[redacted]

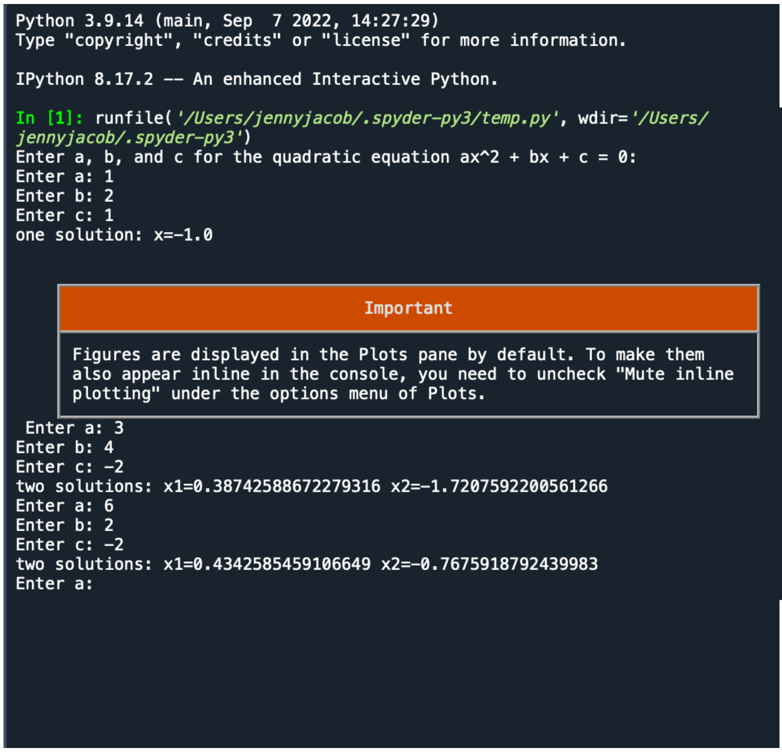
[redacted]

[redacted]

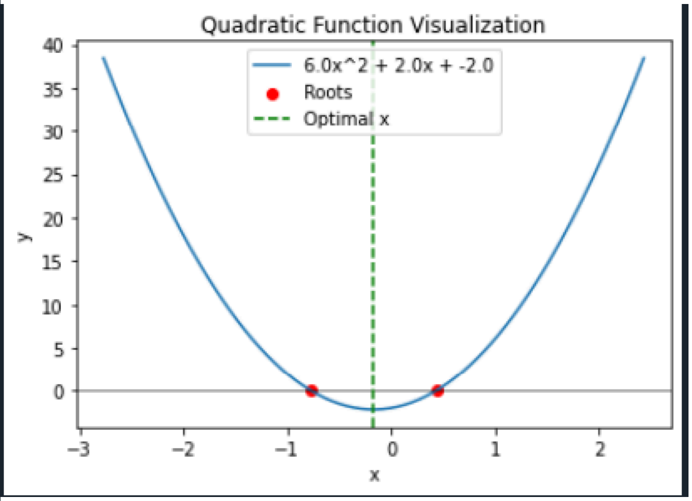
[redacted]

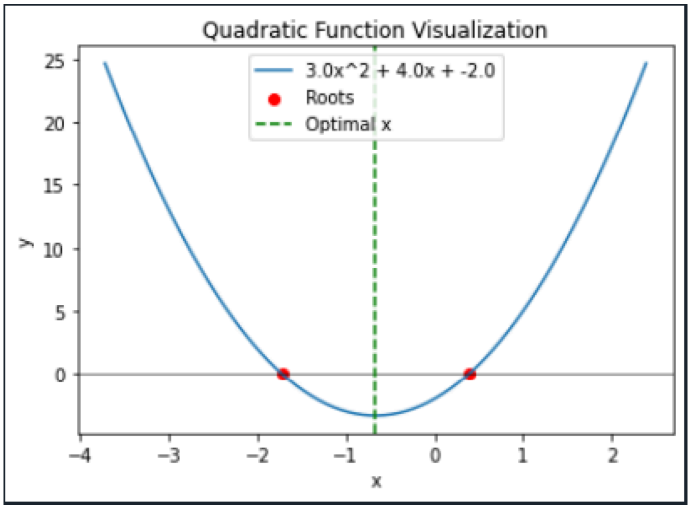
Problem 1. Quadratic Equations:

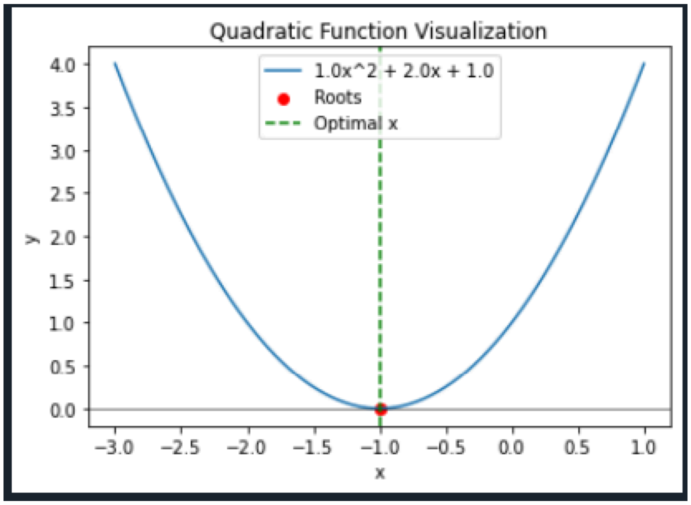
```
import matplotlib.pyplot as plt
import numpy as np
import math
import sys


def discriminant(a, b, c):
    return b**2 - 4*a*c


def calculate_solution(a, b, c):
    d = discriminant(a, b, c)
    if d > 0:
        x1 = (-b + math.sqrt(d)) / (2*a)
        x2 = (-b - math.sqrt(d)) / (2*a)
        print(f'two solutions: x1={x1} x2={x2}')
        return x1, x2
    elif d == 0:
        x = -b / (2*a)
        print(f'one solution: x={x}')
        return x, x
    else:
        print('no real solutions')
        return None, None


def plot_quadratic(a, b, c, roots):
    x_opt = -b / (2*a)

    if roots is not None:
        x_range = np.linspace(min(roots) - 2, max(roots) + 2, 150)
    else:
        x_range = np.linspace(x_opt - 2, x_opt + 2, 150)

    y = a*x_range**2 + b*x_range + c

    plt.plot(x_range, y, label=f"{a}x^2 + {b}x + {c}")

    if roots is not None:
        plt.scatter(roots, [0]*len(roots), color='red', marker='o', label='Roots')

    plt.axhline(0, color='black', linewidth=0.5)
    plt.axvline(x_opt, color='green', linestyle='--', label='Optimal x')

    plt.legend()
    plt.xlabel('x')
    plt.ylabel('y')
    plt.title('Quadratic Function Visualization')
    plt.show()

print('Enter a, b, and c for the quadratic equation ax^2 + bx + c = 0:')

while True:
    try:
        a = float(input('Enter a: '))
        b = float(input('Enter b: '))
        c = float(input('Enter c: '))

        x1, x2 = calculate_solution(a, b, c)
        plot_quadratic(a, b, c, [x1, x2] if x1 is not None else None)

        # Simulate user typing CTRL-Z on Windows to finish the program
        if a == 1 and b == -1 and c == -6:
            sys.exit()

    except ValueError:
        print('Please enter a valid number.')
```



```
Python 3.9.14 (main, Sep  7 2022, 14:27:29)
Type "copyright", "credits" or "license" for more information.

IPython 8.17.2 -- An enhanced Interactive Python.

In [1]: runfile('/Users/jennyjacob/.spyder-py3/temp.py', wdir='/Users/
jennyjacob/.spyder-py3')
Enter a, b, and c for the quadratic equation ax^2 + bx + c = 0:
Enter a: 1
Enter b: 2
Enter c: 1
one solution: x=-1.0


                              Important

    Figures are displayed in the Plots pane by default. To make them
    also appear inline in the console, you need to uncheck "Mute inline
    plotting" under the options menu of Plots.
 Enter a: 3
Enter b: 4
Enter c: -2
two solutions: x1=0.38742588672279316 x2=-1.7207592200561266
Enter a: 6
Enter b: 2
Enter c: -2
two solutions: x1=0.4342585459106649 x2=-0.7675918792439983
Enter a:
```

