

CSEN1131P - SOFTWARE ENGINEERING LAB

EXPERIEMENT-1

Implement weather modeling using the quadratic solution in stages: hard-coding variables keyboard input, read from a file, for a single set of input, multiple sets of inputs. save all versions, debug, fix problems, create a Github account.

Your Github Link: <https://github.com/sneha22004>

Programs link from your account:

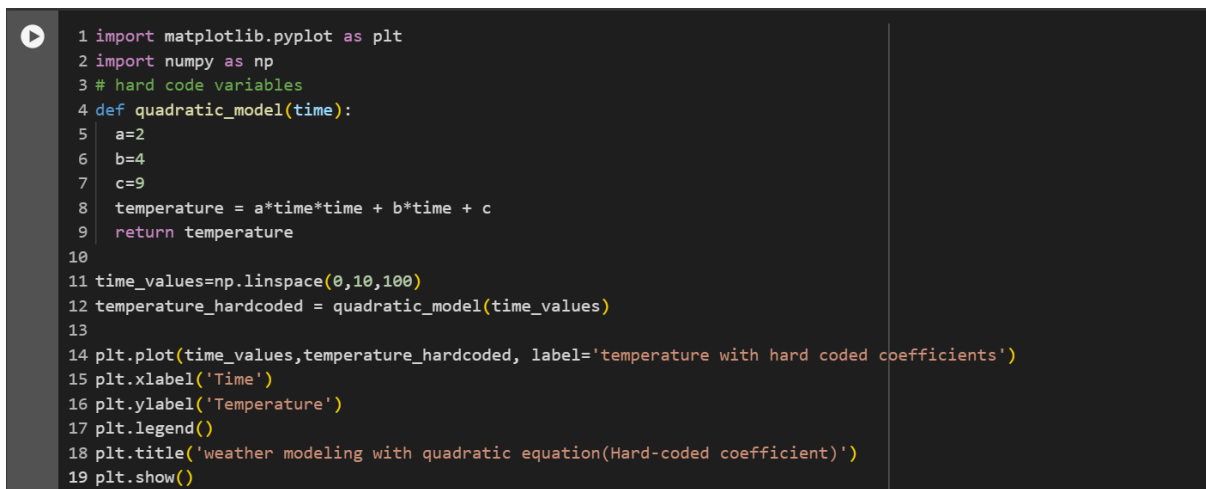
Name of the Student: SNEHASRI MOTAMARRI

Complete Roll No: VU21CSEN0100242

TASK -1

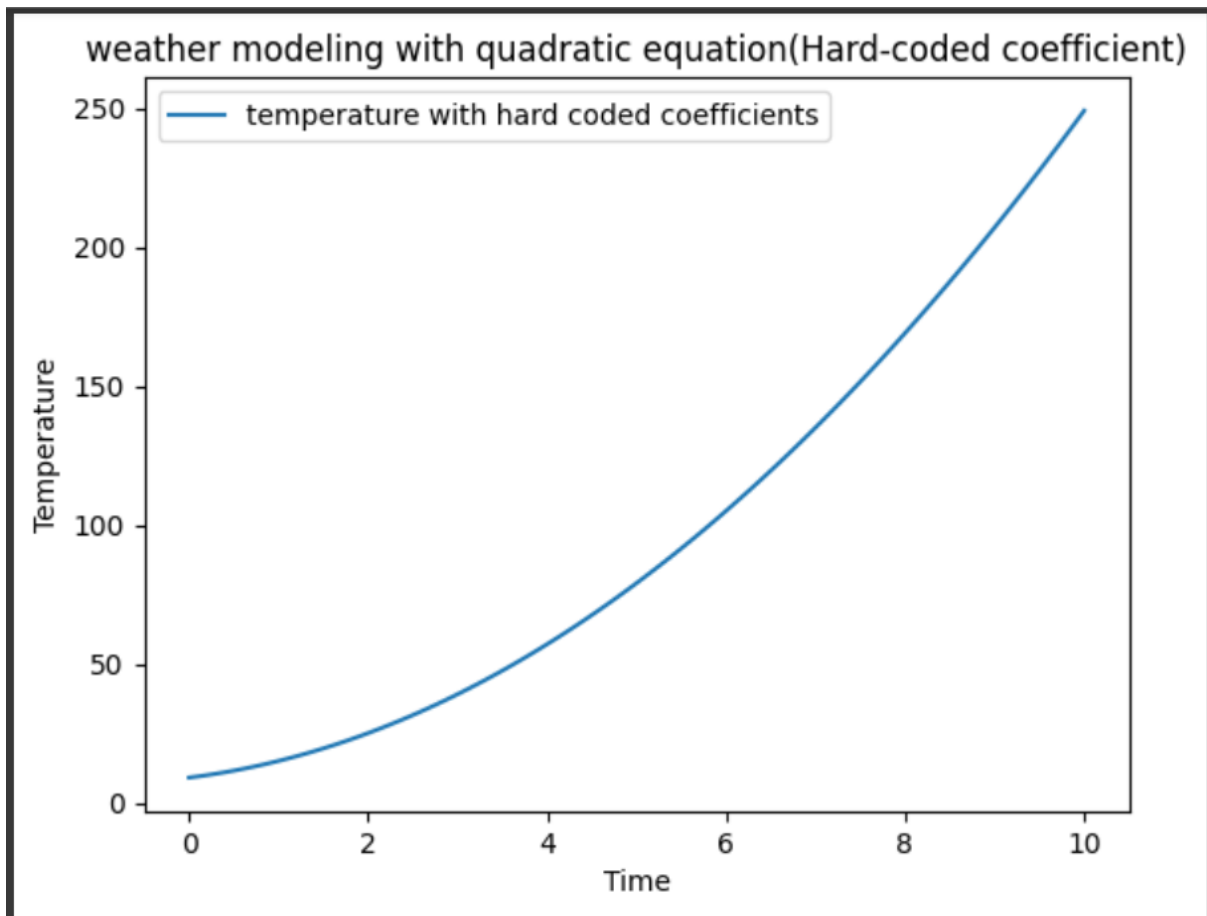
Implement weather modeling using the quadratic solution with hard coding variables (fixed variable values give in program)

Screenshot of program



```
1 import matplotlib.pyplot as plt
2 import numpy as np
3 # hard code variables
4 def quadratic_model(time):
5     a=2
6     b=4
7     c=9
8     temperature = a*time*time + b*time + c
9     return temperature
10
11 time_values=np.linspace(0,10,100)
12 temperature_hardcoded = quadratic_model(time_values)
13
14 plt.plot(time_values,temperature_hardcoded, label='temperature with hard coded coefficients')
15 plt.xlabel('Time')
16 plt.ylabel('Temperature')
17 plt.legend()
18 plt.title('weather modeling with quadratic equation(Hard-coded coefficient)')
19 plt.show()
```

