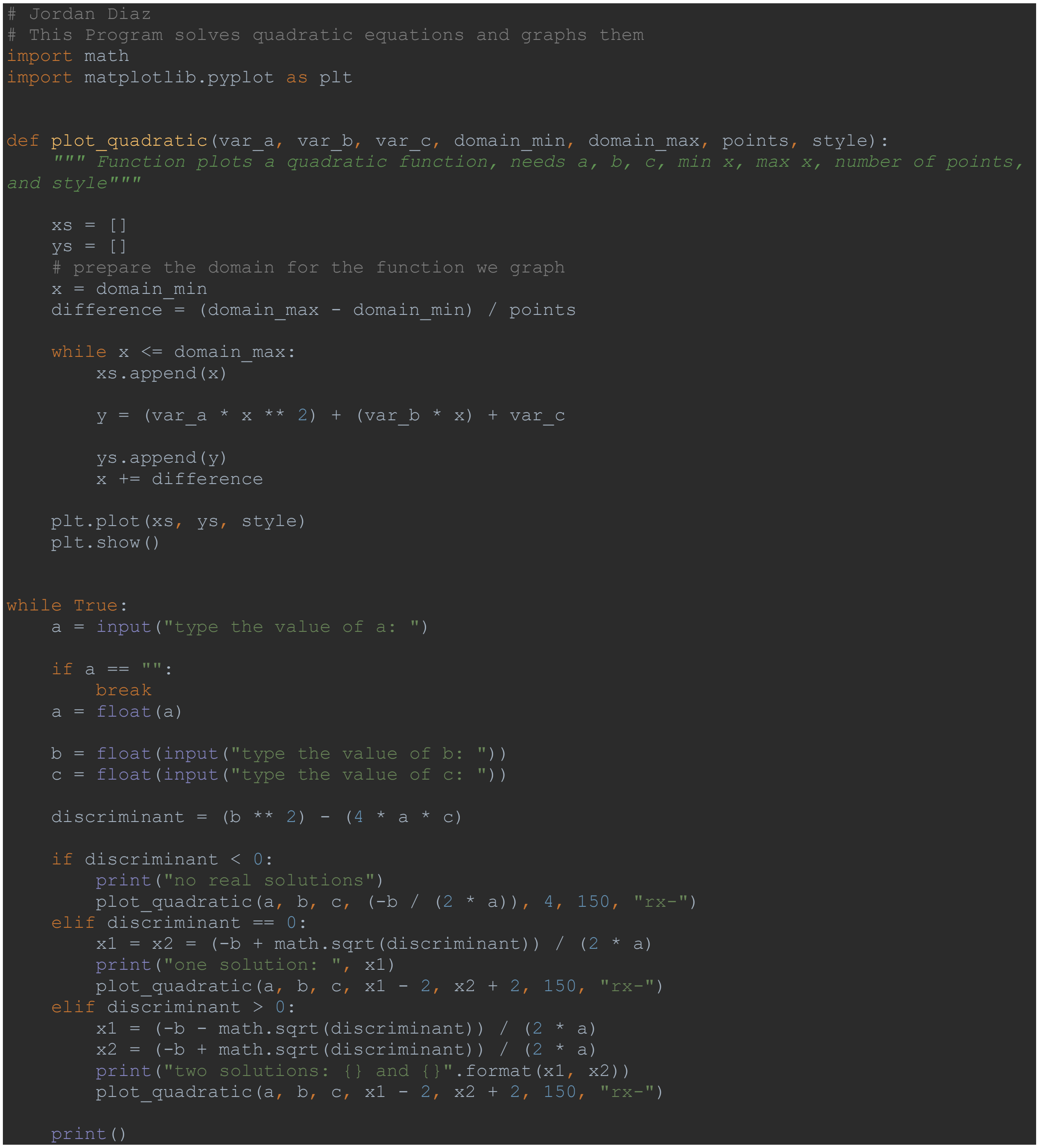
Python Programming

Homework 1

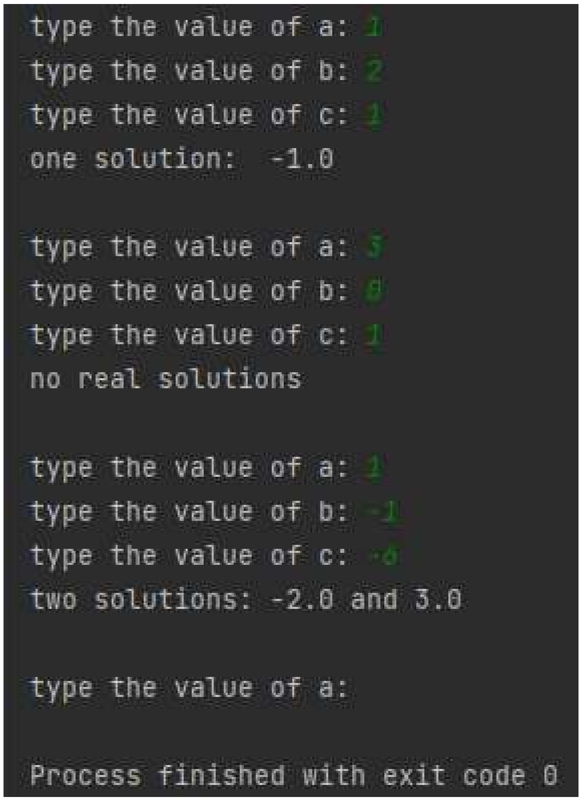
Jordan Diaz

Solution #1:



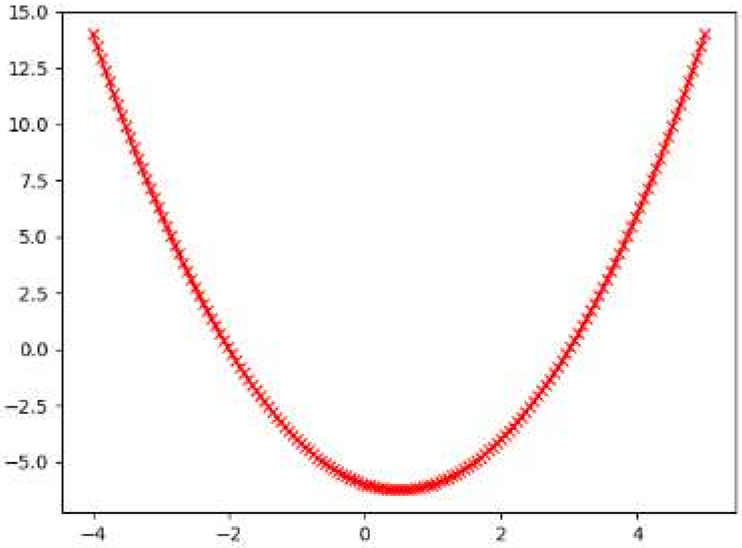
```python
# Jordan Diaz
# This Program solves quadratic equations and graphs them
import math
import matplotlib.pyplot as plt


def plot_quadratic(var_a, var_b, var_c, domain_min, domain_max, points, style):
    """ Function plots a quadratic function, needs a, b, c, min x, max x, number of points,
and style"""

    xs = []
    ys = []
    # prepare the domain for the function we graph
    x = domain_min
    difference = (domain_max - domain_min) / points

    while x <= domain_max:
        xs.append(x)

        y = (var_a * x ** 2) + (var_b * x) + var_c

        ys.append(y)
        x += difference

    plt.plot(xs, ys, style)
    plt.show()


while True:
    a = input("type the value of a: ")

    if a == "":
        break
    a = float(a)

    b = float(input("type the value of b: "))
    c = float(input("type the value of c: "))

    discriminant = (b ** 2) - (4 * a * c)

    if discriminant < 0:
        print("no real solutions")
        plot_quadratic(a, b, c, (-b / (2 * a)), 4, 150, "rx-")
    elif discriminant == 0:
        x1 = x2 = (-b + math.sqrt(discriminant)) / (2 * a)
        print("one solution: ", x1)
        plot_quadratic(a, b, c, x1 - 2, x2 + 2, 150, "rx-")
    elif discriminant > 0:
        x1 = (-b - math.sqrt(discriminant)) / (2 * a)
        x2 = (-b + math.sqrt(discriminant)) / (2 * a)
        print("two solutions: {} and {}".format(x1, x2))
        plot_quadratic(a, b, c, x1 - 2, x2 + 2, 150, "rx-")

    print()
```