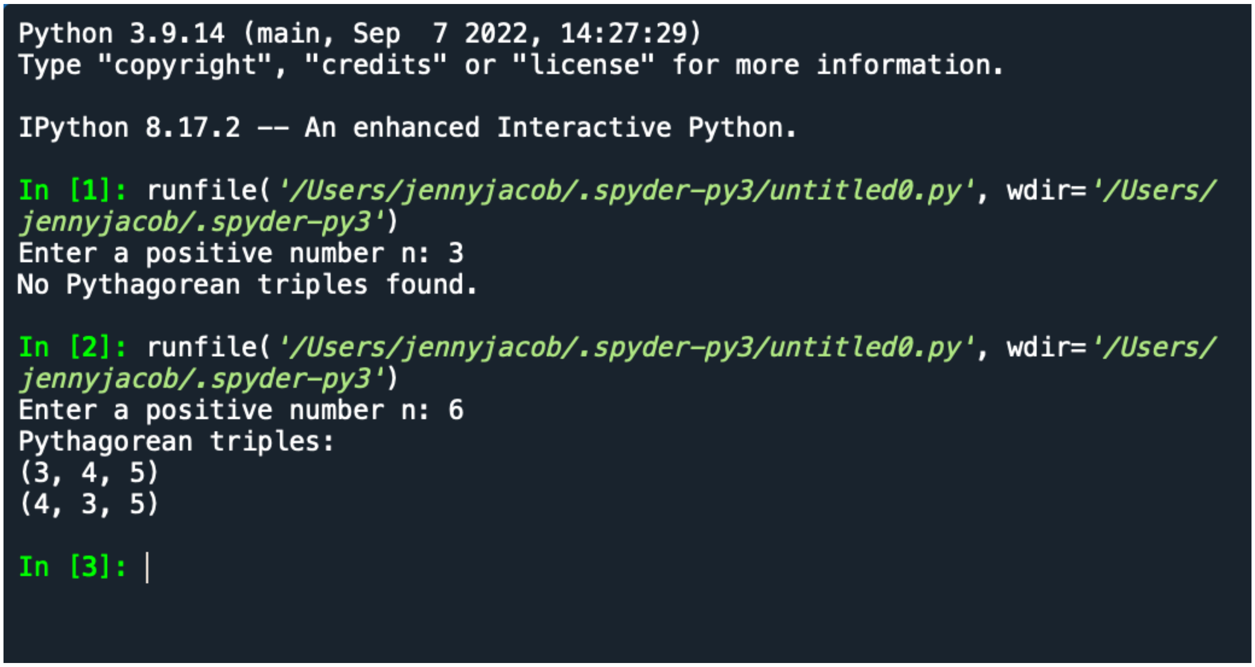






Problem 2. Pythagorean Numbers:

```
def find_Pythagorean(n):
    pythagorean_triples = []
    for a in range(1, n+1):
        for b in range(1, n+1):
            c = (a**2 + b**2)**0.5
            if c.is_integer() and c <= n:
                pythagorean_triples.append((a, b, int(c)))
    return pythagorean_triples


def main():
    try:
        n = int(input("Enter a positive number n: "))
        if n <= 0:
            print("Please enter a positive number.")
            return


        triples = find_Pythagorean(n)


        if not triples:
            print("No Pythagorean triples found.")
        else:
            print("Pythagorean triples:")
            for triple in triples:
                print(triple)

    except ValueError:
        print("INot a positive number. Please enter a positive number.")

if __name__ == "__main__":
    main()
```

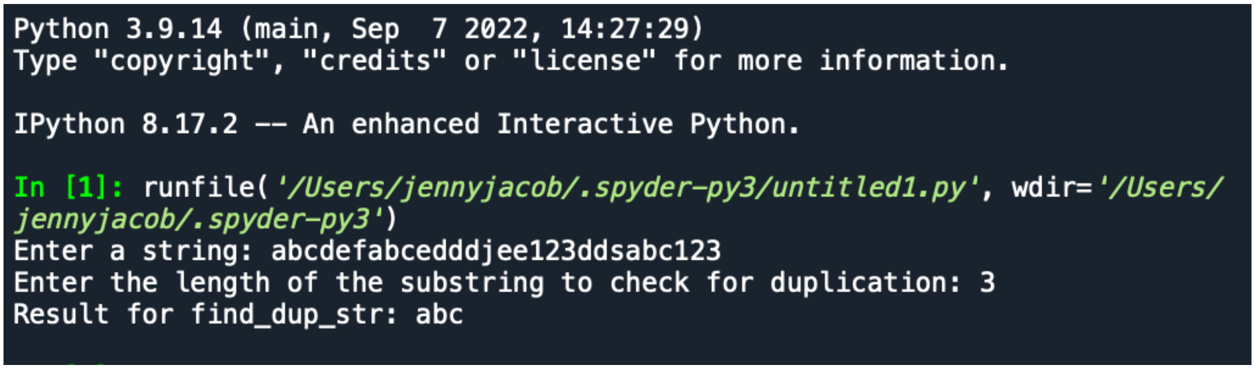


```
Python 3.9.14 (main, Sep  7 2022, 14:27:29)
Type "copyright", "credits" or "license" for more information.

IPython 8.17.2 -- An enhanced Interactive Python.

In [1]: runfile('/Users/jennyjacob/.spyder-py3/untitled0.py', wdir='/Users/
jennyjacob/.spyder-py3')
Enter a positive number n: 3
No Pythagorean triples found.

In [2]: runfile('/Users/jennyjacob/.spyder-py3/untitled0.py', wdir='/Users/
jennyjacob/.spyder-py3')
Enter a positive number n: 6
Pythagorean triples:
(3, 4, 5)
(4, 3, 5)

In [3]: |
```