Screenshot of plot



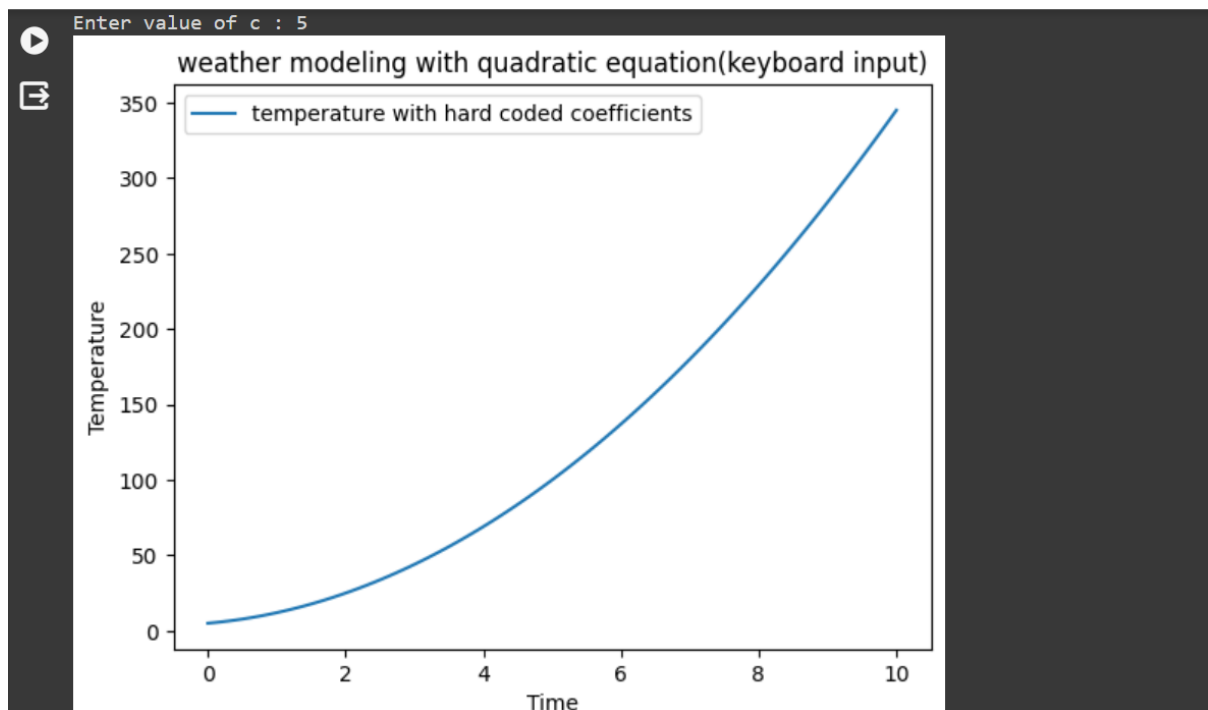
TASK -2

Implement weather modeling using the quadratic solution with hard coding variables through keyboard input by user

Screenshot of program

```
[2] 1 import matplotlib.pyplot as plt
2 import numpy as np
3 # hard code variables
4 def quadratic_model(a,b,c,time):
5     temperature = a*time*time + b*time + c
6     return temperature
7
8 time_values=np.linspace(0,10,100)
9 a=int(input("Enter value of a : "))
10 b=int(input("Enter value of b : "))
11 c=int(input("Enter value of c : "))
12 temperature_hardcoded = quadratic_model(a,b,c,time_values)
13
14 plt.plot(time_values,temperature_hardcoded, label='temperature with hard coded coefficients')
15 plt.xlabel('Time')
16 plt.ylabel('Temperature')
17 plt.legend()
18 plt.title('weather modeling with quadratic equation(keyboard input)')
19 plt.show()
```