Terminal Session for problem #1:

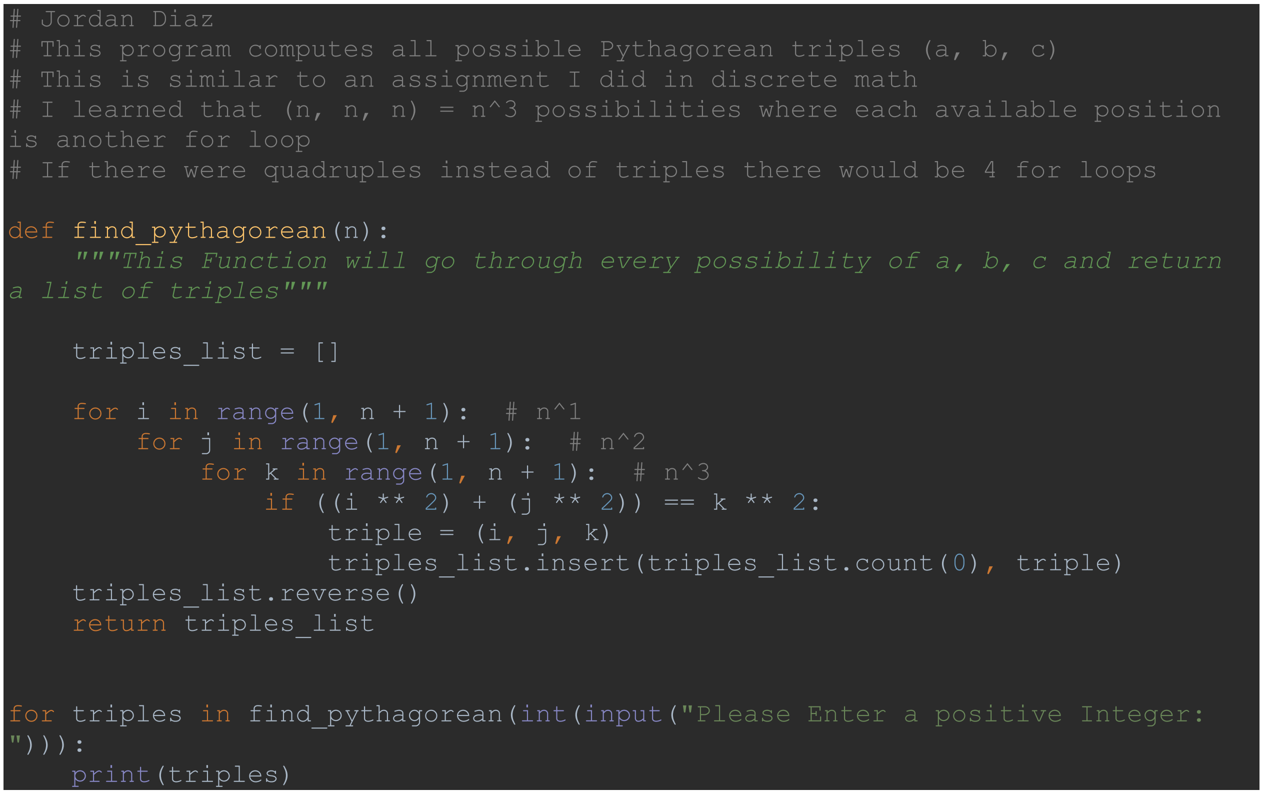


```
type the value of a: 1
type the value of b: 2
type the value of c: 1
one solution:  -1.0

type the value of a: 5
type the value of b: 0
type the value of c: 1
no real solutions

type the value of a: 1
type the value of b: -1
type the value of c: -6
two solutions: -2.0 and 3.0

type the value of a:

Process finished with exit code 0
```

