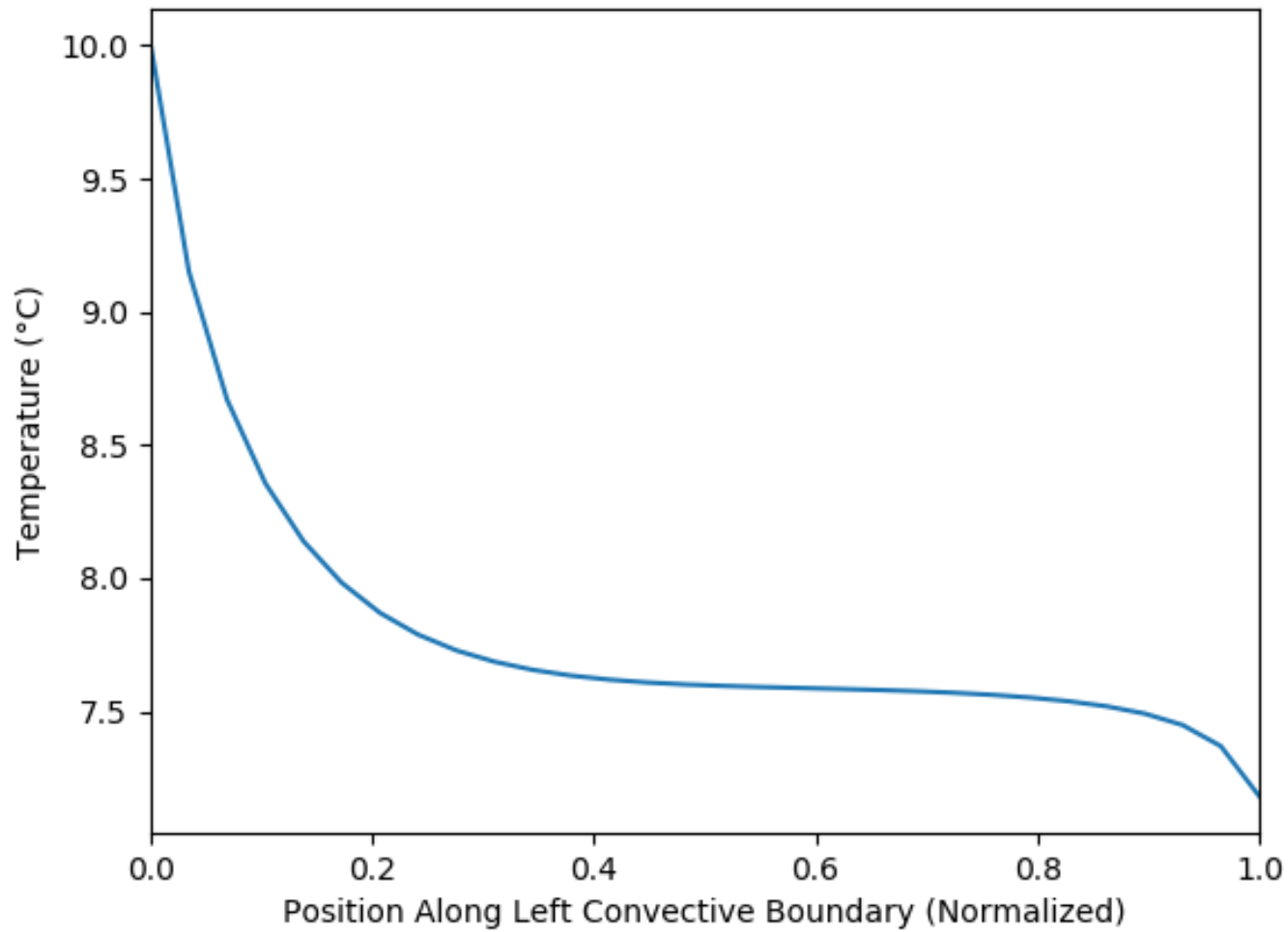