Screenshot of plot



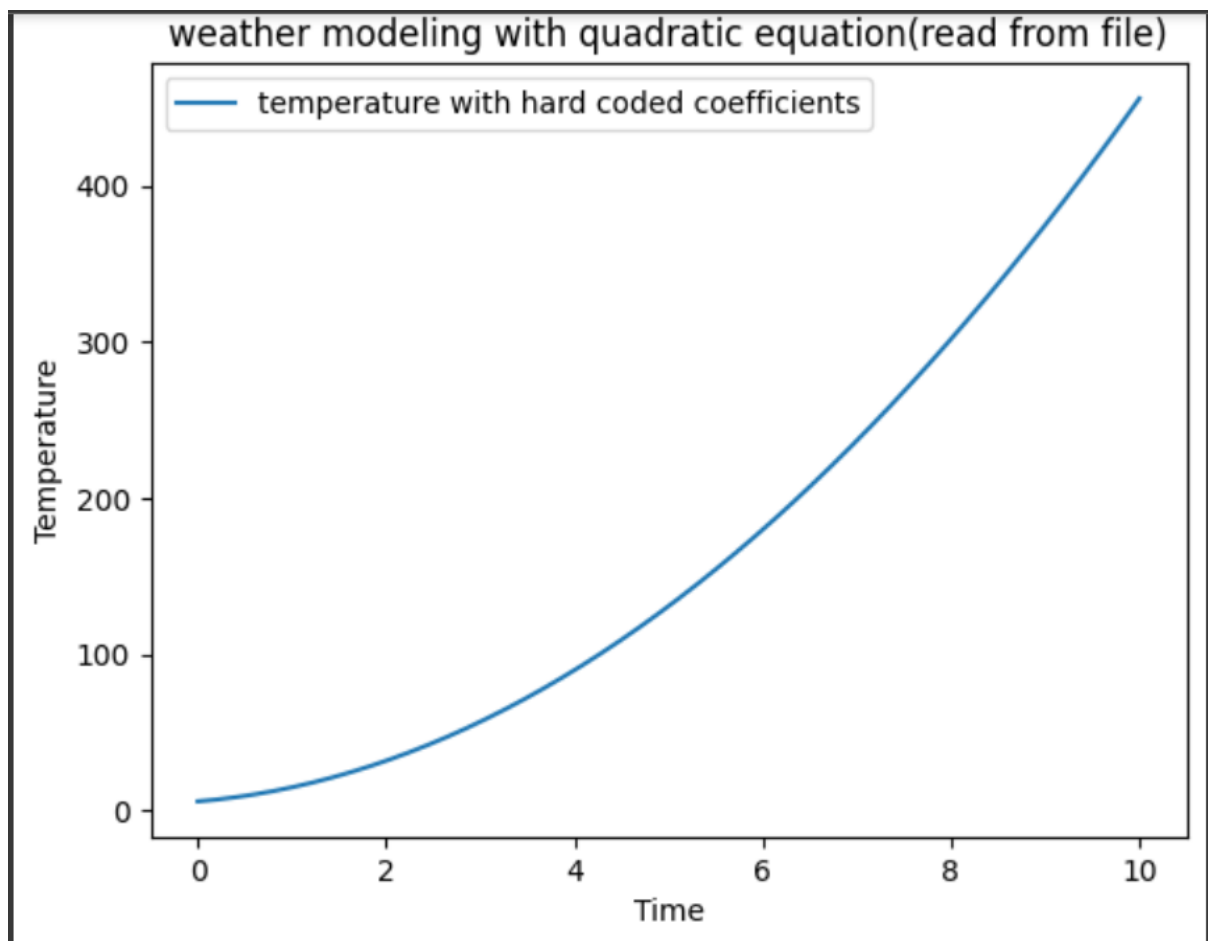
TASK -3

Implement weather modeling using the quadratic solution with hard coding variables read from a file

Screenshot of program

```
1 f = open("a.txt", "r")
2
3 import matplotlib.pyplot as plt
4 import numpy as np
5 # hard code variables
6 def quadratic_model(f,time):
7     a=int(f.readline())
8     b=int(f.readline())
9     c=int(f.readline())
10    temperature = a*time*time + b*time + c
11    return temperature
12
13 time_values=np.linspace(0,10,50)
14 temperature_hardcoded = quadratic_model(f,time_values)
15
16 plt.plot(time_values,temperature_hardcoded, label='temperature with hard coded coefficients')
17 plt.xlabel('Time')
18 plt.ylabel('Temperature')
19 plt.legend()
20 plt.title('weather modeling with quadratic equation(read from file)')
21 plt.show()
```