Graph for a = 1, b = -1, c = -6

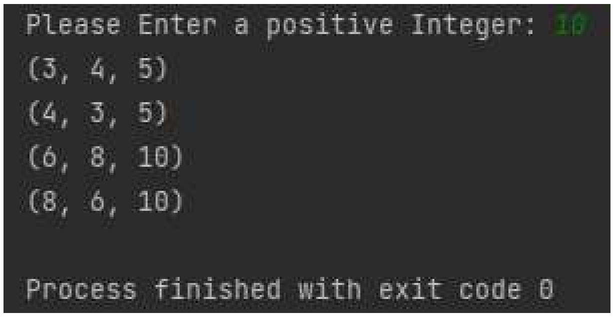


Solution #2:



```
# Jordan Diaz
# This program computes all possible Pythagorean triples (a, b, c)
# This is similar to an assignment I did in discrete math
# I learned that (n, n, n) = n^3 possibilities where each available position
is another for loop
# If there were quadruples instead of triples there would be 4 for loops

def find_pythagorean(n):
    """This Function will go through every possibility of a, b, c and return
a list of triples"""

    triples_list = []

    for i in range(1, n + 1):  # n^1
        for j in range(1, n + 1):  # n^2
            for k in range(1, n + 1):  # n^3
                if ((i ** 2) + (j ** 2)) == k ** 2:
                    triple = (i, j, k)
                    triples_list.insert(triples_list.count(0), triple)
    triples_list.reverse()
    return triples_list


for triples in find_pythagorean(int(input("Please Enter a positive Integer:
"))):
    print(triples)
```

Terminal Session for problem #2

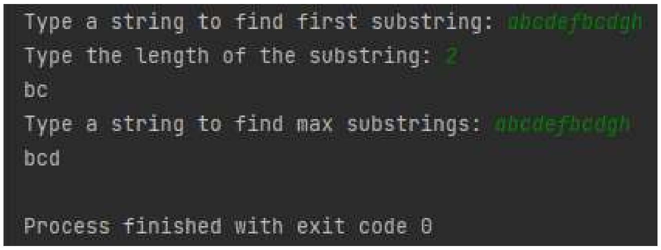


```
Please Enter a positive Integer: 10
(3, 4, 5)
(4, 3, 5)
(6, 8, 10)
(8, 6, 10)

Process finished with exit code 0
```

Solution #3:

```python
# Jordan Diaz
# This Program finds duplicate substring and max duplicate substring
# Cannot use str functions

def find_dup_str(s, n):
    """ This Function determines returns the first occurring substring of
length n (if any) """

    if n >= len(s) or n < 1:
        return ""

    for i in range(0, len(s) - 1):
        curr_str = s[i: i + n]

        for k in range(i + 1, len(s) - 1):
            next_str = s[k:k + n]

            if curr_str == next_str:
                return curr_str
    return ""


def find_max_dup(s):
    """ This Function returns the largest duplicate substring"""

    curr_str = ""
    n = len(s) // 2

    while curr_str == "":
        curr_str = find_dup_str(s, n)
        n -= 1

        if n <= 0:
            break

    return curr_str


print(find_dup_str(input("Type a string to find first substring: "),
int(input("Type the length of the substring: "))))
print(find_max_dup(input("Type a string to find max substrings: ")))
```

Terminal Session for problem #3:

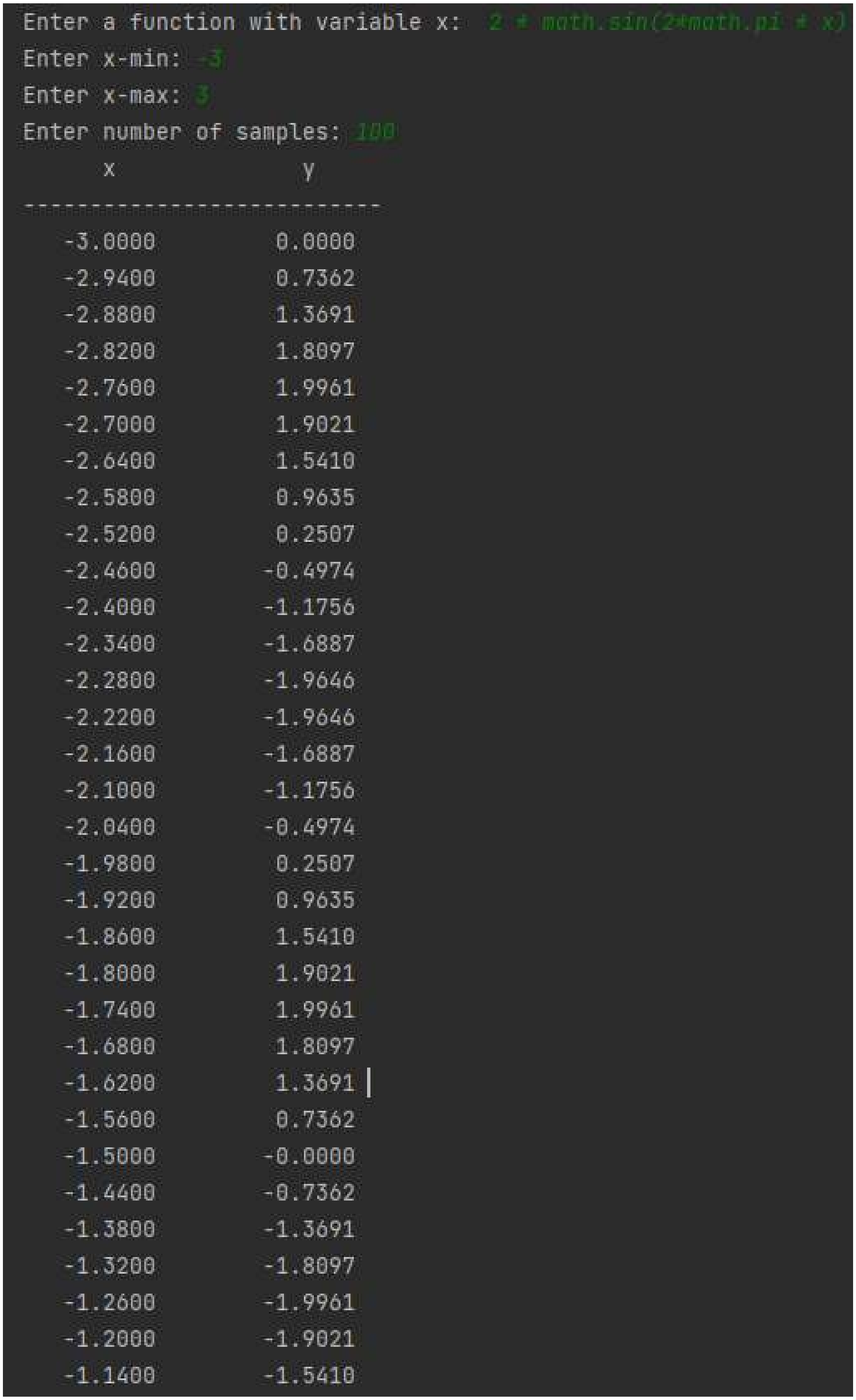


```
Type a string to find first substring: abcdefbcdgh
Type the length of the substring: 2
bc
Type a string to find max substrings: abcdefbcdgh
bcd

Process finished with exit code 0
```

Solution #4:

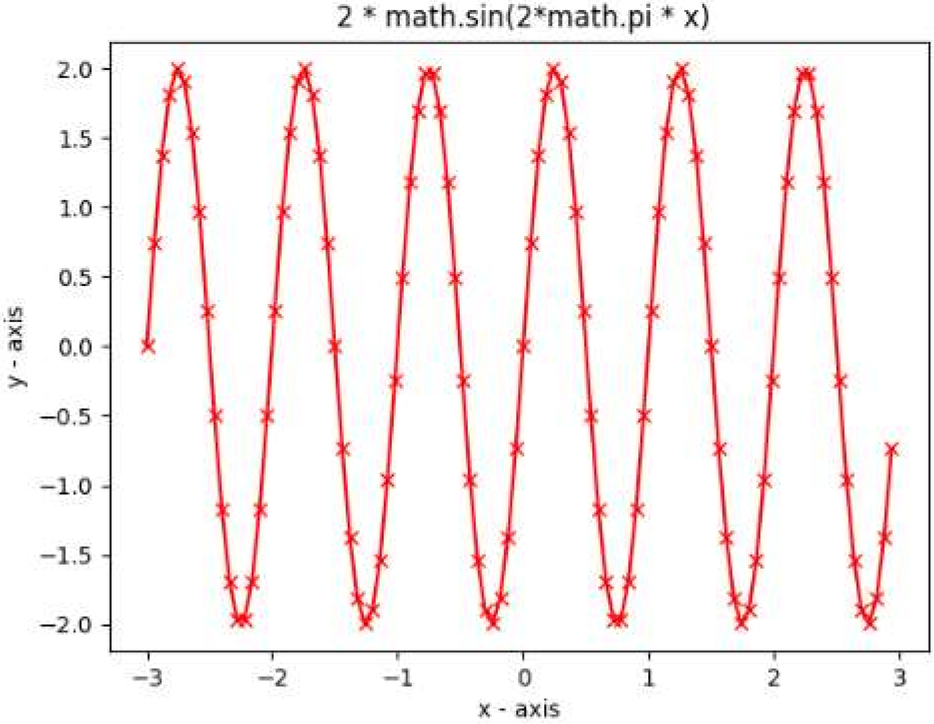
```python
# Jordan Diaz
# This program will plot any function evaluated from the terminal with a
table
import math
import matplotlib.pyplot as plotter


def plot_function(fun_str, domain, points, style):
    """ Function plots a given function"""

    xs = []
    ys = []
    # prepare the domain for the function we graph
    x = domain[0]
    difference = (domain[1] - domain[0]) / points

    while x <= domain[1]:
        xs.append(x)
        x += difference

    for x in xs:
        y = eval(fun_str)
        ys.append(y)

    plotter.plot(xs, ys, style)
    plotter.xlabel("x - axis")
    plotter.ylabel("y - axis")
    plotter.title(fun_str)
    plotter.show()
    print("{:>7s}     {:>10s} ".format("x", "y"))
    print("---------------------------")
    for x in xs:
        y = eval(fun_str)
        print("{:10.4f}     {:10.4f} ".format(x, y))


plot_function(input("Enter a function with variable x: "),
              (float(input("Enter x-min: ")), float(input("Enter x-max: "))),
              int(input("Enter number of samples: ")),
              "rx-")
```

Terminal Session problem #4: (too long to show all)



```
Enter a function with variable x:  2 * math.sin(2*math.pi * x)
Enter x-min: -3
Enter x-max: 3
Enter number of samples: 100
      x              y
---------------------------
   -3.0000         0.0000
   -2.9400         0.7362
   -2.8800         1.3691
   -2.8200         1.8097
   -2.7600         1.9961
   -2.7000         1.9021
   -2.6400         1.5410
   -2.5800         0.9635
   -2.5200         0.2507
   -2.4600        -0.4974
   -2.4000        -1.1756
   -2.3400        -1.6887
   -2.2800        -1.9646
   -2.2200        -1.9646
   -2.1600        -1.6887
   -2.1000        -1.1756
   -2.0400        -0.4974
   -1.9800         0.2507
   -1.9200         0.9635
   -1.8600         1.5410
   -1.8000         1.9021
   -1.7400         1.9961
   -1.6800         1.8097
   -1.6200         1.3691
   -1.5600         0.7362
   -1.5000        -0.0000
   -1.4400        -0.7362
   -1.3800        -1.3691
   -1.3200        -1.8097
   -1.2600        -1.9961
   -1.2000        -1.9021
   -1.1400        -1.5410
```

Graph for Solution #4 Terminal input:


2 * math.sin(2*math.pi * x)
y - axis
x - axis