Problem 3. Duplicated Substrings:

a)

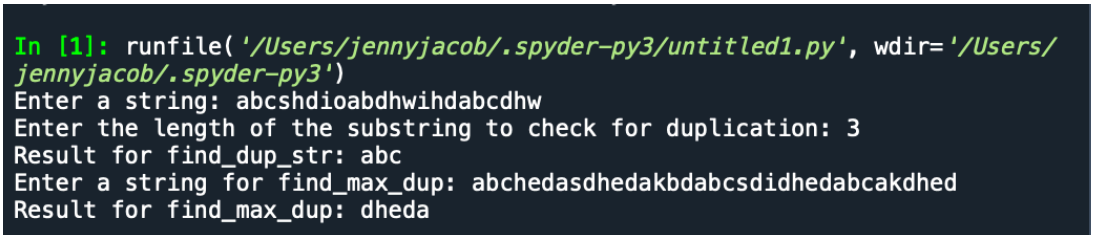
```
def find_dup_str(s, n):
    for i in range(len(s) - n + 1):
        substring = s[i:i+n]
        rest_of_string = s[i+n:]
        if substring in rest_of_string:
            return substring
    return ""


# Testing the function
if __name__ == "__main__":
    s_input = input("Enter a string: ")
    n_input = int(input("Enter the length of the substring to check for duplication: "))


    result = find_dup_str(s_input, n_input)
    print(f"Result for find_dup_str: {result}")
```



```
Python 3.9.14 (main, Sep  7 2022, 14:27:29)
Type "copyright", "credits" or "license" for more information.

IPython 8.17.2 -- An enhanced Interactive Python.

In [1]: runfile('/Users/jennyjacob/.spyder-py3/untitled1.py', wdir='/Users/
jennyjacob/.spyder-py3')
Enter a string: abcdefabcedddjee123ddsabc123
Enter the length of the substring to check for duplication: 3
Result for find_dup_str: abc
```

b)

```
def find_max_dup(s):
    max_length = 0
    max_duplicate = ""

    for length in range(1, len(s)):
        duplicate = find_dup_str(s, length)
        if duplicate:
            max_length = length
            max_duplicate = duplicate

    return max_duplicate

# Testing find_max_dup
if __name__ == "__main__":
    s_input_b = input("Enter a string for find_max_dup: ")

    result_b = find_max_dup(s_input_b)
    print(f"Result for find_max_dup: {result_b}")
```