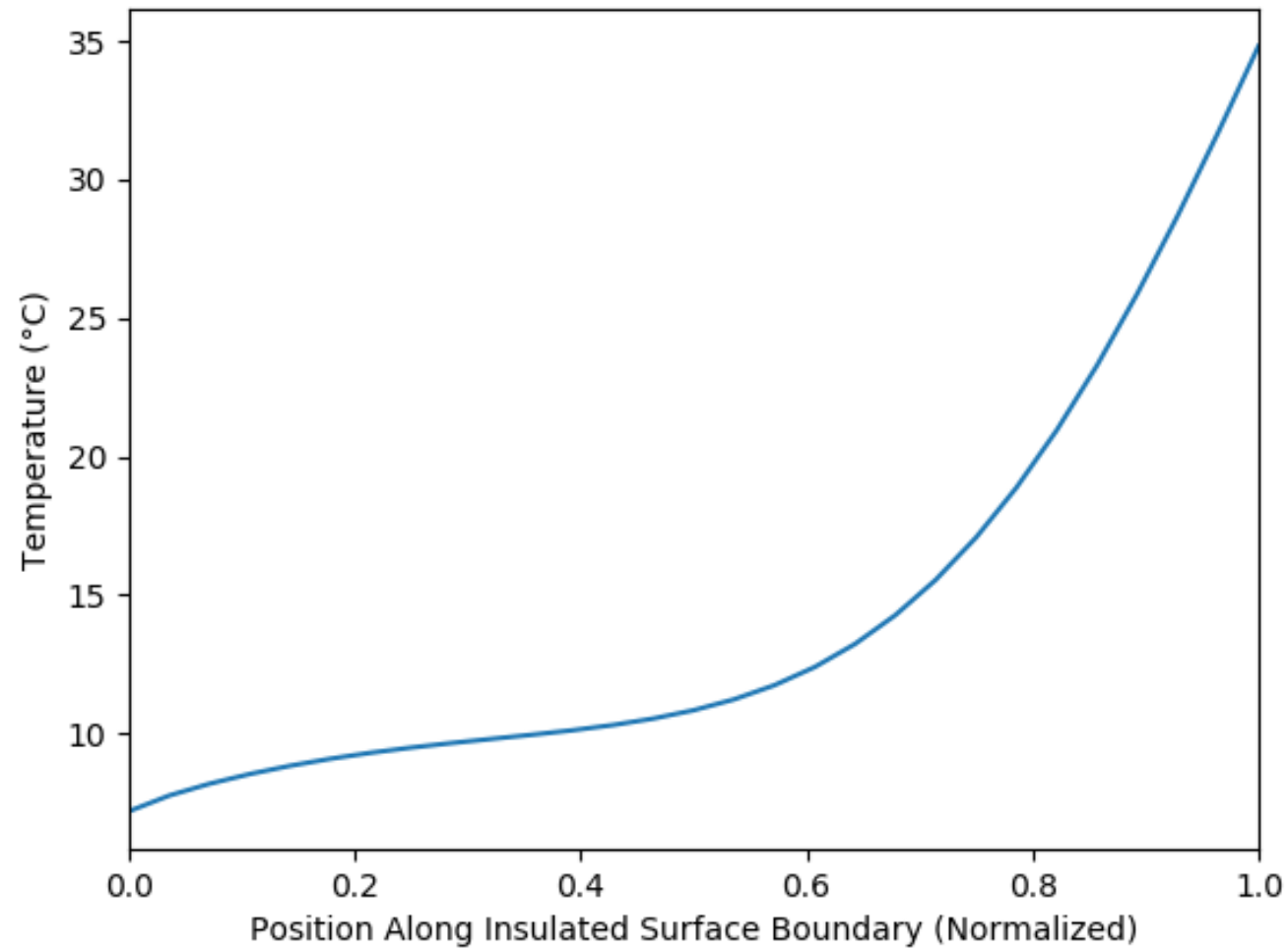


