# Barron Clarence and Neal Jarzen Data Clean and Graphing of Heart Rate and SpO2 for Lab 6

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
#Reads the data sheet
hrO2 = pd.read_csv(r"C:\Users\c3myb\OneDrive\Documents\College Files\CPE
hrO2
```

Out[2]:

| | HRvalue | ts | HRvalid | SpO2valid | SpO2value |
|---|---|---|---|---|---|
| **0** | 93 | 1680296553234 | True | True | 99.169194 |
| **1** | 93 | 1680296680823 | True | True | 98.794776 |
| **2** | 93 | 1680296794691 | True | True | 98.752554 |
| **3** | 93 | 1680296893713 | True | True | 99.727416 |
| **4** | 93 | 1680297178186 | True | True | 97.990506 |
| **...** | ... | ... | ... | ... | ... |
| **61** | 107 | 1680296723730 | True | True | 98.651946 |
| **62** | 107 | 1680297035948 | True | True | 96.727986 |
| **63** | 107 | 1680297249208 | True | True | 97.802616 |
| **64** | 83 | 1680296609760 | True | True | 99.59255399999999 |
| **65** | 83 | 1680296950750 | True | True | 99.169194 |

66 rows × 5 columns

In [3]:

```
#Shows the info of the heart rate
hrO2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 66 entries, 0 to 65
Data columns (total 5 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   HRvalue    66 non-null     object
 1   ts         66 non-null     int64 
 2   HRvalid    66 non-null     bool  
 3   SpO2valid  66 non-null     bool  
 4   SpO2value  66 non-null     object
dtypes: bool(2), int64(1), object(2)
memory usage: 1.8+ KB
```

In [4]:

```
#Shows the amount of valid data in the data set
print ('There are',hrO2['HRvalid'].sum(), 'valid data')
print ('There are',hrO2['SpO2valid'].sum(), 'valid data')
```

```
There are 58 valid data
There are 58 valid data
```

In [5]:

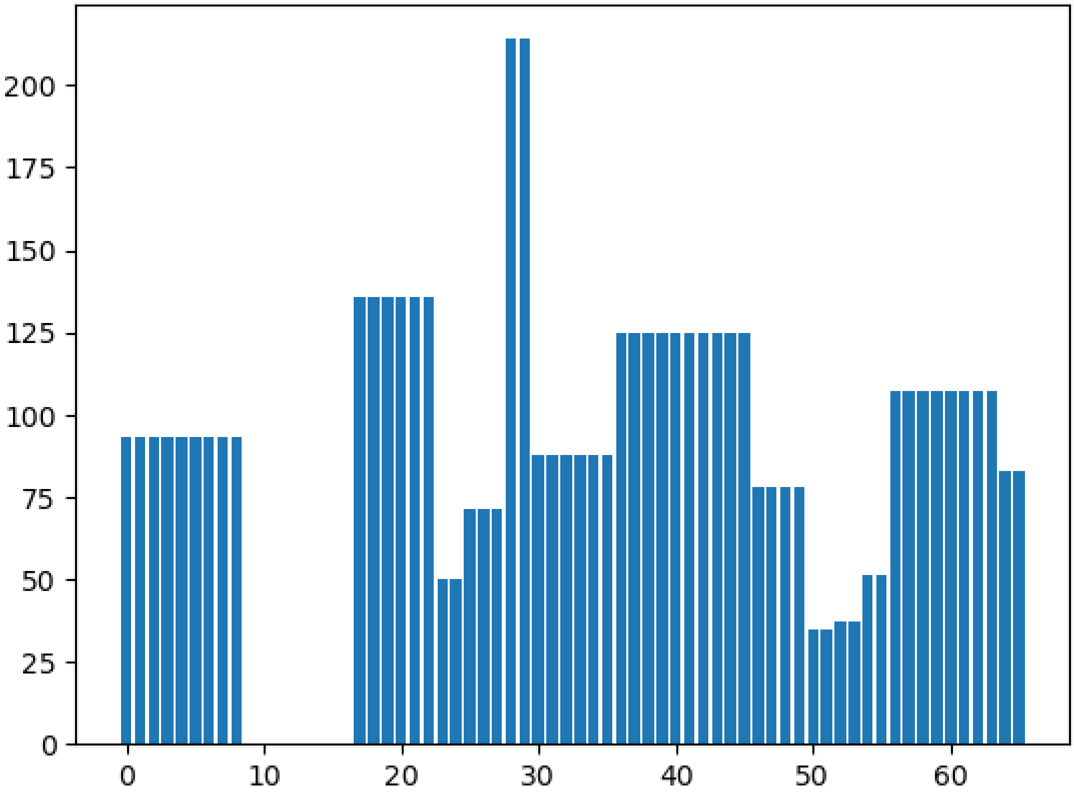
```
#Displays the values of the raw data
hr_1 = pd.to_numeric(hrO2['HRvalue'], errors='coerce')
hr_1
```

Out[5]:

```
0      93.0
1      93.0
2      93.0
3      93.0
4      93.0
      ...
61    107.0
62    107.0
63    107.0
64     83.0
65     83.0
Name: HRvalue, Length: 66, dtype: float64
```

In [6]:

```
#Plots a bar graph of the raw data
plt.bar(hr_1.index, hr_1)
plt.show()
```



In [7]:

```
#Removes the NaN values in the data set
hr_1.dropna(inplace=True)
hr_1
```

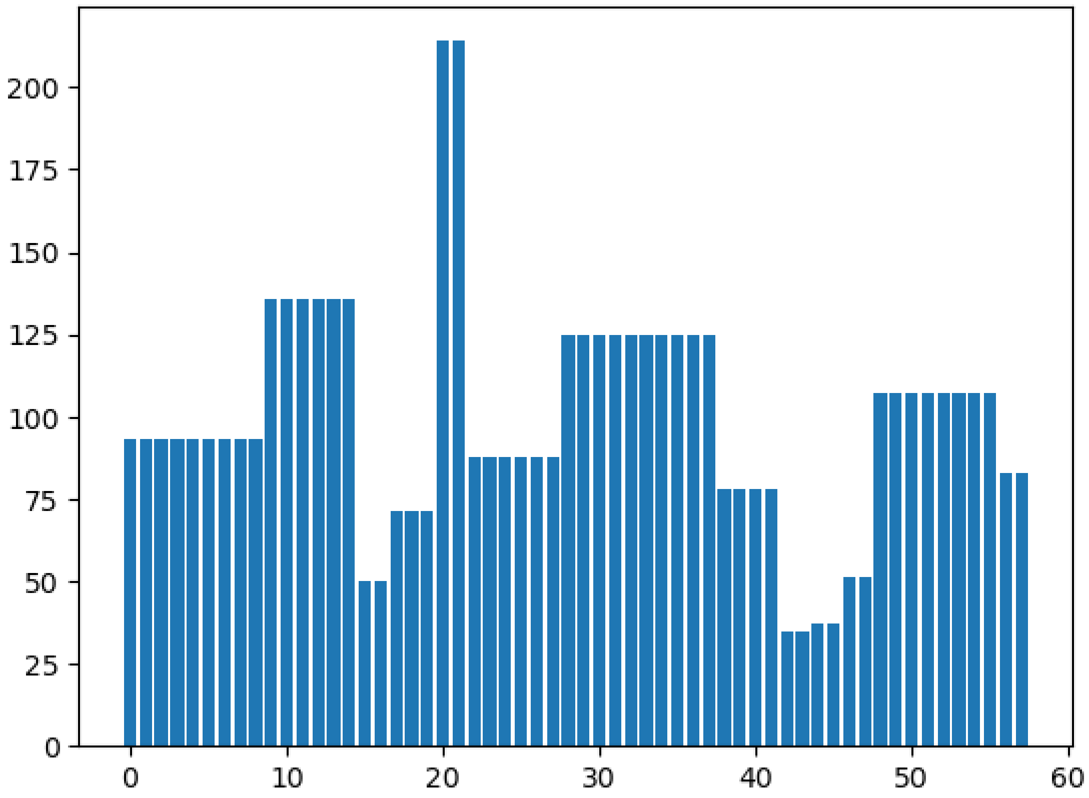
Out[7]:

```
0      93.0
1      93.0
2      93.0
3      93.0
4      93.0
5      93.0
6      93.0
7      93.0
8      93.0
17    136.0
18    136.0
19    136.0
```