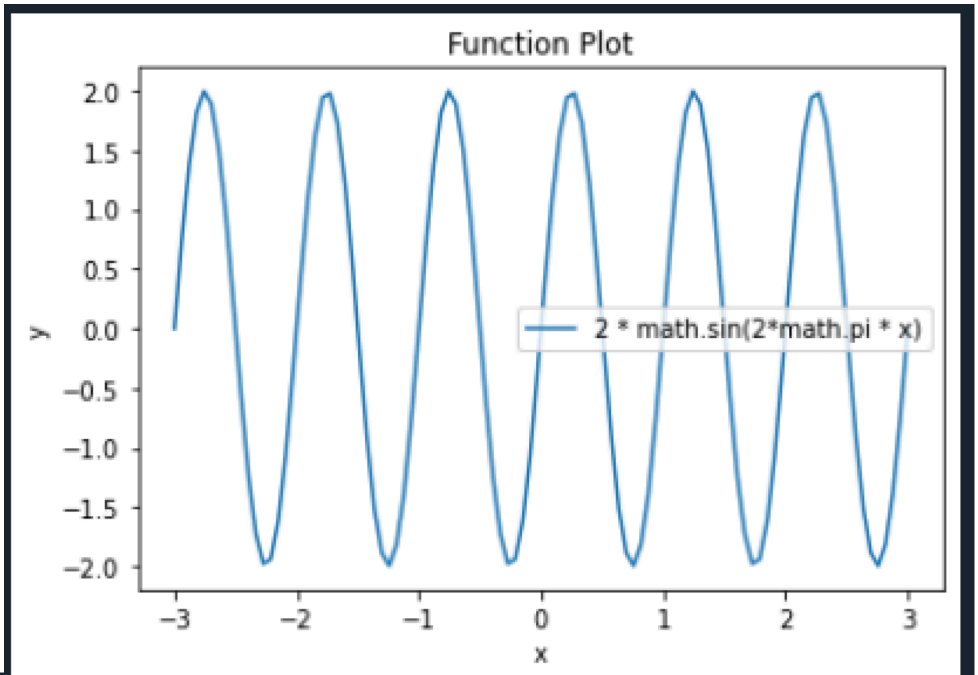


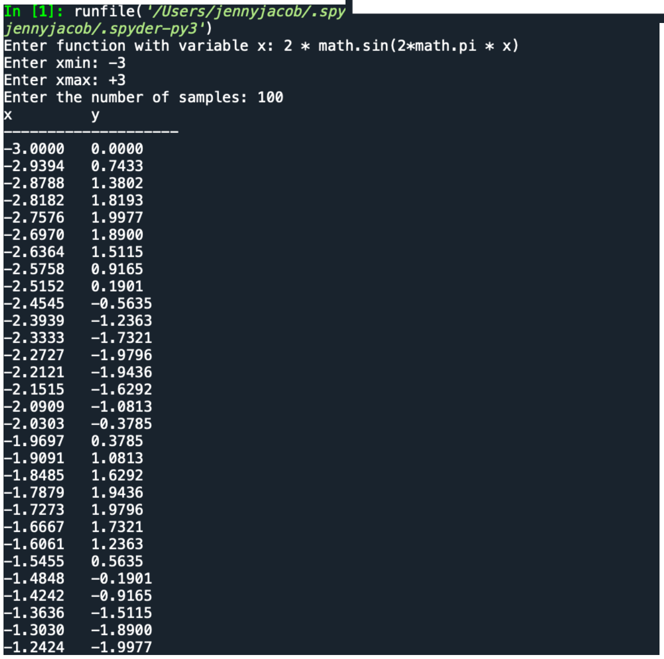
```
In [1]: runfile('/Users/jennyjacob/.spyder-py3/untitled1.py', wdir='/Users/
jennyjacob/.spyder-py3')
Enter a string: abcshdioabdhwihdabcdhw
Enter the length of the substring to check for duplication: 3
Result for find_dup_str: abc
Enter a string for find_max_dup: abchedasdhedakbdabcsdidhedabcakdhed
Result for find_max_dup: dheda
```

Problem 4. Function Visualization:

```
import matplotlib.pyplot as plt

import math


def plot_function(fun_str, domain, ns):

    xmin, xmax = domain

    xs = [xmin + (xmax - xmin) * i / (ns - 1) for i in range(ns)]

    ys = []


    print(f"{'x':<10}{'y':<10}")

    print("-" * 20)


    for x in xs:

        y = eval(fun_str)

        ys.append(y)

        print(f"{x:<10.4f}{y:<10.4f}")


    plt.plot(xs, ys, label=f"{fun_str}")

    plt.xlabel('x')

    plt.ylabel('y')

    plt.title('Function Plot')

    plt.legend()

    plt.show()
```

```
fun_str = input("Enter function with variable x: ")

xmin = float(input("Enter xmin: "))

xmax = float(input("Enter xmax: "))

ns = int(input("Enter the number of samples: "))


plot_function(fun_str, (xmin, xmax), ns)
```





```
In [1]: runfile('/Users/jennyjacob/.spy
jennyjacob/.spyder-py3')
Enter function with variable x: 2 * math.sin(2*math.pi * x)
Enter xmin: -3
Enter xmax: +3
Enter the number of samples: 100
x         y
--------------------
-3.0000   0.0000
-2.9394   0.7433
-2.8788   1.3802
-2.8182   1.8193
-2.7576   1.9977
-2.6970   1.8900
-2.6364   1.5115
-2.5758   0.9165
-2.5152   0.1901
-2.4545   -0.5635
-2.3939   -1.2363
-2.3333   -1.7321
-2.2727   -1.9796
-2.2121   -1.9436
-2.1515   -1.6292
-2.0909   -1.0813
-2.0303   -0.3785
-1.9697   0.3785
-1.9091   1.0813
-1.8485   1.6292
-1.7879   1.9436
-1.7273   1.9796
-1.6667   1.7321
-1.6061   1.2363
-1.5455   0.5635
-1.4848   -0.1901
-1.4242   -0.9165
-1.3636   -1.5115
-1.3030   -1.8900
-1.2424   -1.9977
```

(Not all variables listed in photo)