Screenshot of plot



TASK -4

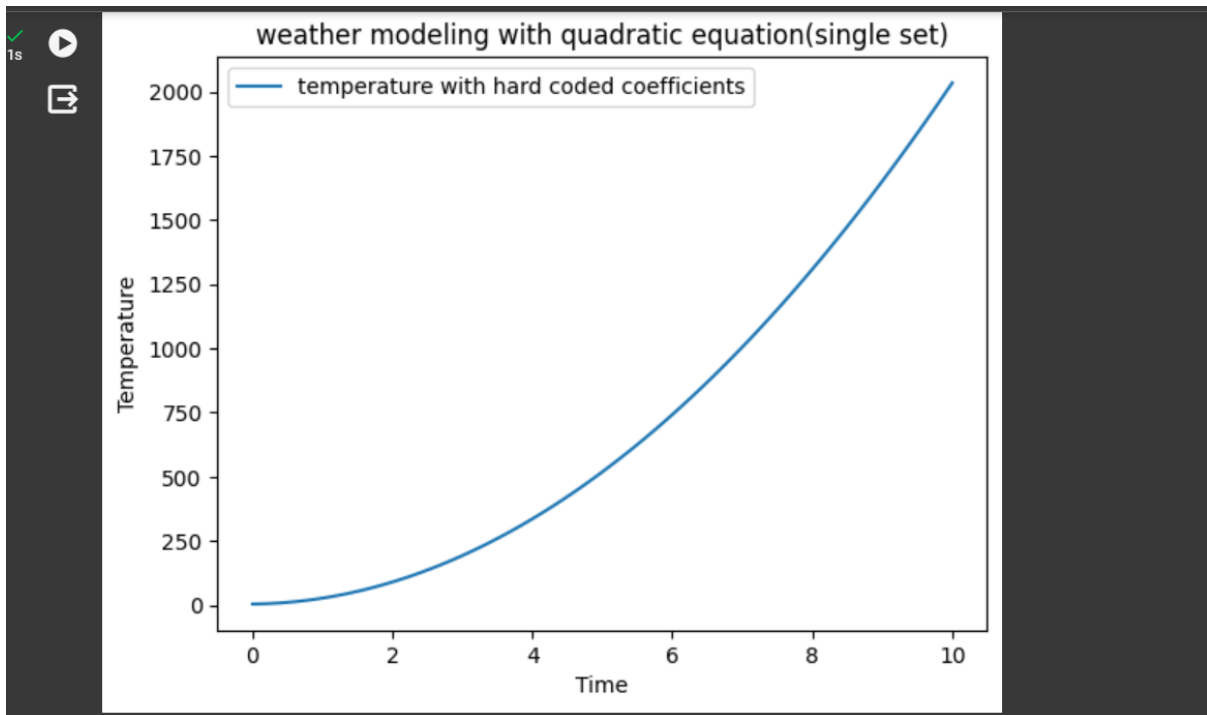
Implement weather modeling using the quadratic solution with hard coding variables for multiple sets of inputs

(multiple sets of variable values given in program or file or user input and print all sets of graphs in one plot and specify those sets values)

Screenshot of program

```
✓ [4] 1 import matplotlib.pyplot as plt
1s    2 import numpy as np
      3 # hard code variables
      4 def quadratic_model(time):
      5     a,b,c=[20,3,4]
      6     temperature = a*time*time + b*time + c
      7     return temperature
      8
      9 time_values=np.linspace(0,10,100)
     10 temperature_hardcoded = quadratic_model(time_values)
     11
     12 plt.plot(time_values,temperature_hardcoded, label='temperature with hard coded coefficients')
     13 plt.xlabel('Time')
     14 plt.ylabel('Temperature')
     15 plt.legend()
     16 plt.title('weather modeling with quadratic equation(single set)')
     17 plt.show()
```

Screenshot of plot



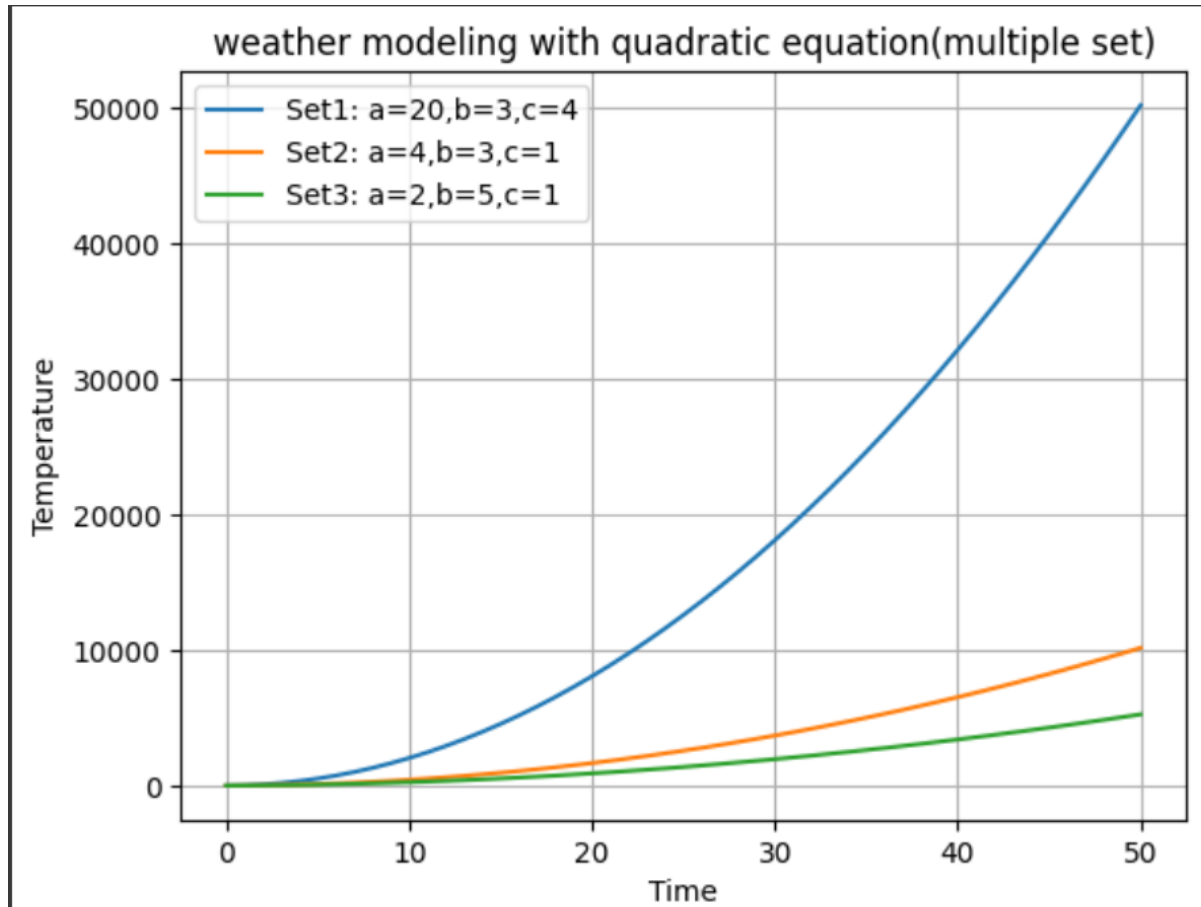
TASK -5

Implement weather modeling using the quadratic solution with combination of hard coding variables (fixed) and user input through keyboard and specify the type of graphs with values

Screenshot of program

```
[7] 1 import matplotlib.pyplot as plt
    2 import numpy as np
    3 # hard code variables
    4 def quadratic_model(time,a,b,c):
    5     temperature = a*time*time + b*time + c
    6     return temperature
    7 list = [(20,3,4),(4,3,1),(2,5,1)]
    8 time_values=np.arange(0,51,1)
    9
   10 for i,(a,b,c) in enumerate(list):
   11     temperature_values = quadratic_model(time_values,a,b,c)
   12     label=f'Set{i+1}: a={a},b={b},c={c}'
   13     plt.plot(time_values,temperature_values, label=label)
   14
   15
   16 plt.xlabel('Time')
   17 plt.ylabel('Temperature')
   18 plt.legend()
   19 plt.grid(True)
   20 plt.title('weather modeling with quadratic equation(multiple set)')
   21 plt.show()
```

Screenshot of plot



Description of libraries methods or functions from the above all programs

Matplotlib:

Matplotlib is a popular Python library for creating static, animated, and interactive visualizations in Python. It provides a wide variety of plots and charts, including line plots, scatter plots, bar plots, histograms, and more. Matplotlib is highly customizable and widely used in the fields of data science, machine learning, and scientific computing.

NumPy:

NumPy is another fundamental library for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these elements. NumPy is often used in conjunction with Matplotlib to handle numerical data efficiently.

Now, let's discuss some common Matplotlib functions:

xlabel and ylabel:

xlabel: This function is used to set the label for the x-axis.

ylabel: This function is used to set the label for the y-axis.