21053250

February 2, 2024

[33]:
```
import numpy as np
```

1 Experiment: 1

1.1 basic mathematical operations

[34]:
```
a=np.array([1,2,3])
b=np.array([5,6,7])
```

[48]:
```
print(np.add(a,b))
```

```
[ 6  8 10]
```

[36]:
```
print(np.subtract(a,b))
```

```
[-4 -4 -4]
```

[37]:
```
print(np.multiply(a,b))
```

```
[ 5 12 21]
```

[38]:
```
print(np.divide(a,b))
```

```
[0.2        0.33333333 0.42857143]
```

1.2 indexing and slicing

[39]:
```
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9])
print(f"Element at index 2: {arr[2]}")
```

```
Element at index 2: 3
```

[40]:
```
print(f"Elements from index 2 to 5: {arr[2:5]}")
print(f"Elements up to index 4(0-3): {arr[:4]}")
print(f"Elements from index 3 to end: {arr[3:]}")
print(f"Elements from index 0 to 6 with step 2: {arr[0:6:2]}")
```

```
Elements from index 2 to 5: [3 4 5]
Elements up to index 4(0-3): [1 2 3 4]
Elements from index 3 to end: [4 5 6 7 8 9]
Elements from index 0 to 6 with step 2: [1 3 5]
```

1.3 mean median mode

[41]:
```
m=np.mean(arr)
print(f"mean :{m}")
```

```
mean :5.0
```

[42]:
```
m=np.median(arr)
print(f"median :{m}")
```

```
median :5.0
```

[43]:
```
#no mode in nummpy
from spicy import stats
m1=stats.mode(arr)
print(f"mode :{m1}")
```

```
mode :ModeResult(mode=1, count=1)
```

1.4 reshape and transpose

[44]:
```
arr1 = np.array([[1, 2, 3], [4, 5, 6]])
print(f"Original Array:\n{arr1}")
print("array shape",arr1.shape)
reshaped = arr1.reshape(3, 2)
print(f"\nReshaped Array:\n{reshaped}")
```

```
Original Array:
[[1 2 3]
 [4 5 6]]
array shape (2, 3)

Reshaped Array:
[[1 2]
 [3 4]
 [5 6]]
```

[45]:
```
print(f"Original Array:\n{arr1}")
print("array shape",arr1.shape)
reshaped = arr1.reshape(3, 2)
transpose = arr1.T
print(f"\nTransposed Array:\n{transpose}")
```

```
Original Array:
[[1 2 3]
 [4 5 6]]
array shape (2, 3)

Transposed Array:
[[1 4]
```

```
 [2 5]
 [3 6]]
```

2 Experiment: 2

2.1 Problem: Analyzing Temperature Data

2.2 Data Preparation:

[49]:
```
hours = 7 * 24
temperatures = np.random.uniform(low=10, high=40, size=hours)
print(temperatures)
```

```
[12.79723679 15.9390336  28.4389492  23.13570278 10.36857002 13.801452
 36.6473176  28.77187254 37.60542531 15.09102694 20.50739181 38.95869177
 37.66720782 26.68780893 20.8272973  26.48775269 19.35838836 15.35309767
 26.97122802 17.06944262 35.32431564 21.88670802 18.19661364 10.69957841
 30.06936788 19.73463576 28.22847992 12.67294459 23.08576082 39.62030726
 37.37386289 16.44946996 16.57687611 20.22610173 25.15213132 29.31599451
 15.39506214 35.93289212 34.12729332 24.52906728 26.51692999 31.57924465
 38.90159804 10.01921822 38.63568046 34.83241467 27.37329301 29.20406669
 27.53739163 30.57704382 24.87966879 26.14827777 24.14335992 31.49497406
 24.49446408 13.84161841 25.01584937 39.04164769 14.04473927 19.7340438
 15.72521668 27.13192401 18.48354579 12.02645598 13.13375771 16.05359792
 36.33473724 38.69912068 38.67223531 27.70013403 10.41073488 11.99330682
 29.12688066 33.74437976 27.61777121 14.83777872 33.2046393  13.33401181
 13.44081398 32.06556411 14.88423575 32.29290102 33.21935073 23.99065912
 12.18564806 16.84315476 15.48641794 19.51478002 10.59533535 24.07642362
 20.098843   36.00652803 22.88867323 24.25971936 31.55630298 12.38383082
 22.73669001 32.28813098 39.31375375 36.14426565 17.24829606 13.78887778
 33.31208009 20.10650815 15.85781362 12.65112124 29.87128258 20.28378584
 15.38745686 37.8574109  37.02166063 36.0868734  11.56341845 29.50290635
 38.40182984 13.42584702 14.09589611 29.67700637 34.53712439 35.12534433
 26.32758777 31.69150308 25.51394117 22.68834409 31.09623706 14.63646066
 15.43418264 14.87187507 17.00262829 18.52533397 29.74631679 27.55473292
 21.17905762 37.85767406 28.55252927 24.01663071 24.58238611 23.39451198
 36.09429798 12.56604672 25.28896082 23.61883948 36.16864077 38.73797756
 34.13895653 22.58221556 26.80899998 31.17975551 23.75500507 31.69849293
 18.40548778 16.20639796 20.89659145 13.45492352 21.65226191 27.26753314
 15.56359005 11.21535606 12.67038212 11.18246568 39.0697866  22.52848269
 19.2266959  20.15497323 11.80154034 12.25162171 27.28471552 19.64266813]
```

[50]:
```
temperatures.shape
```

[50]:
```
(168,)
```

2.3 Data Analysis:

[52]:
```
np.mean(temperatures)
```

[52]:
```
24.185644453632897
```

[53]:
```
np.max(temperatures)
```

[53]:
```
39.620307263361276
```

[54]:
```
np.min(temperatures)
```

[54]:
```
10.01921822173169
```

[55]:
```
np.std(temperatures)
```

[55]:
```
8.702171712877846
```

2.4 Time-based Analysis:

[91]:
```
temperatures=temperatures.reshape(7,24)
```

[58]:
```
temperatures.shape
```

[58]:
```
(7, 24)
```

[59]:
```
print(temperatures)
```

```
[[12.79723679 15.9390336  28.4389492  23.13570278 10.36857002 13.801452
  36.6473176  28.77187254 37.60542531 15.09102694 20.50739181 38.95869177
  37.66720782 26.68780893 20.8272973  26.48775269 19.35838836 15.35309767
  26.97122802 17.06944262 35.32431564 21.88670802 18.19661364 10.69957841]
 [30.06936788 19.73463576 28.22847992 12.67294459 23.08576082 39.62030726
  37.37386289 16.44946996 16.57687611 20.22610173 25.15213132 29.31599451
  15.39506214 35.93289212 34.12729332 24.52906728 26.51692999 31.57924465
  38.90159804 10.01921822 38.63568046 34.83241467 27.37329301 29.20406669]
 [27.53739163 30.57704382 24.87966879 26.14827777 24.14335992 31.49497406
  24.49446408 13.84161841 25.01584937 39.04164769 14.04473927 19.7340438
  15.72521668 27.13192401 18.48354579 12.02645598 13.13375771 16.05359792
  36.33473724 38.69912068 38.67223531 27.70013403 10.41073488 11.99330682]
 [29.12688066 33.74437976 27.61777121 14.83777872 33.2046393  13.33401181
  13.44081398 32.06556411 14.88423575 32.29290102 33.21935073 23.99065912
  12.18564806 16.84315476 15.48641794 19.51478002 10.59533535 24.07642362
  20.098843   36.00652803 22.88867323 24.25971936 31.55630298 12.38383082]
 [22.73669001 32.28813098 39.31375375 36.14426565 17.24829606 13.78887778
  33.31208009 20.10650815 15.85781362 12.65112124 29.87128258 20.28378584
  15.38745686 37.8574109  37.02166063 36.0868734  11.56341845 29.50290635
  38.40182984 13.42584702 14.09589611 29.67700637 34.53712439 35.12534433]
 [26.32758777 31.69150308 25.51394117 22.68834409 31.09623706 14.63646066
  15.43418264 14.87187507 17.00262829 18.52533397 29.74631679 27.55473292
  21.17905762 37.85767406 28.55252927 24.01663071 24.58238611 23.39451198
  36.09429798 12.56604672 25.28896082 23.61883948 36.16864077 38.73797756]
 [34.13895653 22.58221556 26.80899998 31.17975551 23.75500507 31.69849293
  18.40548778 16.20639796 20.89659145 13.45492352 21.65226191 27.26753314
  15.56359005 11.21535606 12.67038212 11.18246568 39.0697866  22.52848269
  19.2266959  20.15497323 11.80154034 12.25162171 27.28471552 19.64266813]]
```

[65]:
```
a=np.mean(temperatures,axis =1)
for i in range(1,8):
    print(f"mean of day {i} is: {a[i-1]}")
```

```
mean of day 1 is: 23.274671227667834
mean of day 2 is: 26.89802888914225
mean of day 3 is: 23.63824356946475
mean of day 4 is: 22.818943473315773
mean of day 5 is: 26.095224183668687
mean of day 6 is: 25.297779024368396
mean of day 7 is: 21.276620807802605
```

[68]:
```
a=np.argmax(temperatures,axis =1)
for i in range(1,8):
    print(f"max temp at day {i} is at: {a[i-1]} hour")
```

```
max temp at day 1 is at: 11 hour
max temp at day 2 is at: 5 hour
max temp at day 3 is at: 9 hour
max temp at day 4 is at: 19 hour
max temp at day 5 is at: 2 hour
max temp at day 6 is at: 23 hour
max temp at day 7 is at: 16 hour
```

2.5 Extreme Temperature Events:

[75]:
```
t= 25
et = np.where(temperatures > t)
n= len(et[0])
print(f"Indices when temperature exceeds {t}: {et}")
print(f"Number of occurrences when temperature exceeds {t}: {n}")
```

```
Indices when temperature exceeds 25: (array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 3, 3, 3,
       3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 5,
       5, 5, 5, 5, 6, 6, 6, 6, 6, 6, 6], dtype=int64), array([ 2,  6,  7,  8,
11, 12, 13, 15, 18, 20,  0,  2,  5,  6, 10, 11, 13,
       14, 16, 17, 18, 20, 21, 22, 23,  0,  1,  3,  5,  8,  9, 13, 18, 19,
       20, 21,  0,  1,  2,  4,  7,  9, 10, 19, 22,  1,  2,  3,  6, 10, 13,
       14, 15, 17, 18, 21, 22, 23,  0,  1,  2,  4, 10, 11, 13, 14, 18, 20,
       22, 23,  0,  2,  3,  5, 11, 16, 22], dtype=int64))
Number of occurrences when temperature exceeds 25: 77
```

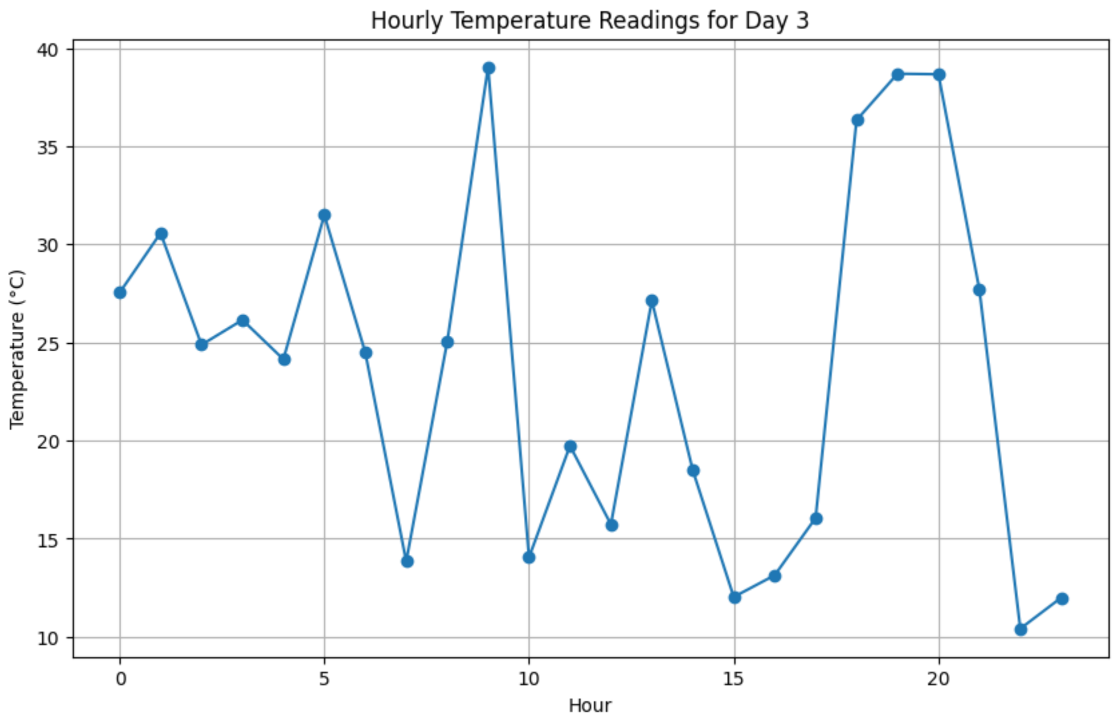
2.6 Temperature Fluctuations:

[76]:
```python
temp_deltas = np.diff(temperatures)
max_increase_hour = np.argmax(temp_deltas)
max_decrease_hour = np.argmin(temp_deltas)

print(f"Hour with max positive temperature change: {max_increase_hour}")
print(f"Hour with max negative temperature change: {max_decrease_hour}")
```

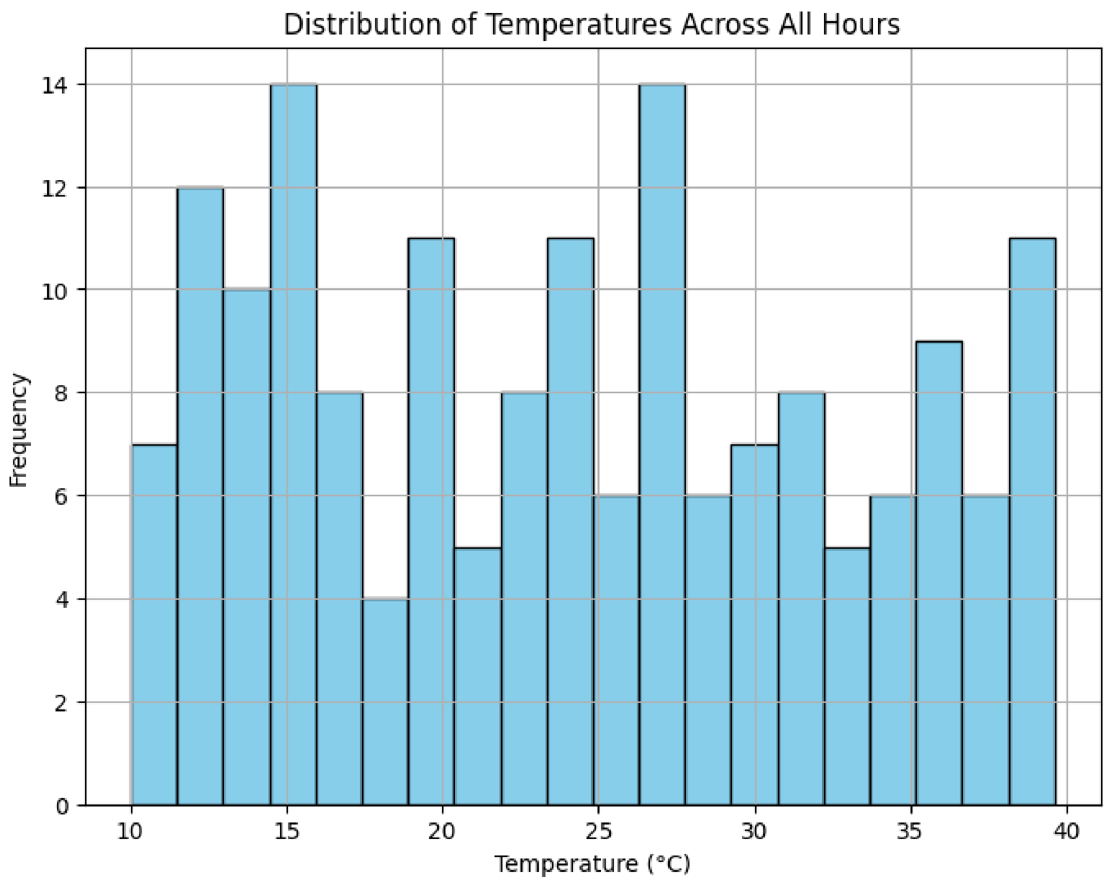
```
Hour with max positive temperature change: 42
Hour with max negative temperature change: 41
```

2.7 Visualization:

[95]:
```python
import matplotlib.pyplot as plt
selected_day = 2

plt.figure(figsize=(10, 6))
plt.plot(np.arange(24), temperatures[selected_day], marker='o')
plt.title("Hourly Temperature Readings for Day " + str(selected_day + 1))
plt.xlabel("Hour")
plt.ylabel("Temperature (°C)")
plt.grid(True)
plt.show()
```



[94]:
```
plt.figure(figsize=(8, 6))
plt.hist(temperatures.flatten(), bins=20, color='skyblue', edgecolor='black')  #␣
↪Flatten the temperatures array
plt.title("Distribution of Temperatures Across All Hours")
plt.xlabel("Temperature (°C)")
plt.ylabel("Frequency")
plt.grid(True)
plt.show()
```



[ ]:
```
```