

Problem 1

```
problem1.py > ...
1 import numpy as np
2 import pandas as pd
3
4 ## not my work, used for testing
5 import csv
6 import random
7
8 def generate_random_csv(filename, num_rows):
9     with open(filename, 'w', newline='') as csvfile:
10         fieldnames = ['x', 'y']
11         writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
12         writer.writeheader()
13         for _ in range(num_rows):
14             writer.writerow({'x': random.randint(-200, 200), 'y': random.randint(-200, 200)})
15
16 if __name__ == '__main__':
17     filename = "random_data.csv"
18     num_rows = 100 # Change this to the desired number of rows
19     generate_random_csv(filename, num_rows)
20     print("CSV file: " + filename + " generated with " + str(num_rows) + " rows.")
21
22 ## My work
23
24 data = pd.read_csv("random_data.csv")
25 dataframe = pd.DataFrame(data, columns=['x', 'y'])
26 valid = (dataframe['x'] > 0) & (dataframe['y'] < 100)
27 dataframe = dataframe[valid]
28 dataframe.to_csv('trimmed.csv')
29
30 #This will take y values with value less than 100, regardless of magnitude. If filtering of y values based on magnitude is 0
31 #use an absolute value function on the y condition
32
33
```

```
trimmed.csv
1 x,y
2 3,28,38
3 9,11,86
4 13,89,-157
5 18,11,-87
6 23,127,26
7 28,5,-4
8 31,181,62
9 34,85,-84
10 39,54,56
11 40,126,-34
12 51,26,-159
13 83,197,-2
14 55,6,-101
15 56,21,92
16 59,119,-52
17 64,-4,-98
18 68,100,-48
19 69,78,-157
20 72,75,-104
21 73,68,22
22 74,100,77
23 75,4,51
24 77,62,61
25 83,171,25
26 86,152,36
27 89,116,22
28 90,16,-32
29 95,153,-54
30 96,65,68
```

```
random_data.csv
1 x,y
2 -116,176
3 -25,-128
4 -192,-54
5 26,38
6 -72,32
7 -128,-99
8 -33,17
9 65,137
10 -82,33
11 11,86
12 -107,28
13 -4,197
14 -16,-139
15 89,-157
16 -176,143
17 -77,176
18 39,108
19 -161,-123
20 11,-87
21 -168,61
22 -69,-105
23 135,109
24 -189,183
25 127,26
26 -103,143
27 -121,146
28 -91,-45
29 105,162
30 5,-4
31 -99,-185
32 -32,37
33 181,62
34 -51,-94
35 93,189
36 85,-84
37 -148,108
38 158,186
39 -26,-123
40 -95,-8
41 54,56
42 126,-34
43 88,199
44 -14,4
45 -184,-69
46 -110,142
47 -78,120
48 -48,9
49 -200,19
50 -68,-39
51 -182,-37
52 -169,-27
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

plea@ultrasound-124:~/Documents/School Vault/Computing for Physicists/Homework 3 \$ /bin/python3 ~/home/plea/Documents/School Vault/Computing for Physicists/Homework 3/problem1.py

CSV file: random_data.csv generated with 100 rows

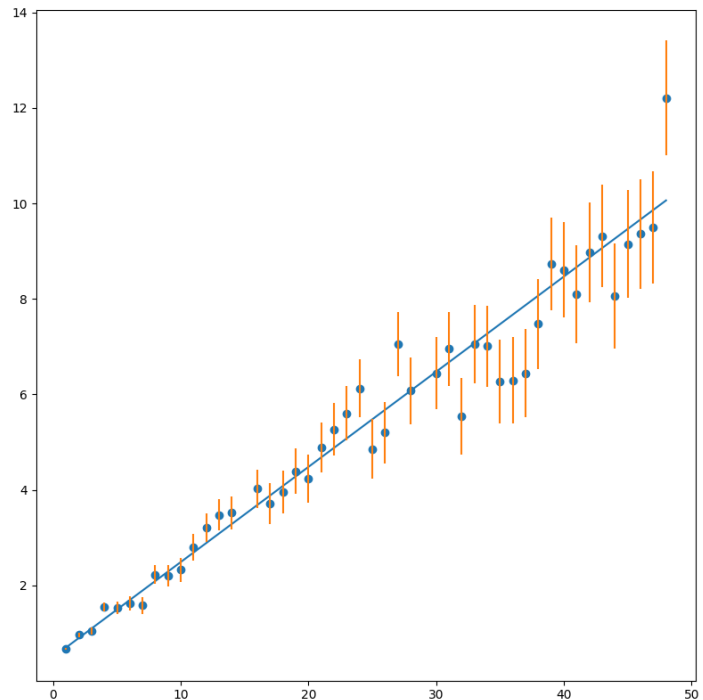
Problem 2