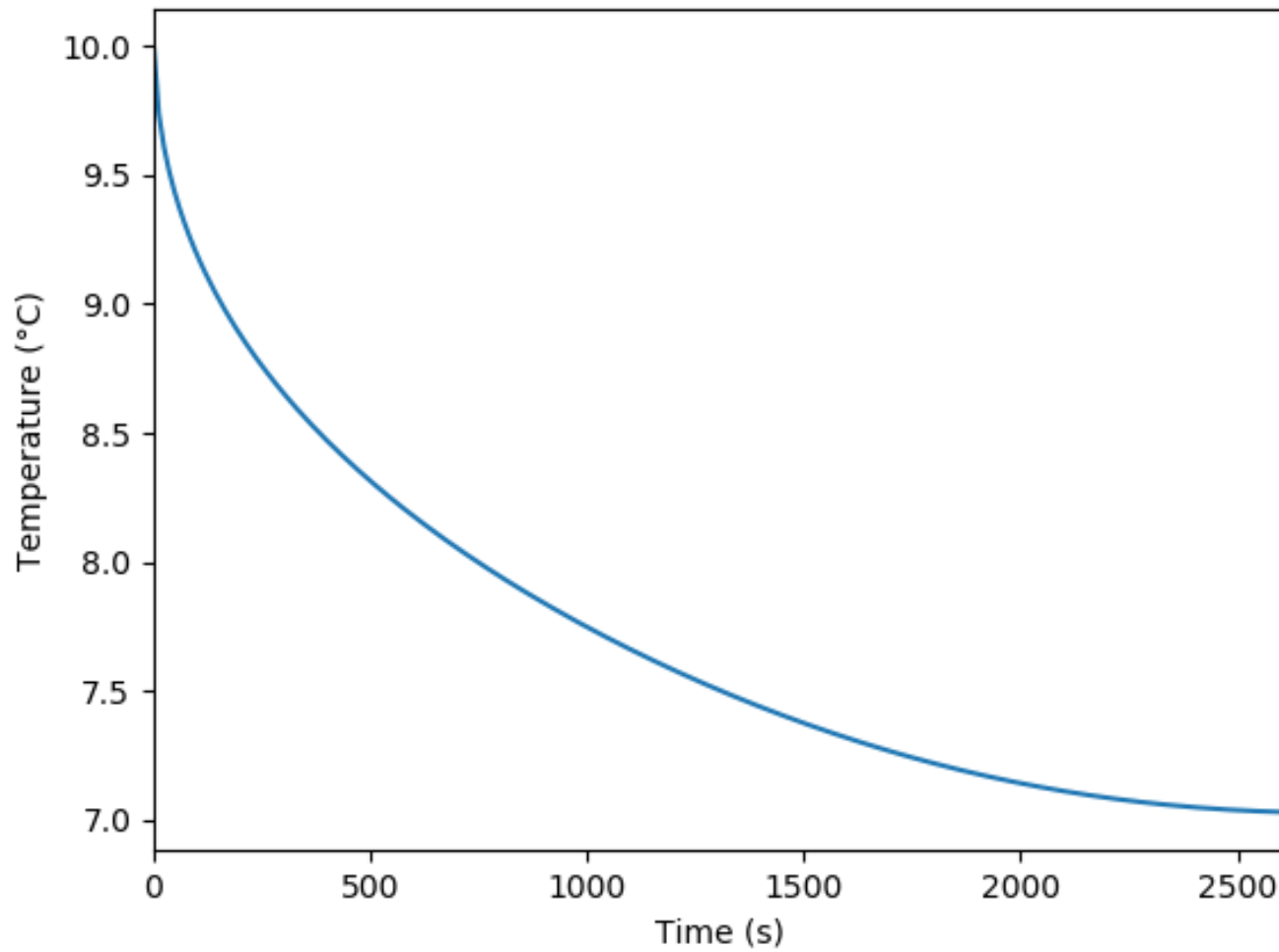
Temperature Along the Left Convective Boundary
Bottom to Top; 100 Time Steps