```
20    136.0
21    136.0
22    136.0
23     50.0
24     50.0
25     71.0
26     71.0
27     71.0
28    214.0
29    214.0
30     88.0
31     88.0
32     88.0
33     88.0
34     88.0
35     88.0
36    125.0
37    125.0
38    125.0
39    125.0
40    125.0
41    125.0
42    125.0
43    125.0
44    125.0
45    125.0
46     78.0
47     78.0
48     78.0
49     78.0
50     35.0
51     35.0
52     37.0
53     37.0
54     51.0
55     51.0
56    107.0
57    107.0
58    107.0
59    107.0
60    107.0
61    107.0
62    107.0
63    107.0
64     83.0
65     83.0
Name: HRvalue, dtype: float64
```

In [8]:

```
#Gets rid of the values that have been removed with the last code block
hr_1.reset_index(drop=True, inplace=True)
hr_1
plt.bar(hr_1.index, hr_1)
plt.show()
```



In [9]:

```
#Displays the values of the Sp02
O2_1 = pd.to_numeric(hrO2['SpO2value'], errors='coerce')
O2_1
```

Out[9]:

```
0      99.169194
1      98.794776
2      98.752554
3      99.727416
4      97.990506
         ...
61     98.651946
62     96.727986
63     97.802616
64     99.592554
65     99.169194
Name: SpO2value, Length: 66, dtype: float64
```

In [10]:

```
#Cleans the data of the Sp02 of the NaN numbers and removes the indexes c
O2_1.dropna(inplace=True)
O2_1.reset_index(drop=True, inplace=True)
O2_1
```

Out[10]:

```
0      99.169194
1      98.794776
2      98.752554
3      99.727416
4      97.990506
5      98.651946
6      99.016626
7      97.118136
8      98.500104
9      97.399800
10     97.802616
11     98.794776
12     99.824664
13     96.235050
14     96.960936
```

```
15    97.399800
16    96.727986
17    98.169384
18    99.053400
19    98.889096
20    97.335786
21    97.990506
22    99.053400
23    99.169194
24    96.486024
25    98.794776
26    96.235050
27    97.990506
28    98.651946
29    98.651946
30    99.169194
31    99.542250
32    99.758856
33    98.928594
34    99.657000
35    98.928594
36    98.651946
37    97.184874
38    99.016626
39    98.500104
40    98.339250
41    99.771114
42    98.452434
43    99.345144
44    96.960936
45    96.486024
46    98.752554
47    99.053400
48    99.855786
49    98.500104
50    98.651946
51    99.053400
52    97.605714
53    98.651946
54    96.727986
55    97.802616
56    99.592554
57    99.169194
Name: SpO2value, dtype: float64
```

In [11]:

```
#Replaces the outliners with the median value and displays the aggregated
median = hr_1.median()
hr_1[hr_1 > 150] = median
hr_1[hr_1 < 50] = median
hr_1.describe()
```

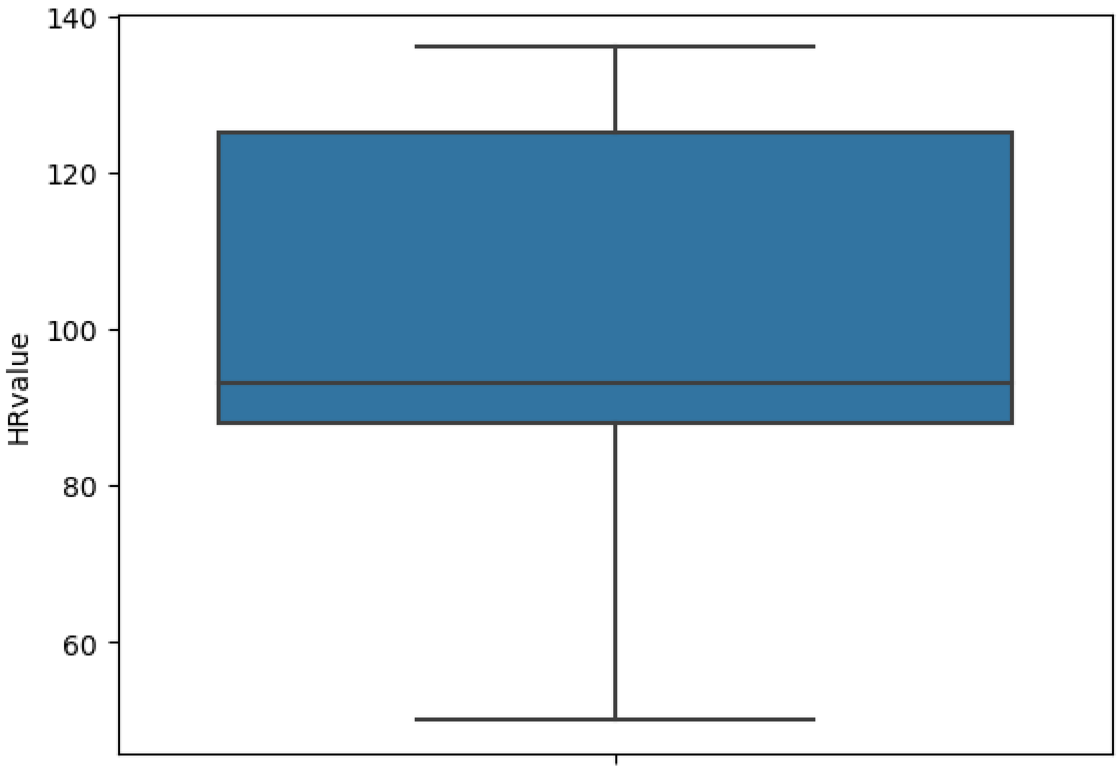
Out[11]:

```
count     58.000000
mean      98.931034
std       23.241570
min       50.000000
25%       88.000000
50%       93.000000
75%      125.000000
max      136.000000
Name: HRvalue, dtype: float64
```

In [12]:

```
#Shows a boxplot of the data in beteween 150 and 50 BPM
hr_2 = hr_1[(hr_1 <= 150) & (hr_1 >=50)]
```

```
sns.boxplot(y=hr_2)
plt.show()
```



In [13]:

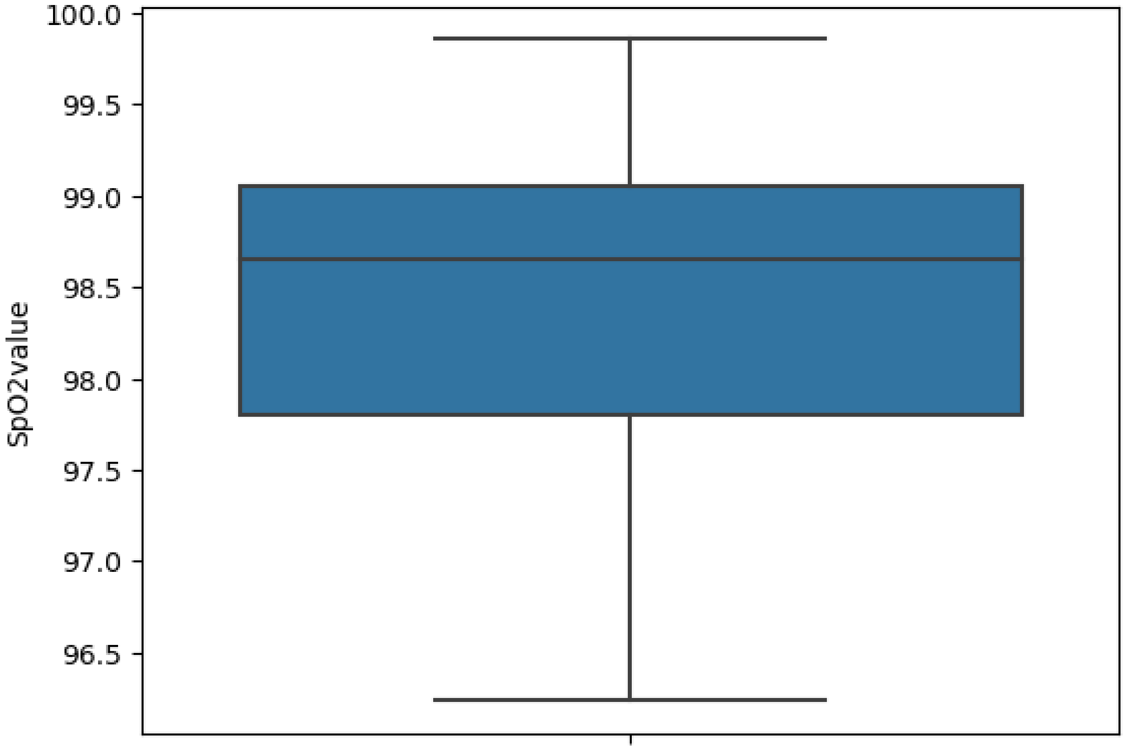
```
#Displays the aggregate data of the Sp02 values
O2_1.describe()
```

Out[13]:

```
count    58.000000
mean     98.404277
std       0.992875
min      96.235050
25%      97.802616
50%      98.651946
75%      99.053400
max      99.855786
Name: SpO2value, dtype: float64
```

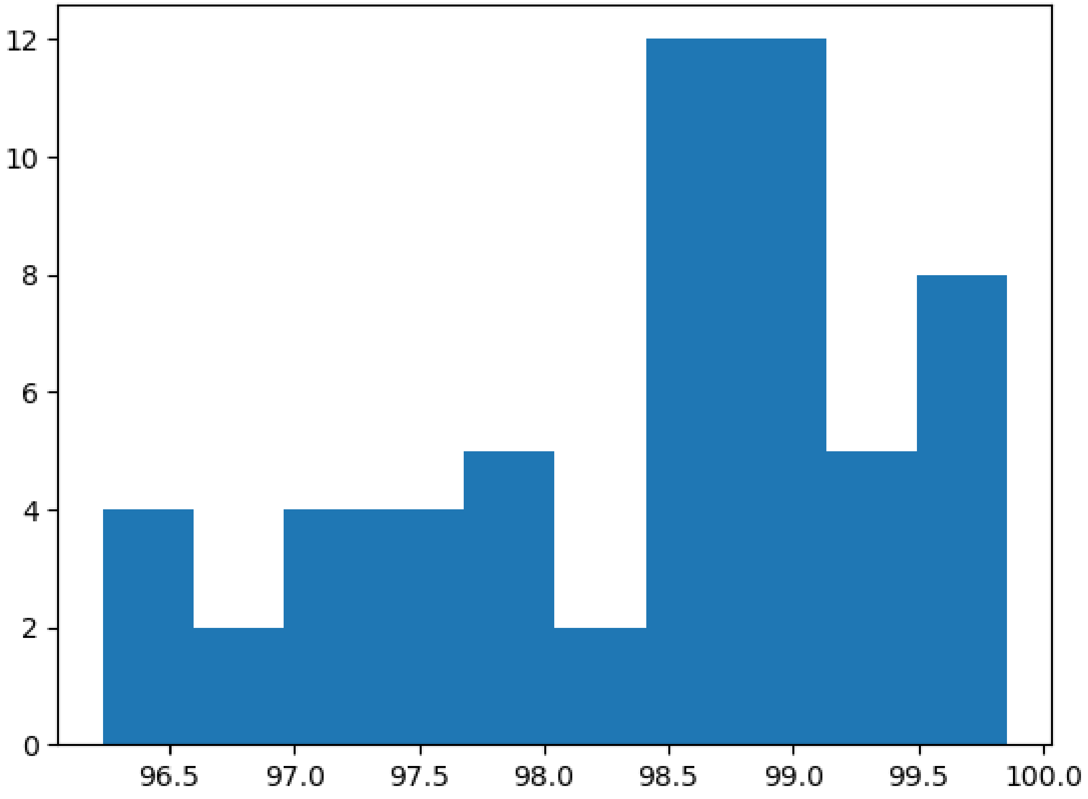
In [14]:

```
#Displays the BoxPlot of the Sp02 data that is not lower than 80.
O2_2 = O2_1[O2_1 > 80]
sns.boxplot(y=O2_2)
plt.show()
```



In [15]:

```
#Displays a bar graph of the Sp02 values.
plt.hist(O2_2, bins=10)
plt.show()
```



In [ ]: