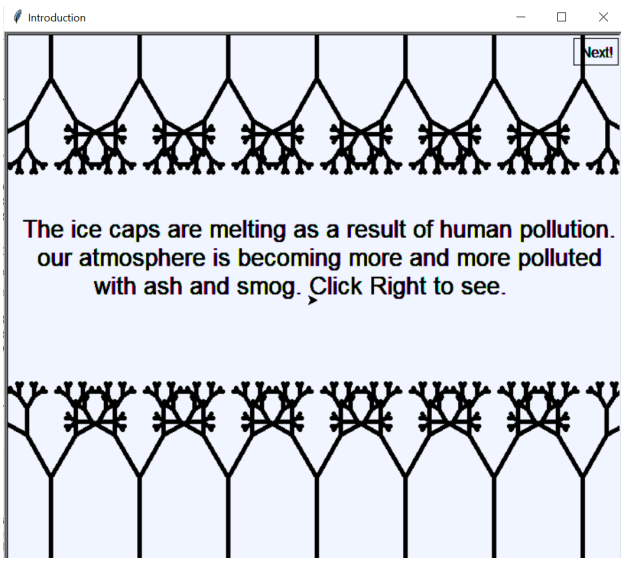
Abstract of Interactive Environmental Artwork and Your Theme:
This project is inspired by the ever-evolving problem of gas emissions. It is causing extreme temperatures and melting of the polar ice caps. This project effectively demonstrates the current situation of melting ice, elevating water levels, and atmospheric pollution, as well as the effect that these have on arctic populations. These are exhibited through Turtle and Zelle graphic modules, with various other concepts incorporated, such as L-systems and recursive functions. Object creation is used throughout the project as well.

Environmental Artwork Design Sketch Image: (from Jill Pelto lab)

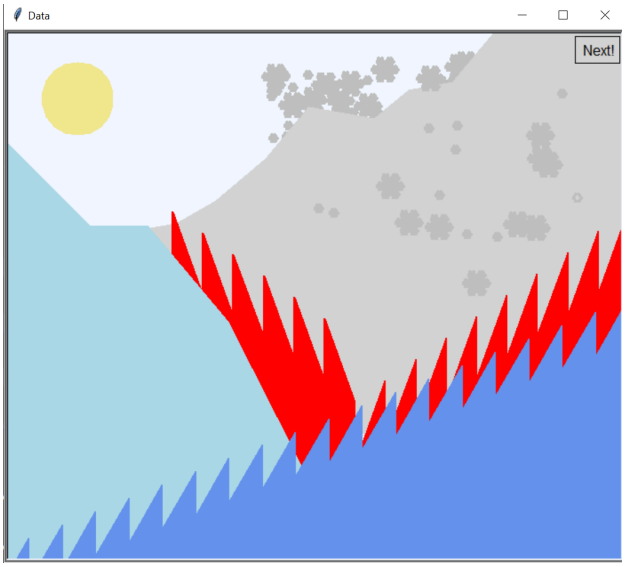


Required Images and Results Sections:

Required Image:	Results: Describe what course concepts and skills you specifically used to create each scene ****PROGRAM RUNS FROM MAIN.PY****
Required Image: Scene 1 	This scene was made primarily with turtle methods that we have used throughout the semester. Commands like forward, right/left, penup/pendown, goto, and many more were used to draw here. Additionally, text was written using the turtle write command. I also used recursion to create the design around the edges of the screen. The recursive functions use turtle commands to draw them. Finally, this scene, along with all the following ones, uses button I/O to navigate through the program. Additionally, all the turtle screens in this project use key press commands to exit the program.
Required Image: Scene 2	This scene is similar to the last in it's structure and design. It uses all the same turtle commands, recursion, and I/O techniques to create what is displayed. Turtle write command was used to give some information in the form of text.



Required Image:
Scene 3

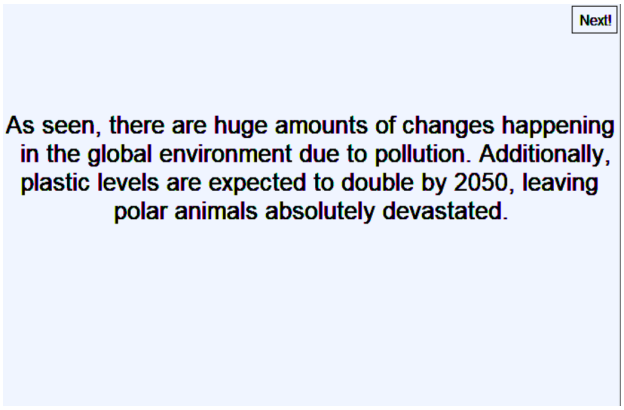


Turtle skills

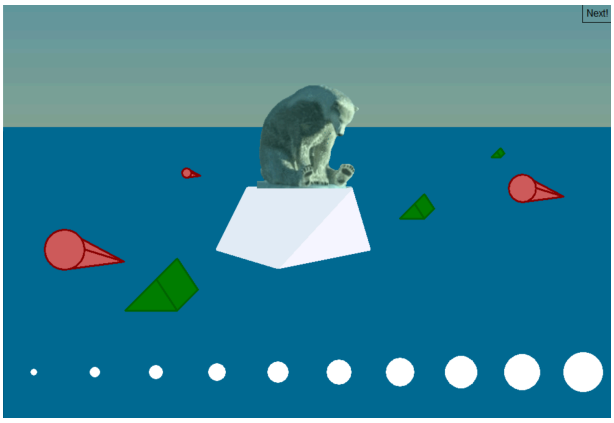
This scene uses turtle commands again to draw the various shapes. The light blue demonstrates icebergs melting, and the dark blue is water levels elevating in that upwards trend as seen. The snowflakes correspond to atmospheric pollution and smog buildup, and they are made with L-systems, using a variation of the Koch-snowflake. Classes are used to set up the L-system, which contains all the mutators and accessors (setters and getters). There is also a turtle interpreter in the zip file which reads the .txt file and builds the L-system from the base and rule given in turtle.

Extension Scenes (if any, though not required)

This scene is similar to the first and second. It again uses all the same turtle commands and I/O techniques to create what is displayed.



This scene is built using Zelle graphics. I utilized the shape methods and imported an image as well. I also used a fairly simple equation to make the gradient in the sky. The circles at the bottom represent levels of pollution in the ocean which are anticipated to increase detrimentally. They also have some animation and the "Next!" button shows up after this animation finishes.



This scene is also made using Zelle graphics. It is simpler than the last in terms of what is shown. It gives some concluding remarks and can be closed at any time by pressing the button in the top right. Additionally, I made used the same equation from the sky gradient in the last scene to make a cool background gradient in this one.

Personal Reflection:

How does this project demonstrate how much you have learned about introductory CS concepts, good programming practices, computational problem solving?

This project serves as a culmination of everything I have learned in intro CS. My project reflects good organizational skills, and while I could have made some things more efficiently, my project is also fairly efficient. I use encapsulation to make functions shorter and more efficient. I use Turtle throughout the project and Zelle as well. I also use more difficult concepts like recursion and L-systems. While these were very difficult, I was able to conceptualize and incorporate them into my program. I also use while and for loops in my project, along with conditionals and if-else statements.

Knowing what you know now, is there anything you might have done differently in this course during the semester?

Knowing what I know now if I were to do anything differently in the semester, I would make sure to come to all classes. While I was able to submit all projects and do extensions on many, I think I could have made my projects even better had I come to certain classes. This would have made certain concepts easier to understand and incorporate into my work. Otherwise, I am very happy with my work this semester and am super proud of myself and my progression through computer science.

Sources of support for this project and/or for the course:
Who helped you achieve your final project, who supported you over the course of the semester? It is important to acknowledge and thank people who have helped us along the way. If you had a TA or other supporter this semester, in addition to including it into this report, consider writing them a short email of gratitude. It will mean more than you know!

I would like to thank Professor Doore for her help and support throughout the semester. I also would like to thank the TAs and my lab partner Tracy. They all helped significantly!

Sources:
CS151 A – Google Classroom: Lectures, Colaboraties, and Readings
https://trinket.io/docs/colors
https://th-thumbnailer.cdn-si-edu.com/-lJ46q8XgForRnGHWmg_OodVKzY=/fit-in/1

072x0/https://tf-cmsv2-smithsonianmag-media.s3.amazonaws.com/filer/a7/dd/a7dd4409-2692-4048-8f84-007dc9e14ac6/landscape_of_change.jpg
https://mcsp.wartburg.edu/zelle/python/graphics/graphics.pdf
https://www.w3schools.com/python/python_variables_global.asp