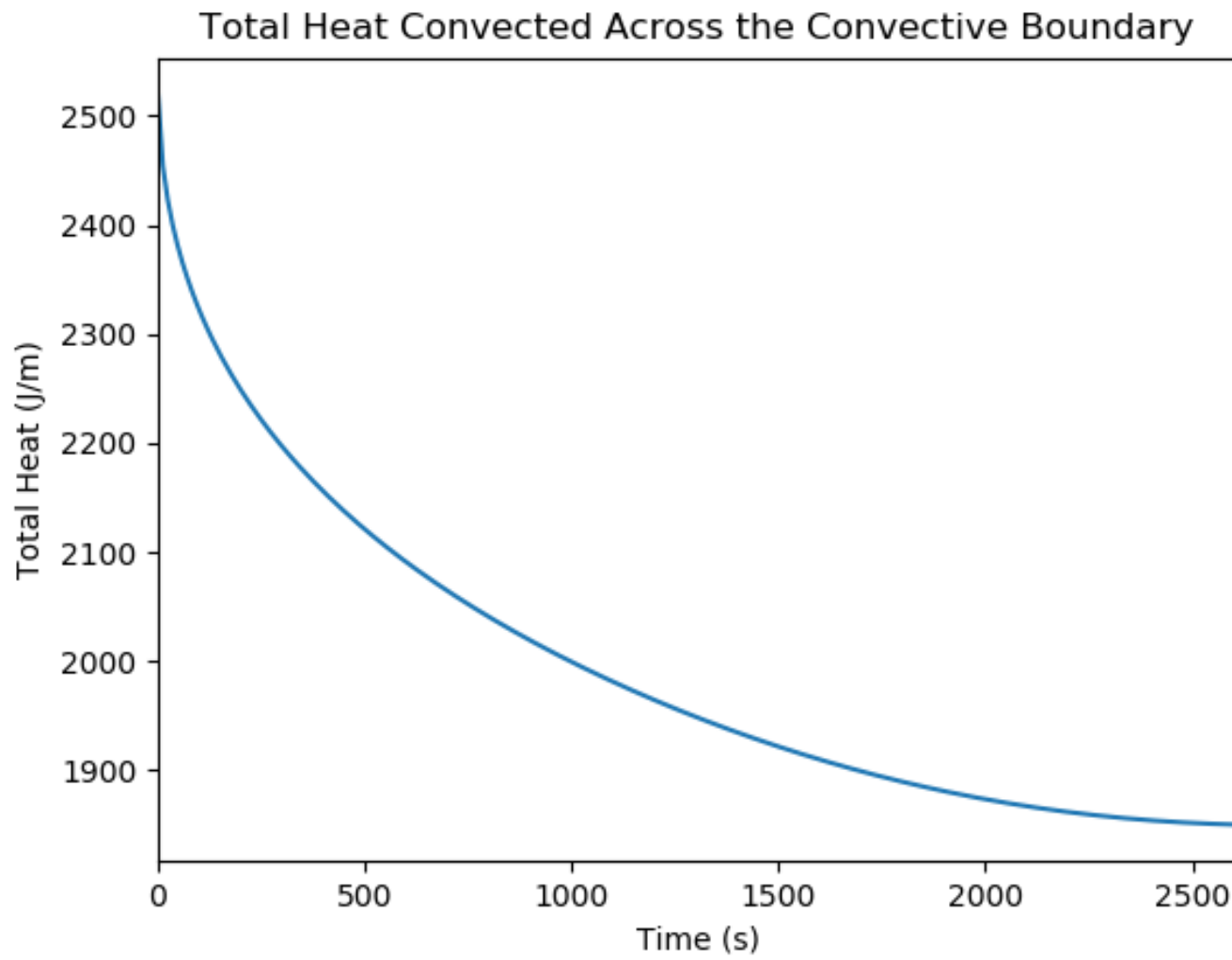


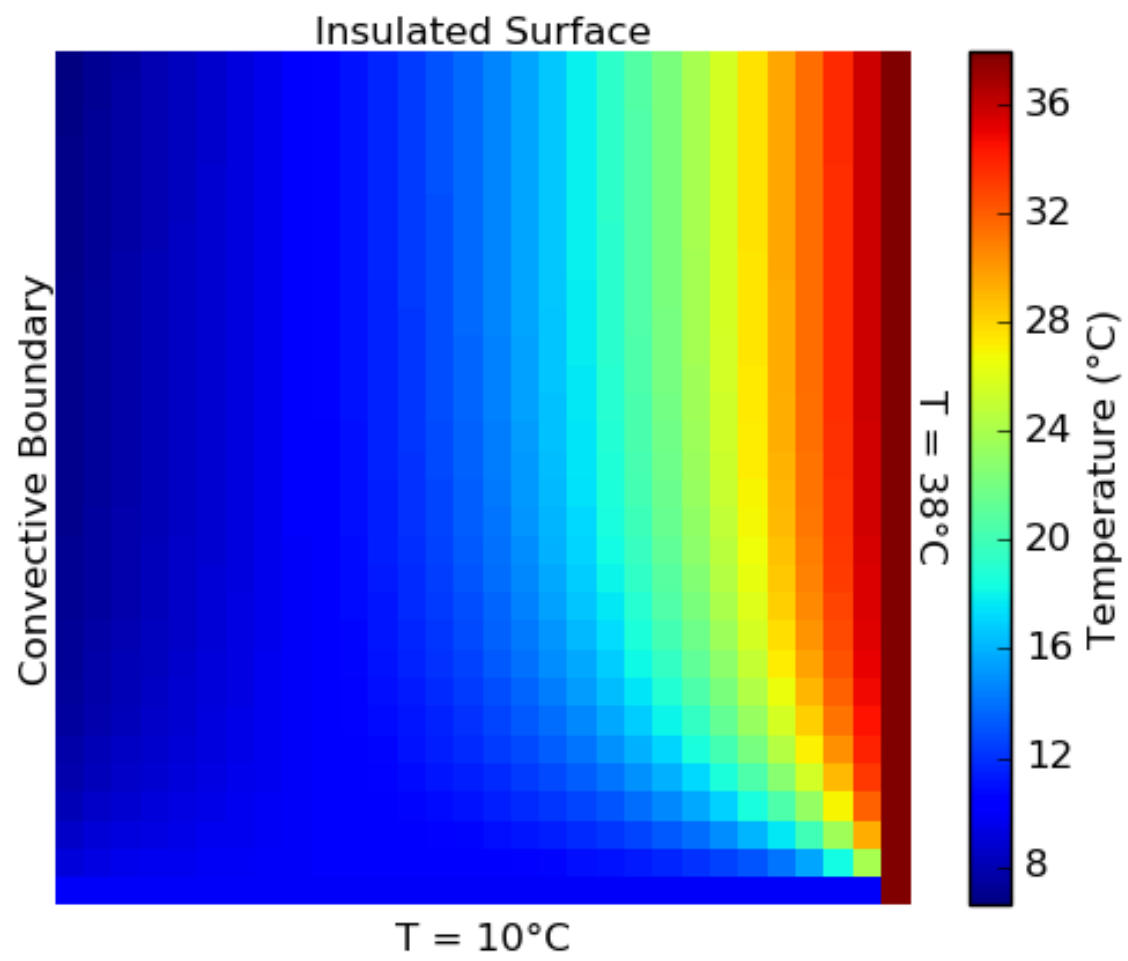
Temperature Along the Insulated Surface Boundary
Left to Right; 100 Time Steps



Temperature At A Specific Point (Halfway Up Convective Boundary) Until Steady State Reached







1 PSet-5.py

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 fileName = "Problem-Set-5"
5
6 num_time_steps = 100
7 delta = 0.01          # 1 cm
8
9 # Grid points:
10 height = 30
11 width = 30
12
13 side_length = 0.30    # meters
14
15 # Constants
16 rho = 3000            # kg/m^3
17 c = 840               # J/(kg*C)
18 h = 28                # W/(m^2*C) Convective Heat Transfer Coefficient
19 k = 5.2               # W/(m*C) Thermal Conductivity
20 alpha = k / (rho * c) # m^2/s Thermal Diffusivity
21
22 dt_1 = rho * c * (delta * delta) / (2 * h * delta + 4 * k) # Characteristic time (
23                    convective boundary)
24 dt_2 = (delta * delta) / (4 * alpha)                        # Characteristic time (
25                    internal grid)
26 dt = min(dt_1, dt_2)
27 Fo = alpha * dt / (delta * delta)                          # Fourier Number
28 Bi = h * delta / k                                          # Biot Number
29 T_initial = 10
30 T_right = 38
31 T_inf = 0
32
33 # Create array and initialize to T-initial
```

```

32 data = np.zeros((width, height)) + T_initial
33
34 # Set the right boundary to T_right
35 for j in range(height):
36     data[(width - 1), j] = T_right
37
38 history = [data.copy()]
39
40 error_flag = True
41 error_limit = 1e-4
42 while error_flag:
43     large_error_term_found = False
44
45     data_old = data.copy()
46
47     # Internal Nodes
48     for m in range(1, width - 1):
49         for n in range(1, height - 1):
50             data[m, n] = alpha * dt / (delta * delta) * (data_old[m + 1, n] + data_old[m
- 1, n] + data_old[m, n + 1] + data_old[m, n - 1]) + (1 - 4 * alpha * dt / (delta *
delta)) * data_old[m, n]
51
52     # Convective Boundary Nodes (Left)
53     for n in range(1, height - 1):
54         m = 0
55         data[m, n] = Fo * (2 * Bi * (T_inf - data_old[m, n]) + 2 * data_old[m + 1, n] +
data_old[m, n + 1] + data_old[m, n - 1] - 4 * data_old[m, n]) + data_old[m, n]
56
57     # Insulated Boundary Nodes (Top)
58     for m in range(1, width - 1):
59         n = height - 1
60         data[m, n] = Fo * (2 * data_old[m, n - 1] + data_old[m - 1, n] + data_old[m + 1,
n]) + (1 - 4 * Fo) * data_old[m, n]
61
62     # Exterior Corner with Convection Boundary

```

```

63     m = 0
64     n = height - 1
65     data[m, n] = 2 * Fo * (data_old[m + 1, n] + data_old[m, n - 1] - 2 * data_old[m, n]
        + 2 * Bi * (T_inf - data_old[m, n])) + data_old[m, n]
66
67
68     # Check if reached steady state
69     if not large_error_term_found:
70         error_term = abs(data[m, n] - data_old[m, n]) / data_old[m, n]
71         if (error_term <= error_limit):
72             error_flag = False
73         else:
74             error_flag = True
75             large_error_term_found = True
76
77     history.append(data.copy())
78
79     #print(len(history))
80
81     # Print the data in the console (readable format)
82     #print(np.rot90(data))
83
84     figNum = 1
85     plt.figure(figNum)
86     x = np.linspace(0, 1, height)
87     y = history[num_time_steps][0, :]
88     plt.plot(x, y)
89     plt.xlabel("Position Along Left Convective Boundary (Normalized)")
90     plt.ylabel("Temperature (\N{DEGREE SIGN}C)")
91     plt.suptitle("Temperature Along the Left Convective Boundary")
92     plt.title("Bottom to Top; 100 Time Steps")
93     plt.xlim(0, 1)
94     plt.savefig(fileName + "/images/" + fileName + "-Figure-" + str(figNum) + ".png")
95     plt.show()
96

```



```

97 figNum = 2
98 plt.figure(figNum)
99 x = np.linspace(0, 1, width-1)
100 y = history[num_time_steps][0:(width-1), height - 1]
101 plt.plot(x, y)
102 plt.xlabel("Position Along Insulated Surface Boundary (Normalized)")
103 plt.ylabel("Temperature (\N{DEGREE SIGN}C)")
104 plt.suptitle("Temperature Along the Insulated Surface Boundary")
105 plt.title("Left to Right; 100 Time Steps")
106 plt.xlim(0, 1)
107 plt.savefig(fileName + "/images/" + fileName + "-Figure-" + str(figNum) + ".png")
108 plt.show()
109
110 figNum = 3
111 plt.figure(figNum)
112 history_length = len(history)
113 y = []
114 for state in history:
115     y.append(state[0, 15])
116 x = dt * np.linspace(0, (len(history) - 1), len(history))
117 plt.plot(x, y)
118 plt.xlabel("Time (s)")
119 plt.ylabel("Temperature (\N{DEGREE SIGN}C)")
120 plt.suptitle("Temperature At A Specific Point (Halfway Up Convective")
121 plt.title("Boundary) Until Steady State Reached")
122 plt.xlim(0, max(x))
123 plt.savefig(fileName + "/images/" + fileName + "-Figure-" + str(figNum) + ".png")
124 plt.show()
125
126
127 figNum = 4
128 plt.figure(figNum)
129 q = []
130 for state_num in range(len(history)):
131     m = 0

```

```

132     delta_T = 0
133     for n in range(height):
134         delta_T += history[state_num][m, n] - T_inf
135
136     q_total_state = h * side_length * delta_T
137     q.append(q_total_state)
138
139 x = dt * np.linspace(0, (len(q) - 1), len(q))
140 plt.plot(x, q)
141 plt.xlabel("Time (s)")
142 plt.ylabel("Total Heat (J/m)")
143 plt.title("Total Heat Convected Across the Convective Boundary")
144 plt.xlim(0, max(x))
145 plt.savefig(fileName + "/images/" + fileName + "-Figure-" + str(figNum) + ".png")
146 plt.show()
147
148
149
150 figNum = 5
151 plt.figure(figNum)
152 plt.axes().set_aspect('equal')
153 plt.style.use('classic')
154 data_graphable = np.flipud(np.rot90(data))
155 heatmap = plt.pcolor(data_graphable)
156
157 plt.text(0.5, -0.02, "T = " + str(T_initial) + "\N{DEGREE SIGN}C",
158         horizontalalignment='center',
159         verticalalignment='top',
160         rotation=0,
161         clip_on=False,
162         transform=plt.gca().transAxes)
163 plt.text(0, 0.5, "Convective Boundary",
164         horizontalalignment='right',
165         verticalalignment='center',
166         rotation=90,

```

```

167         clip_on=False,
168         transform=plt.gca().transAxes)
169 plt.text(0.5, 1, "Insulated Surface",
170         horizontalalignment='center',
171         verticalalignment='bottom',
172         rotation=0,
173         clip_on=False,
174         transform=plt.gca().transAxes)
175 plt.text(1, 0.5, "T = " + str(T_right) + "\N{DEGREE SIGN}C",
176         horizontalalignment='left',
177         verticalalignment='center',
178         rotation=270,
179         clip_on=False,
180         transform=plt.gca().transAxes)
181
182 plt.axis("off")
183
184 plt.xlim(0, width)
185 plt.ylim(0, height)
186
187 cbar = plt.colorbar(heatmap)
188 cbar.set_label("Temperature (\N{DEGREE SIGN}C)")
189 plt.clim(np.amin(data), np.amax(data))
190
191 plt.savefig(fileName + "/images/" + fileName + "-Figure-" + str(figNum) + ".png")
192 plt.show()

```