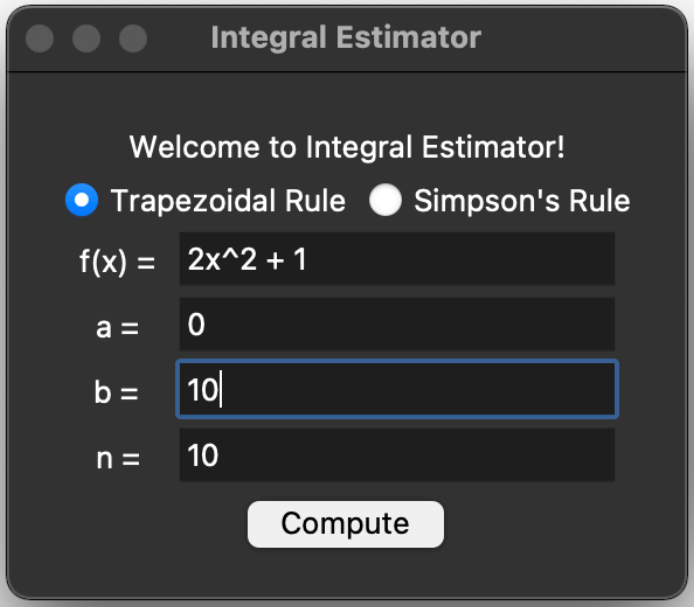
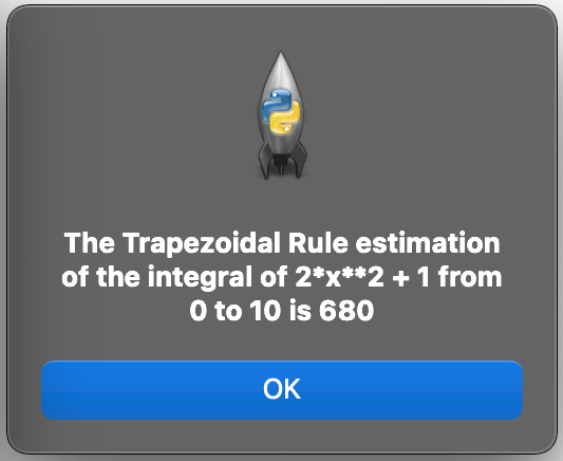
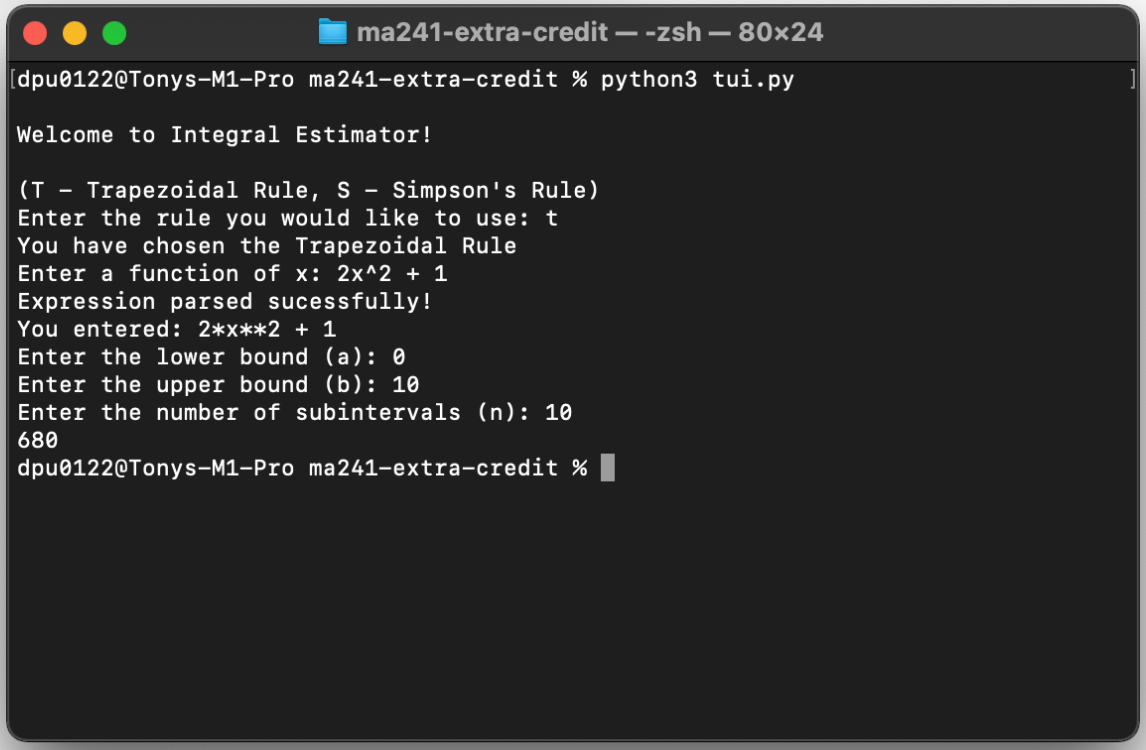
This is a Python program with a graphic user interface and a terminal user interface. The UI's should looks like the following screenshot images:







Python version: Python 3.10.1

Program should work in all Python 3 versions in theory, in case it doesn't use the version listed above.

Python modules required: `tk`, `decimal`, `sympy`

The program may not run without the dependencies listed above.

The main program contains two files, `gui.py` and `tui.py`. `gui.py` is built using the tkinter module. `tui.py` have a command line interface. In case one fails, the other one is used as a backup.

Terminal command for running the program in ma241-extra-credit-submission directory:

`python3 gui.py` or `python3 tui.py`

A standalone macOS .app file is also included in the zip file, for easier execution in macOS. I do not have access to a computer running Windows, so unfortunately a Windows .exe file is not included.

Objective: provide a program (with graphical user interface) that computes the approximation of an integral given an equation, a lower bound, an upper bound, and the number of subintervals using the Trapezoid rule or Simpson's Rule.

The program should accept implicit multiplication and the caret sign. The parsing process should be fairly lenient, parenthesis are allowed, spaces are not required, and it should follow the correct order of operations and simplify the expression automatically. x should be the only variable in the expression, or a warning will be shown.

Backend Methodology:

```
class IntegralEstimator:
      __init__(self, rule, a, b, n, fx):
            rule: “T” or “S”
            a: lower bound
            b: upper bound
            n: number of subintervals
            fx: a list of f(x) values calculated using evaluate_fx
      trap_estimate(self):
            get the sum of all f(x) values using a for-loop:
                  multiply every value by 2 except the first and last
            use the formula (b - a) / n / 2 * sum to compute the result
      simp_estimate(self):
            get the sum of all f(x) values using a for-loop:
                  multiply every second value by 2 or 4 except the first
and last
            use the formula (b - a) / n / 3 * sum to compute the result


class Fx:
      __init__(self, expr):
            expr: a string entered by the user
      is_valid_fx(self):
            try to parse expr using sympy methods and
            return true if successful and false if otherwise
      get_fx(self):
            return the successfully parsed expression
      evaluate_fx(self, x_value):
```

evaluates f(x) given an x value

Class IntegralEstimatorGUI:

run(self):

runs the graphical user interface

interface(self):

sets up the graphical user interface

calculate(self):

retrieve user input from text boxes and radio buttons

check the validity of f(x) using is_valid_fx

check if a is less than b

check if n is a positive integer

(positive even integer for Simpson's Rule)

calculate f(x) at each subinterval and store them in a list

pass rule, a, b, n, and the list of f(x) values into IntegralEstimator

calculate using trap_estimate or simp_estimate depending on the rule

display the result using a message prompt

TUI is composed of sequential code instead of a class but works similar to GUI.

Conclusion: after carefully testing my program, it should handle most user input errors and give an accurate approximation of a definite integral depending on the number of subintervals.

This project took me a long time to finish. I gained more insights in Python and the numerical integration methods. I enjoyed the entire creative process and I hope you've enjoyed using it as much as I did.