```
problem2.py > ...
1 import numpy as np
2 import pandas as pd
3 import scipy as sp
4 import matplotlib.pyplot as plt
5 def linearFunc(x,a,b):
6     return a*x+b
7
8 data = pd.read_excel('HW3lineardata.xlsx')
9 df = pd.DataFrame(data, columns=['x','y','sig'])
10 # Trimming NaNs from dataframe
11 notnull = (df['y'].notnull())
12 df = df[notnull]
13 #Trimming 0 values from sigma
14 not0 = (df['sig'] != 0)
15 df = df[not0]
16 xvals = df['x'].values
17 yvals = df['y'].values
18 sigma = df['sig'].values
19 parameters,covar = sp.optimize.curve_fit(linearFunc, xvals, yvals)
20 print("Best fit parameters \n A = ", parameters[0], ", ",
21 perr = np.sqrt(np.diag(covar))
22 print("Parameter Errors \n A error = ", perr[0], ", B
23
24 plt.scatter(xvals,yvals)
25 plt.plot(xvals, linearFunc(xvals,parameters[0],parameters[1]))
26 plt.errorbar(xvals,yvals,yerr = sigma,ls='none')
27 plt.show()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

plea@ultrasound-124:~/Documents/School Vault/Computing for Physicists/Homework 3 \$ /bin/python3 ~/home/plea/Documents/School Vault/Computing for Physicists/Homework 3/problem2.py

Best fit parameters
A = 0.19930410296747728 , B = 0.49701803257173877
Parameter Errors
A error = 0.00417292996112784 , B error = 0.022245344987520194



Problem 3

```

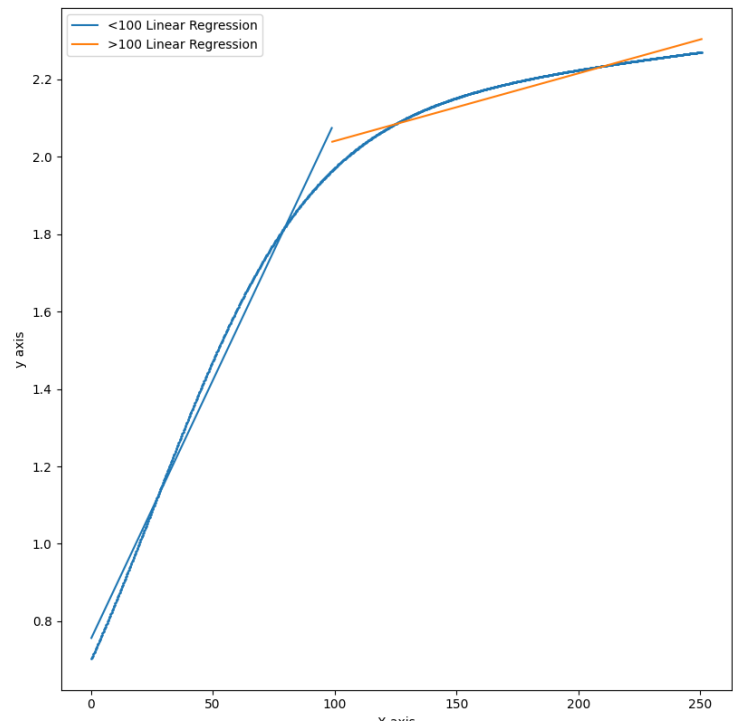
problem3.py > ...
1 import matplotlib.pyplot as plt
2 import numpy as np
3 import scipy as sp
4 import pandas as pd
5 def linearFunc(x,a,b):
6     return a*x+b
7
8 data = pd.read_excel('lin2.xlsx')
9 df = pd.DataFrame(data)
10 xvals = df.iloc[:,0].values
11 yvals = df.iloc[:,1].values
12
13 x = np.where(np.logical_and(xvals>= 99, xvals<= 101))[0]
14
15 xvals2 = xvals[x[0]:]
16 yvals2 = yvals[x[0]:]
17 maxx = x[0].argmax()
18 xvals1 = xvals[:x[maxx]]
19 yvals1 = yvals[:x[maxx]]
20
21 parameters = np.polyfit(xvals1,yvals1,1)
22 parameters1 = np.polyfit(xvals2,yvals2,1)
23
24 plt.scatter(xvals, yvals, s=1)
25 plt.plot(xvals1, linearFunc(xvals1,parameters[0],parameters[1]),color='blue')
26 plt.plot(xvals2, linearFunc(xvals2,parameters1[0],parameters1[1]),color='orange')
27 plt.xlabel("x axis")
28 plt.ylabel("y axis")
29 plt.legend()
30 plt.show()

```

```

plea@ultrasound-124:~/Documents/School Vault/Computing for Physicists/Homework 3$ /bin/python3 /home/plea/Documents/School Vault/Computing for Physicists/Homework 3/problem3.py

```



Problem 4

```

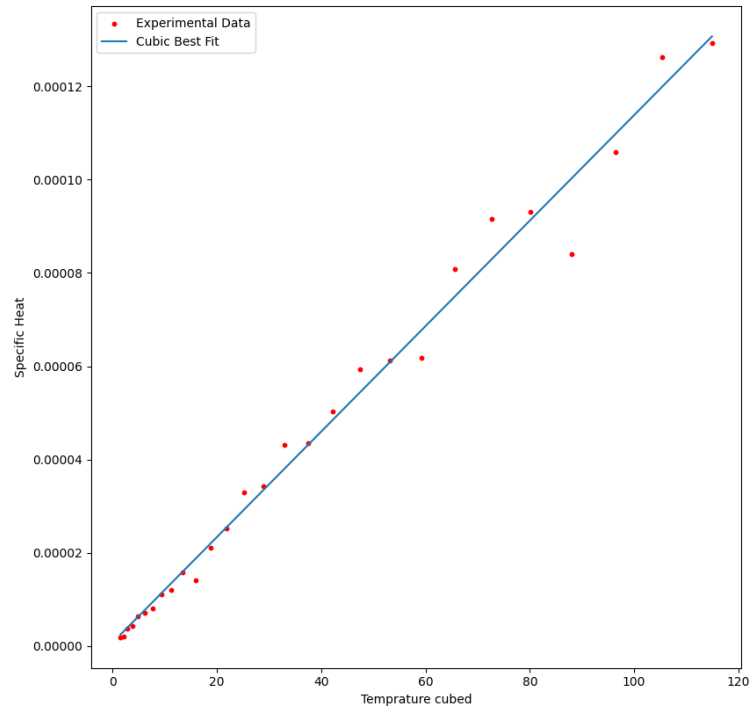
problem4.py > ...
1 import matplotlib.pyplot as plt
2 import numpy as np
3 import scipy as sp
4 import pandas as pd
5 def linear(x,a,b):
6     return a*x + b
7
8 data = pd.read_table('DebyeDat.txt', sep=" ")
9 df = pd.DataFrame(data)
10 temp = df.iloc[:,0].values
11 specheat = df.iloc[:,1].values
12 cubetemp = temp ** 3
13 parameters = np.polyfit(cubetemp,specheat,1)
14 td = np.cbrt(322.77/parameters[0])
15 print("Debye teprature t_d is", td)
16 plt.scatter(cubetemp, specheat, s=10, color='r', label = "Experimental Data")
17 plt.plot(cubetemp, linear(cubetemp, parameters[0],parameters[1]),color='blue')
18 plt.xlabel("Temprature cubed")
19 plt.ylabel("Specific Heat")
20 plt.legend()
21 plt.show()

```

```

plea@ultrasound-124:~/Documents/School Vault/Computing for Physicists/Homework 3$ /bin/python3 /home/plea/Documents/School Vault/Computing for Physicists/Homework 3/problem4.py
Debye teprature t_d is 658.5154112596998

```



Problem 5

```
problem5.py x problem4.py
problem5.py > ...
1 import matplotlib.pyplot as plt
2 import numpy as np
3 import random
4 points = 10 ** 6
5 xcoord = np.random.rand(points) * 2 - 1
6 ycoord = np.random.rand(points) * 2 - 1
7 circleindex = np.where(np.sqrt(xcoord ** 2 + ycoord ** 2) < 1)[0]
8 circlex = xcoord[circleindex]
9 circley = ycoord[circleindex]
10 circlecount = len(circleindex)
11 #Ratio of circle to square
12 pi = circlecount/points * 4
13 print(pi)
14 plt.scatter(xcoord, ycoord)
15 plt.scatter(circlex, circley, color='r')
16 plt.show()

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
/bin/python "/home/plea/Documents/School Vault/Computing for Physicists/Homework 3/problem5.py"
plea@ultrasound-124:~/Documents/School Vault/Computing for Physicists/Homework 3$ /bin/python "/home/plea/Documents/School Vault/Computing for Physicists/Homework 3/problem5.py"
3.140972
```

Figure 1

