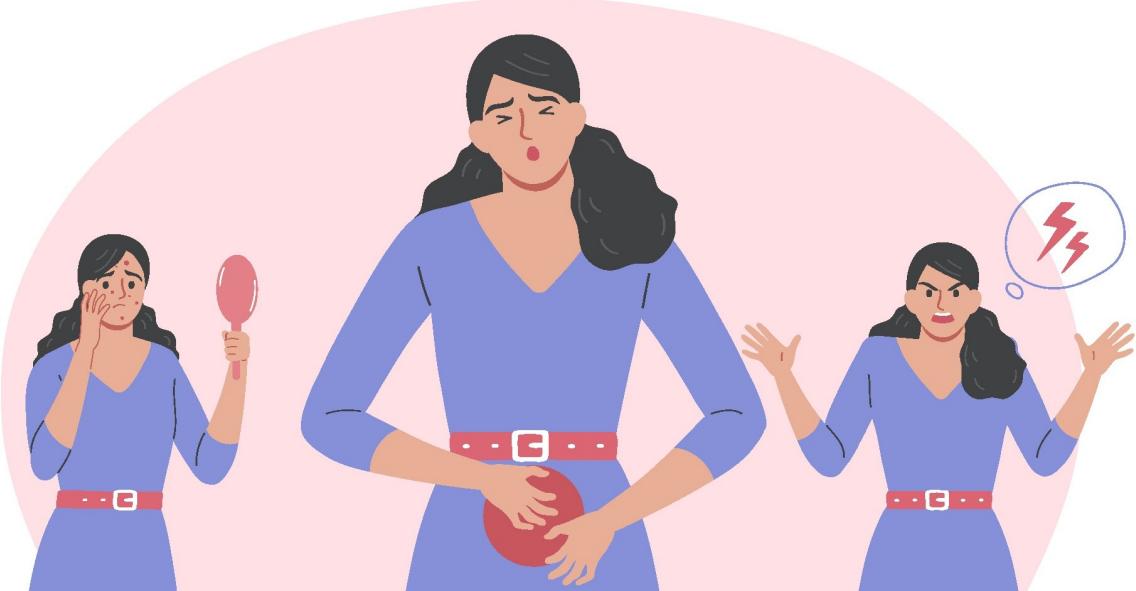


# Various Health Conditions Due to Menstruation (like PCOS)



```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [2]: import warnings
warnings.filterwarnings('ignore')
```

```
In [3]: df = pd.read_csv('periods.csv')
```

```
In [4]: df.head()
```

```
Out[4]:
```

	number_of_peak	Age	Length_of_cycle	Estimated_day_of_ovulation	Length_of_Leutal_Phase	Length_of_menses	Unusual_Bleeding	Height
0	3	18	27	14	9	5	no	5'6
1	4	18	25	17	10	6	yes	5'6
2	2	19	30	17	13	4	no	5'3
3	3	19	28	16	14	6	no	5'1
4	2	19	35	18	15	5	no	5'3

```
In [5]: df.tail()
```

```
Out[5]:
```

	number_of_peak	Age	Length_of_cycle	Estimated_day_of_ovulation	Length_of_Leutal_Phase	Length_of_menses	Unusual_Bleeding	Height
157	2	19	35	16	12	5	no	5'
158	2	18	30	13	14	6	no	5'
159	3	19	40	16	14	6	no	5'
160	2	18	32	15	14	8	no	5'
161	3	19	28	7	14	6	no	5'

```
In [6]: df.shape
```

```

In [6]: df.shape
Out[6]: (162, 13)

In [7]: df.columns
Out[7]: Index(['number_of_peak', 'Age', 'Length_of_cycle',
   'Estimated_day_of_ovulation', 'Length_of_Leutal_Phase',
   'Length_of_menses', 'Unusual_Bleeding', 'Height', 'Weight', 'Income',
   'BMI', 'Mean_of_length_of_cycle', 'Menses_score'],
  dtype='object')

In [8]: df.duplicated().sum()
Out[8]: 81

In [9]: df = df.drop_duplicates()

In [10]: df.isnull().sum()
Out[10]:
number_of_peak      0
Age                 0
Length_of_cycle    0
Estimated_day_of_ovulation 0
Length_of_Leutal_Phase 0
Length_of_menses   0
Unusual_Bleeding   0
Height              0
Weight              0
Income              0
BMI                0
Mean_of_length_of_cycle 0
Menses_score       0
dtype: int64

In [11]: df.info()
<class 'pandas.core.frame.DataFrame'>
Int64Index: 81 entries, 0 to 80
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype  
 ---  --  
 0   number_of_peak   81 non-null     int64  
 1   Age              81 non-null     int64  
 2   Length_of_cycle  81 non-null     int64  
 3   Estimated_day_of_ovulation 81 non-null     int64  
 4   Length_of_Leutal_Phase 81 non-null     int64  
 5   Length_of_menses  81 non-null     int64  
 6   Unusual_Bleeding 81 non-null     object  
 7   Height            81 non-null     object  
 8   Weight            81 non-null     float64 
 9   Income            81 non-null     int64  
 10  BMI               81 non-null     float64 
 11  Mean_of_length_of_cycle 81 non-null     int64  
 12  Menses_score     81 non-null     float64 
dtypes: float64(3), int64(8), object(2)
memory usage: 8.9+ KB

In [12]: df.describe()
Out[12]:
   number_of_peak      Age   Length_of_cycle  Estimated_day_of_ovulation  Length_of_Leutal_Phase  Length_of_menses   Weight   Income
count      81.000000  81.000000        81.000000          81.000000        81.000000        81.000000  81.000000      8
mean      2.518519  19.333333      29.283951         15.135802        12.753086        5.061728  56.696296      8
std       0.709068  1.457738      4.204862         2.359836        3.280743        1.028903  9.500993      8
min       1.000000  14.000000      22.000000         7.000000        7.000000        1.000000  40.000000      8
25%      2.000000  19.000000      27.000000         14.000000        10.000000        5.000000  51.000000      8
50%      2.000000  19.000000      28.000000         15.000000        13.000000        5.000000  55.000000      8
75%      3.000000  20.000000      30.000000         16.000000        14.000000        6.000000  60.000000      8
max      5.000000  25.000000      40.000000         22.000000        30.000000        8.000000  85.000000      8

In [13]: df.nunique()

```

```
Out[13]: number_of_peak      5
Age             8
Length_of_cycle 16
Estimated_day_of_ovulation 12
Length_of_Leutal_Phase   12
Length_of_menses       7
Unusual_Bleeding      4
Height            27
Weight            27
Income            1
BMI              44
Mean_of_length_of_cycle 18
Menses_score      7
dtype: int64
```

```
In [14]: df['Height']
```

```
Out[14]: 0      5 6
1      5 6
2      5 3
3      5 1
4      5'3
...
76     5'3
77     5'6
78     5'3
79     5'6
80     5'2"
Name: Height, Length: 81, dtype: object
```

```
In [15]: df['Height'] = df['Height'].str.strip()
df['Height'] = df['Height'].str.replace(' ', '')
df['Height'] = df['Height'].str.replace("'", '')
df['Height'] = df['Height'].apply(lambda x: f"{x[0]}.{x[2:]}" if "''" not in x else x)

def convert_to_inches(x):
    try:
        if "'" in x:
            feet, inches = map(int, x.split("''"))
            return feet * 12 + inches
        else:
            return pd.to_numeric(x, errors='coerce')
    except ValueError:
        return pd.to_numeric(x, errors='coerce')

df['Height'] = df['Height'].apply(convert_to_inches)
```

```
In [16]: df['Height']
```

```
Out[16]: 0      66.0
1      66.0
2      63.0
3      61.0
4      63.0
...
76     63.0
77     66.0
78     63.0
79     66.0
80     62.0
Name: Height, Length: 81, dtype: float64
```

```
In [17]: df['Unusual_Bleeding'] = df['Unusual_Bleeding'].str.lower()
df['Unusual_Bleeding'] = df['Unusual_Bleeding'].map({'yes': 'Yes', 'no': 'No'})
```

```
In [18]: df['Income'] = df['Income'].apply(lambda x: 'Low' if x == 0 else 'High')
df['Menses_score'] = df['Menses_score'].apply(lambda x: 'Low' if x <= 3 else 'High')
```

```
In [19]: object_columns = df.select_dtypes(include=['object']).columns
print("Object type columns:")
print(object_columns)

numerical_columns = df.select_dtypes(include=['int64', 'float64']).columns
print("\nNumerical type columns:")
print(numerical_columns)

Object type columns:
Index(['Unusual_Bleeding', 'Income', 'Menses_score'], dtype='object')

Numerical type columns:
Index(['number_of_peak', 'Age', 'Length_of_cycle',
       'Estimated_day_of_ovulation', 'Length_of_Leutal_Phase',
       'Length_of_menses', 'Height', 'Weight', 'BMI',
       'Mean_of_length_of_cycle'],
       dtype='object')
```

```
In [20]: def classify_features(df):
    categorical_features = []
```

```

non_categorical_features = []
discrete_features = []
continuous_features = []

for column in df.columns:
    if df[column].dtype == 'object':
        if df[column].nunique() < 10:
            categorical_features.append(column)
        else:
            non_categorical_features.append(column)
    elif df[column].dtype in ['int64', 'float64']:
        if df[column].nunique() < 10:
            discrete_features.append(column)
        else:
            continuous_features.append(column)

return categorical_features, non_categorical_features, discrete_features, continuous_features

```

```
In [21]: categorical, non_categorical, discrete, continuous = classify_features(df)
```

```
In [22]: print("Categorical Features:", categorical)
print("Non-Categorical Features:", non_categorical)
print("Discrete Features:", discrete)
print("Continuous Features:", continuous)
```

```
Categorical Features: ['Unusual_Bleeding', 'Income', 'Menses_score']
Non-Categorical Features: []
Discrete Features: ['number_of_peak', 'Age', 'Length_of_menses']
Continuous Features: ['Length_of_cycle', 'Estimated_day_of_ovulation', 'Length_of_Leutal_Phase', 'Height', 'Weight', 'BMI', 'Mean_of_length_of_cycle']
```

```
In [23]: for i in categorical:
    print(i, ':')
    print(df[i].unique())
    print()
```

```
Unusual_Bleeding :
['No' 'Yes']
```

```
Income :
['Low']
```

```
Menses_score :
['High' 'Low']
```

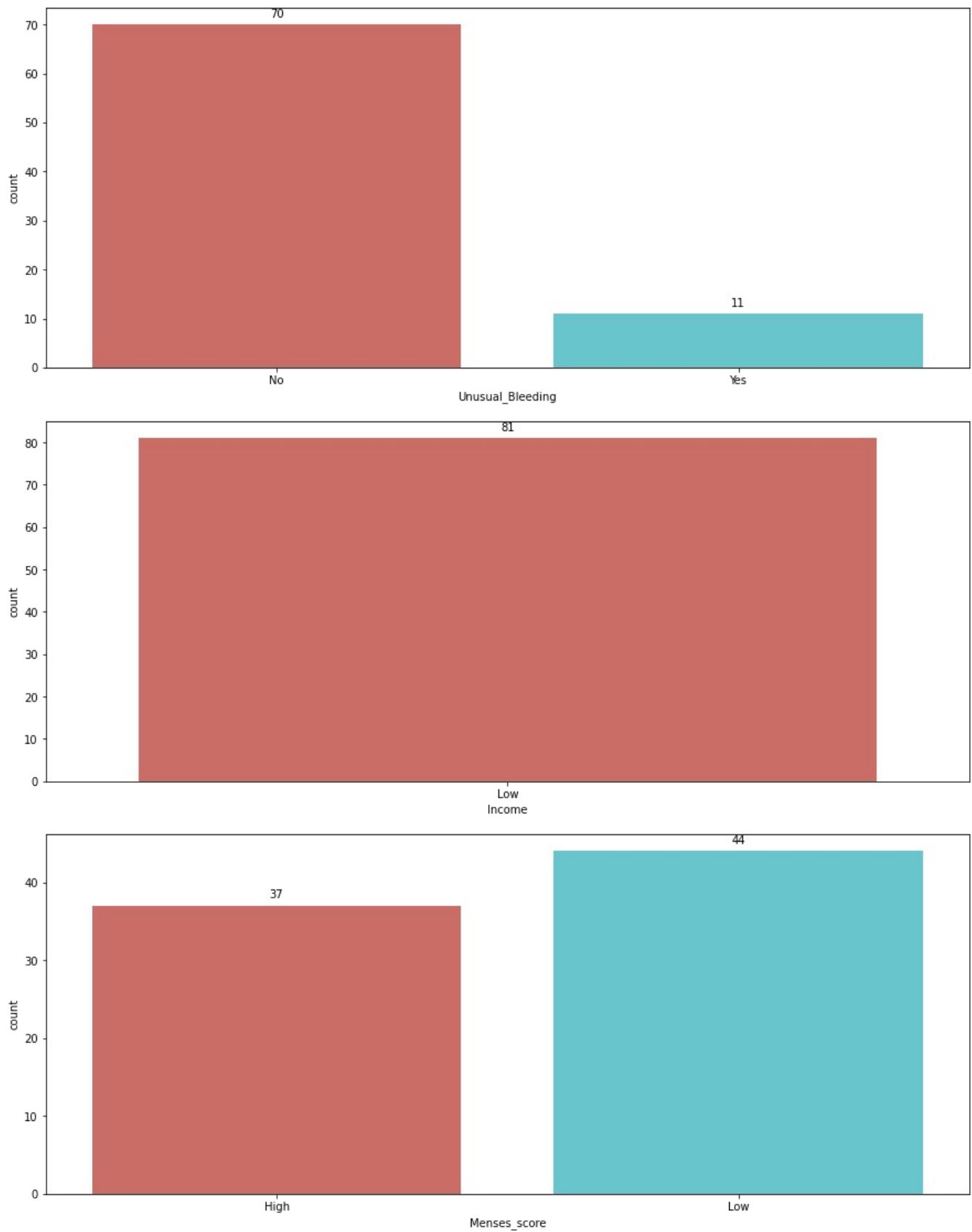
```
In [24]: for i in categorical:
    print(i, ':')
    print(df[i].value_counts())
    print()
```

```
Unusual_Bleeding :
No      70
Yes     11
Name: Unusual_Bleeding, dtype: int64
```

```
Income :
Low     81
Name: Income, dtype: int64
```

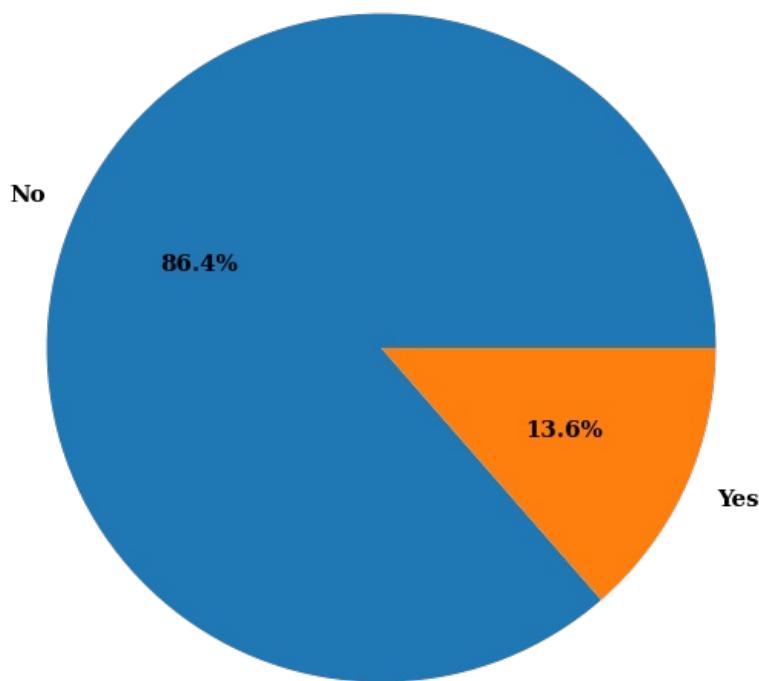
```
Menses_score :
Low     44
High    37
Name: Menses_score, dtype: int64
```

```
In [25]: for i in categorical:
    plt.figure(figsize=(15, 6))
    ax = sns.countplot(x=i, data=df, palette='hls')
    for p in ax.patches:
        ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                    ha='center', va='center', xytext=(0, 10), textcoords='offset points')
    plt.show()
```

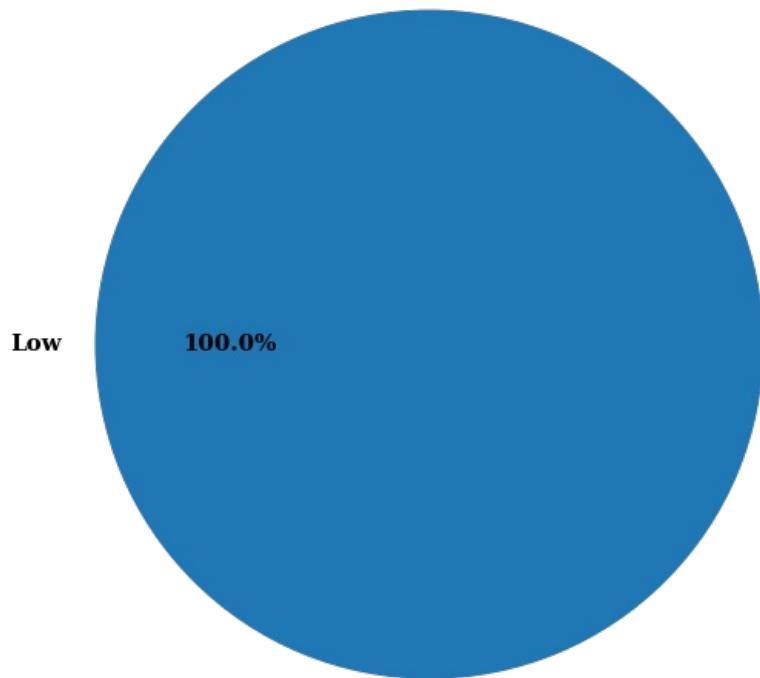


```
In [26]: for i in categorical:
    plt.figure(figsize=(20,10))
    plt.pie(df[i].value_counts(), labels=df[i].value_counts().index, autopct='%1.1f%%', textprops={'fontsize':
        'color': 'black',
        'weight': 'bold',
        'family': 'serif' })
    hfont = {'fontname':'serif', 'weight': 'bold'}
    plt.title(i, size=20, **hfont)
    plt.show()
```

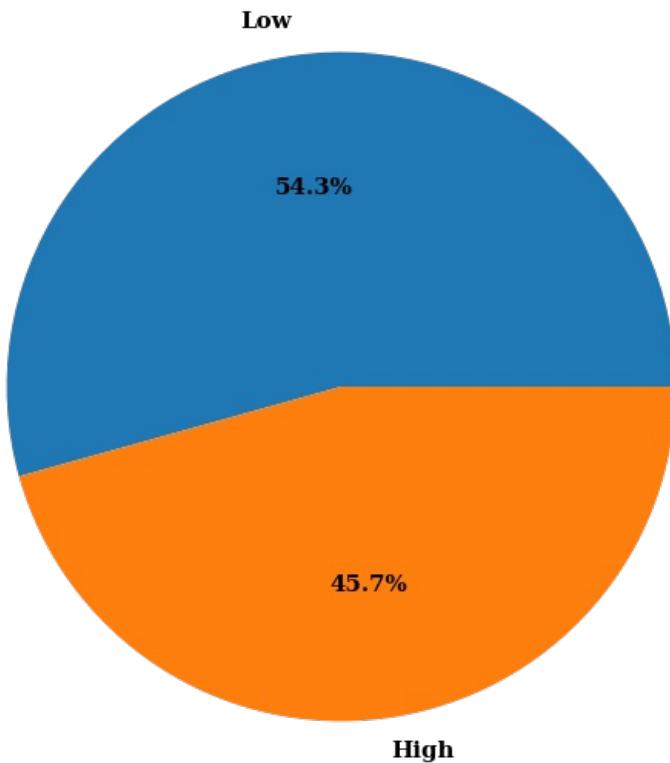
### **Unusual\_Bleeding**



### **Income**



## Menses\_score



```
In [27]: for i in discrete:  
    print(i, ':')  
    print(df[i].unique())  
    print()
```

number\_of\_peak :  
[3 4 2 5 1]

Age :  
[18 19 20 22 23 14 17 25]

Length\_of\_menses :  
[5 6 4 3 1 7 8]

```
In [28]: for i in discrete:  
    print(i, ':')  
    print(df[i].value_counts())  
    print()
```

number\_of\_peak :  
2 39  
3 34  
4 4  
1 3  
5 1  
Name: number\_of\_peak, dtype: int64

Age :  
19 38  
20 23  
18 10  
23 4  
17 3  
22 1  
14 1  
25 1  
Name: Age, dtype: int64

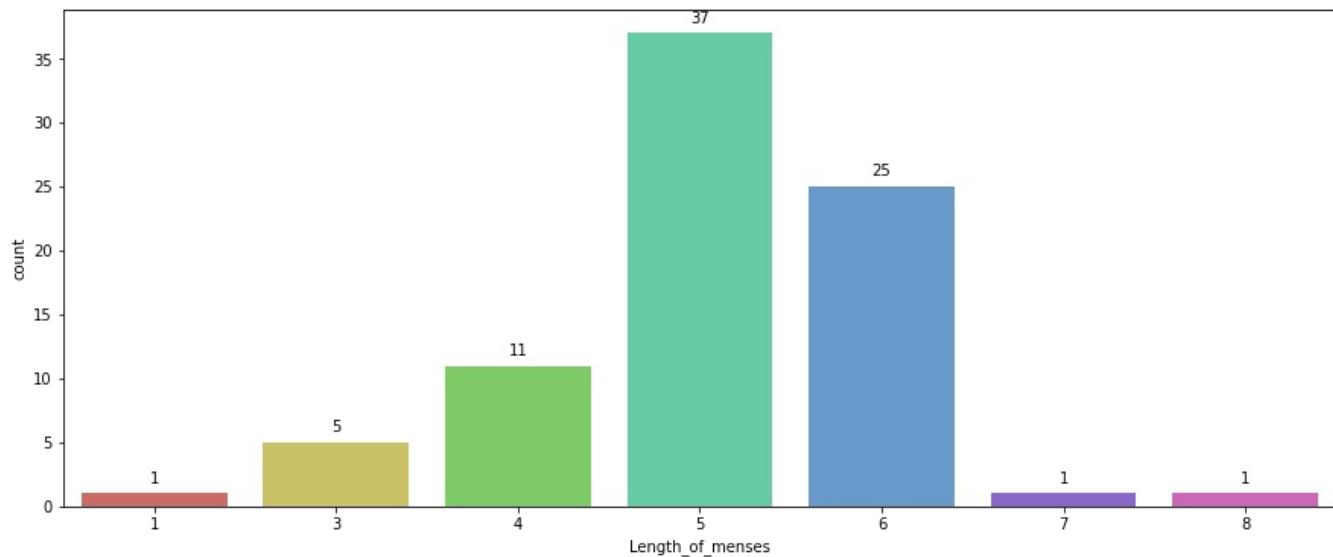
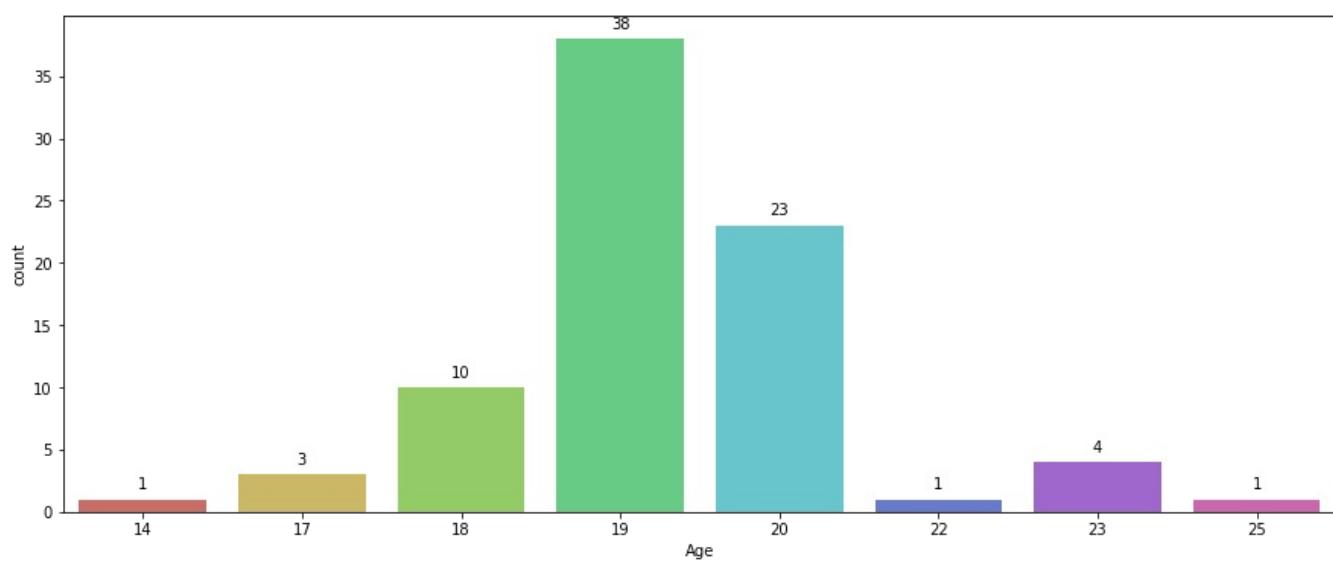
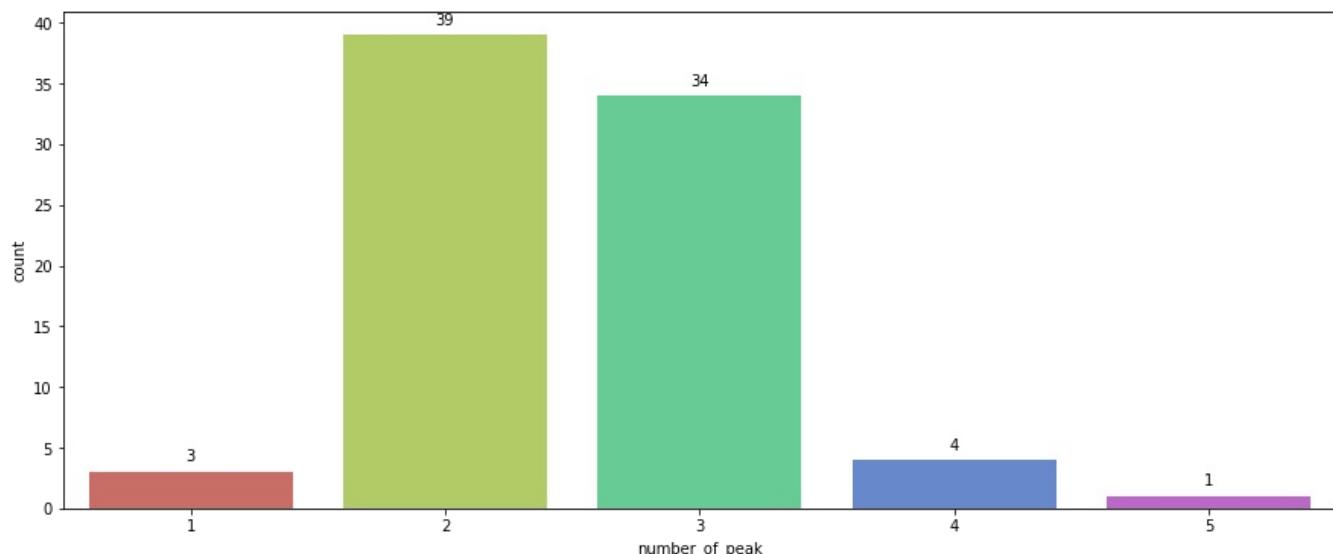
Length\_of\_menses :  
5 37  
6 25  
4 11  
3 5  
1 1  
7 1  
8 1  
Name: Length\_of\_menses, dtype: int64

```
In [29]: for i in discrete:  
    plt.figure(figsize=(15, 6))
```

```

ax = sns.countplot(x=i, data=df, palette='hls')
for p in ax.patches:
    ax.annotate(f'{p.get_height()}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', xytext=(0, 10), textcoords='offset points')
plt.show()

```

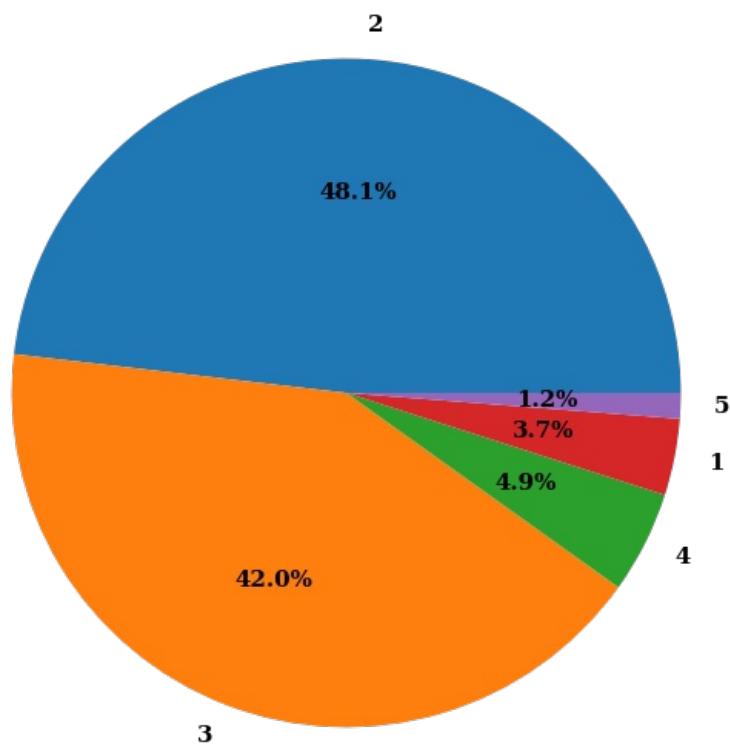


```

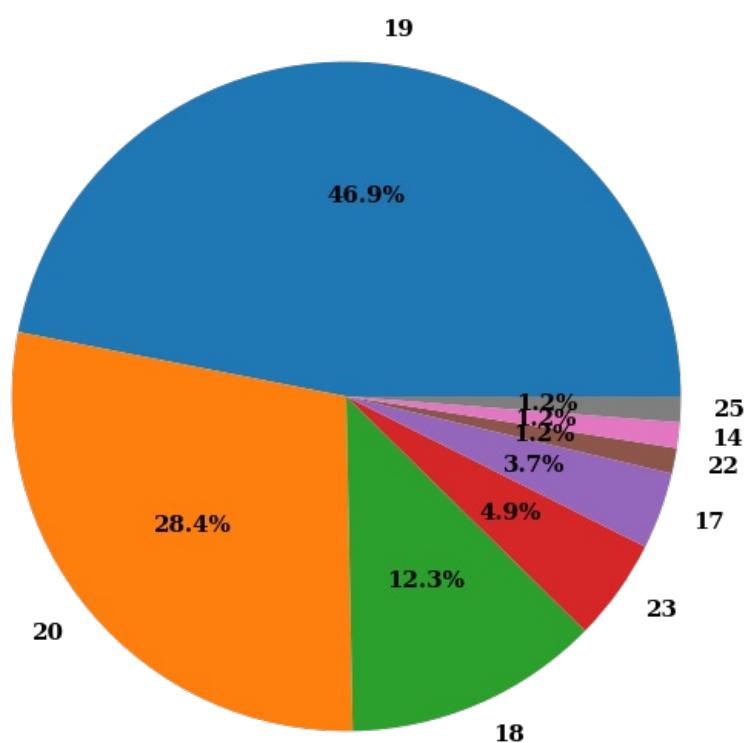
In [30]: for i in discrete:
    plt.figure(figsize=(20,10))
    plt.pie(df[i].value_counts(), labels=df[i].value_counts().index, autopct='%1.1f%%', textprops={'fontsize':
                                                'color': 'black',
                                                'weight': 'bold',
                                                'family': 'serif' })
    hfont = {'fontname':'serif', 'weight': 'bold'}
    plt.title(i, size=20, **hfont)
    plt.show()

```

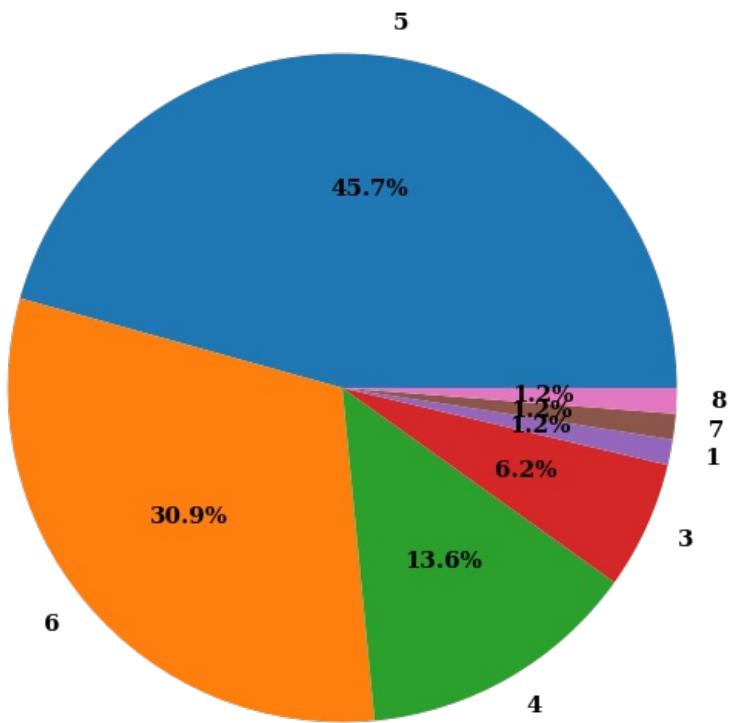
**number\_of\_peak**



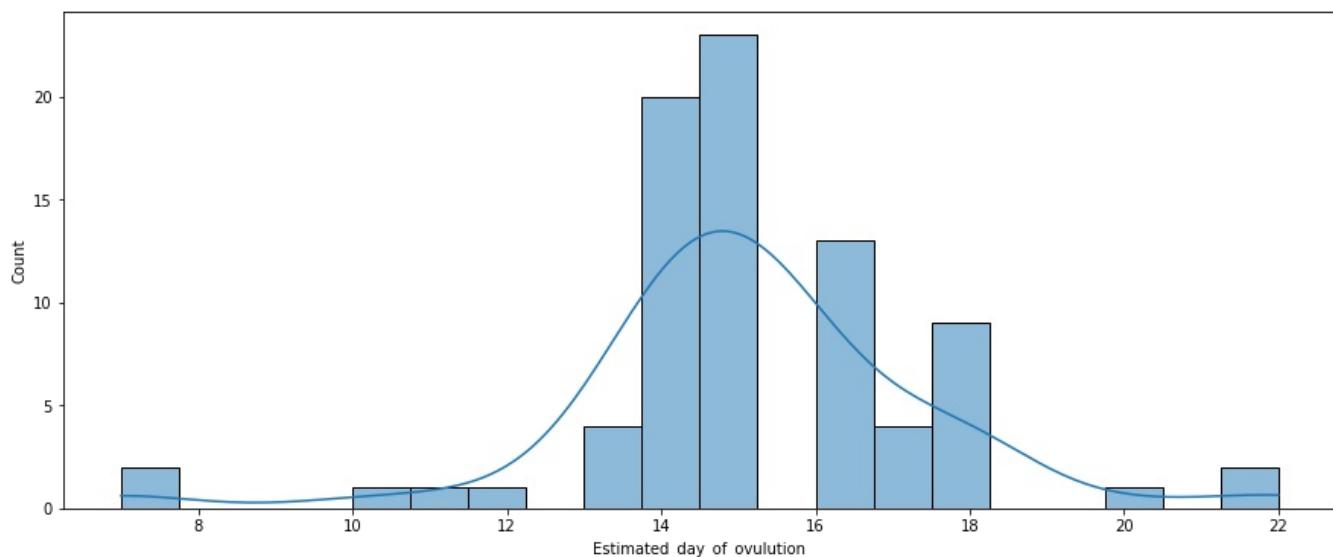
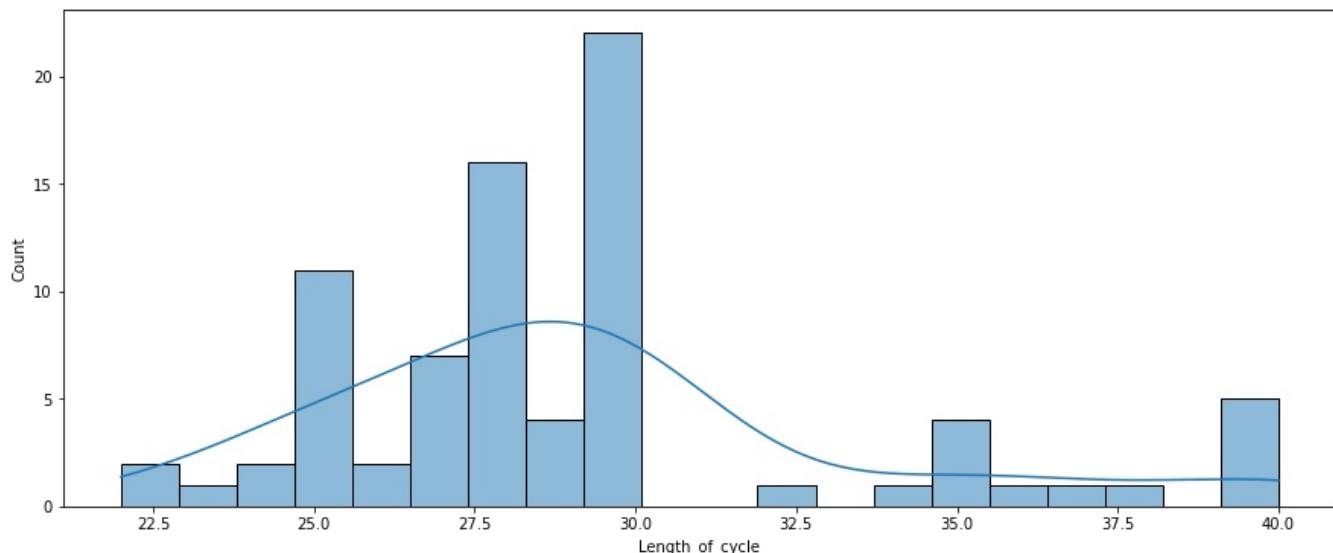
**Age**

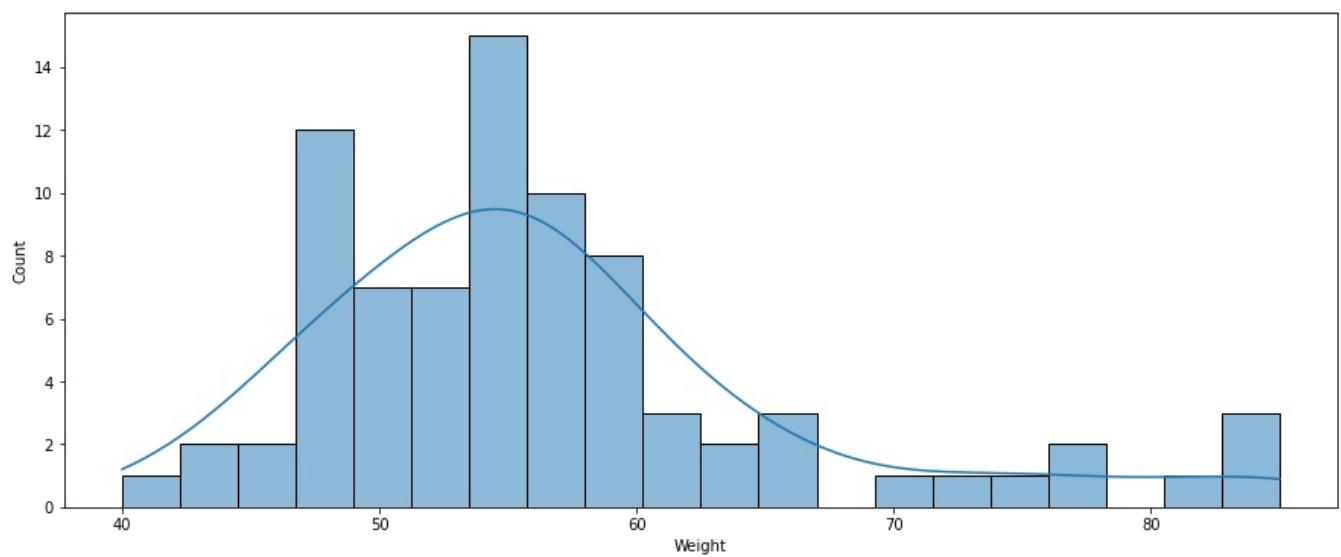
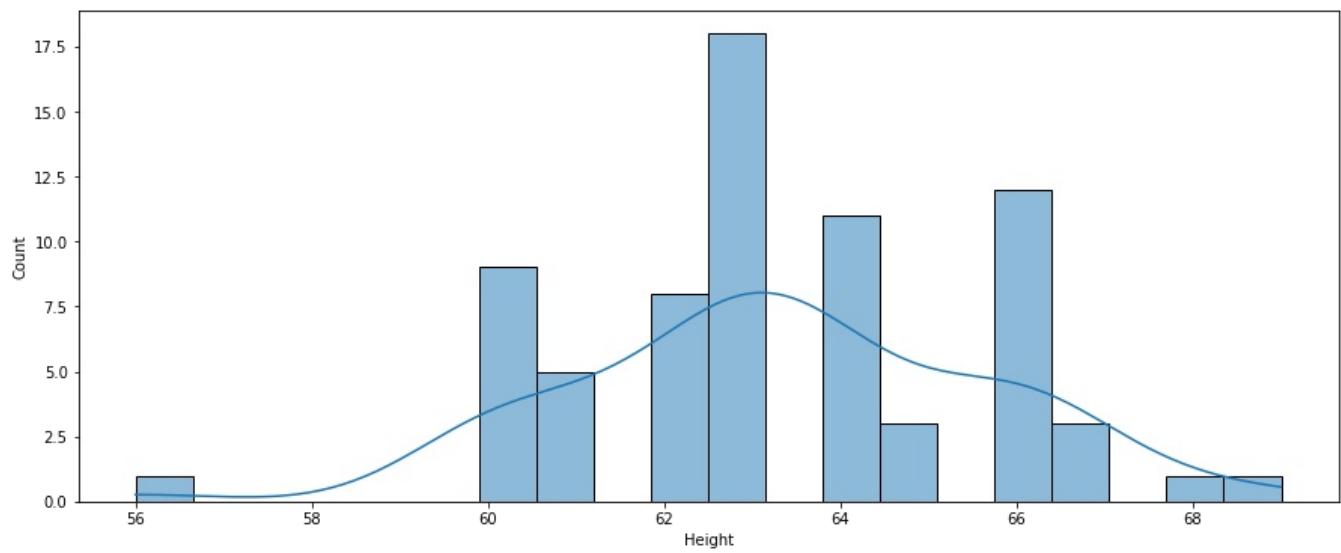
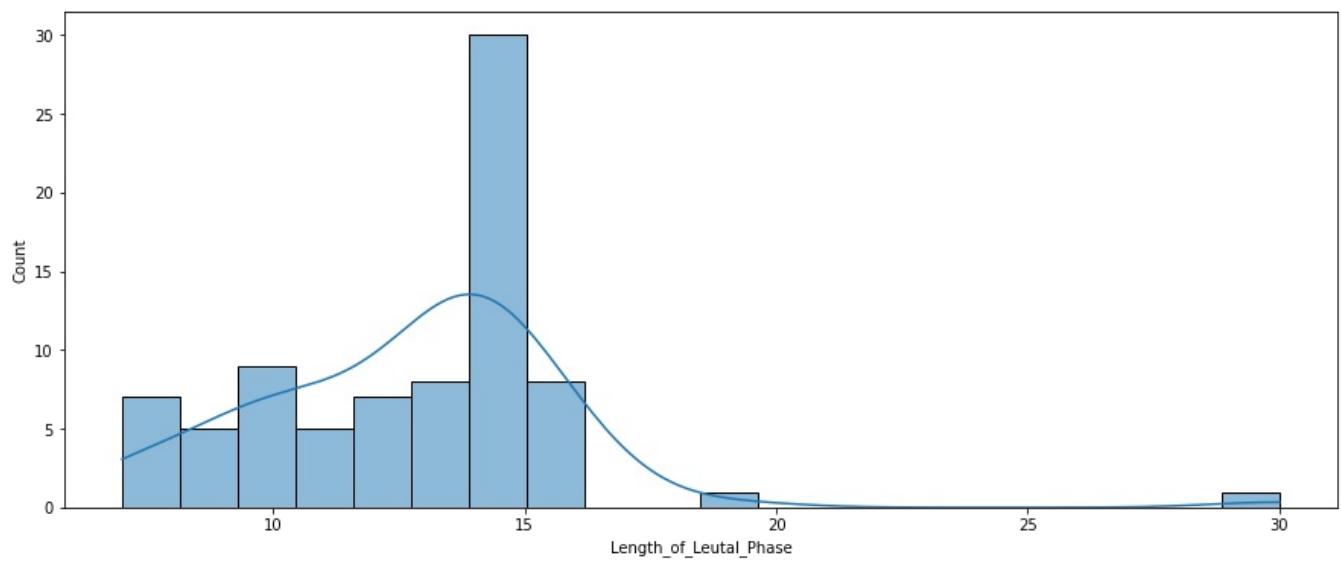


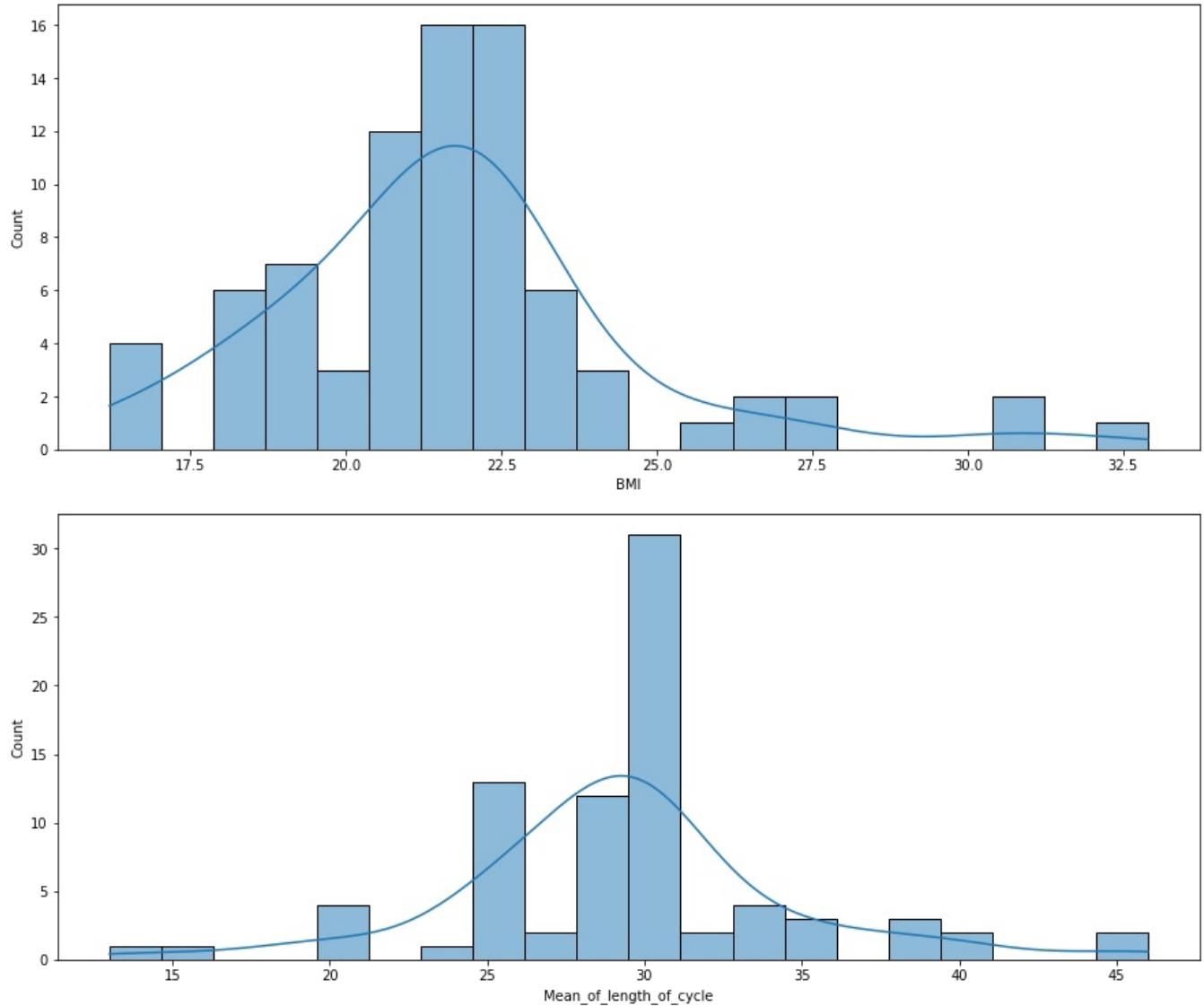
## Length\_of\_menses



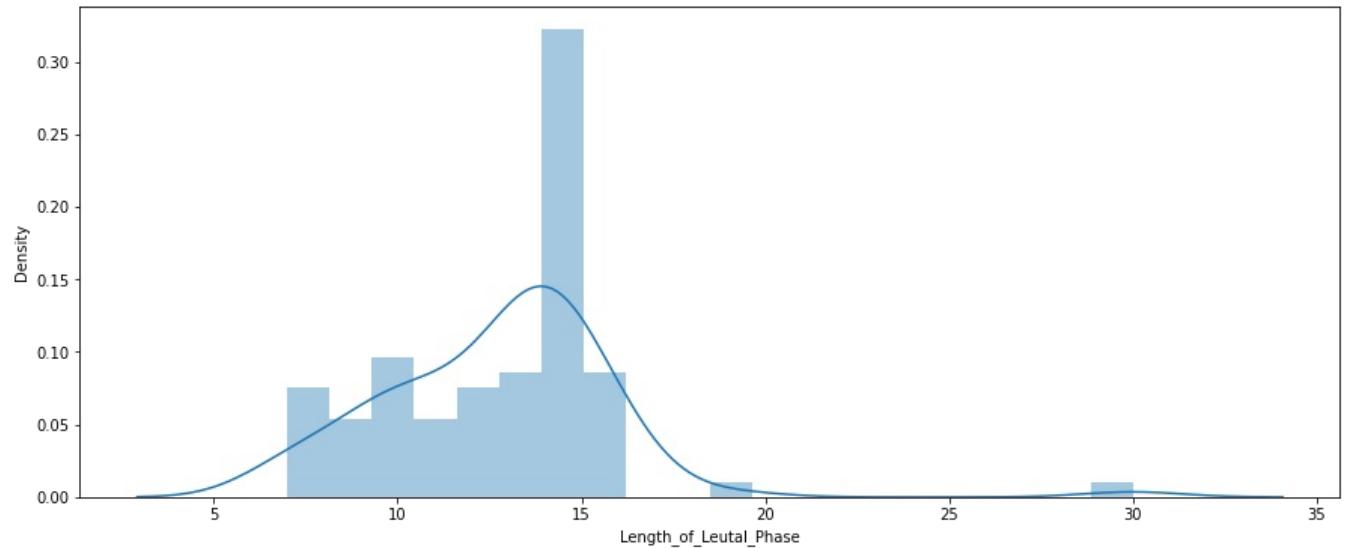
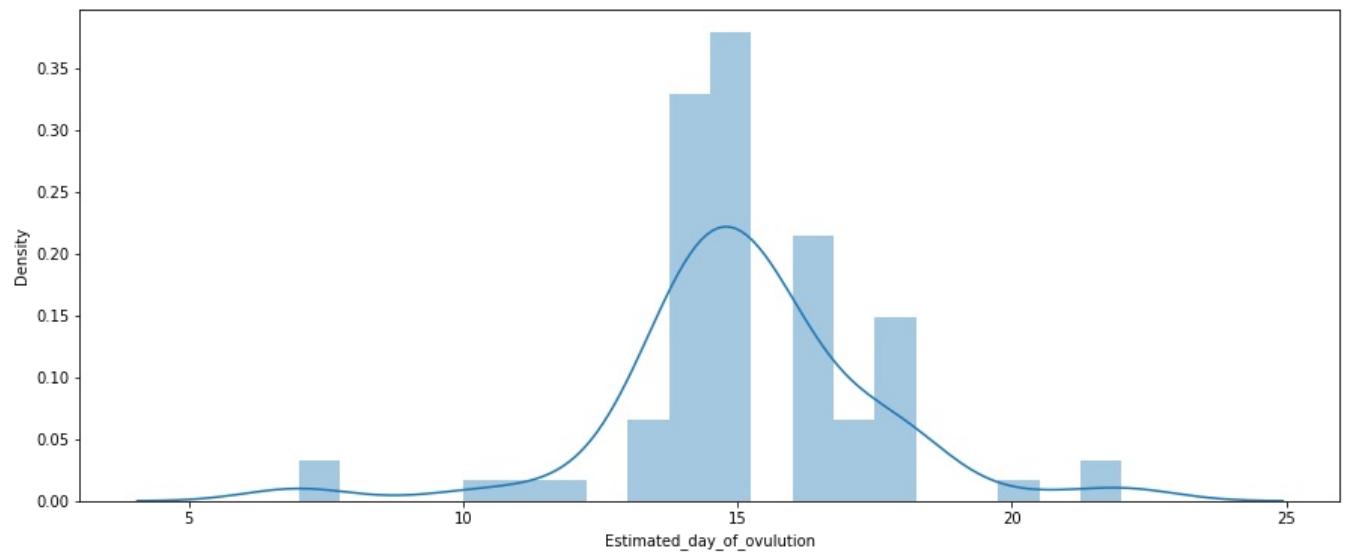
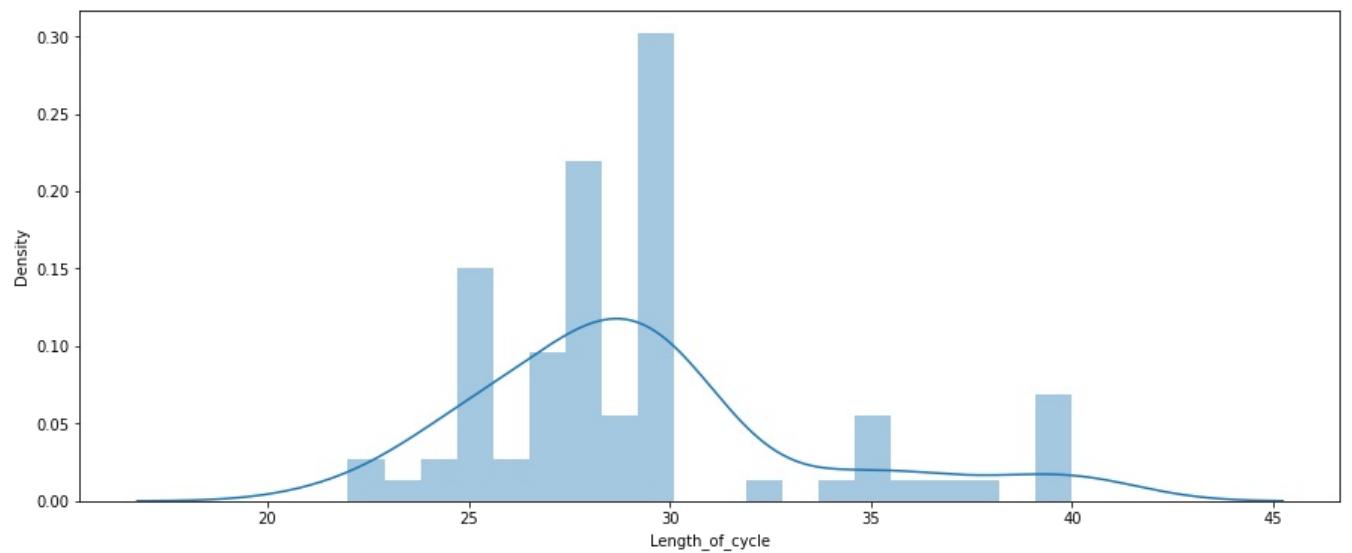
```
In [31]: for i in continuous:
    plt.figure(figsize=(15,6))
    sns.histplot(df[i], bins = 20, kde = True, palette='hls')
    plt.show()
```

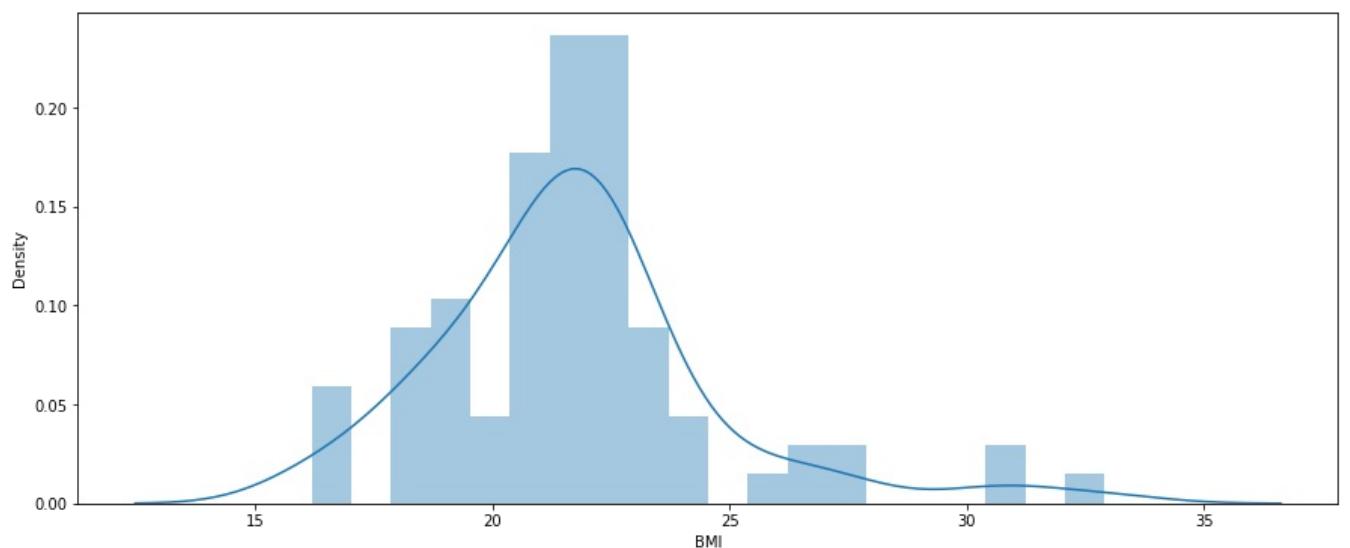
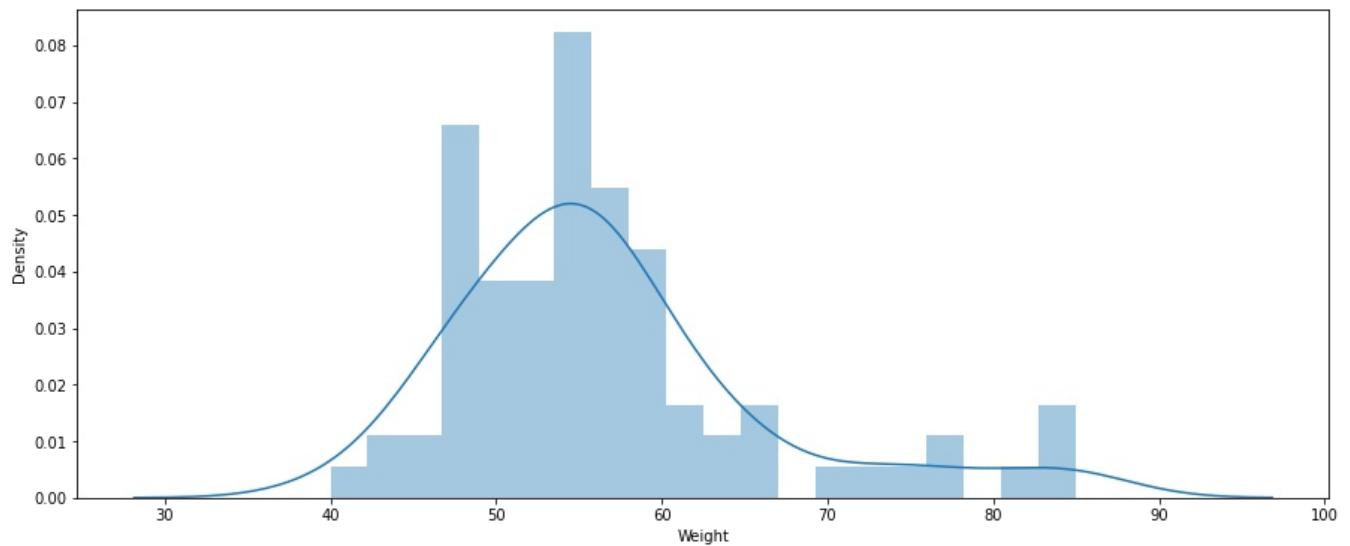
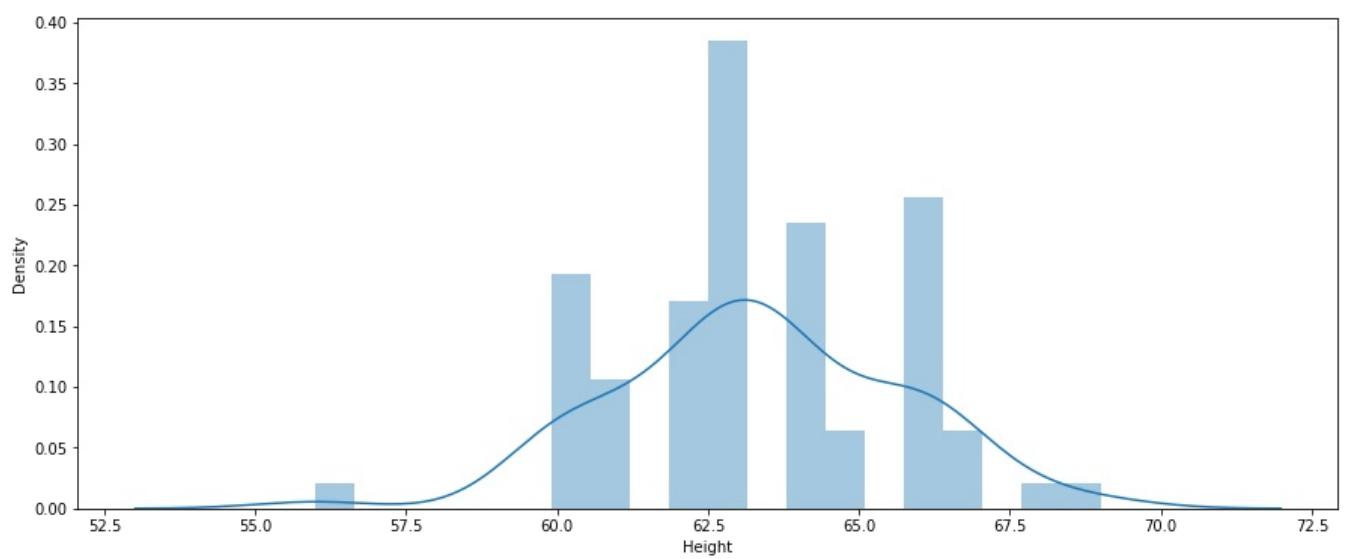


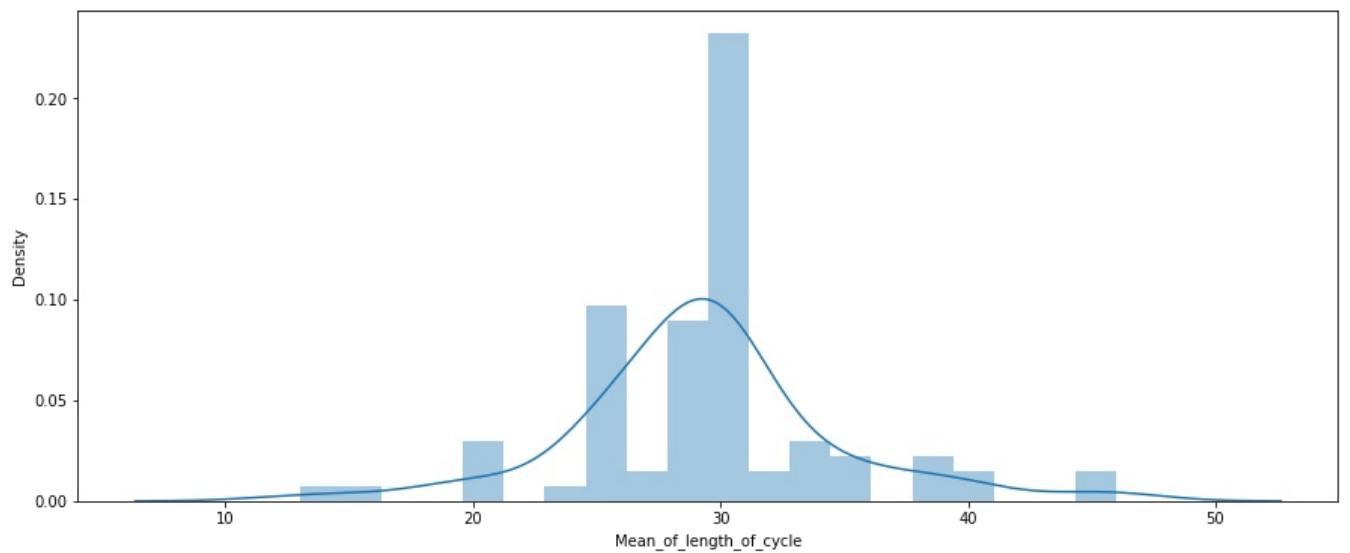




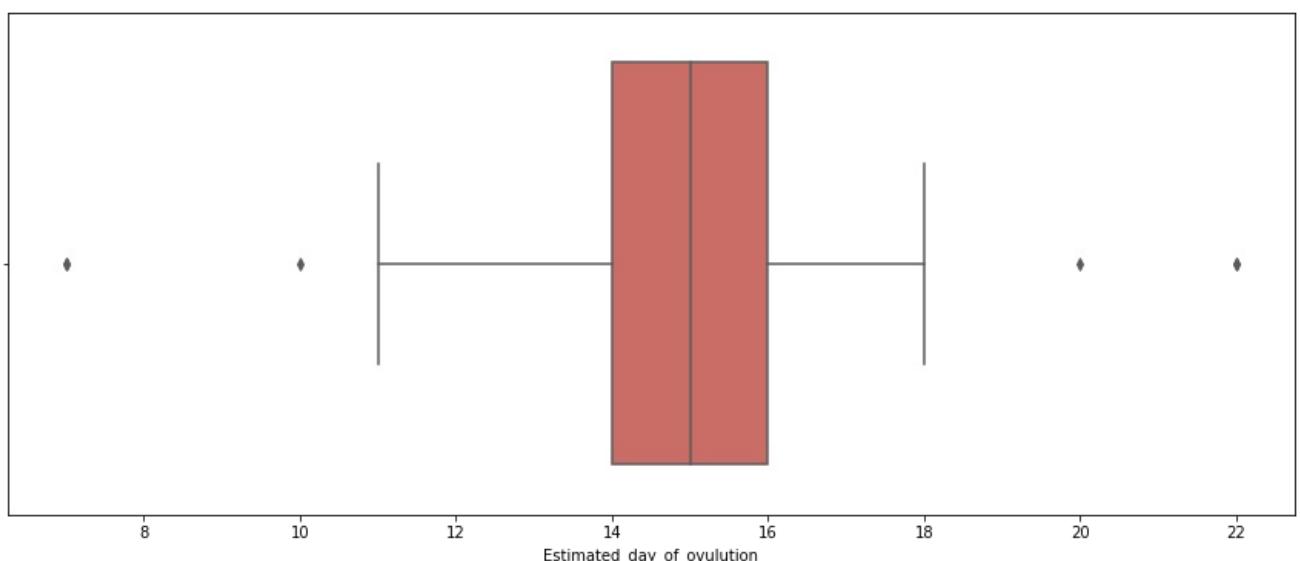
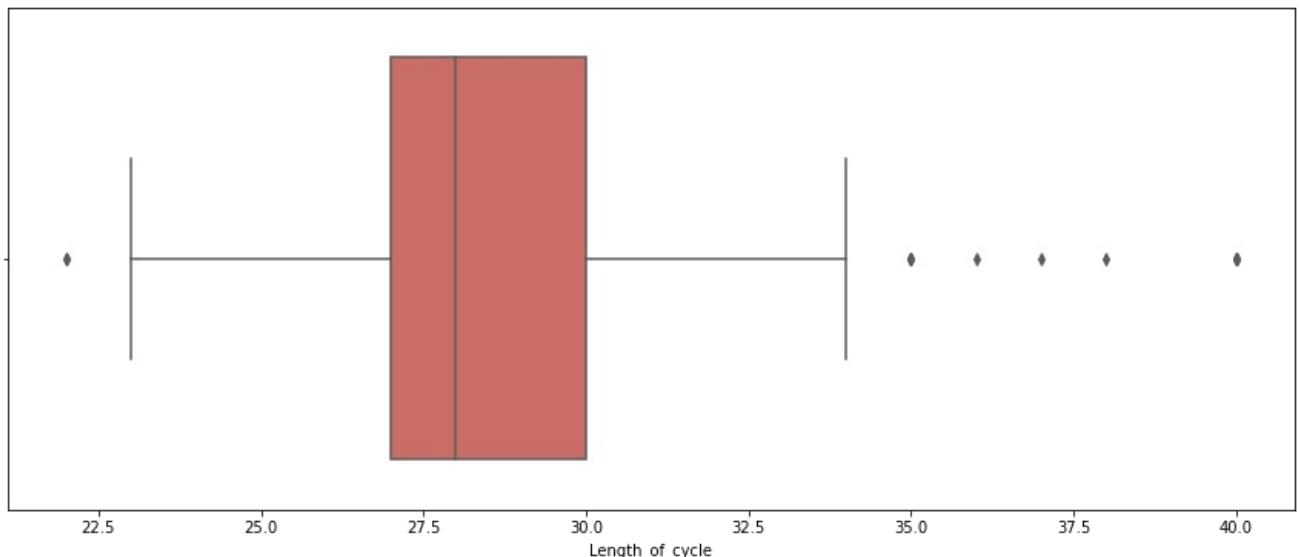
```
In [32]: for i in continuous:
    plt.figure(figsize=(15,6))
    sns.distplot(df[i], bins = 20, kde = True)
    plt.show()
```

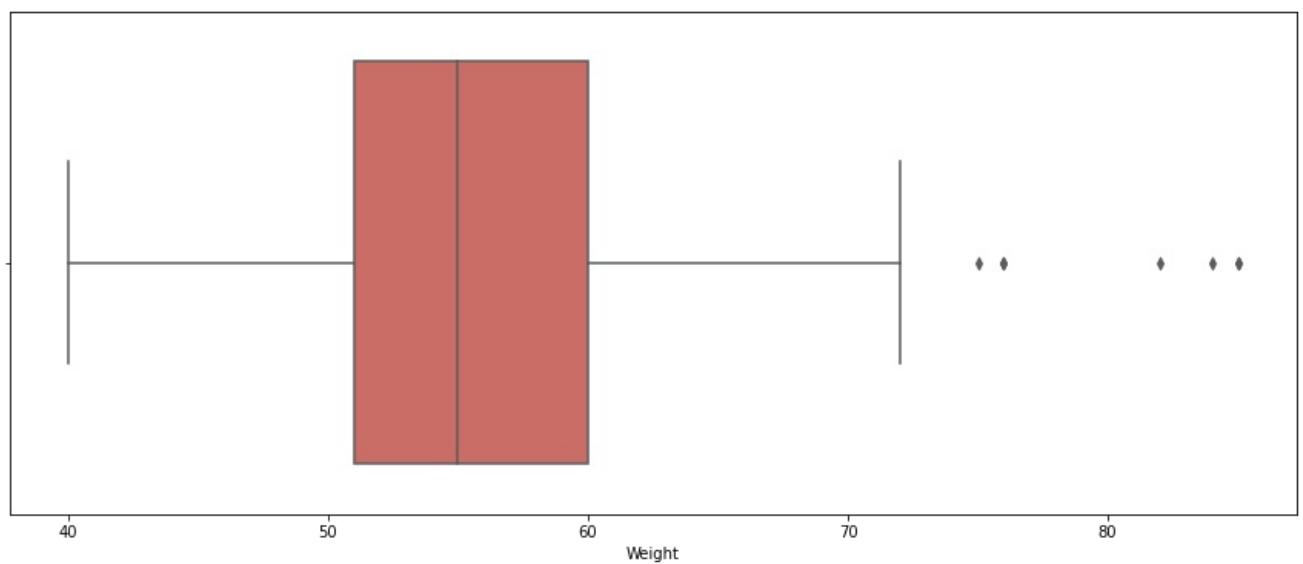
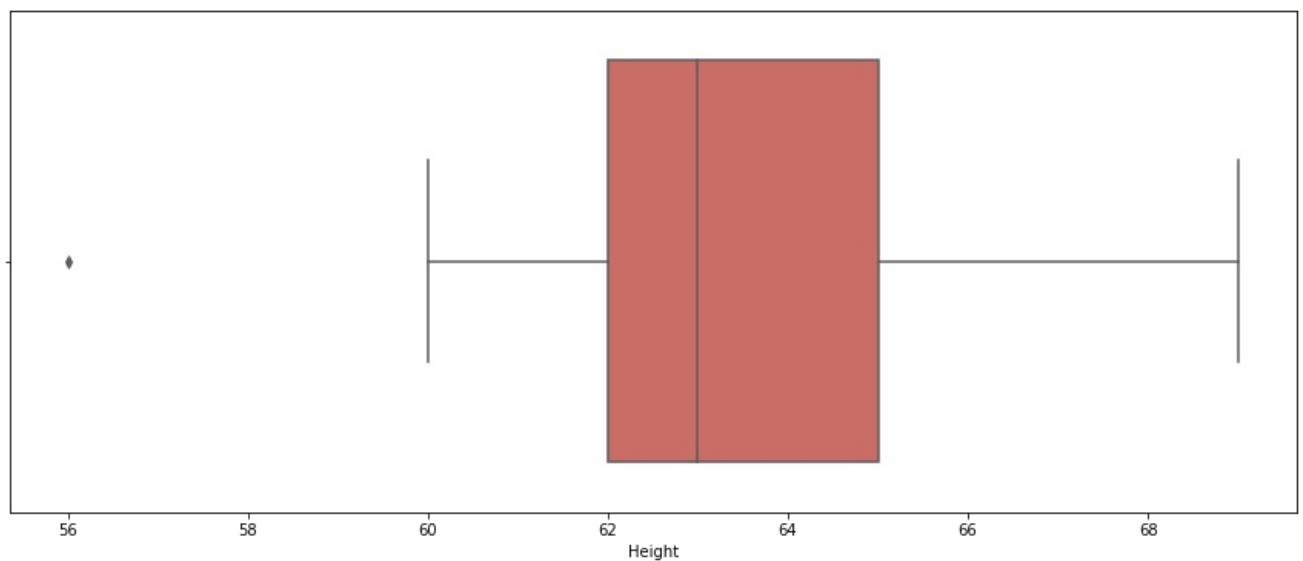
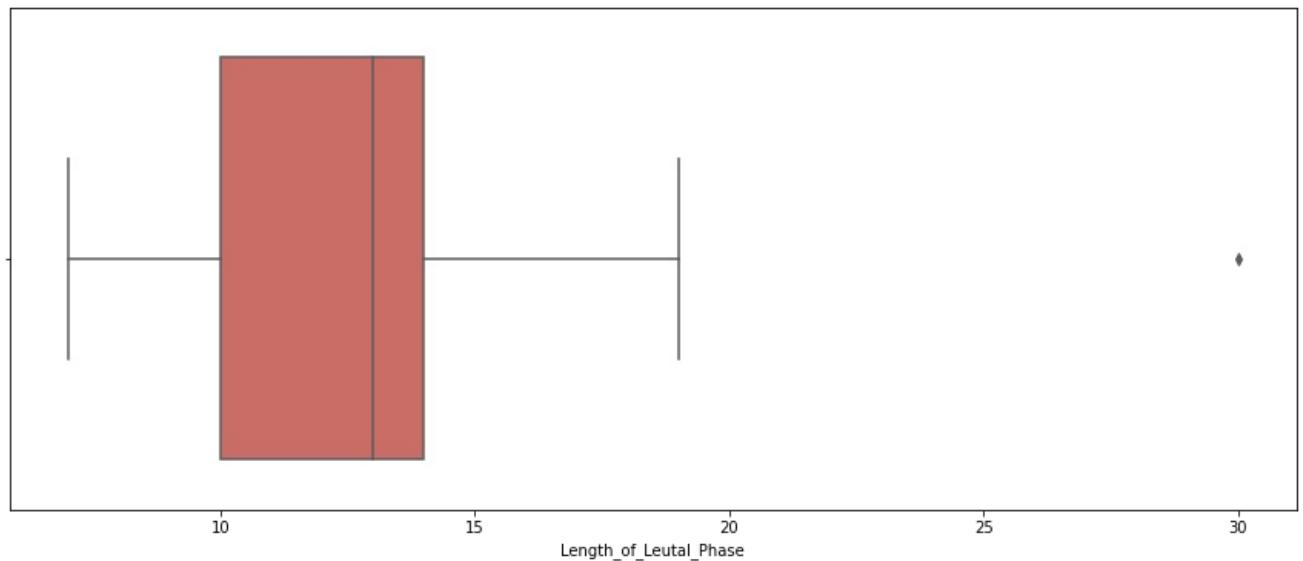


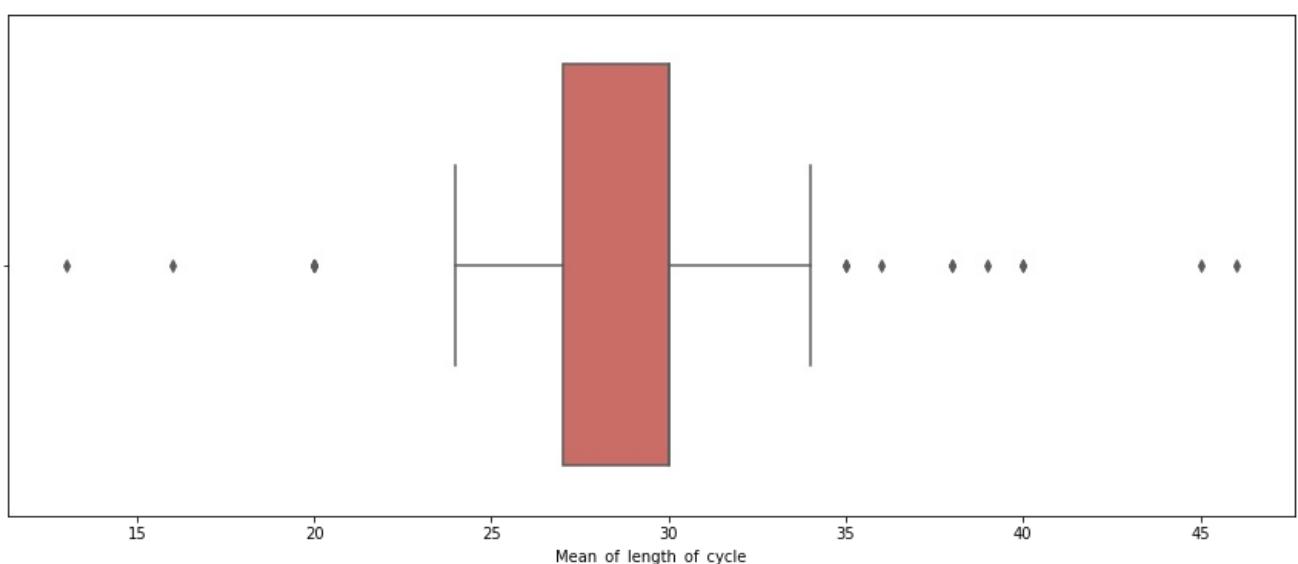
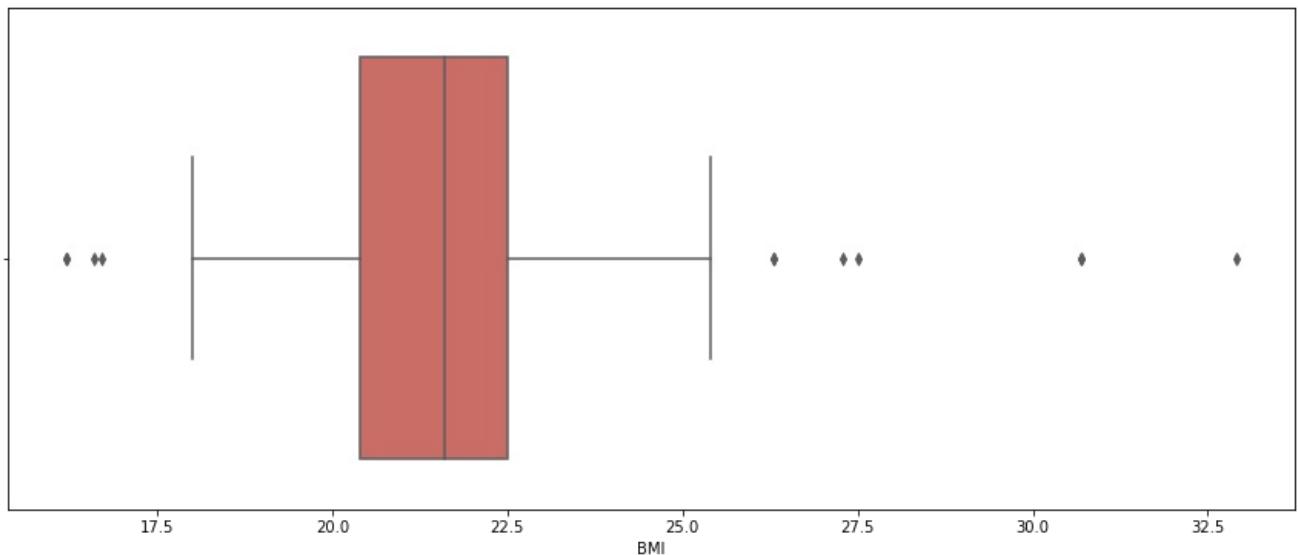




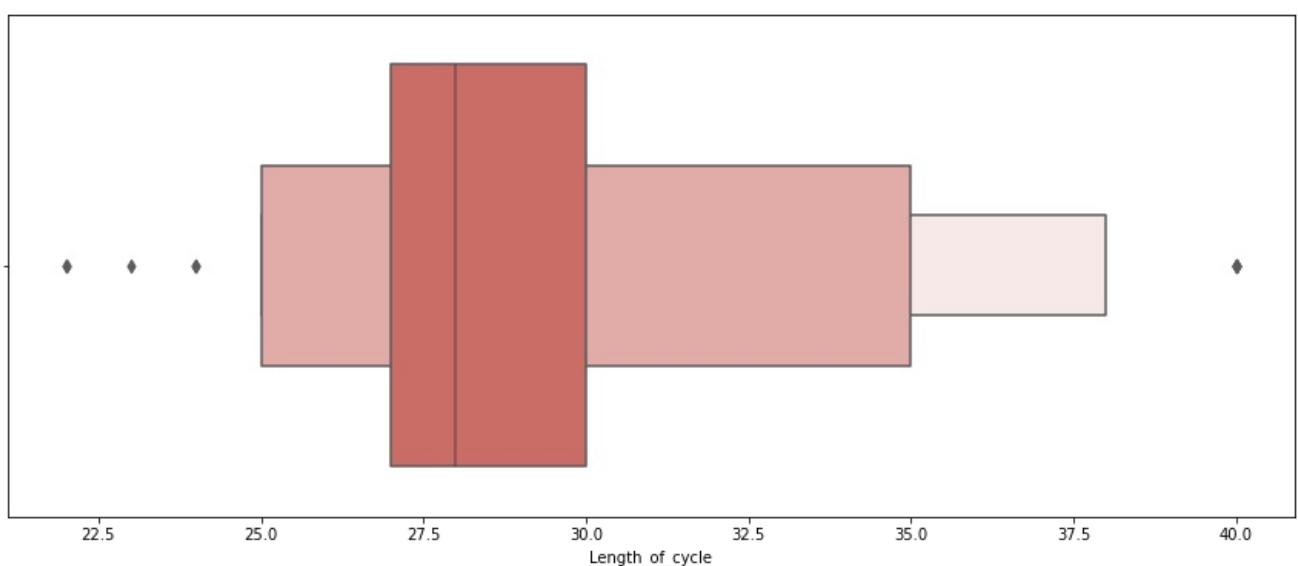
```
In [33]: for i in continuous:  
    plt.figure(figsize=(15,6))  
    sns.boxplot(i, data = df, palette='hls')  
    plt.show()
```

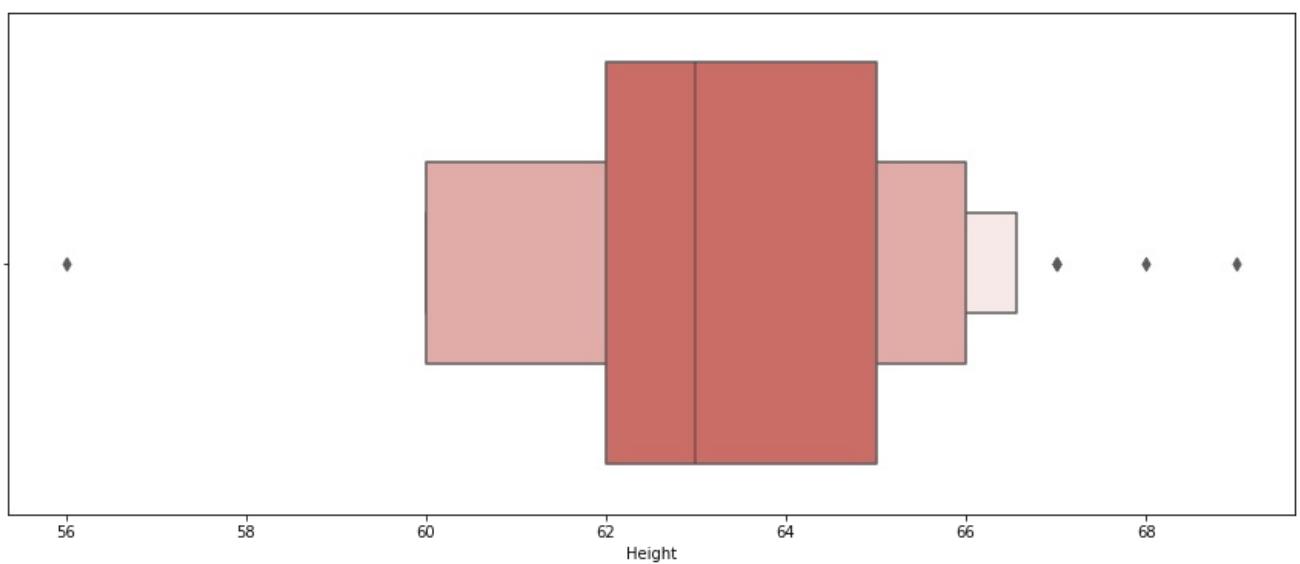
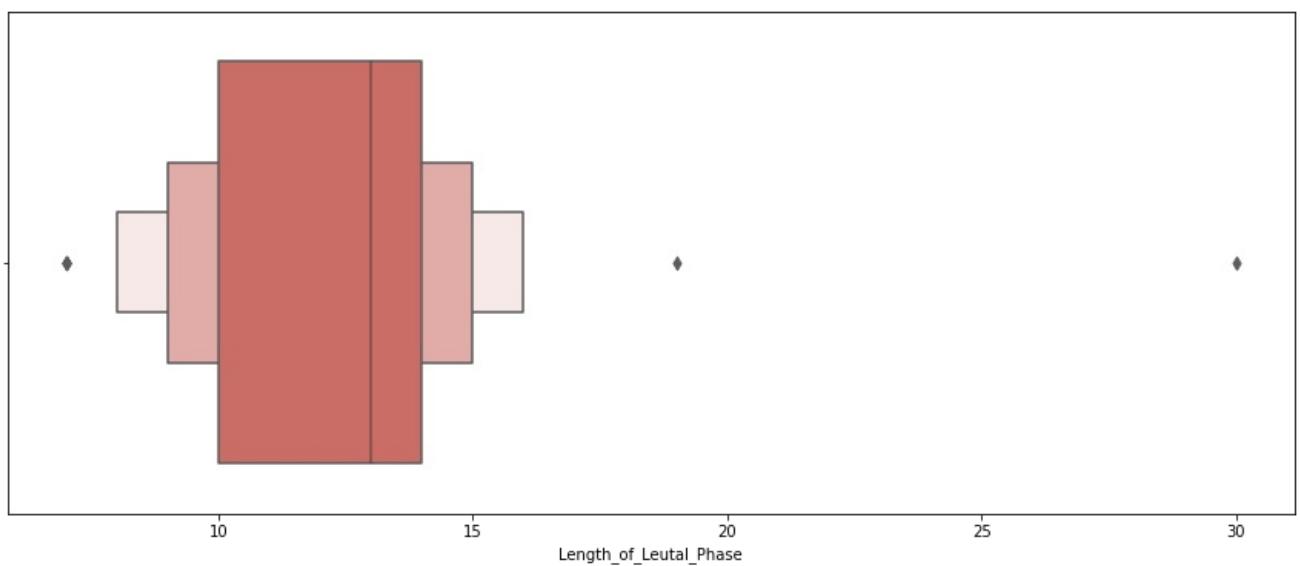
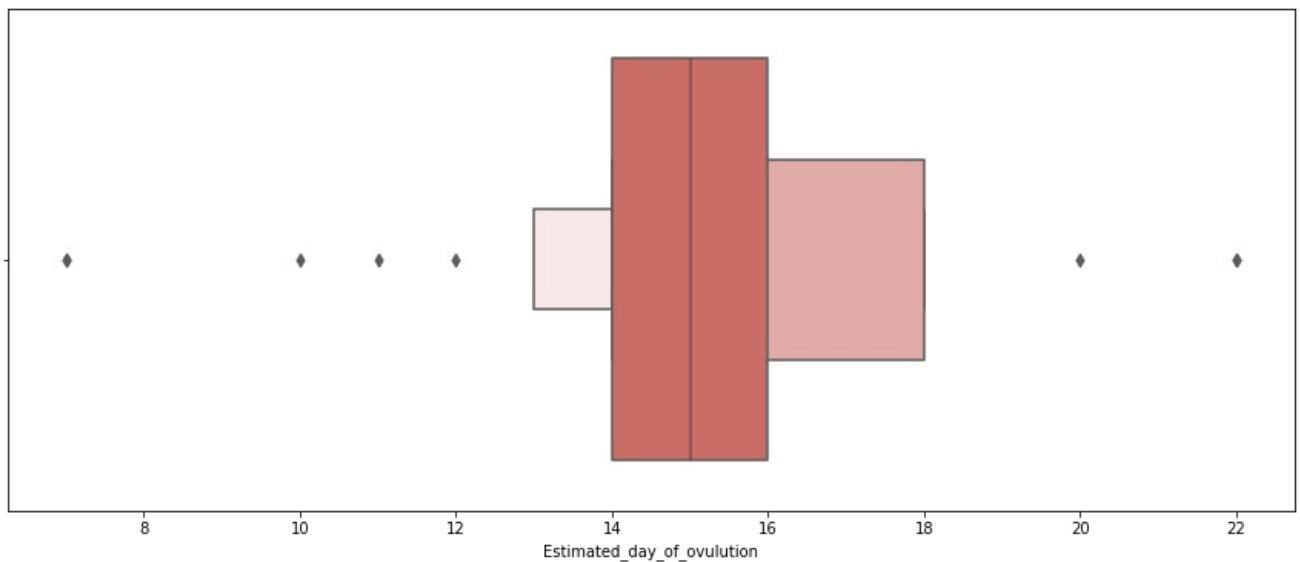


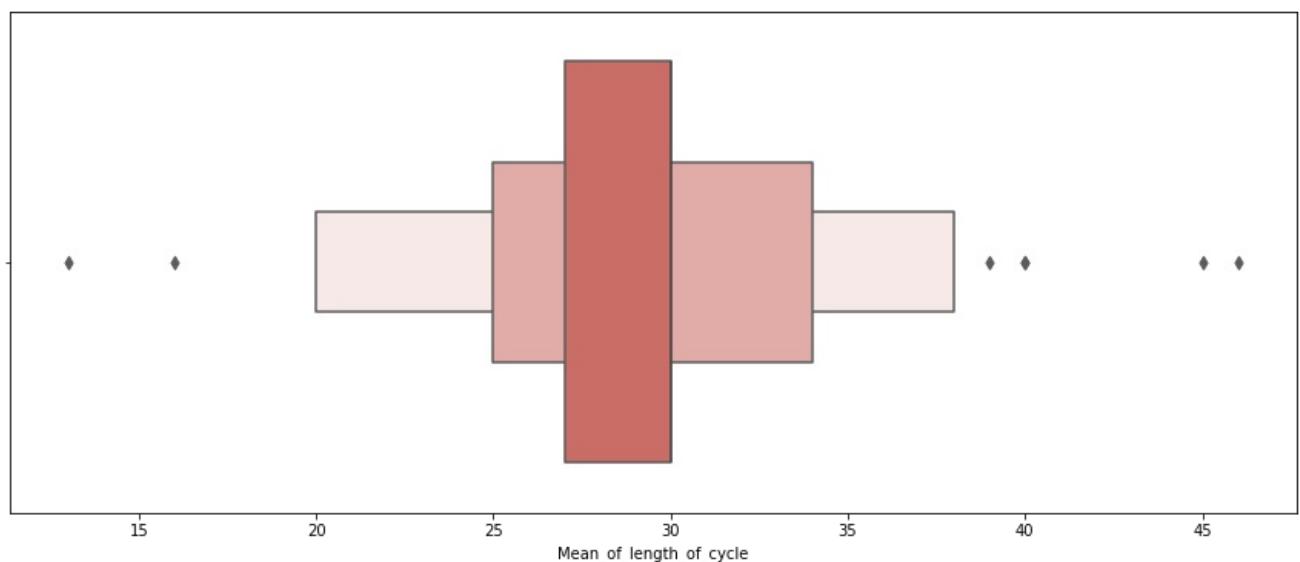
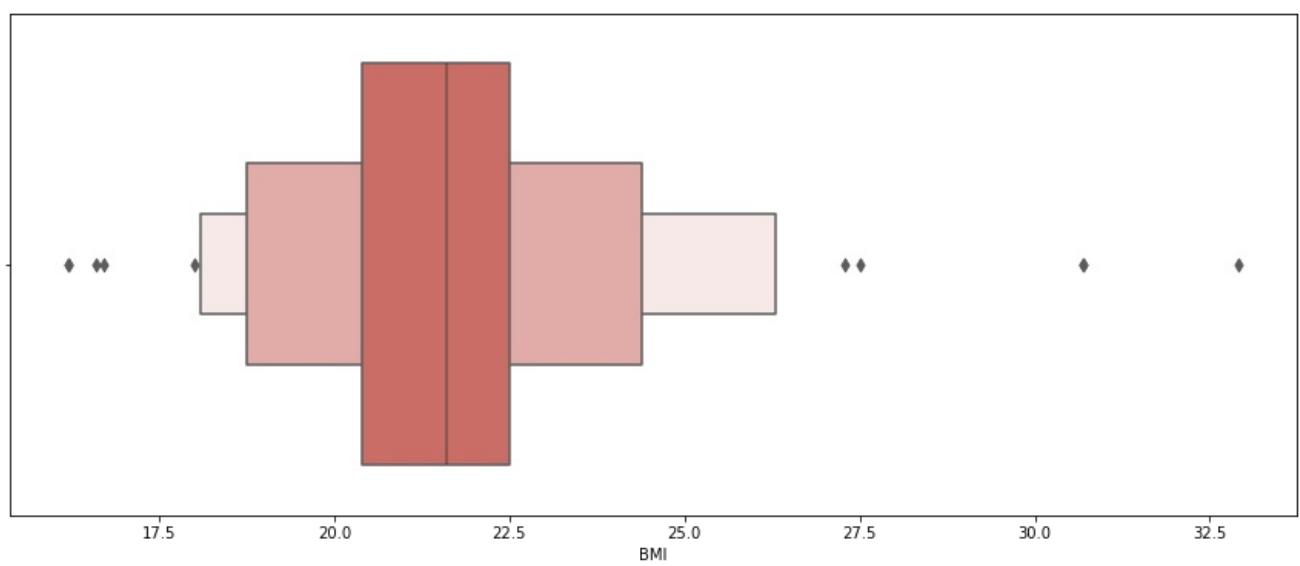
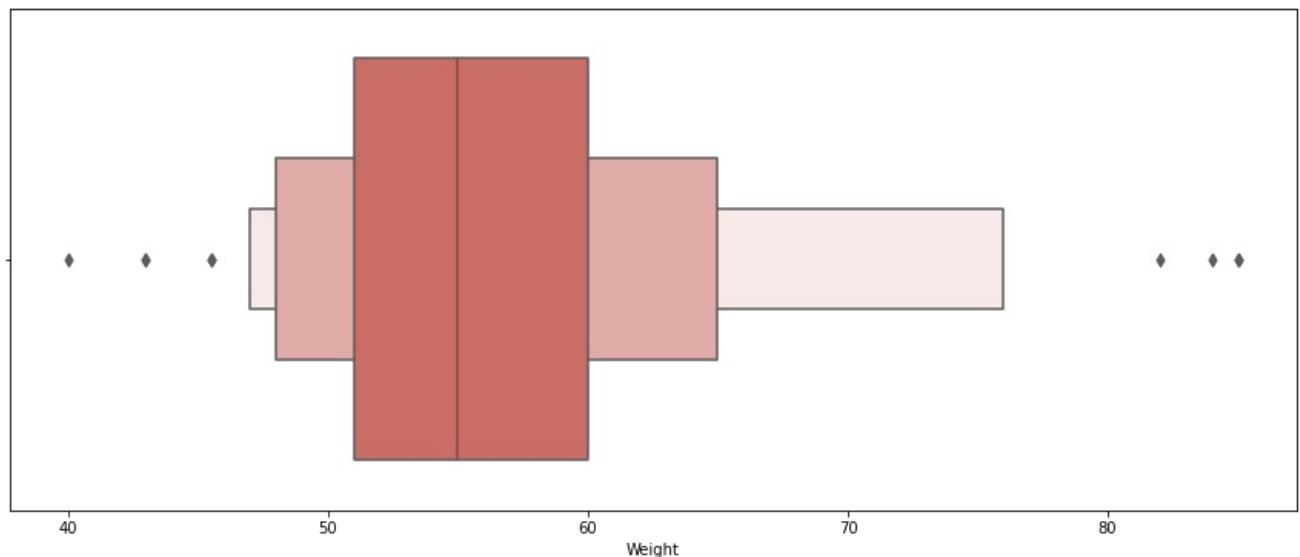




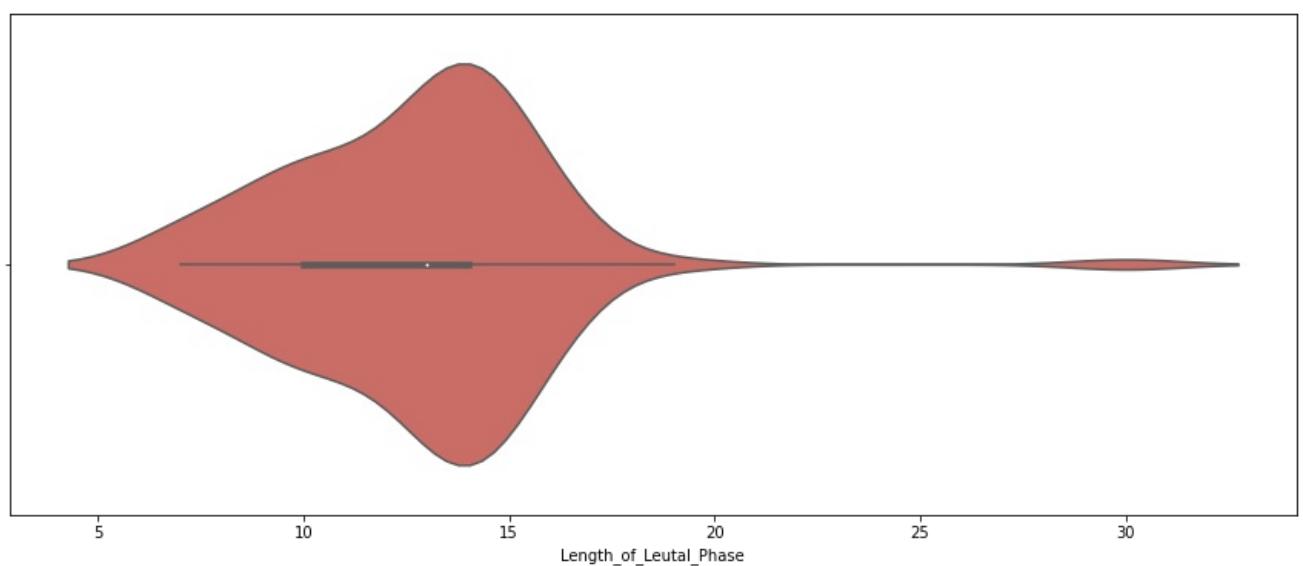
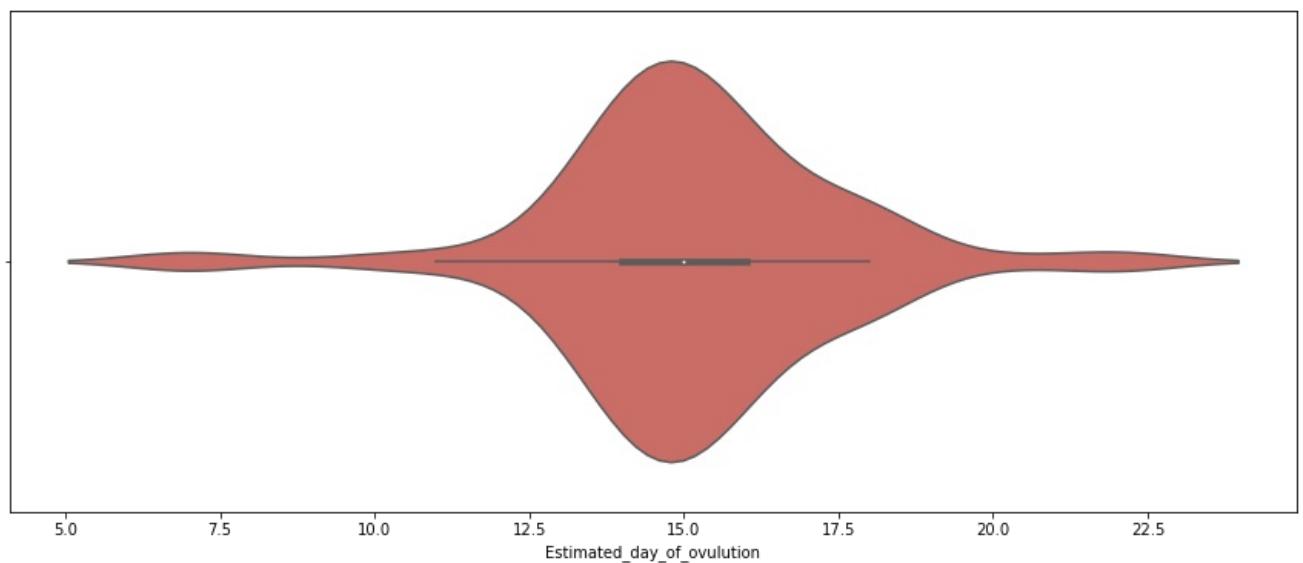
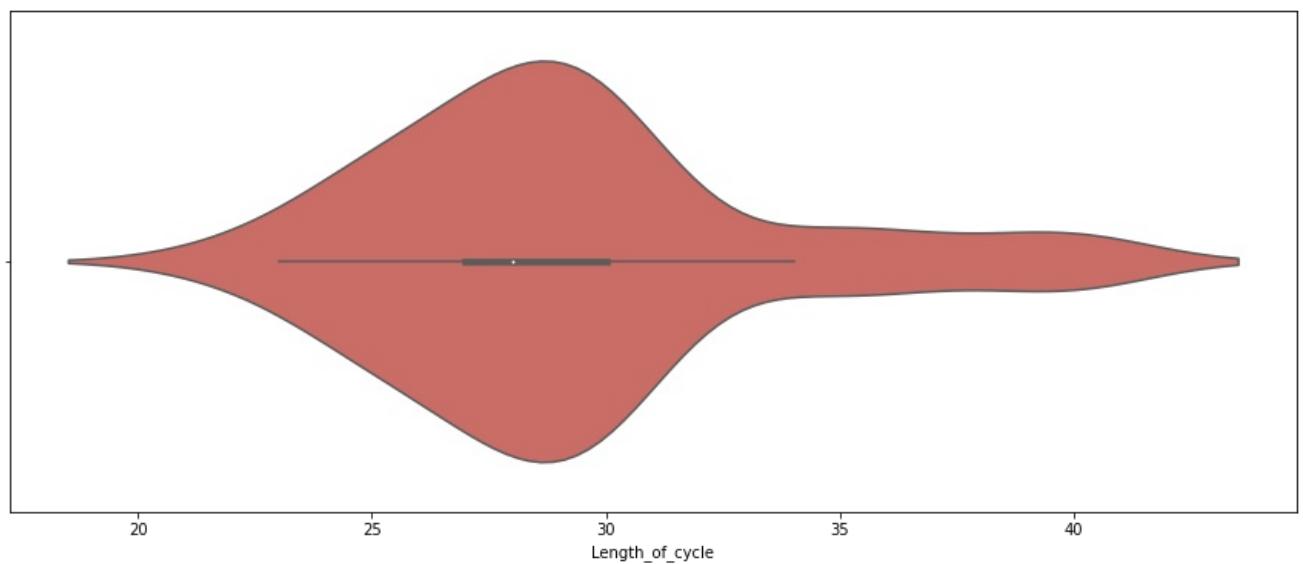
```
In [34]: for i in continuous:  
    plt.figure(figsize=(15,6))  
    sns.boxenplot(i, data = df, palette='hls')  
    plt.show()
```

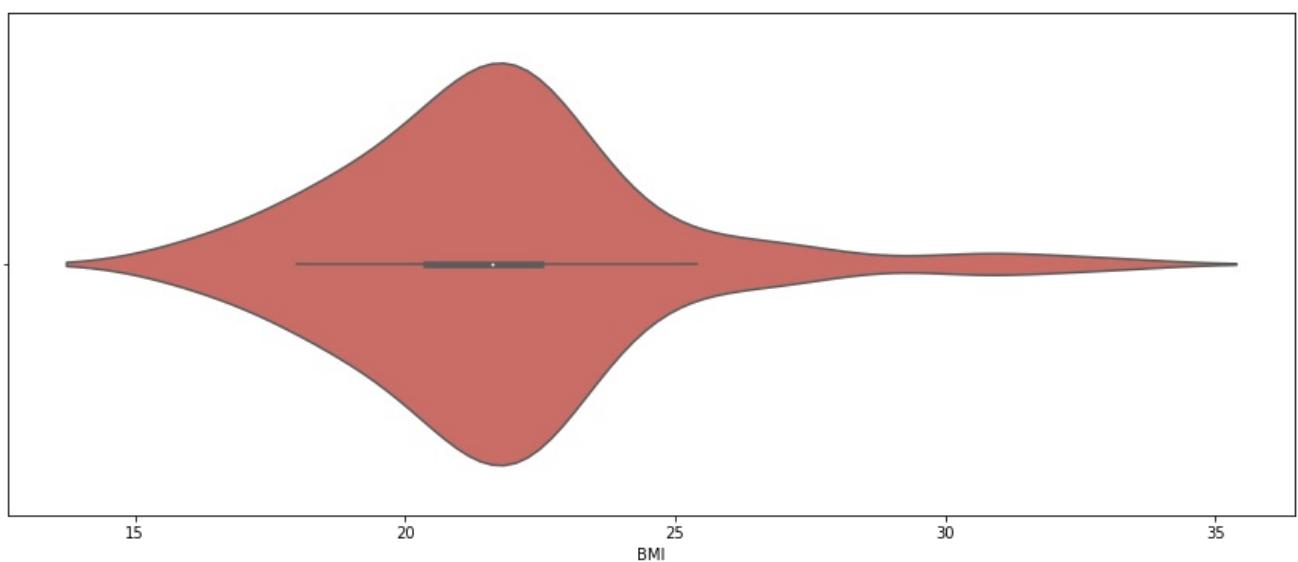
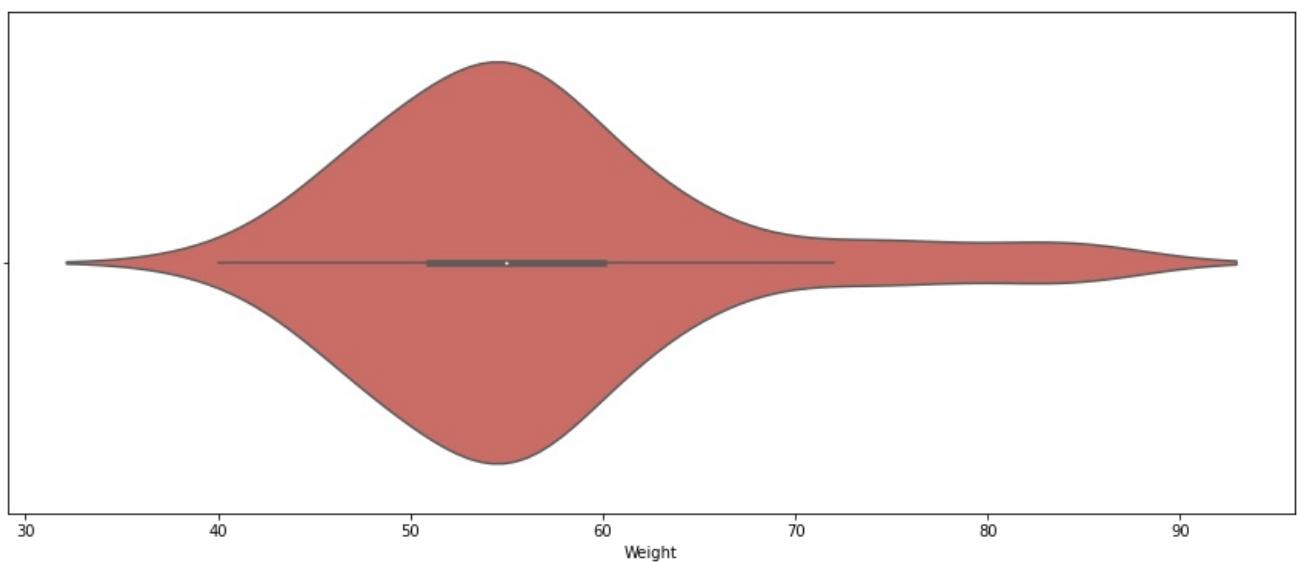
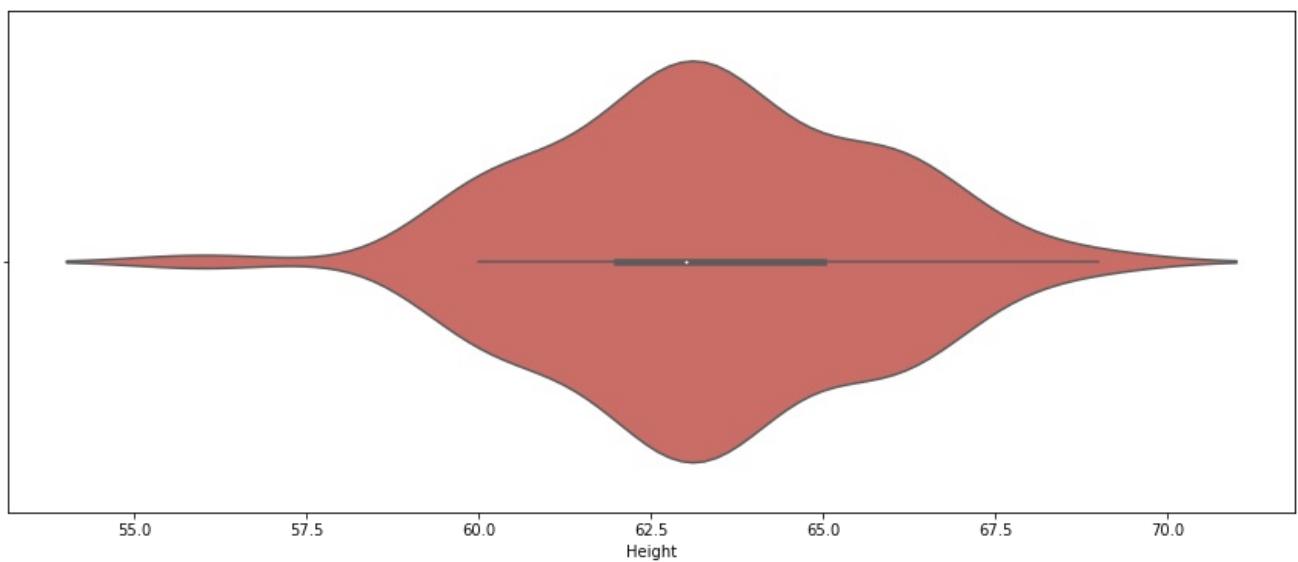


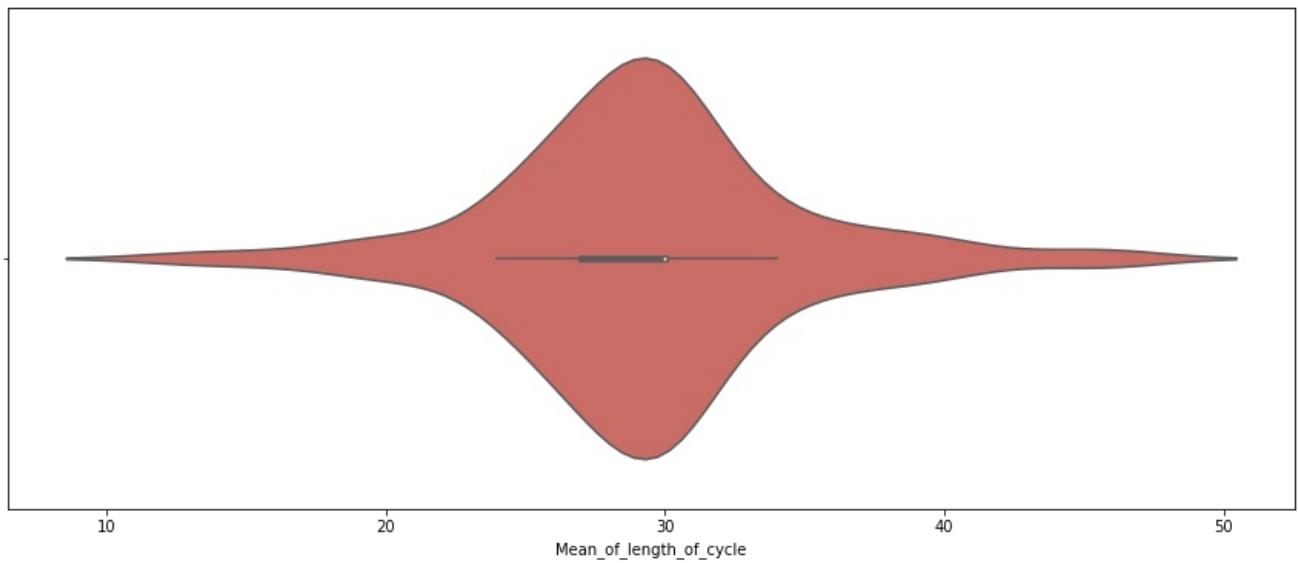




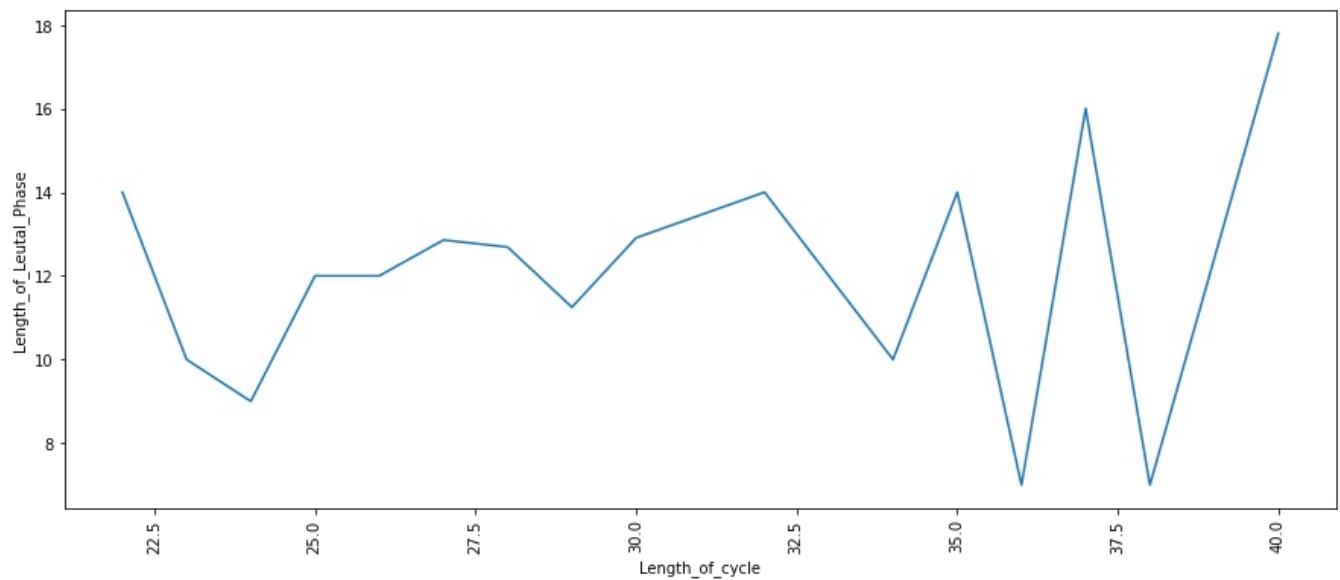
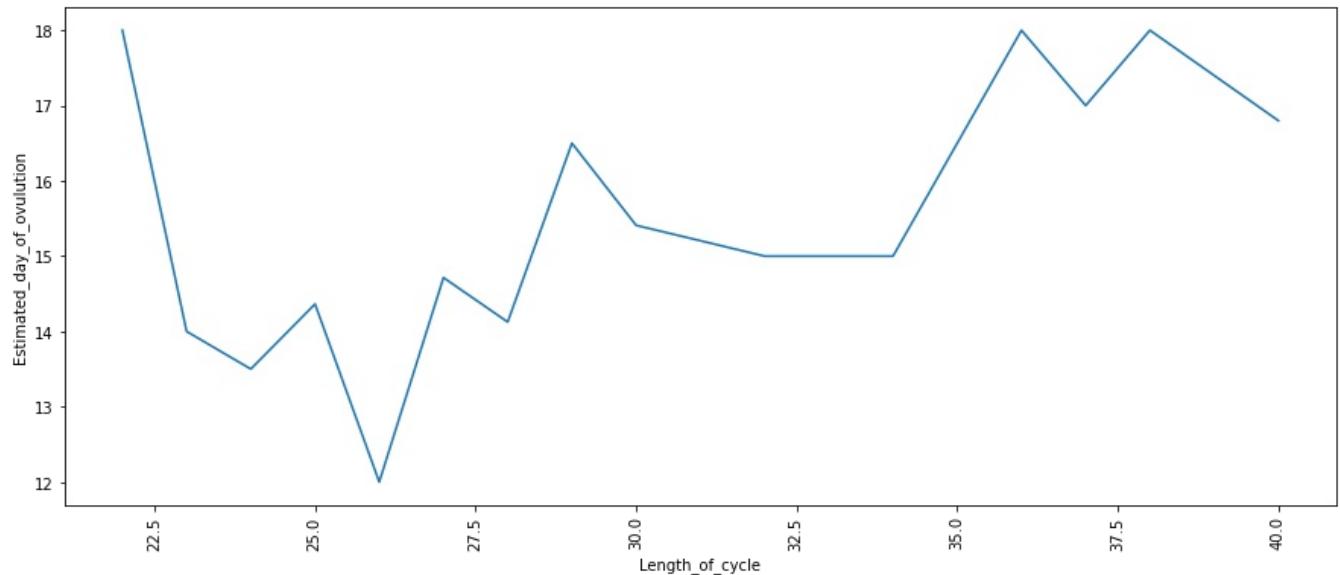
```
In [35]: for i in continuous:  
    plt.figure(figsize=(15,6))  
    sns.violinplot(i, data = df, palette='hls')  
    plt.show()
```

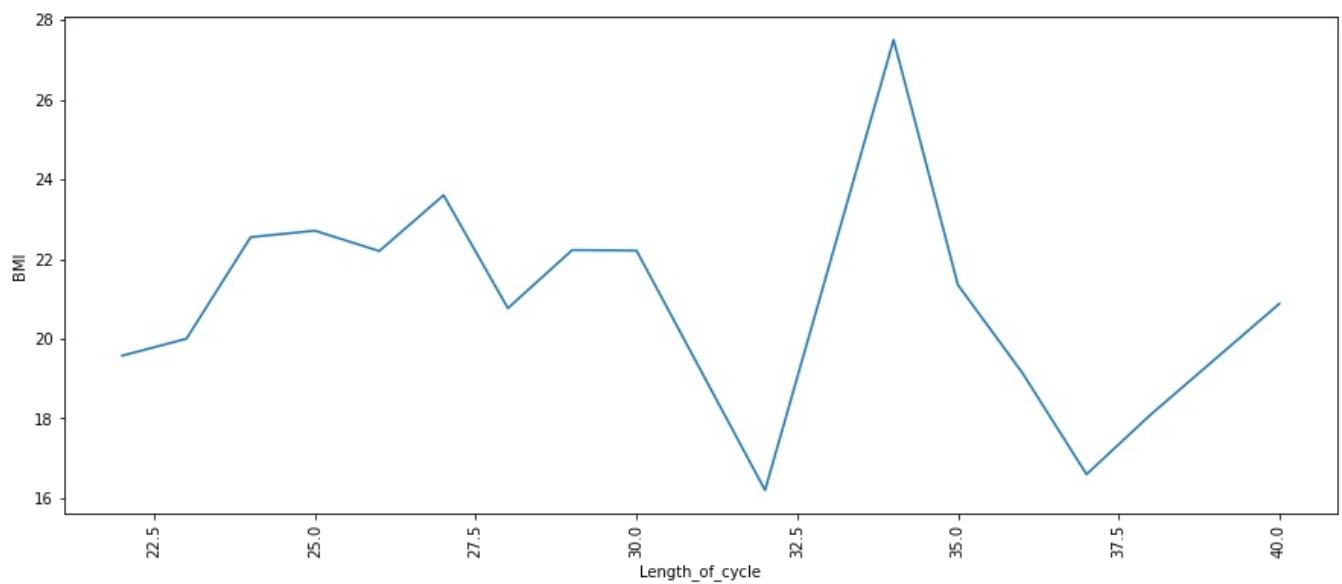
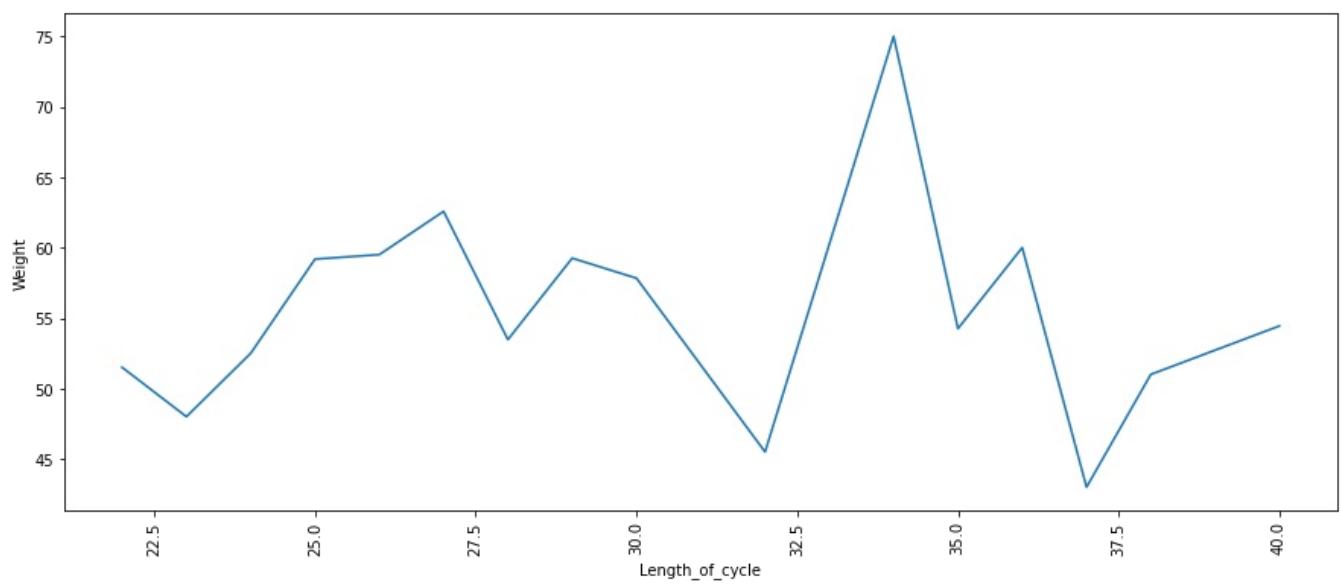
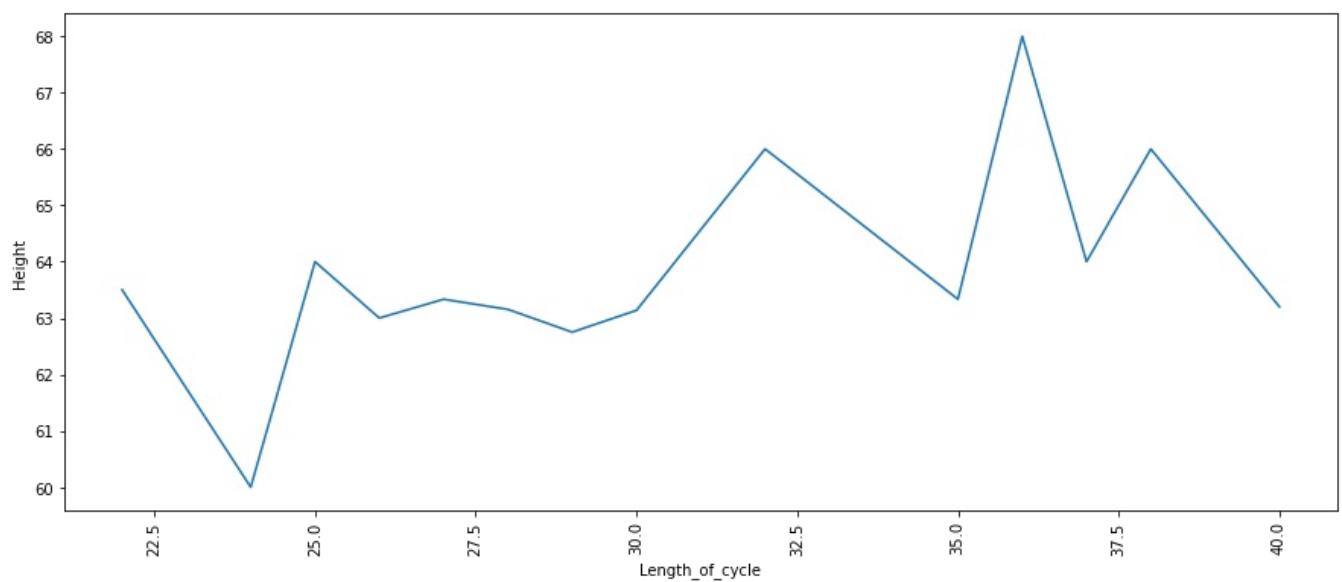


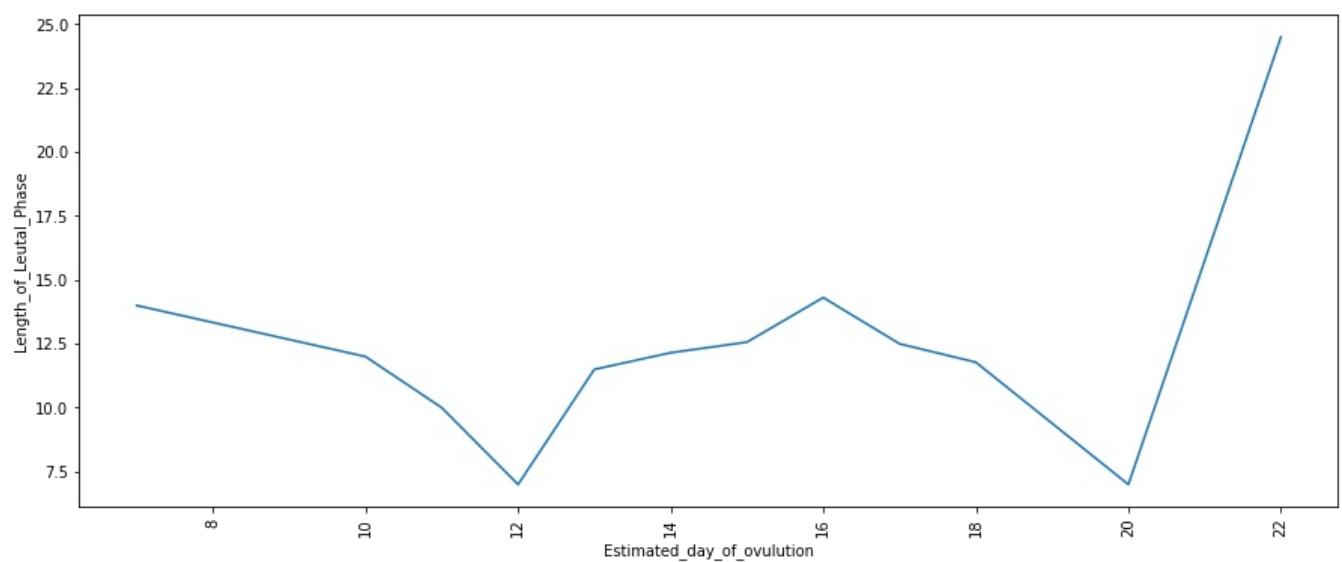
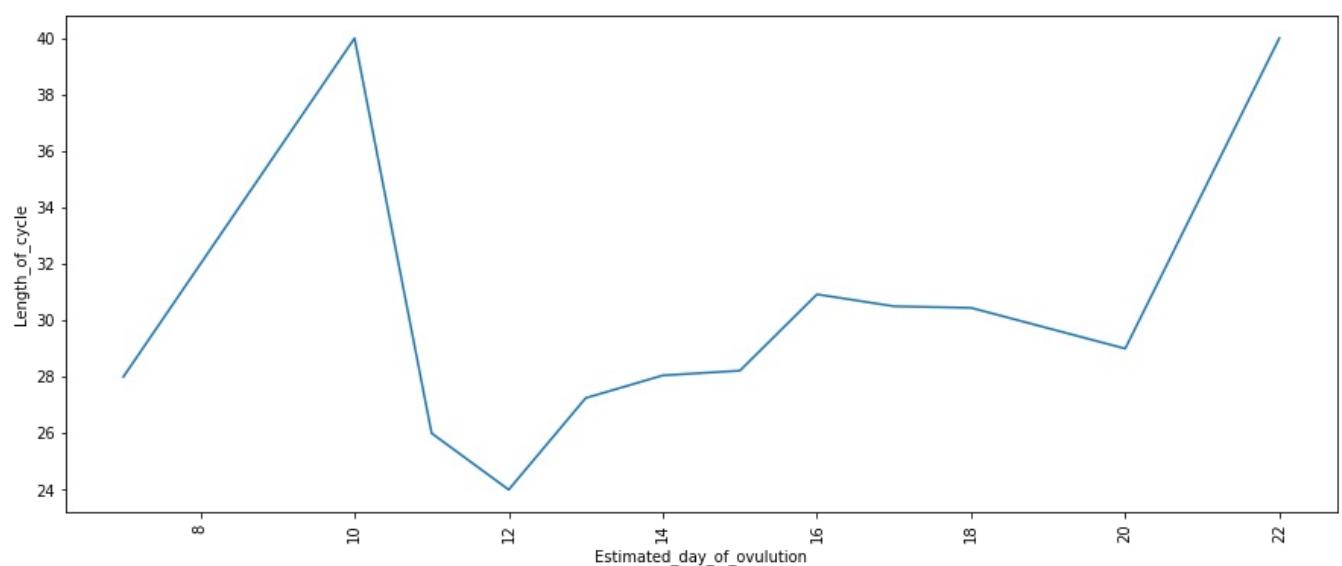
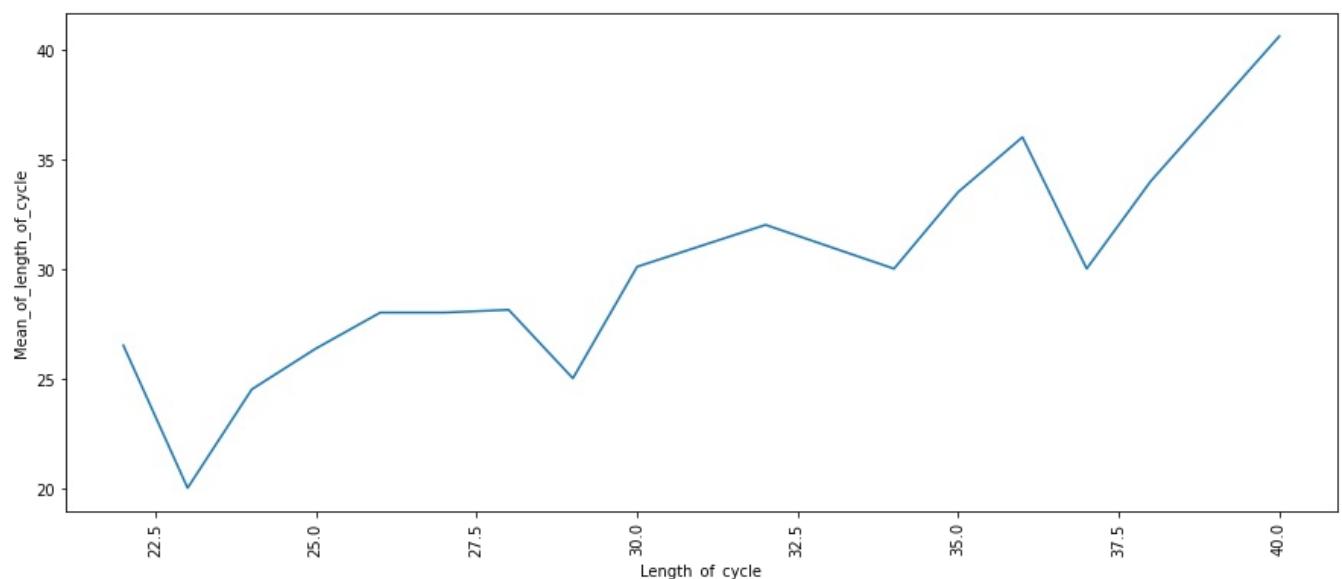


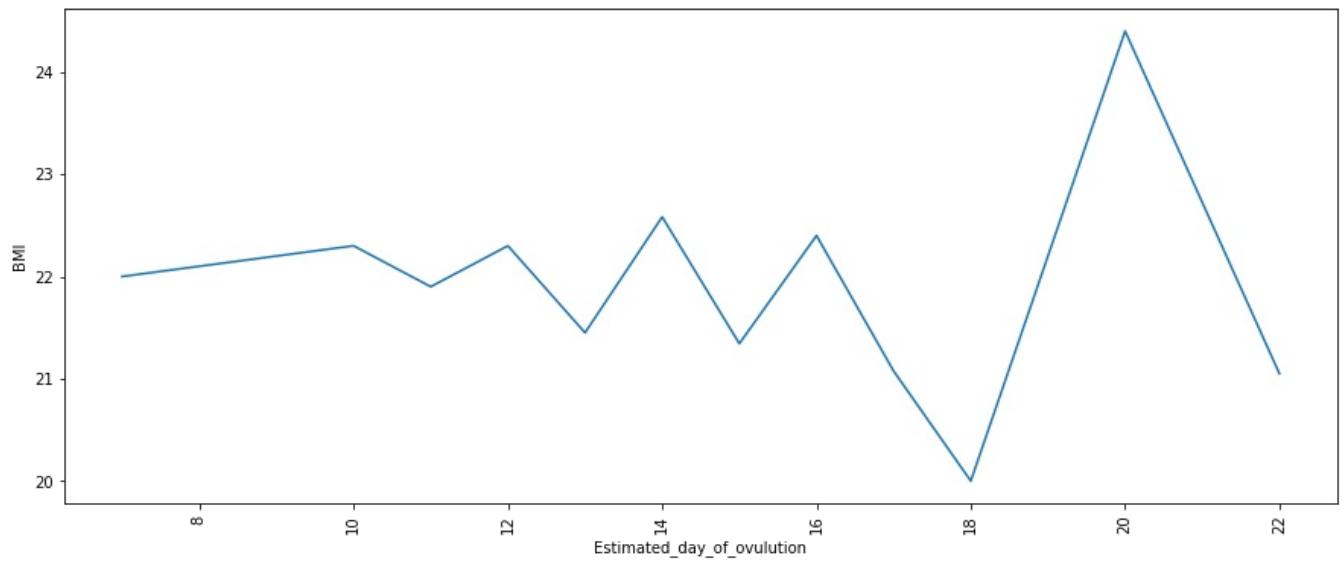
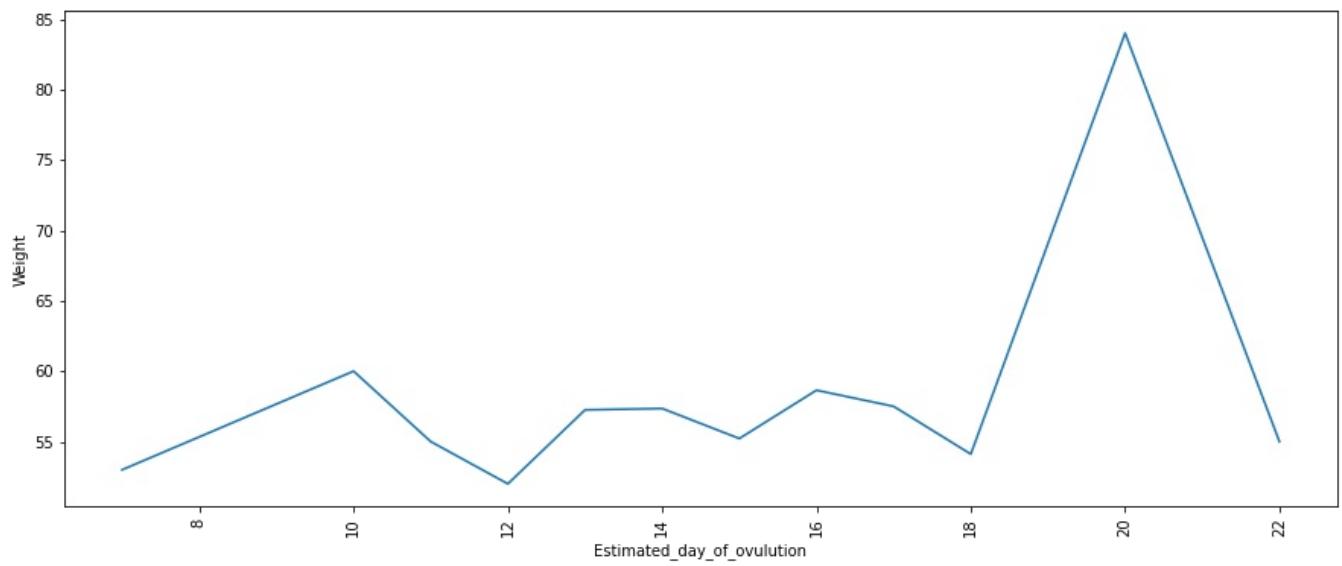
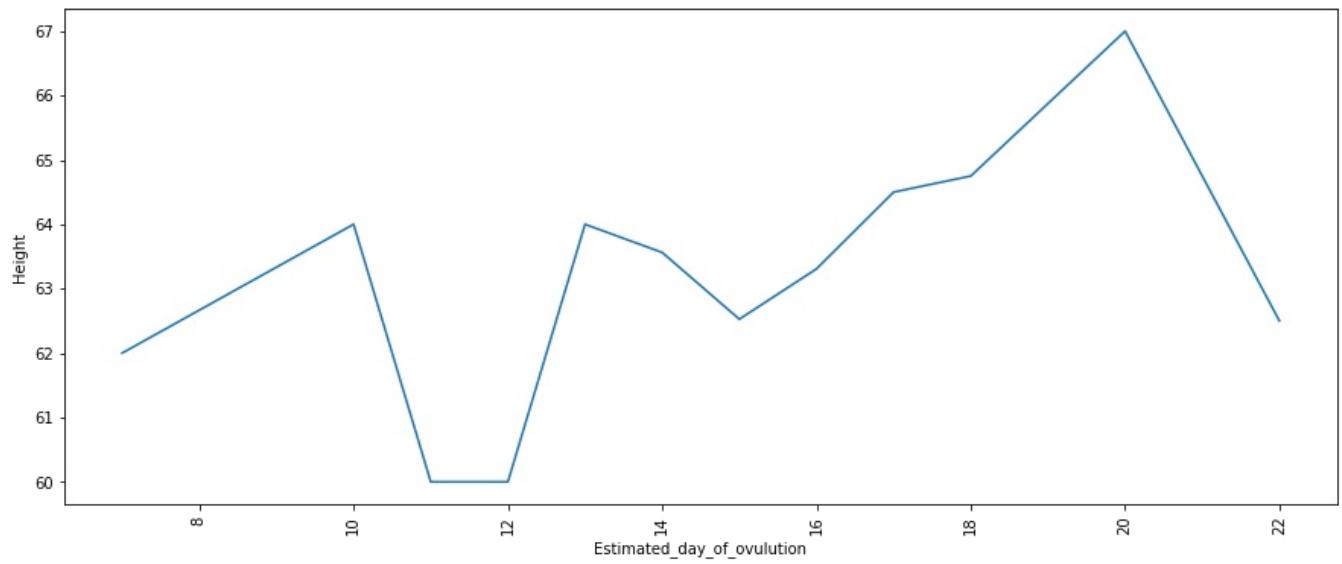


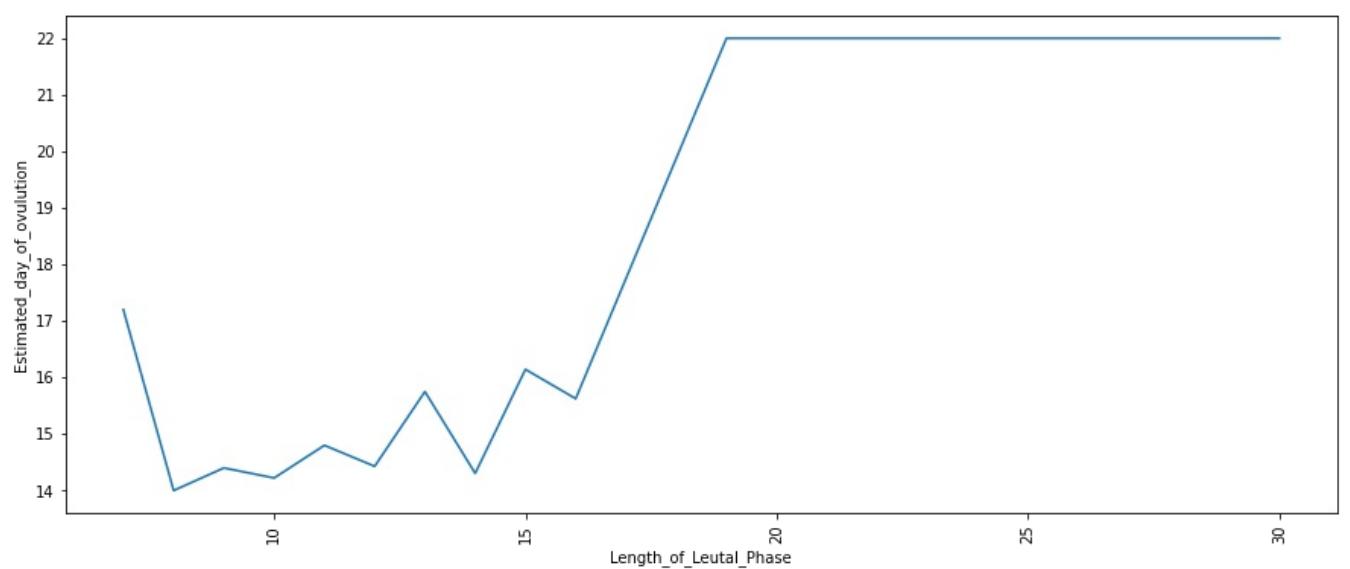
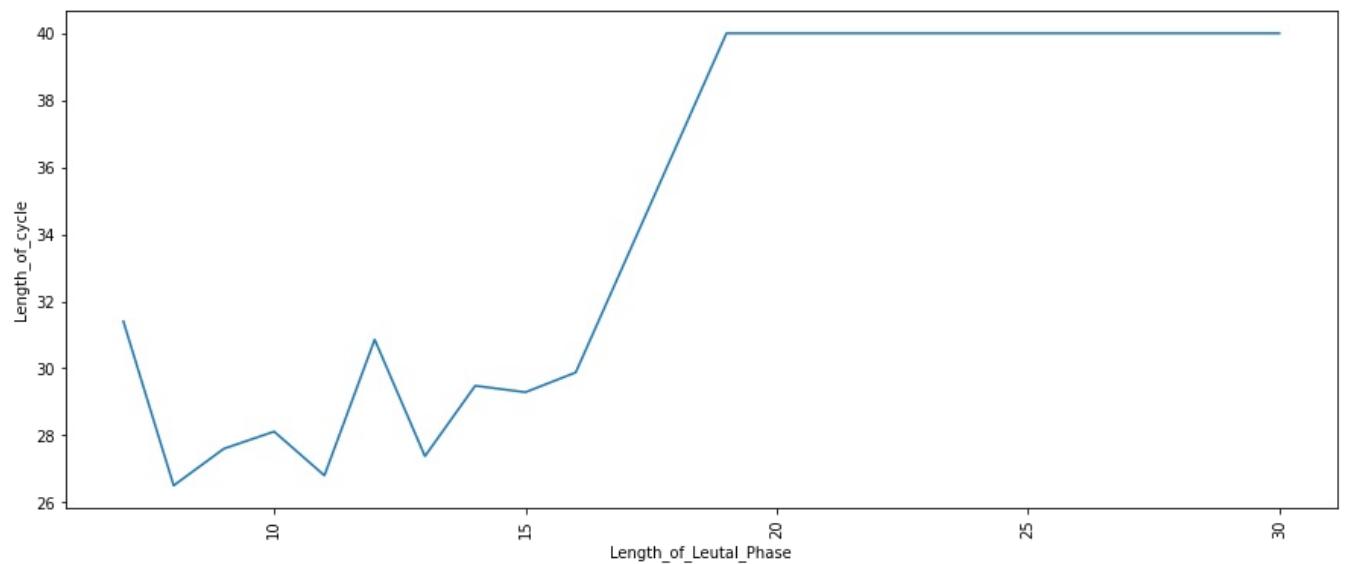
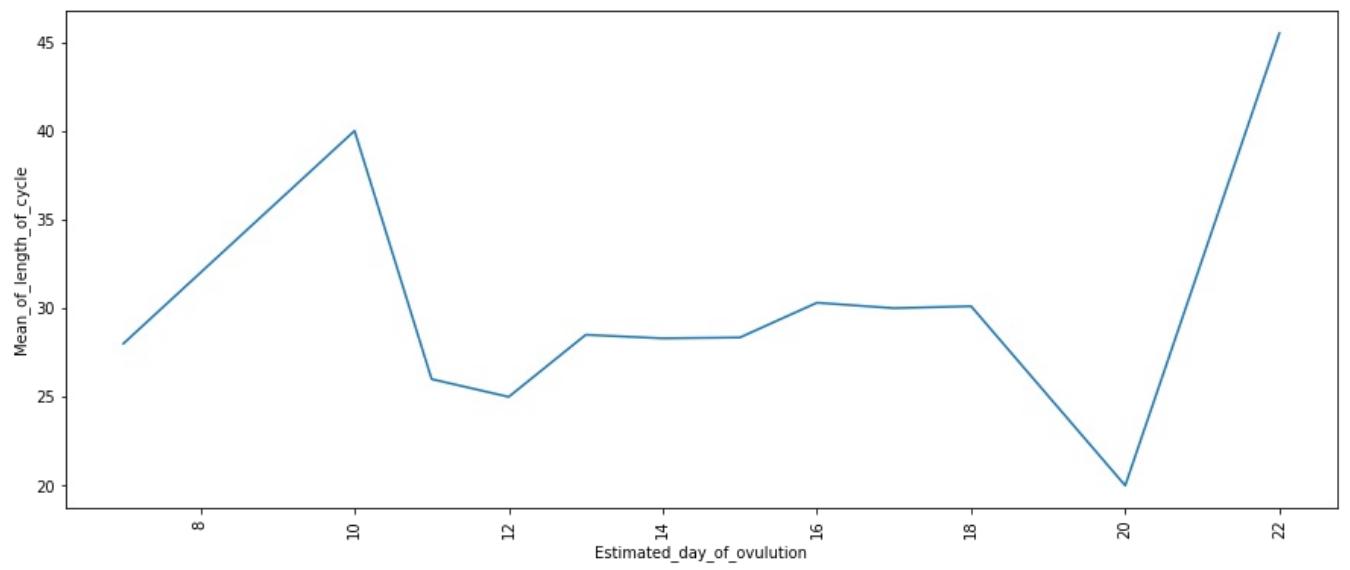
```
In [36]: for i in continuous:  
    for j in continuous:  
        if i != j:  
            plt.figure(figsize=(15,6))  
            sns.lineplot(x = i, y = j, data = df, ci = None, palette='hls')  
            plt.xticks(rotation = 90)  
            plt.show()
```

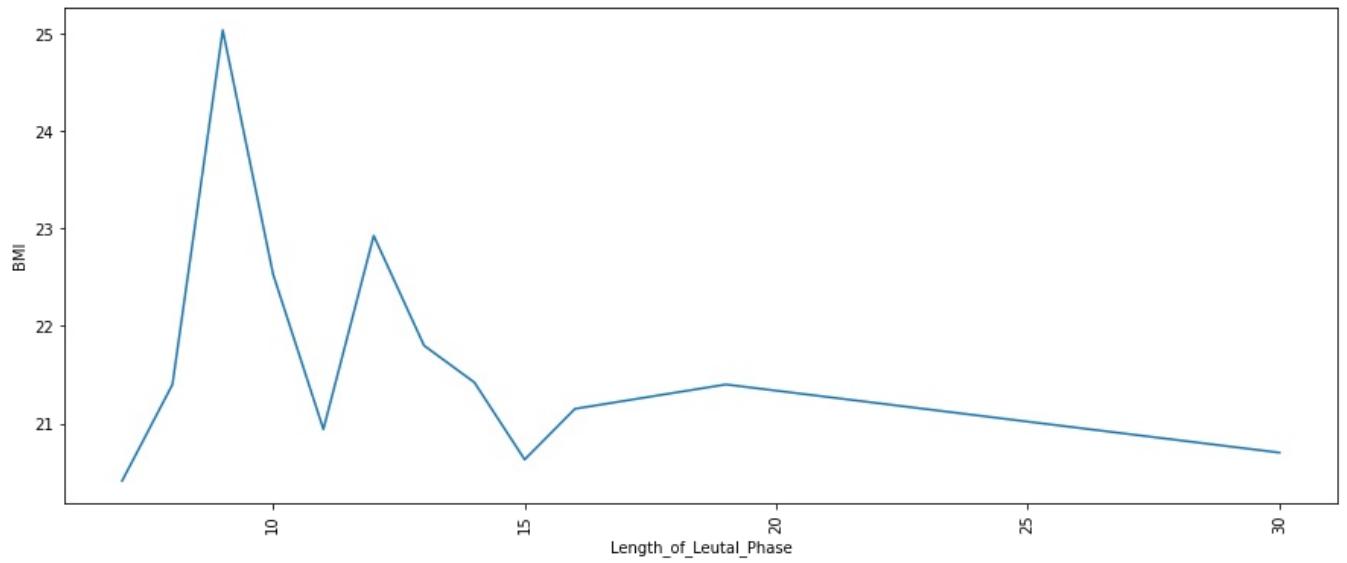
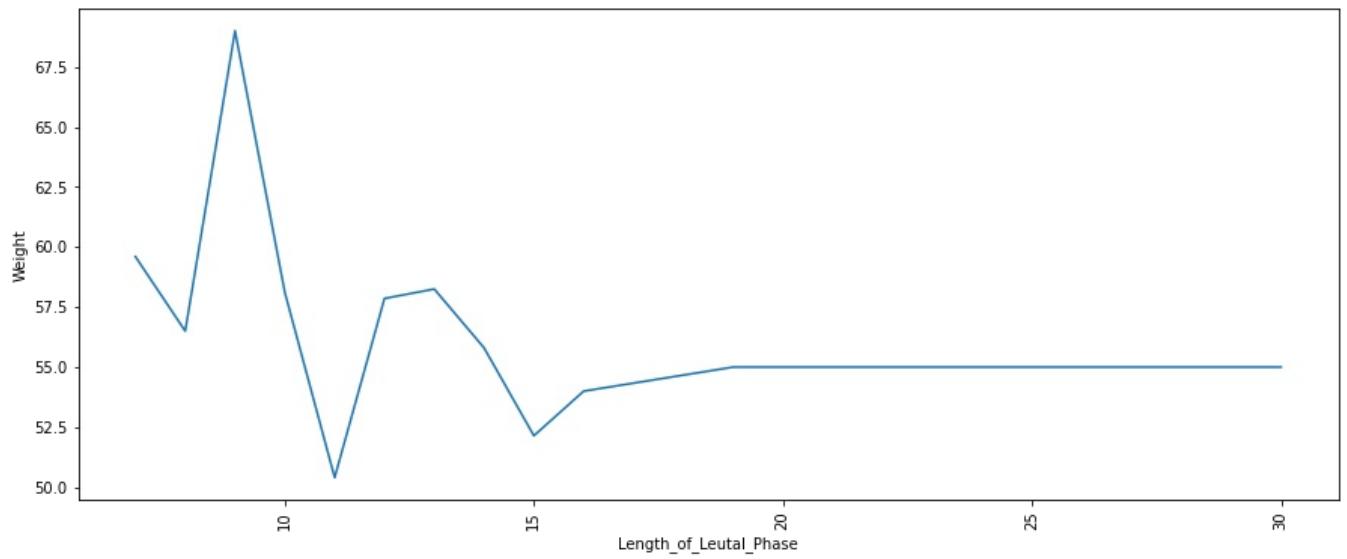
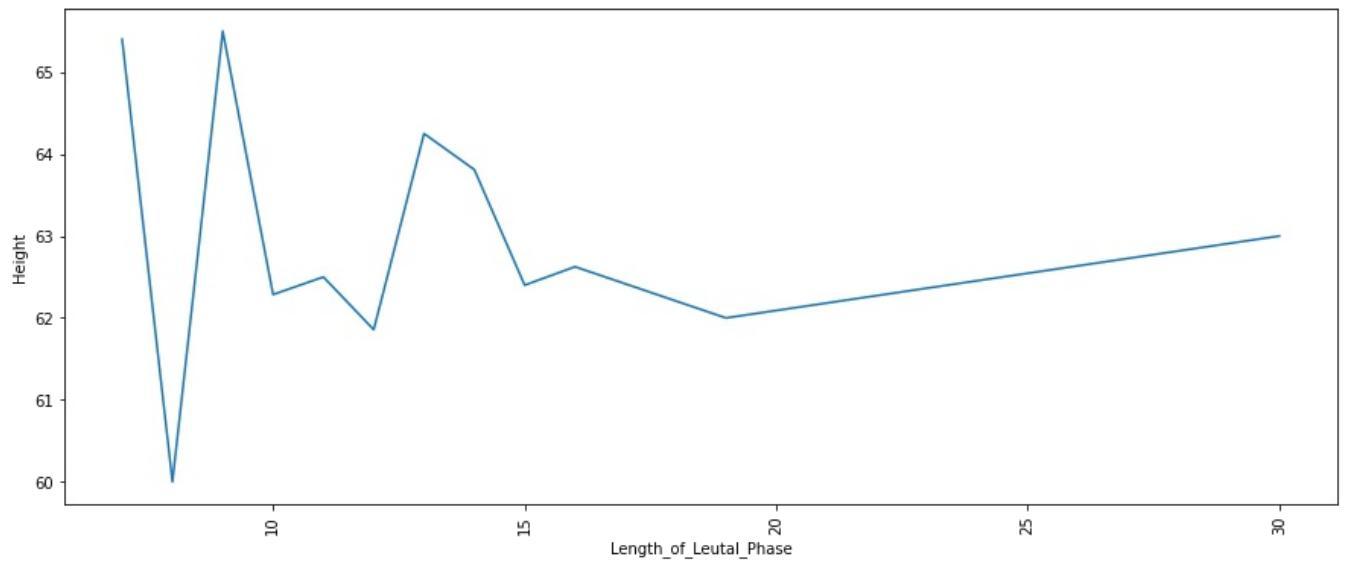


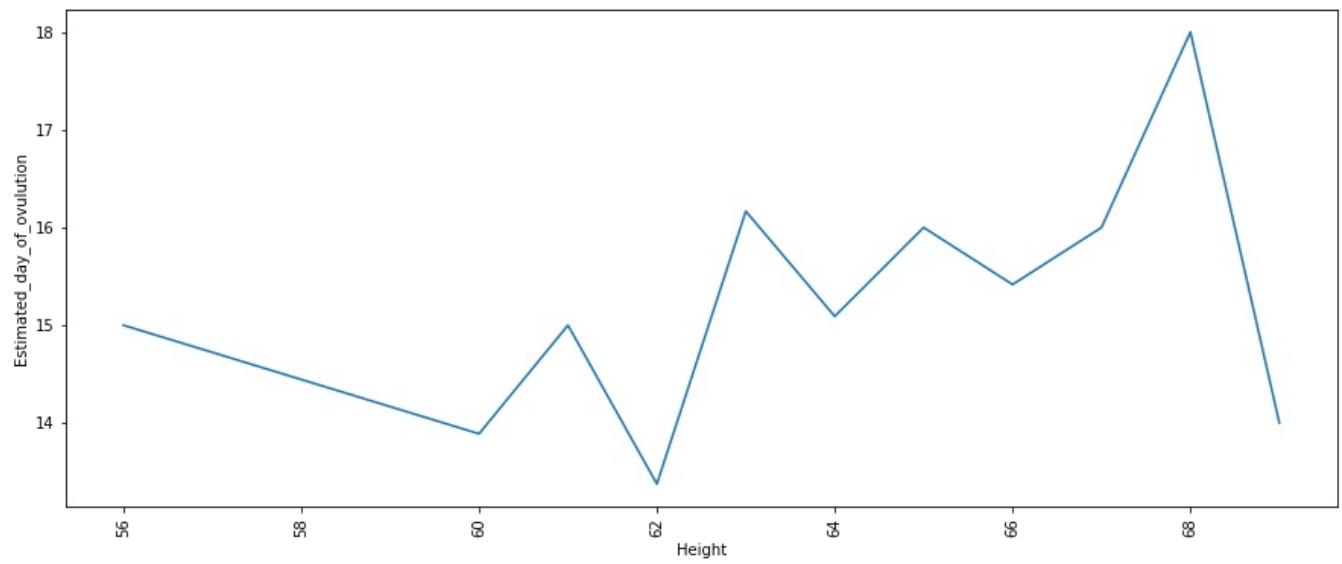
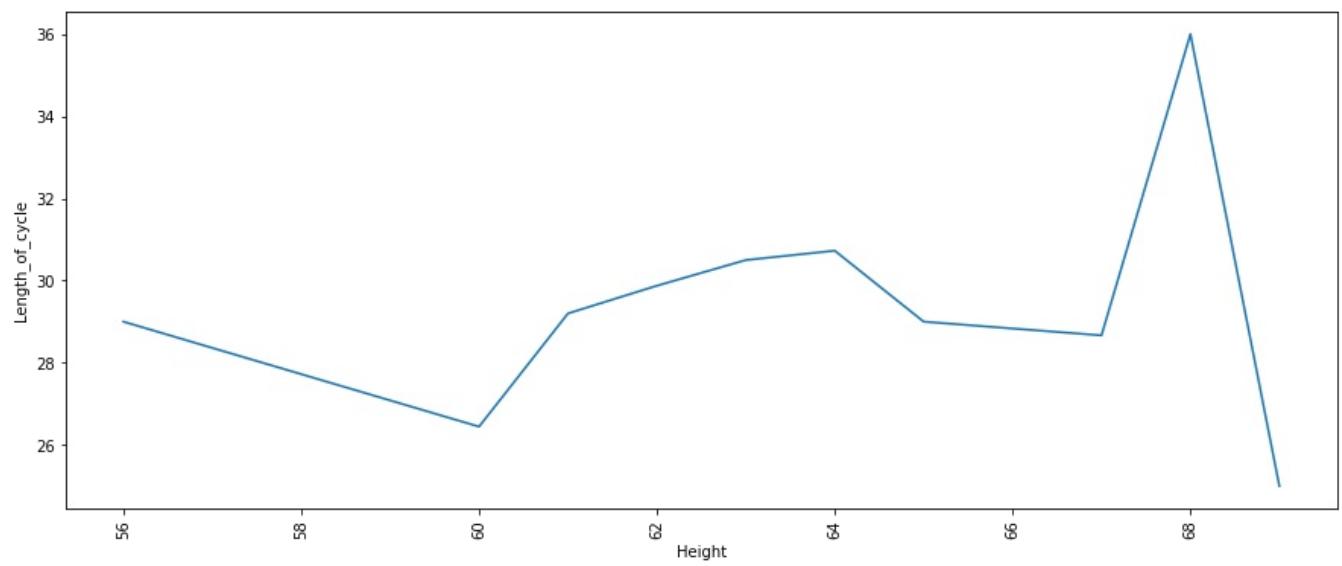
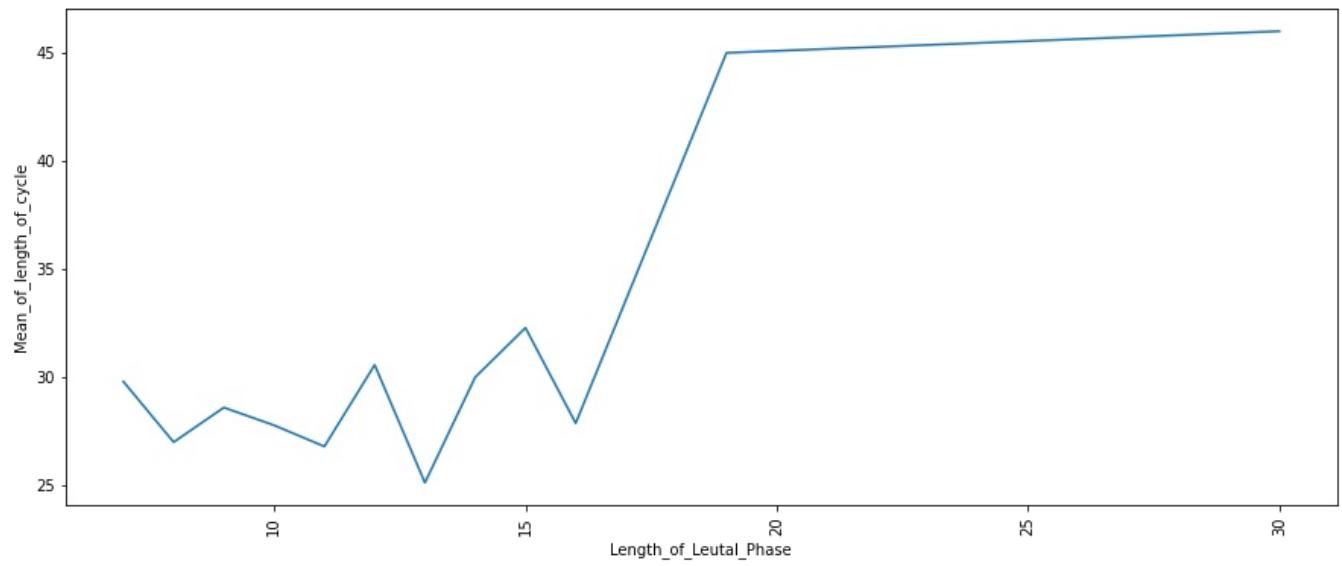


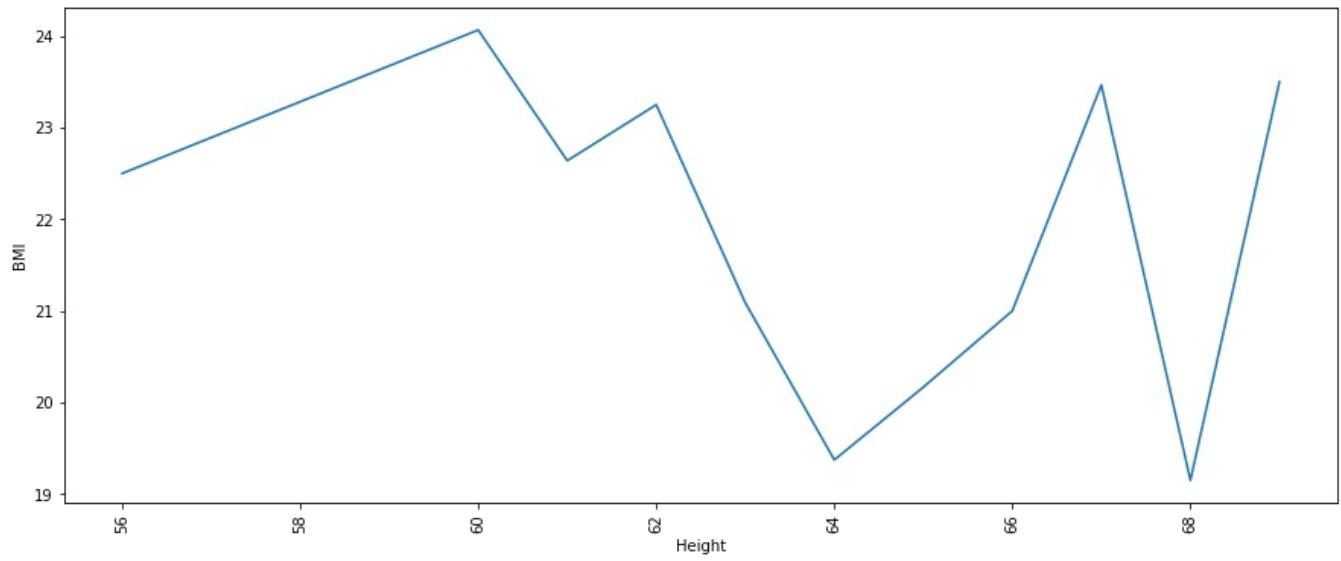
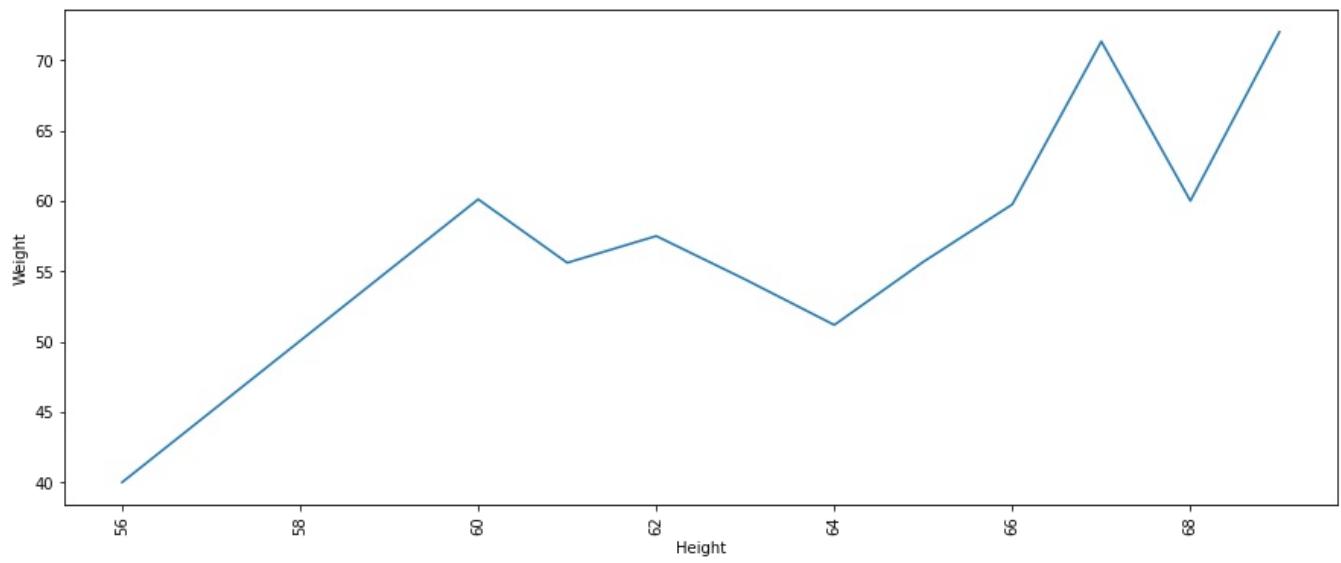
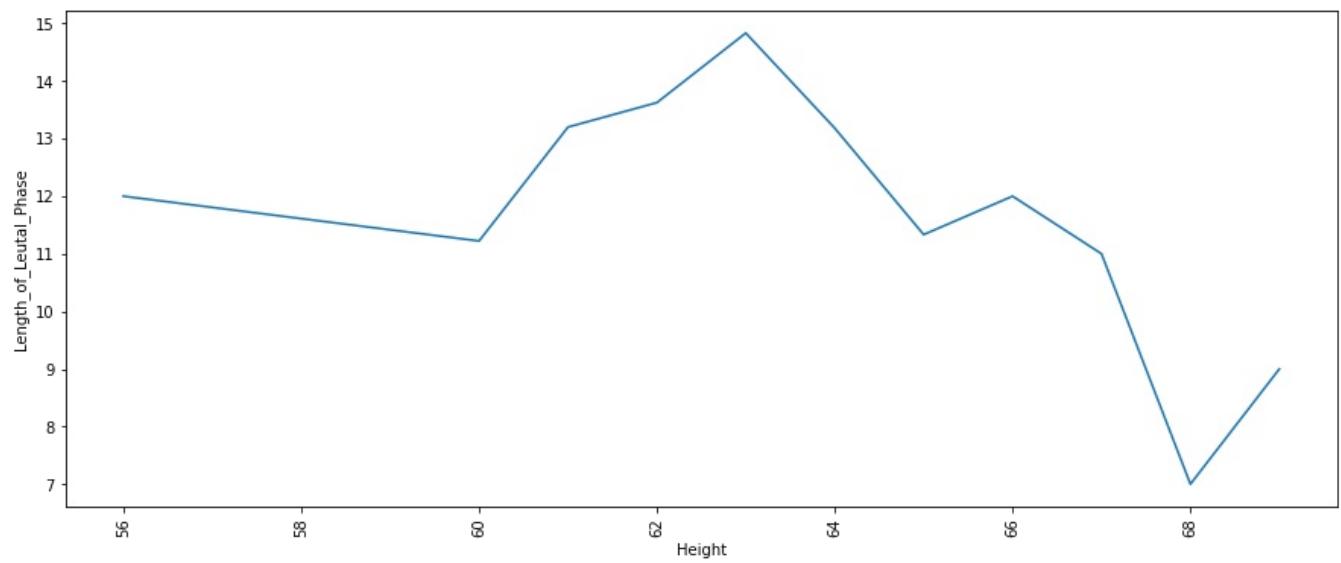


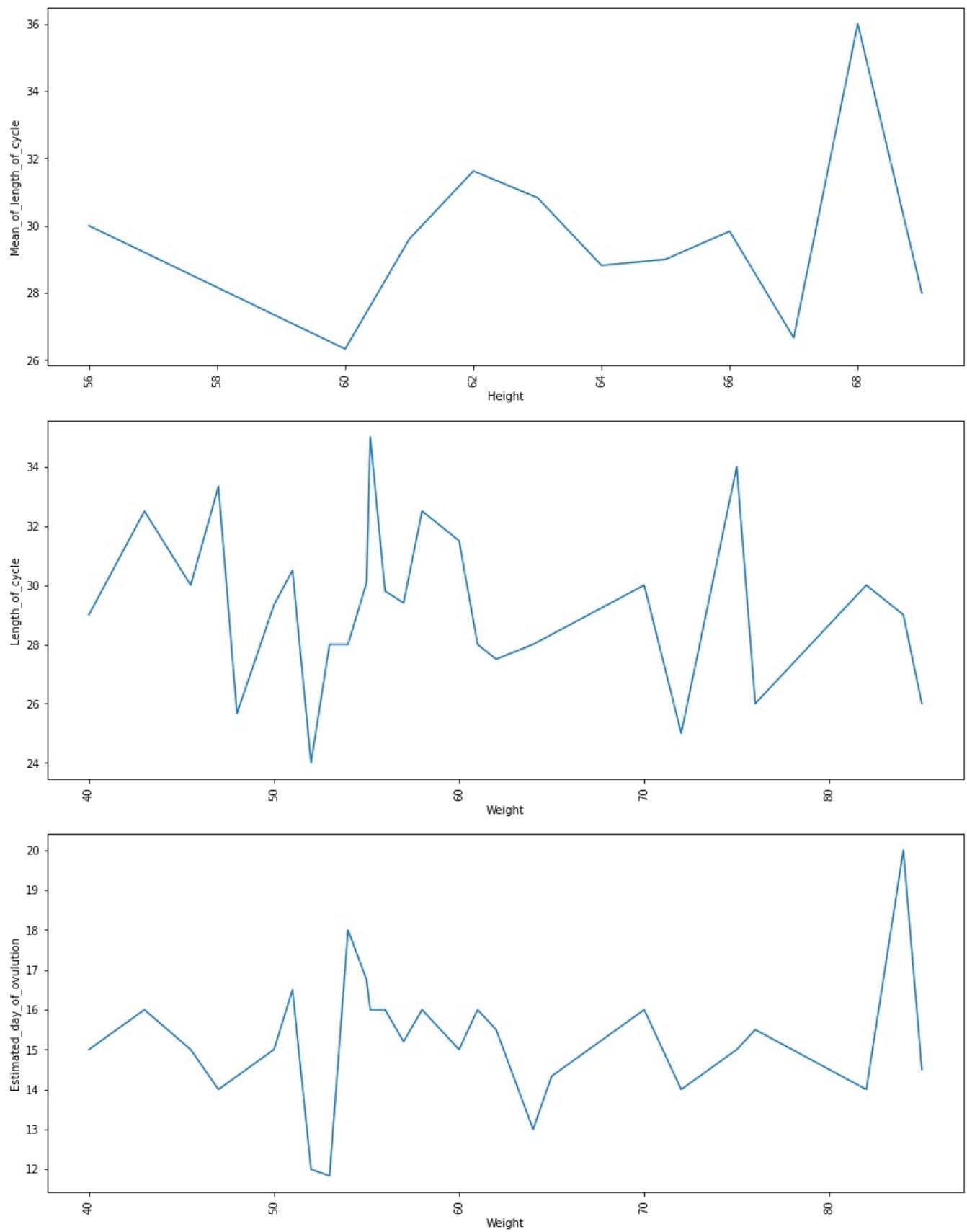


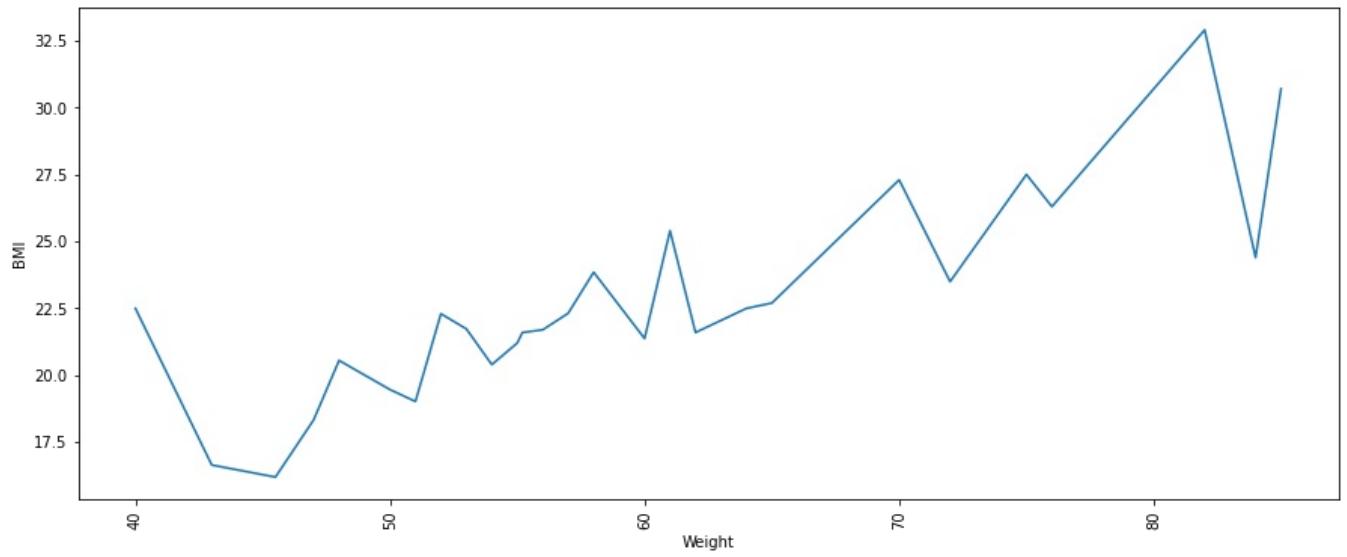
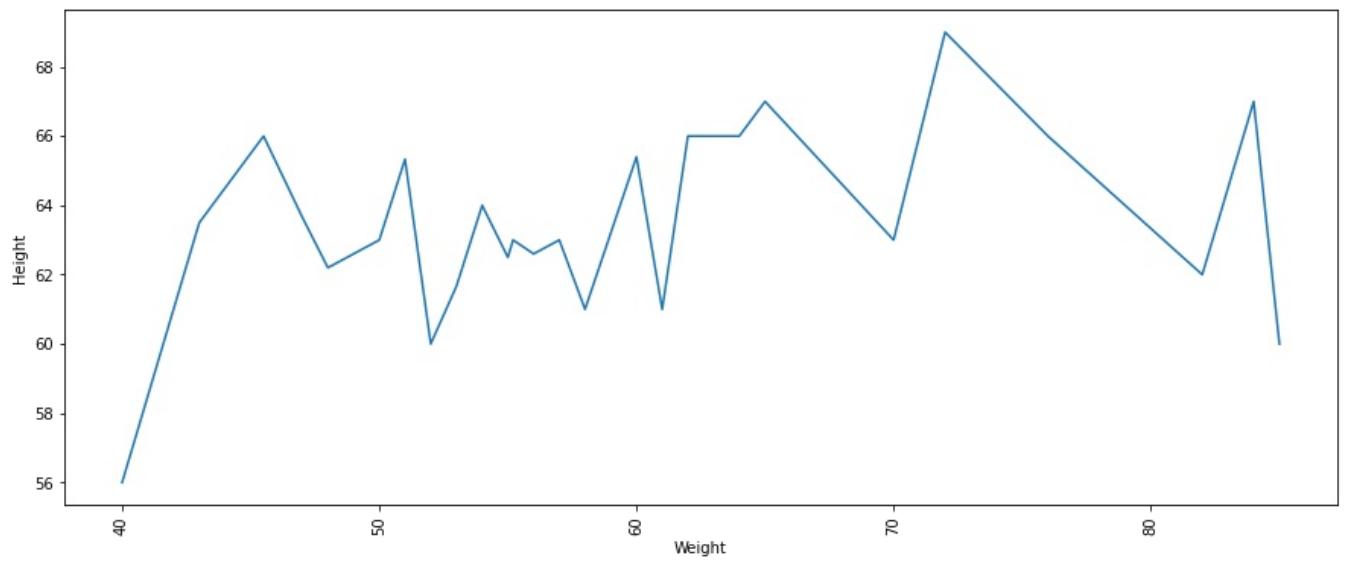
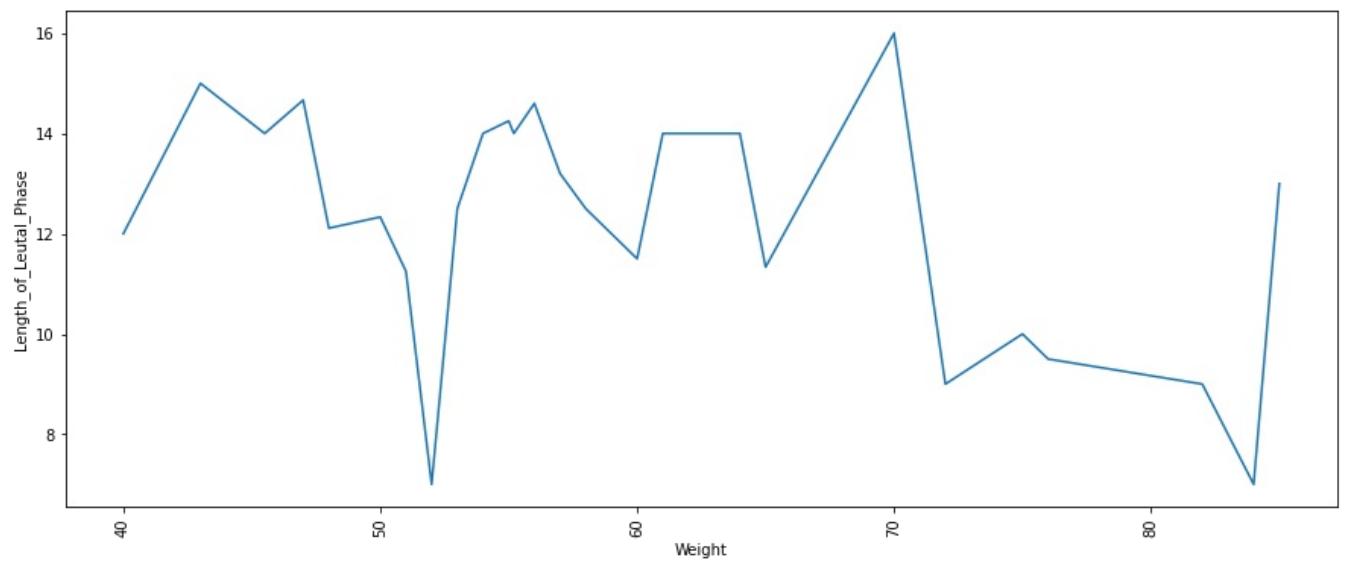


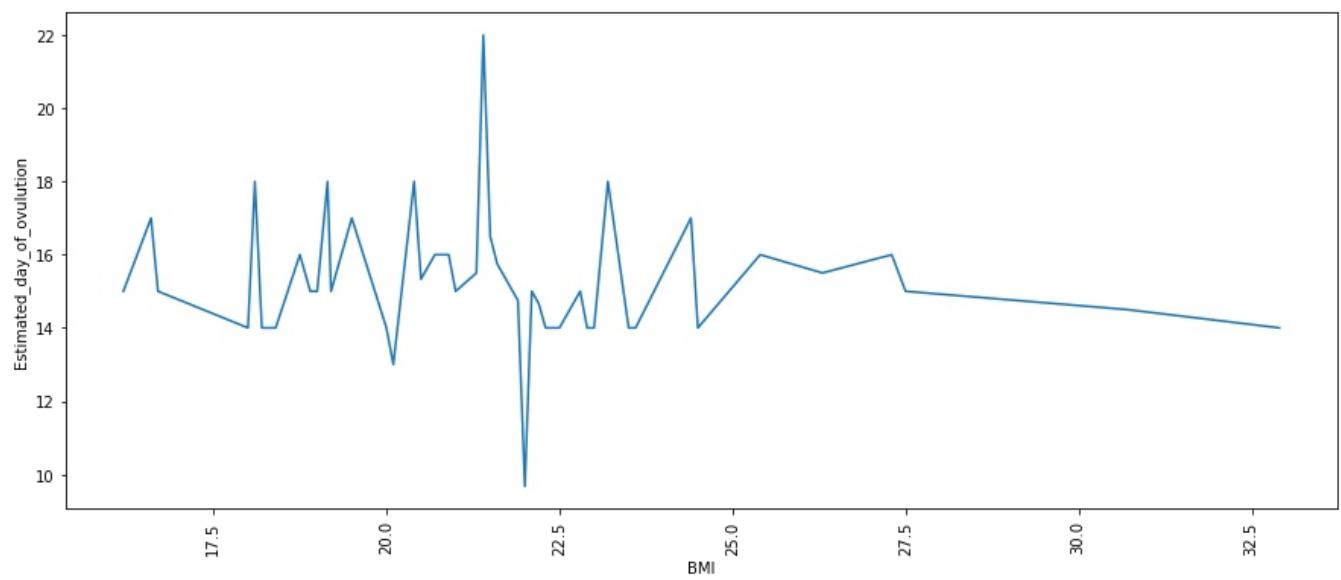
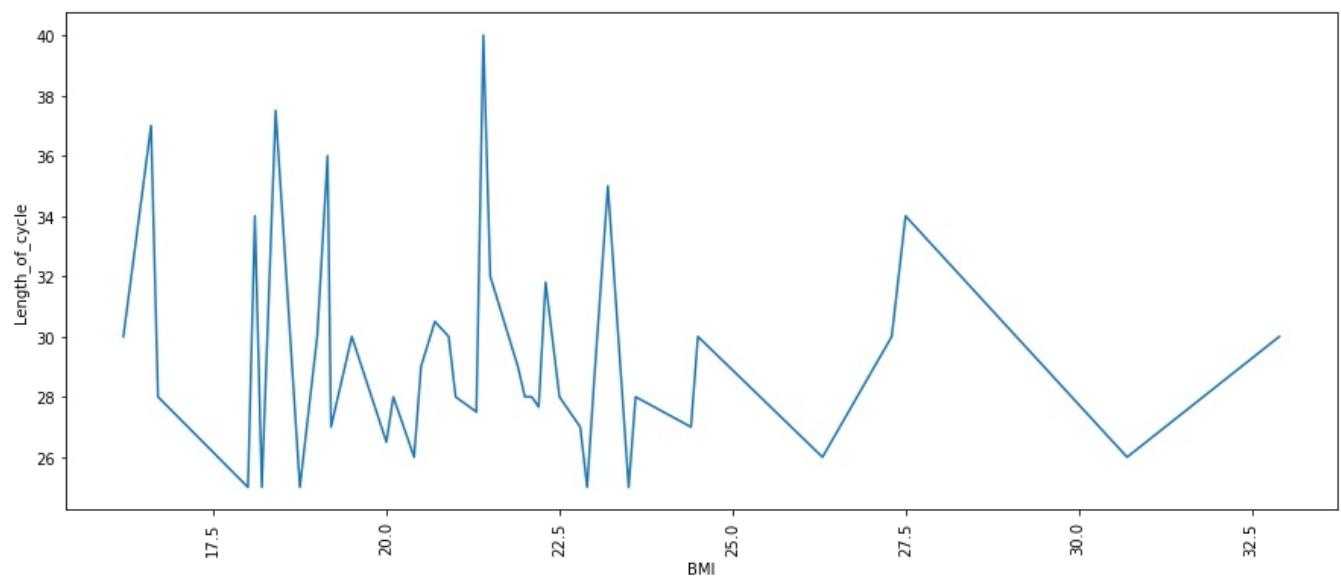
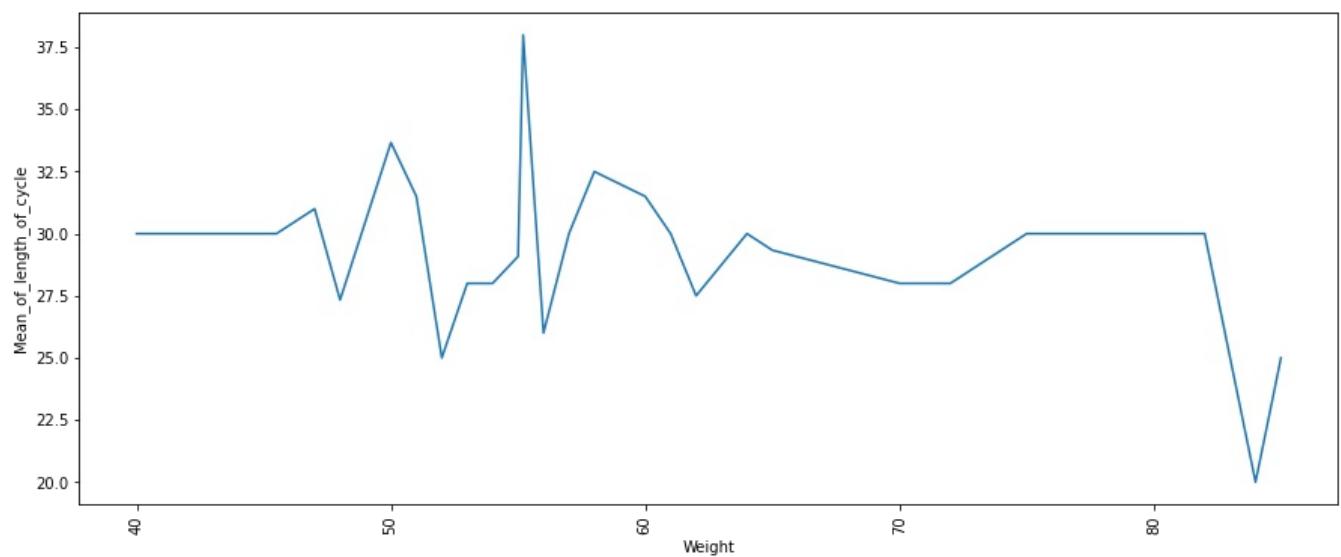


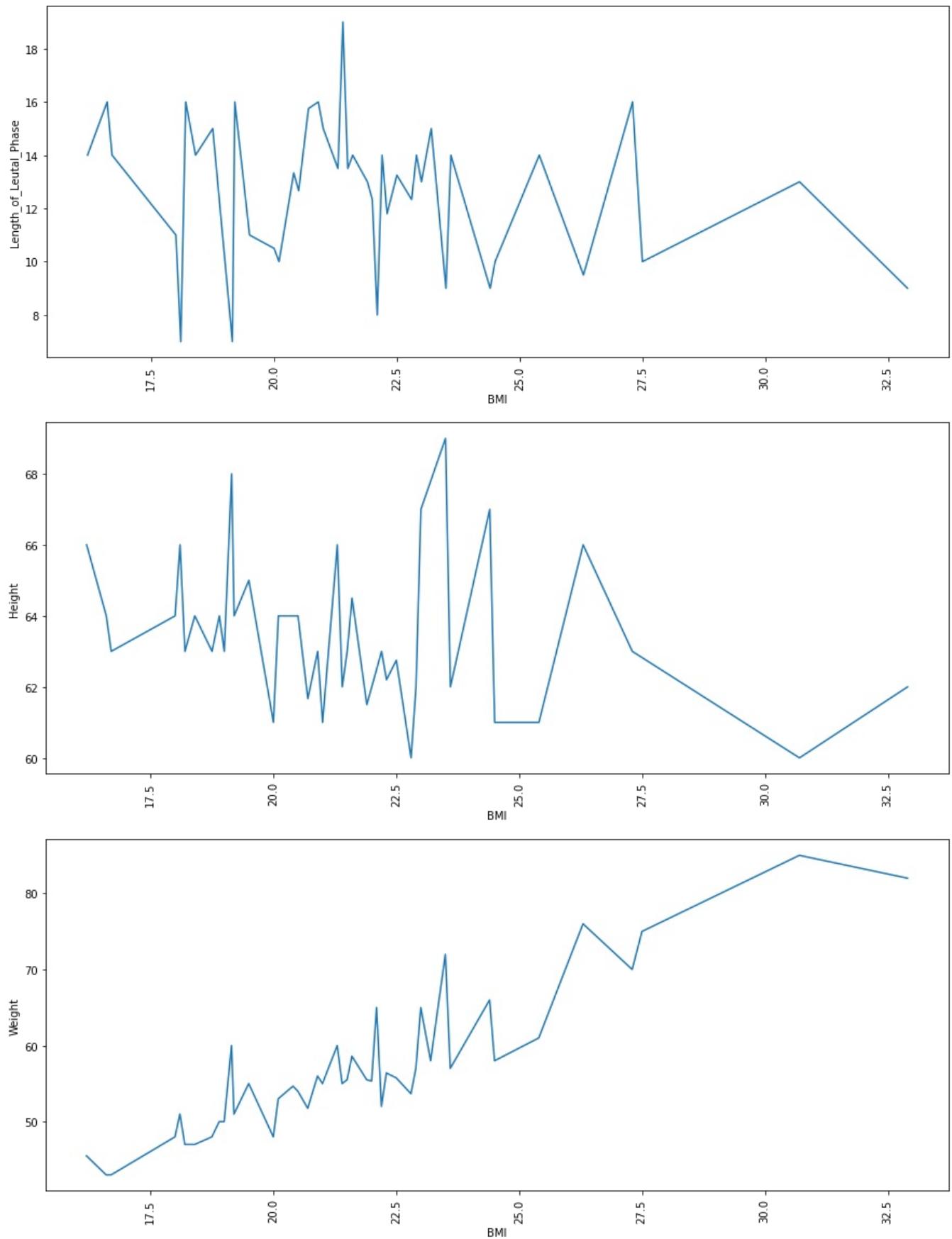


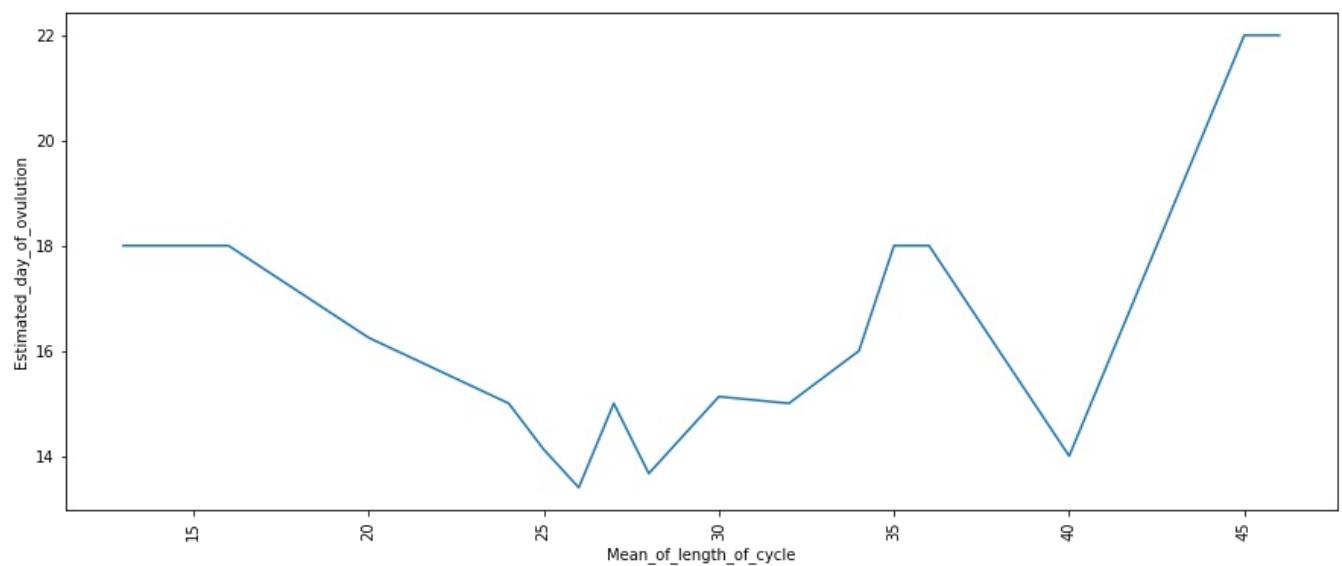
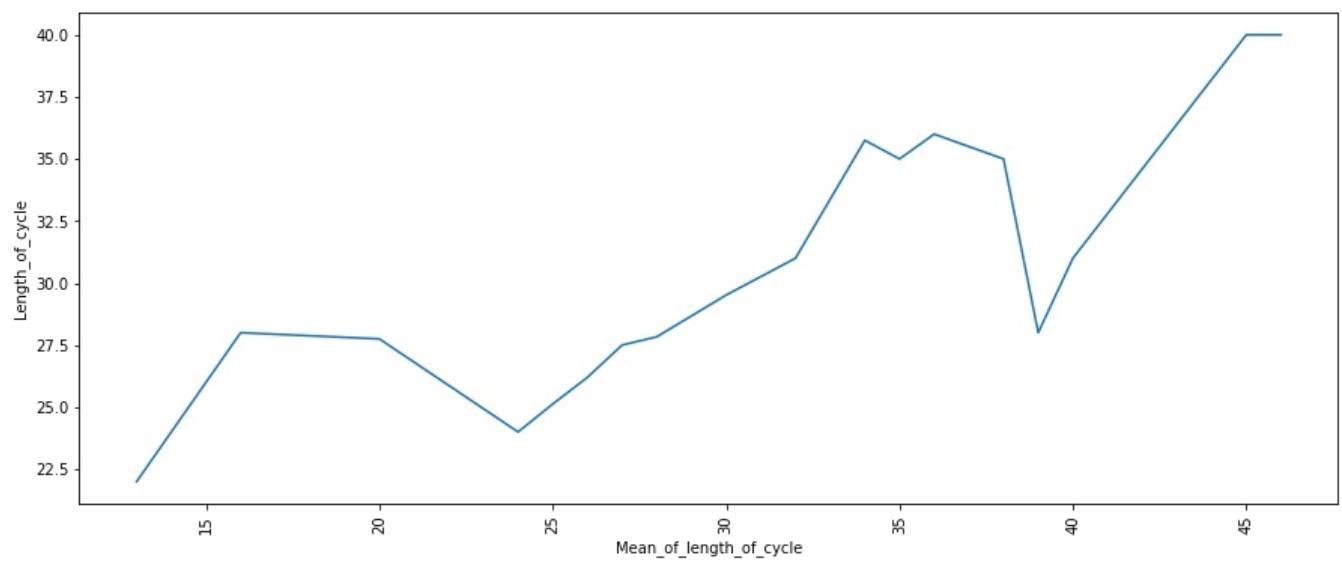
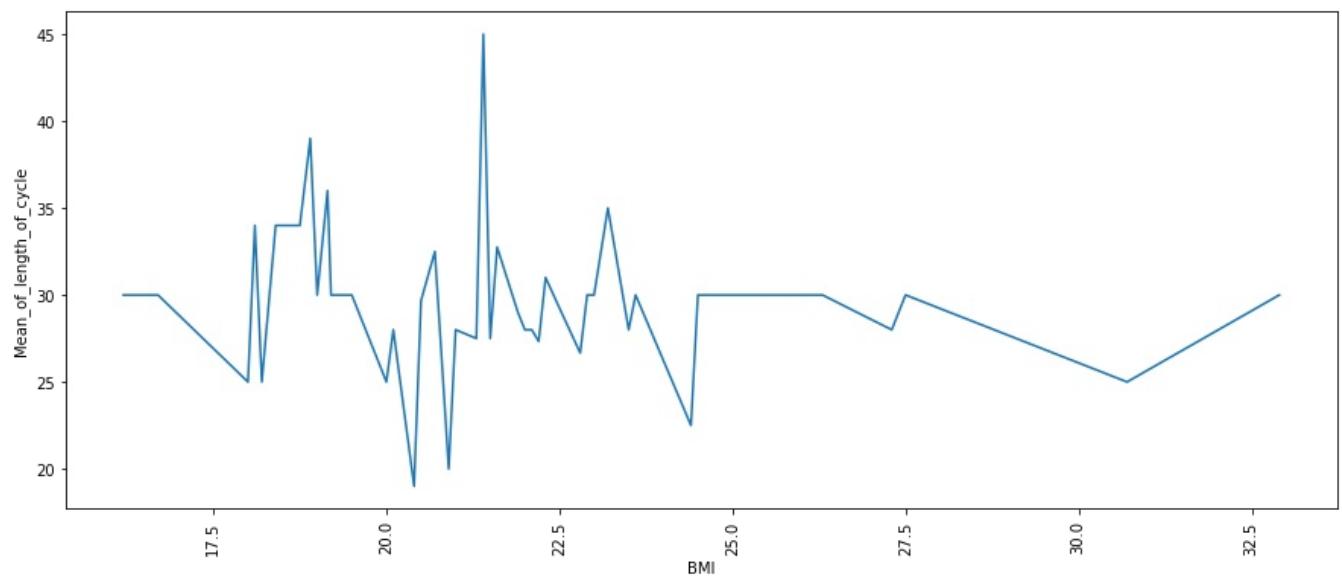


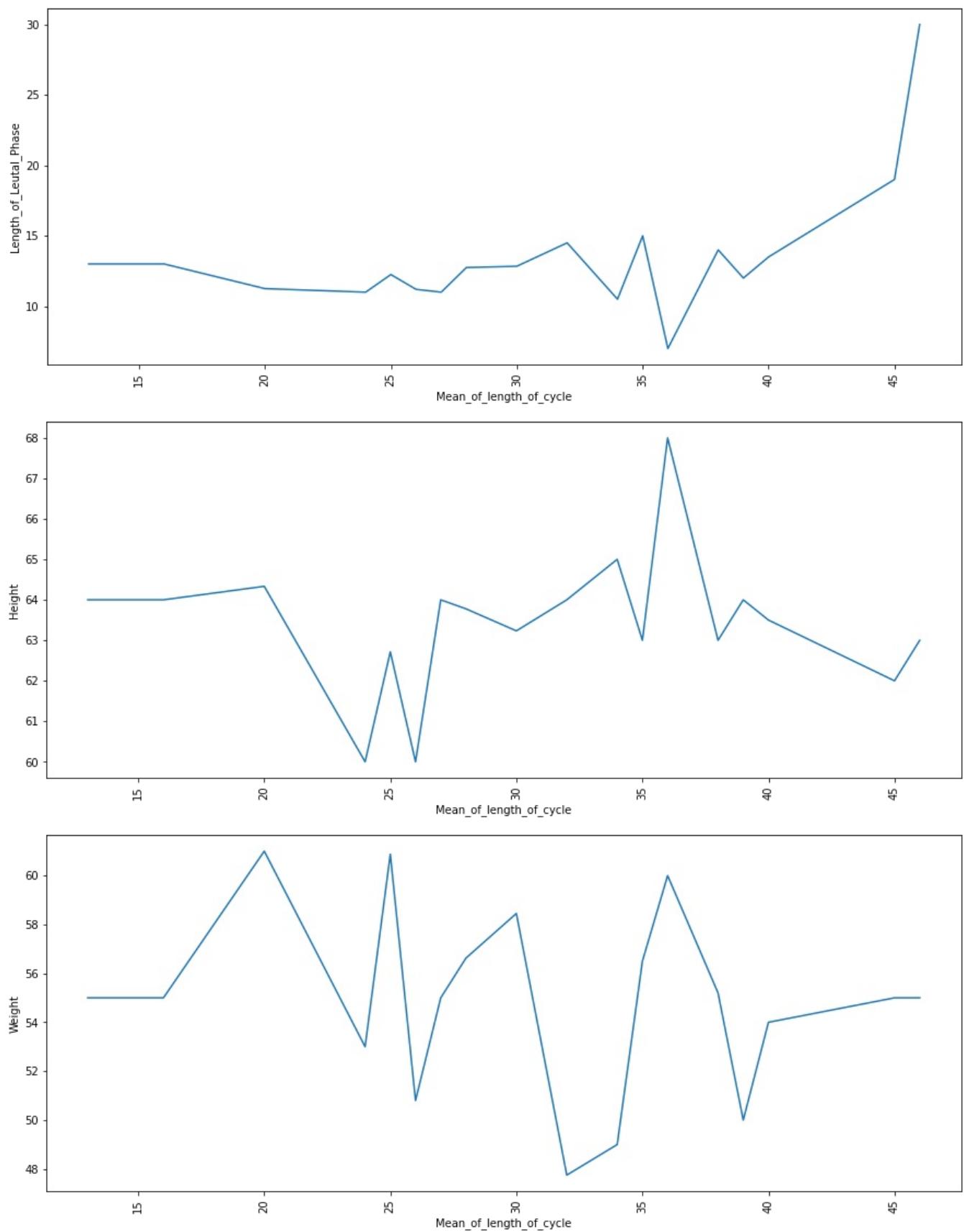


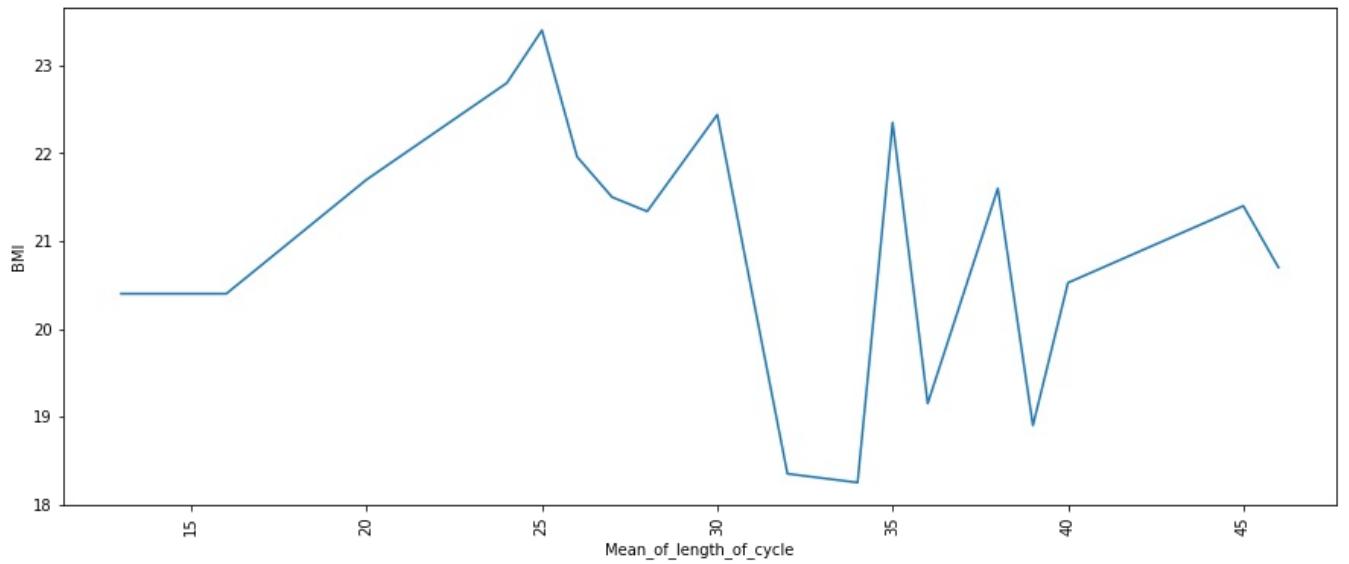




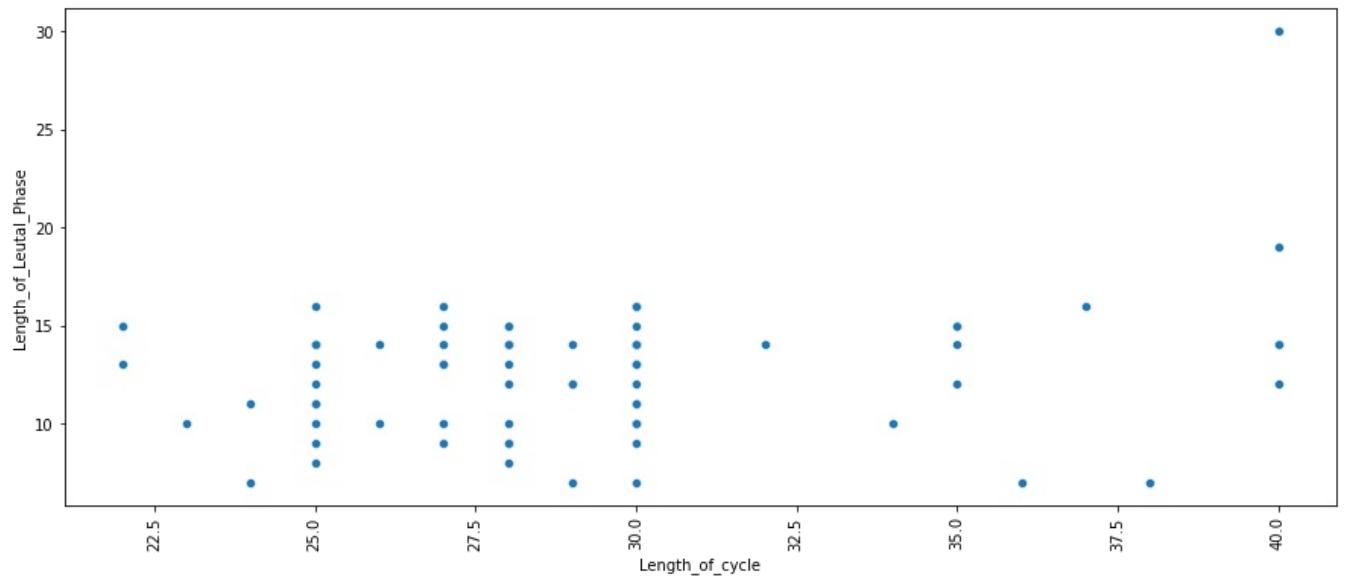
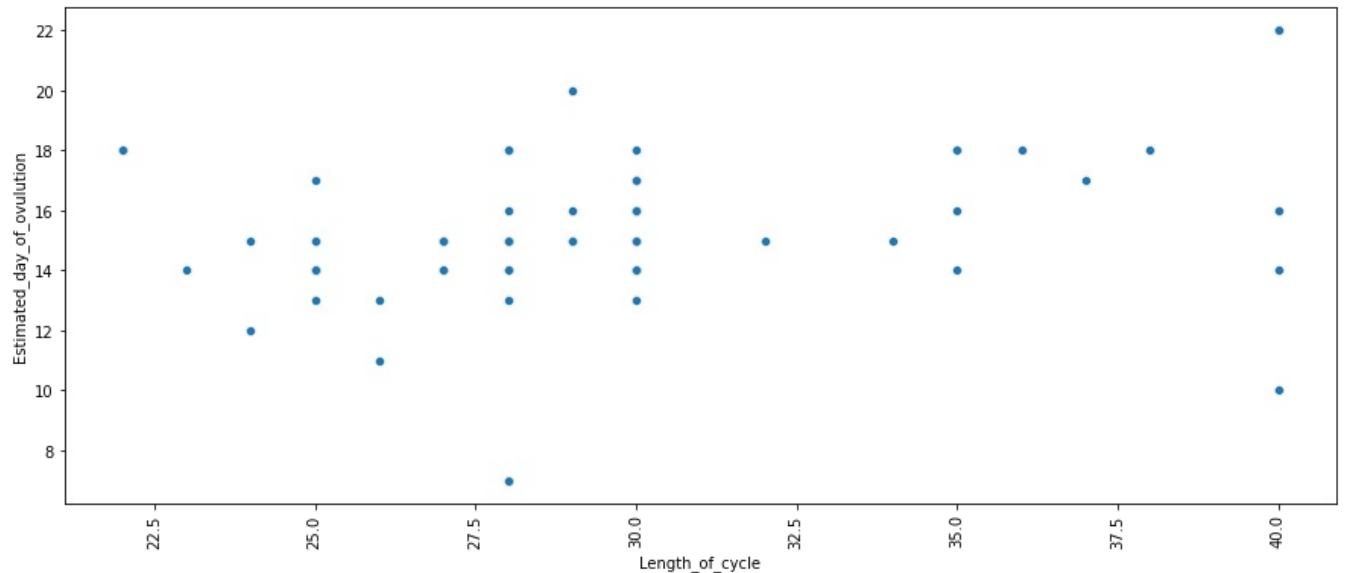


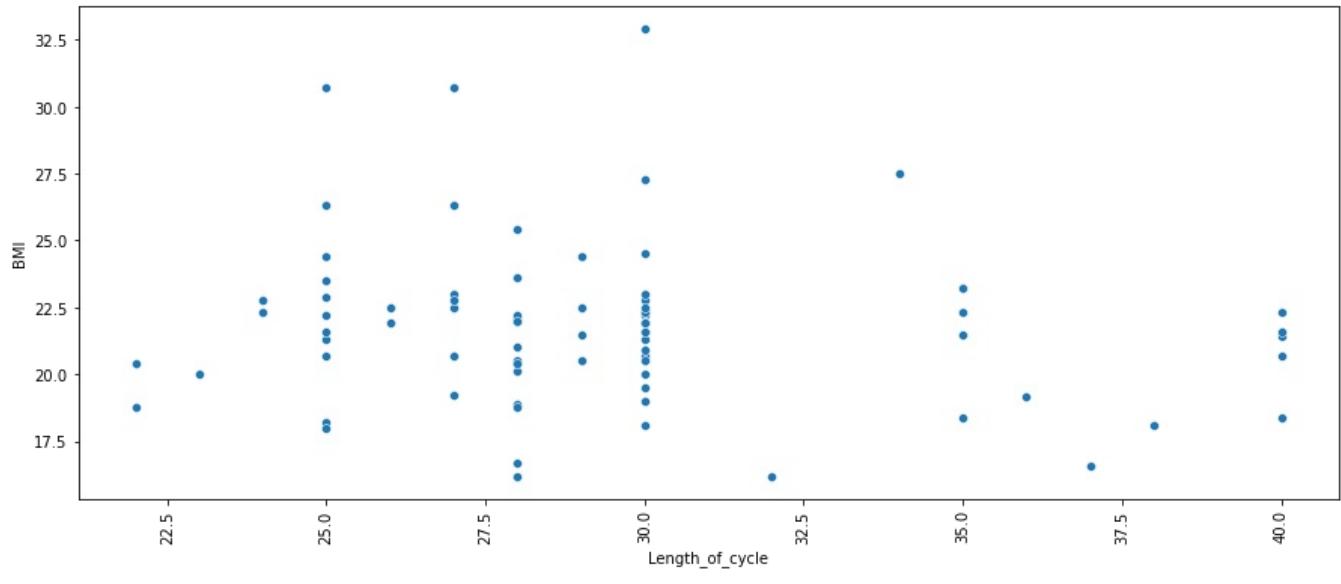
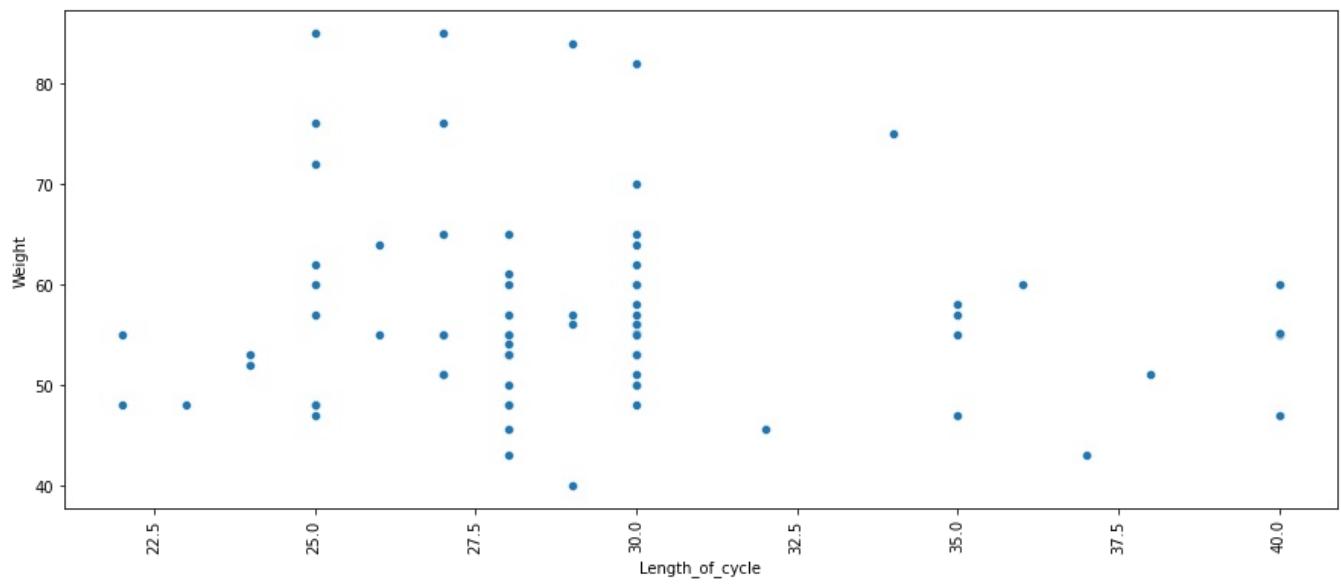
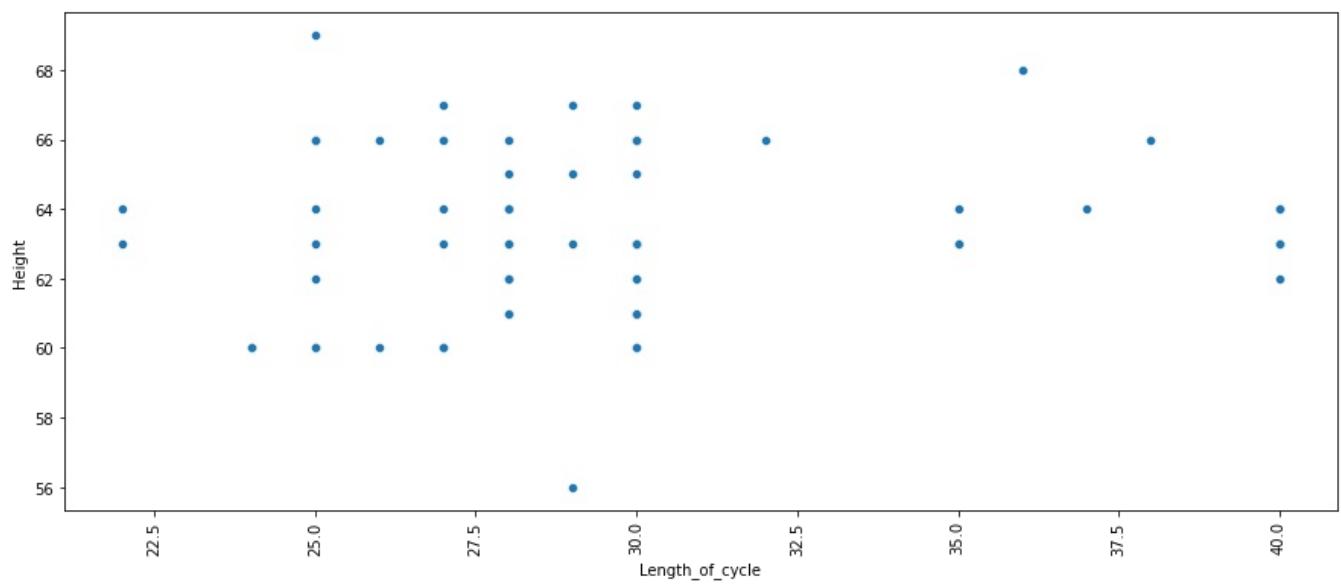


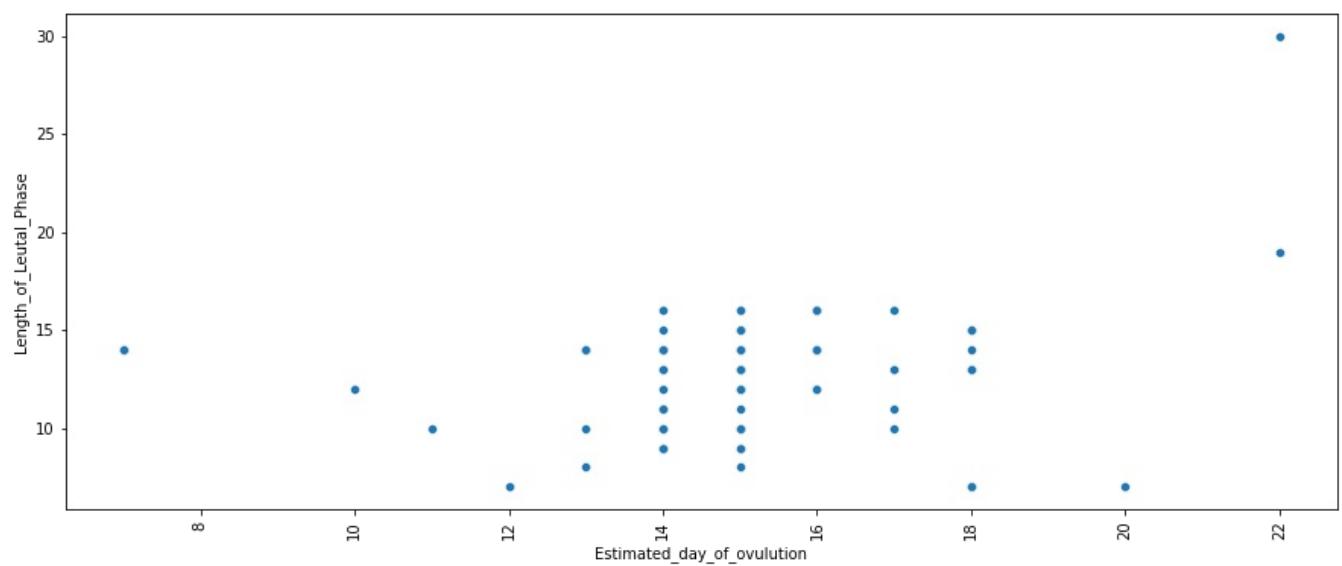
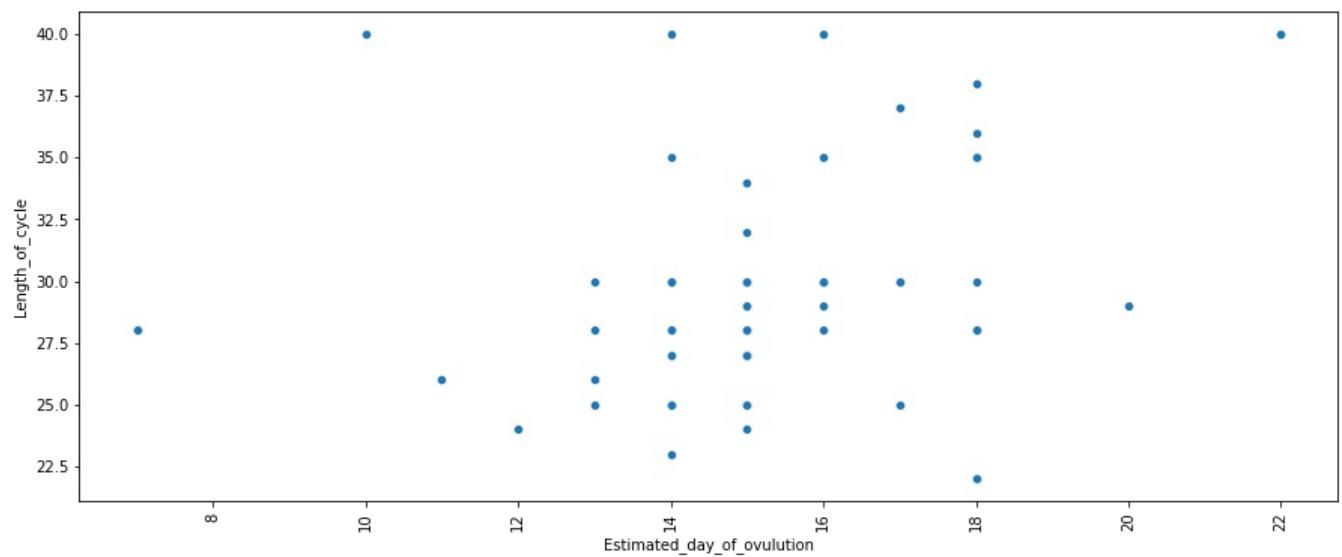
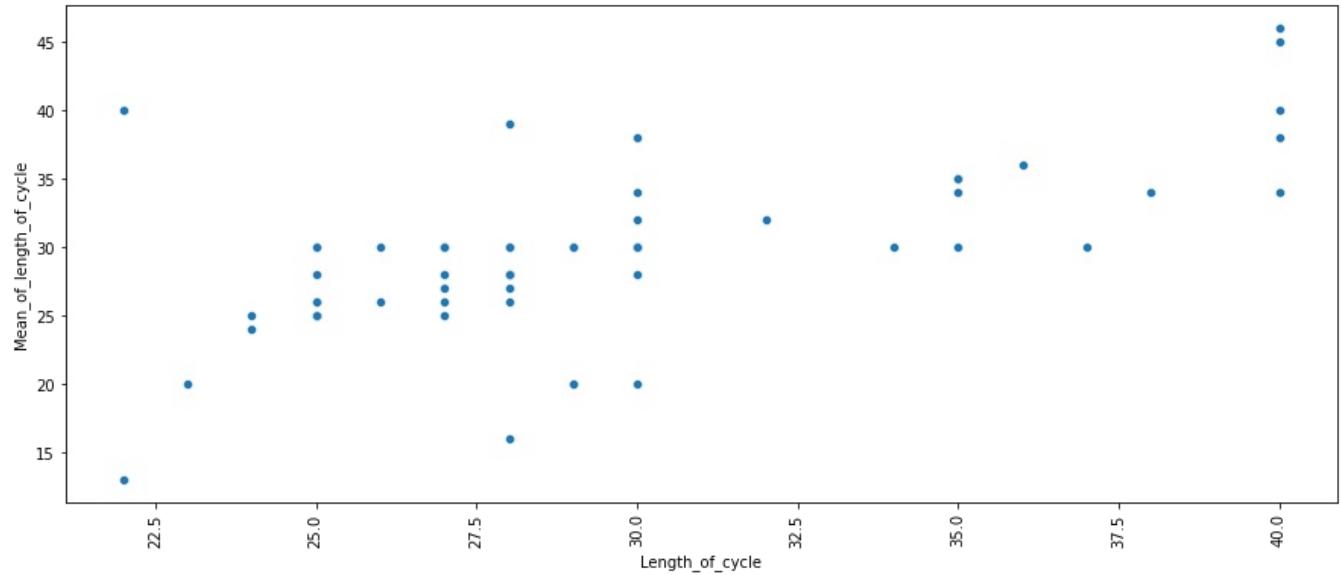


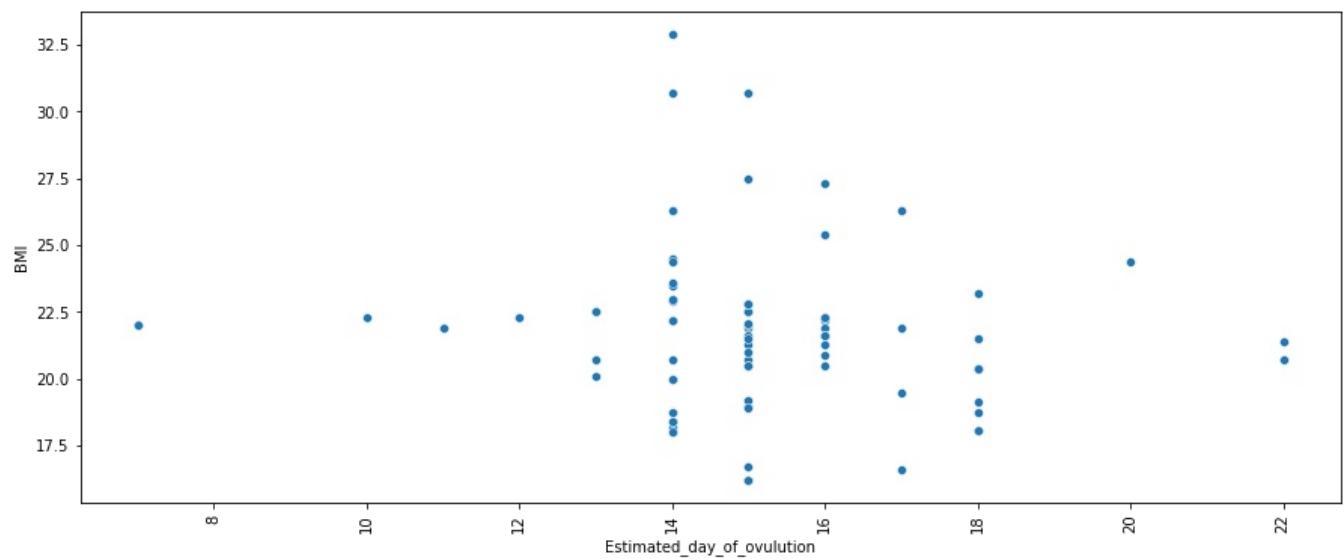
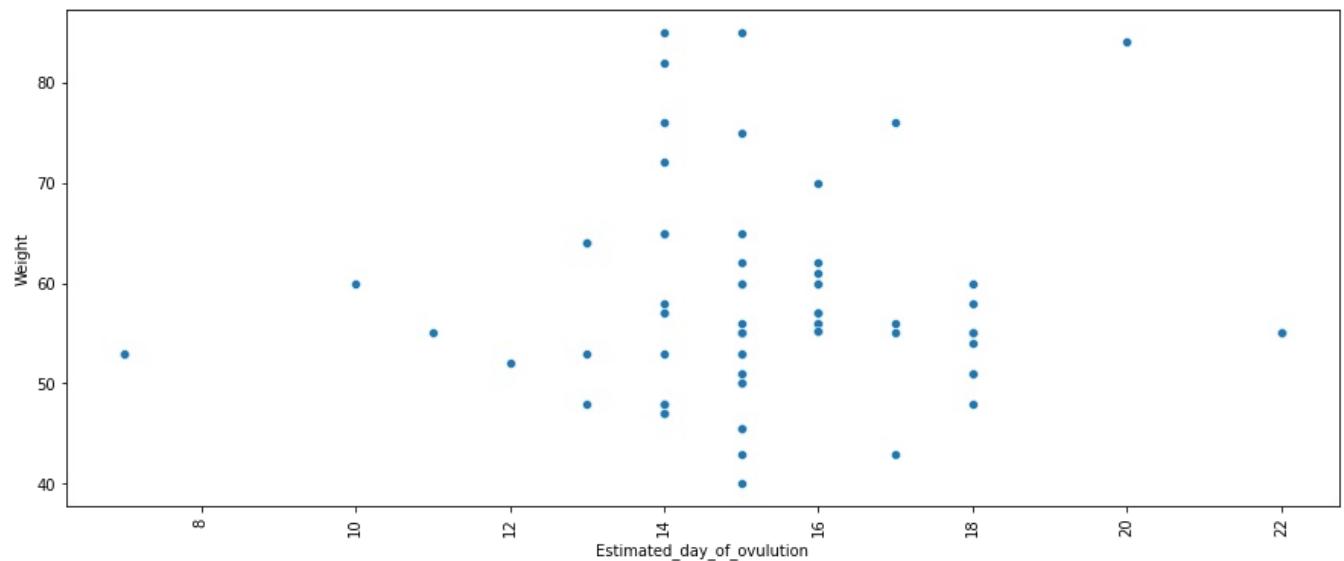
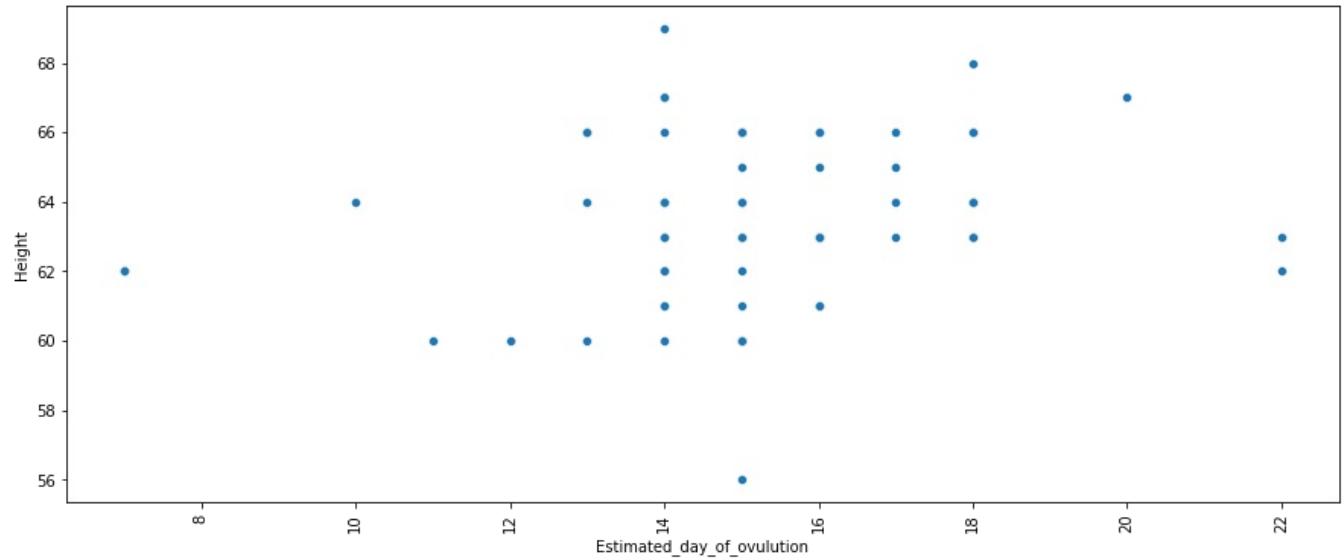


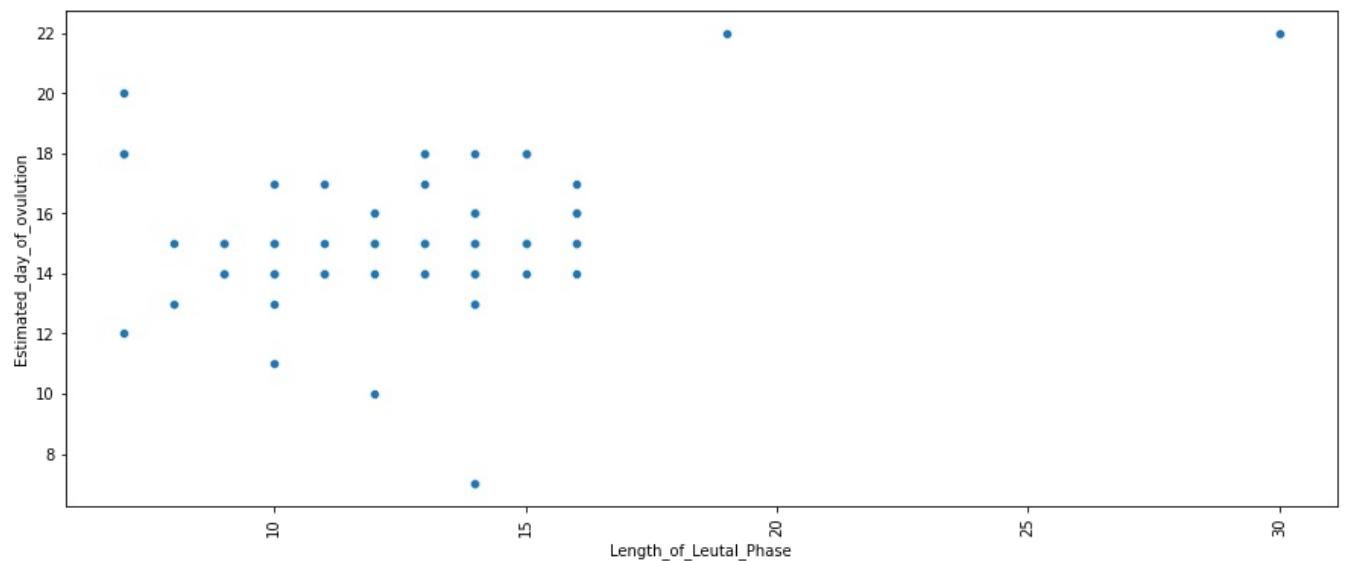
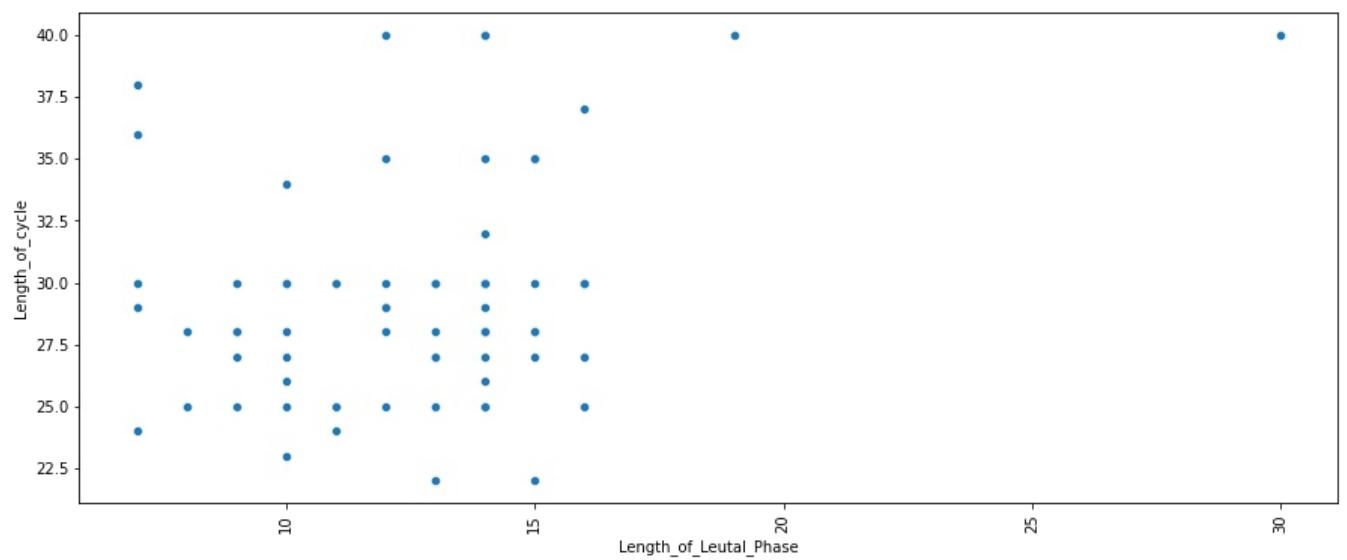
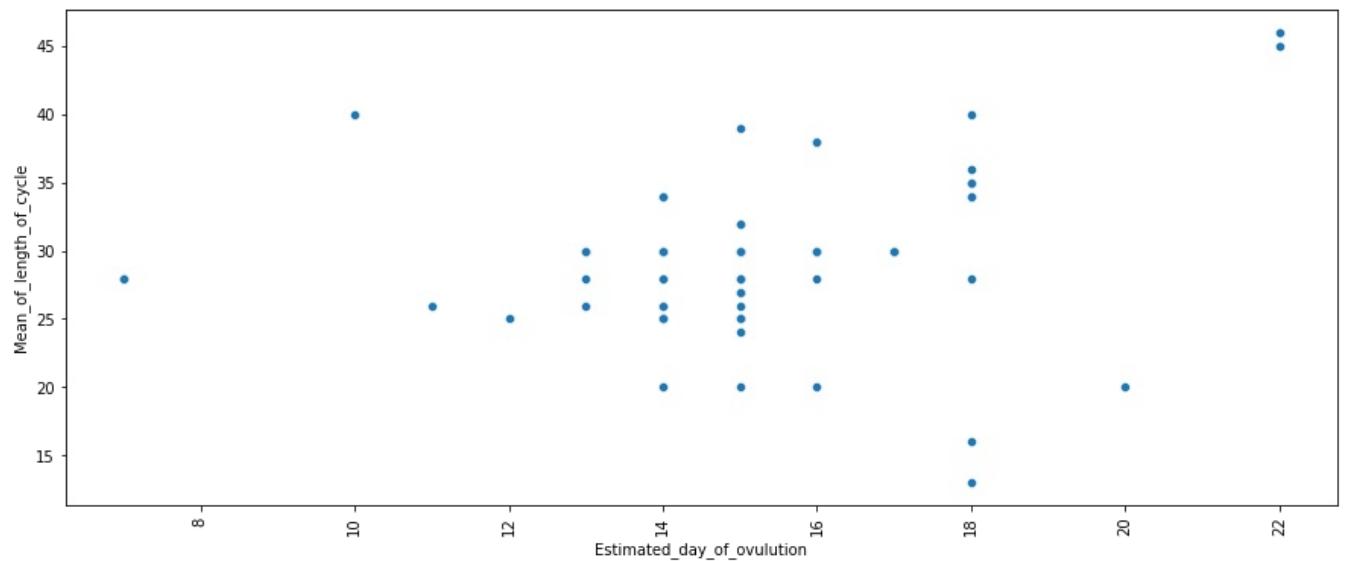
```
In [37]: for i in continuous:
    for j in continuous:
        if i != j:
            plt.figure(figsize=(15,6))
            sns.scatterplot(x = i, y = j, data = df, ci = None, palette='hls')
            plt.xticks(rotation = 90)
            plt.show()
```

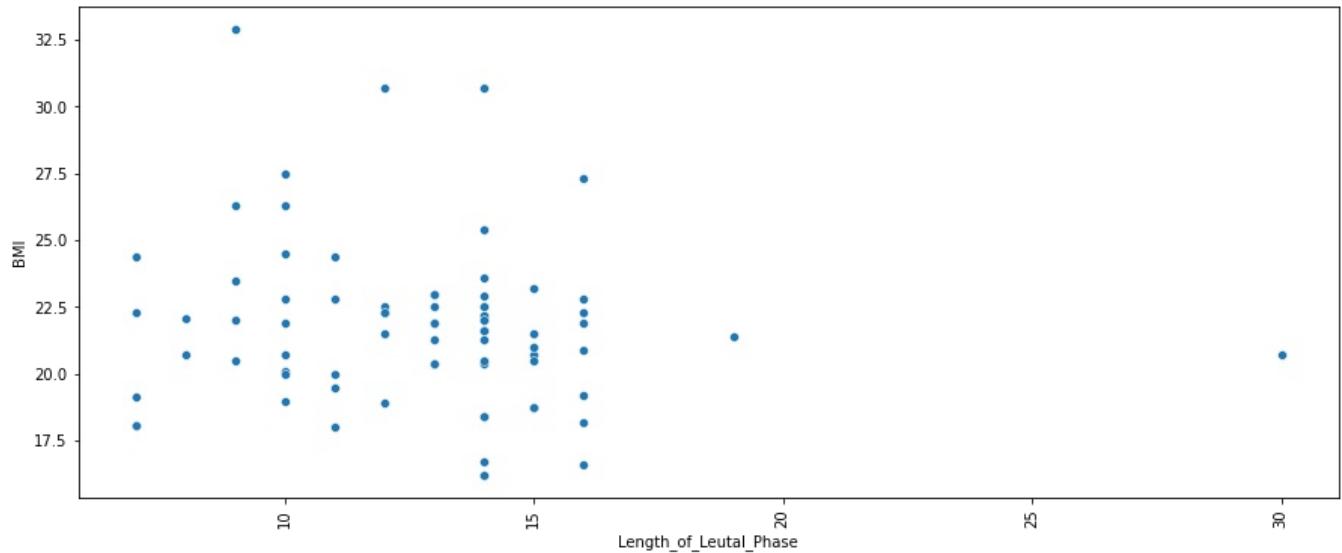
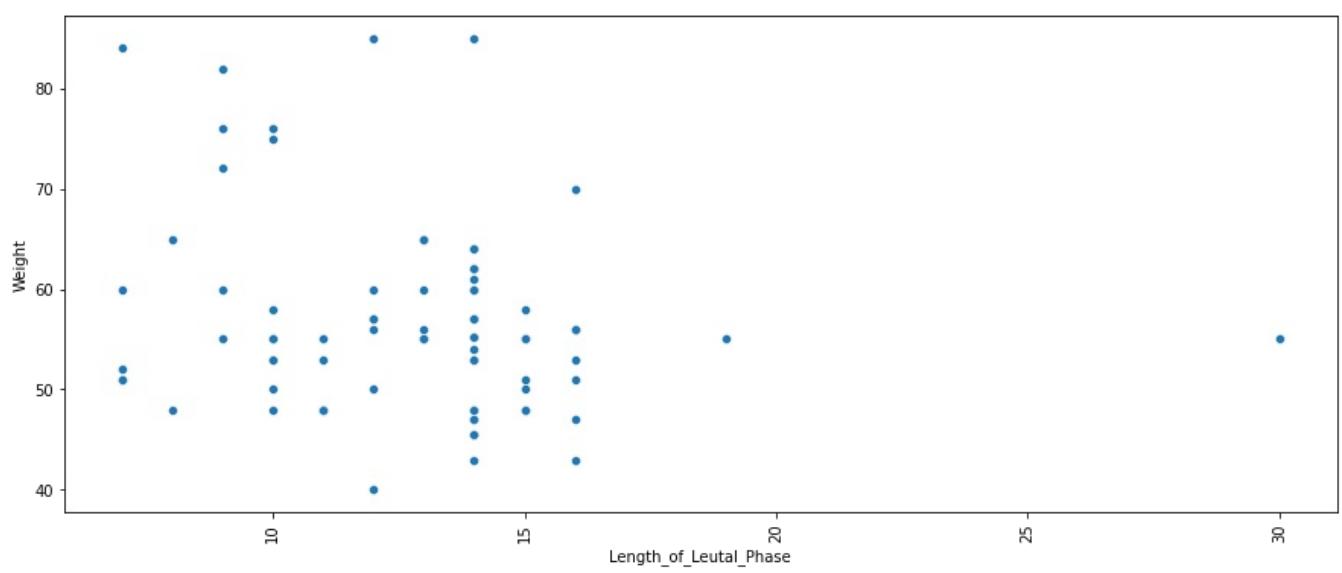
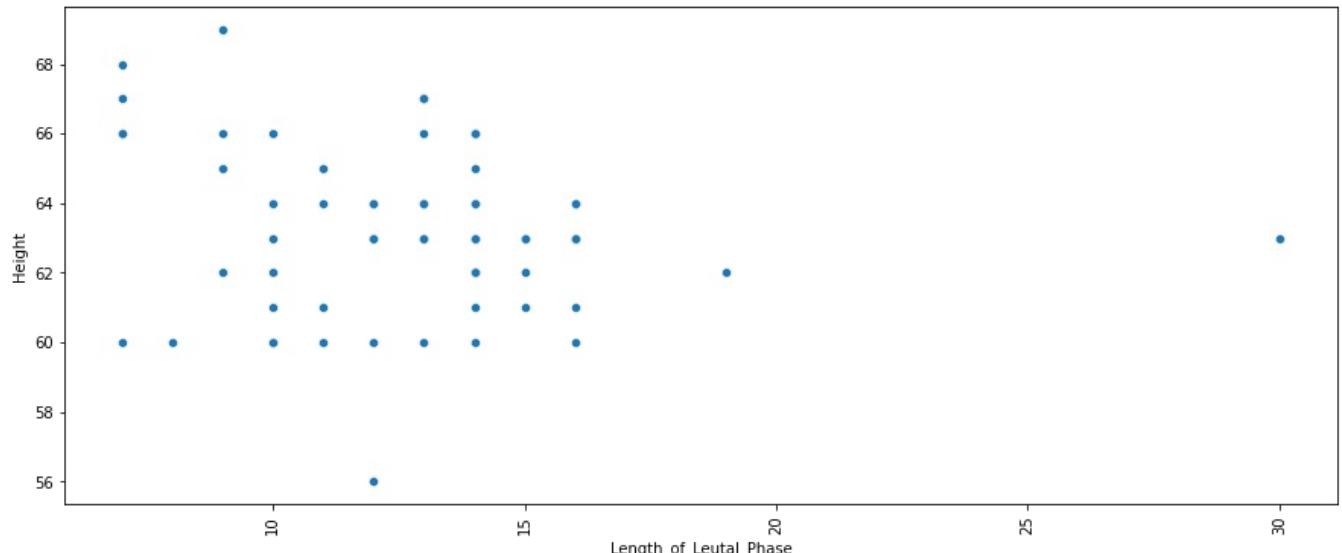


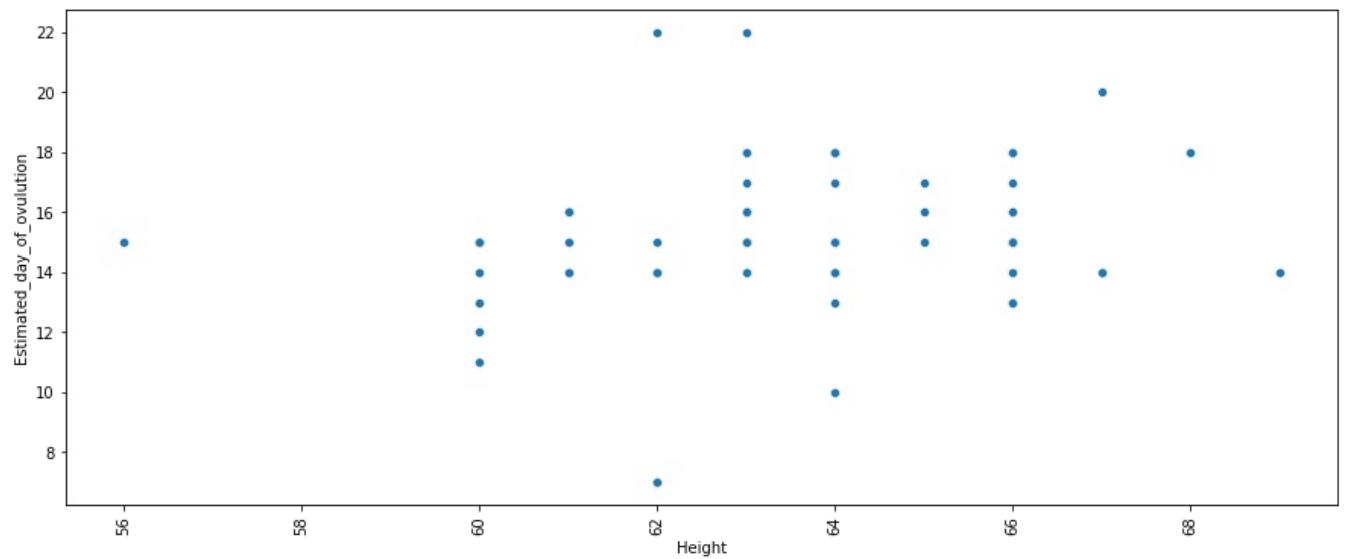
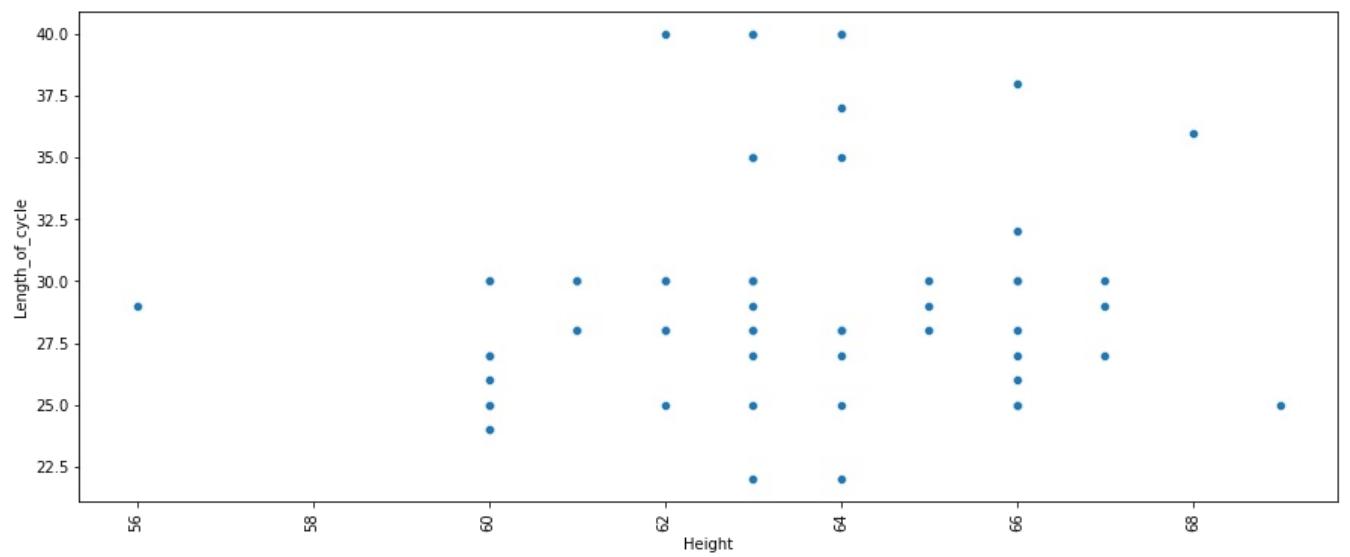
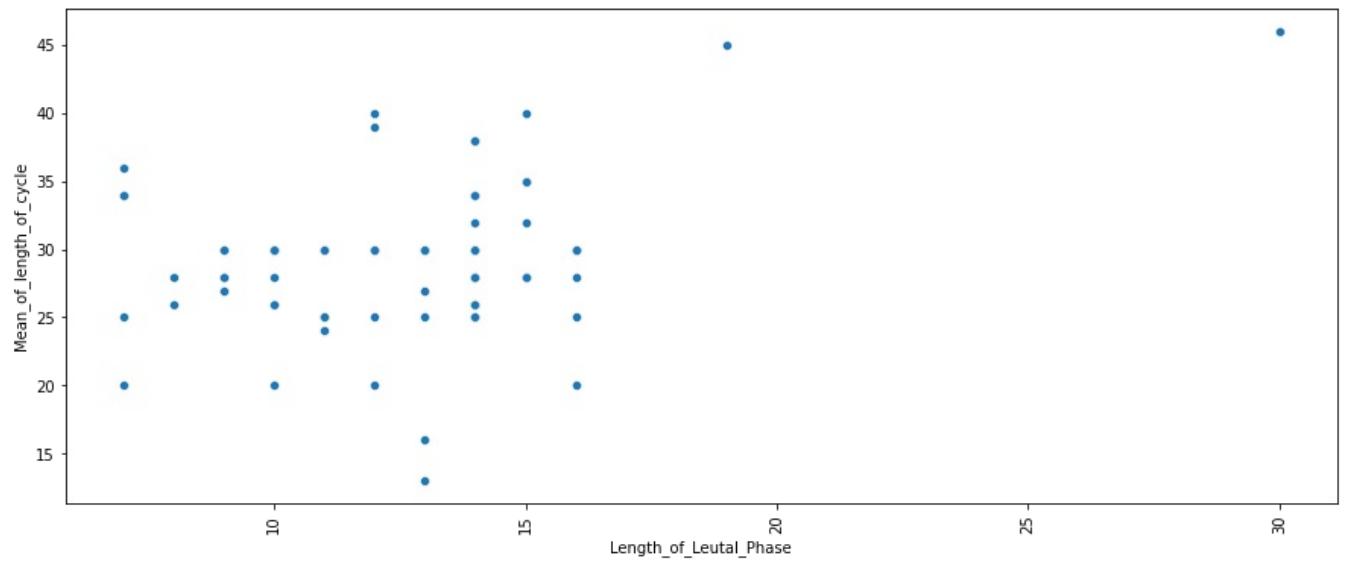


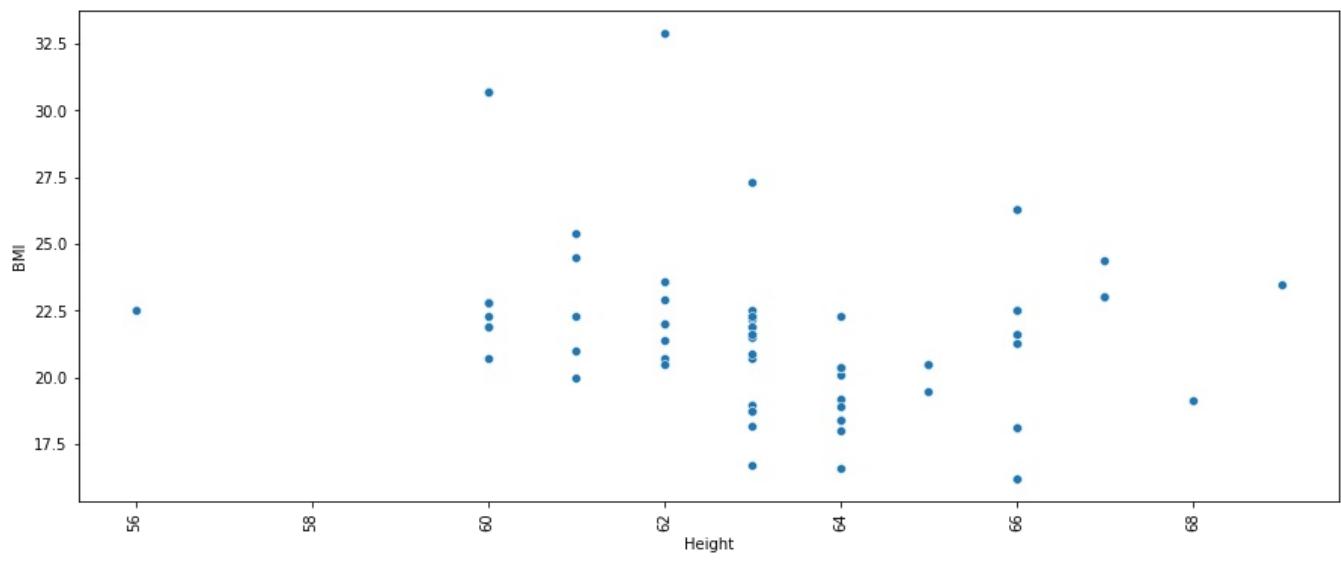
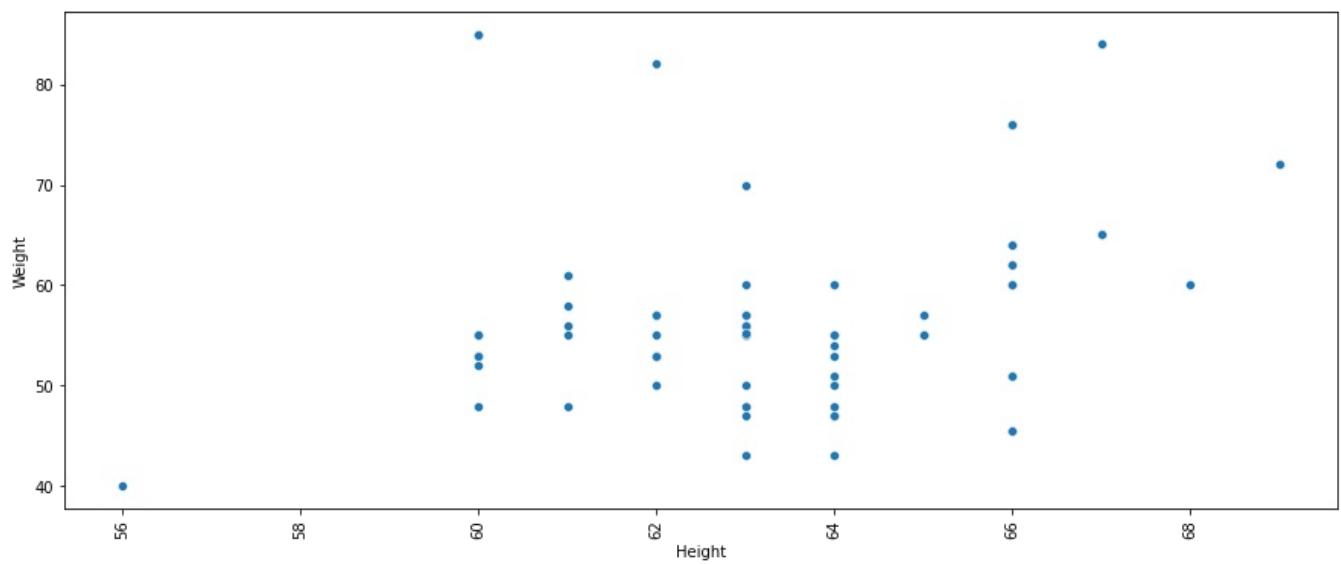
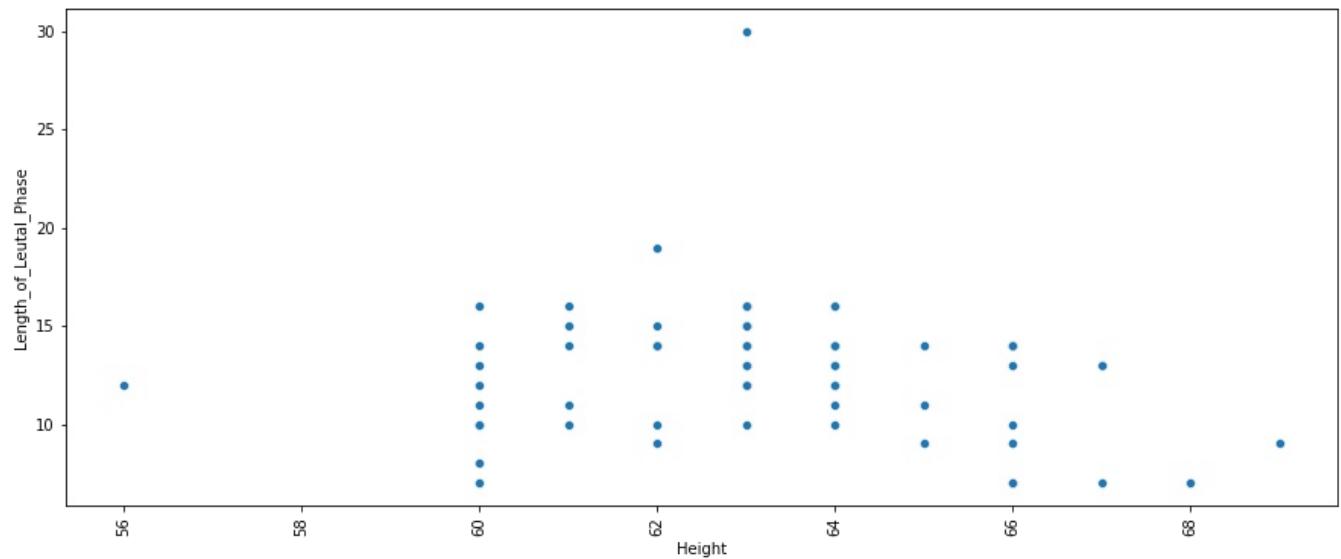


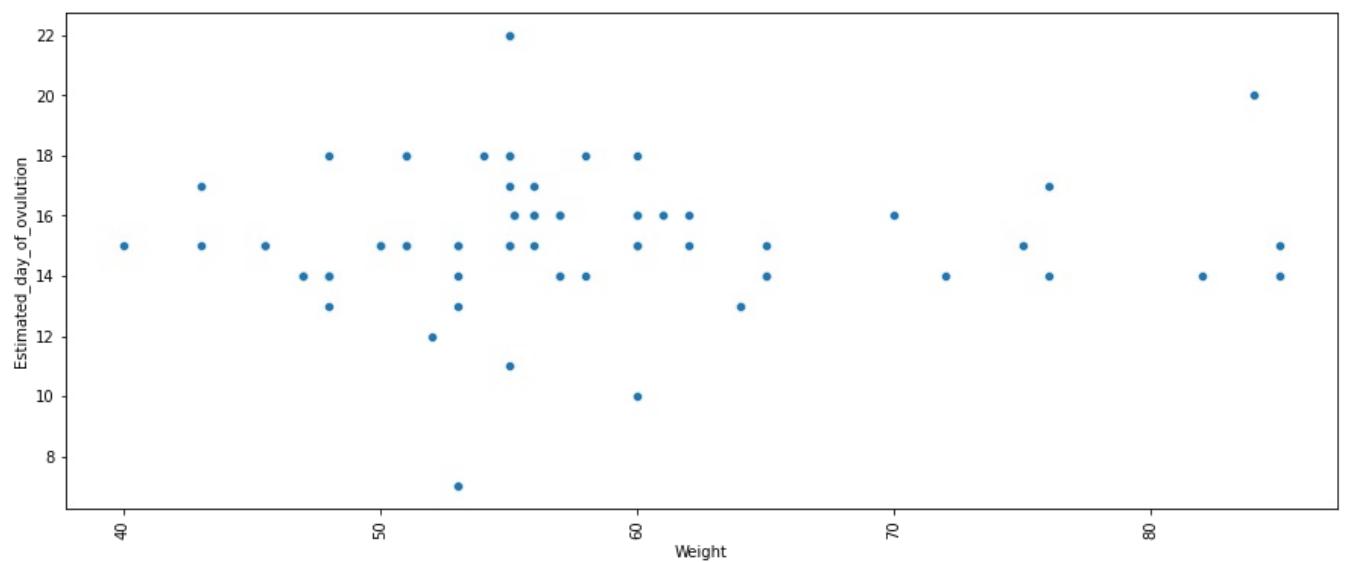
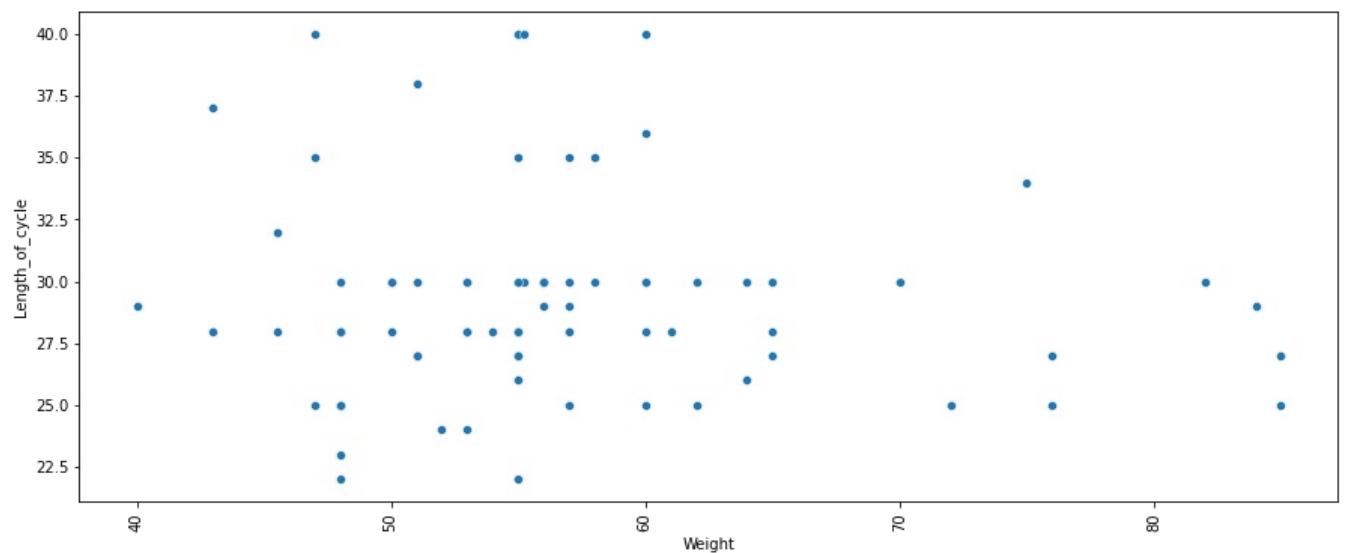
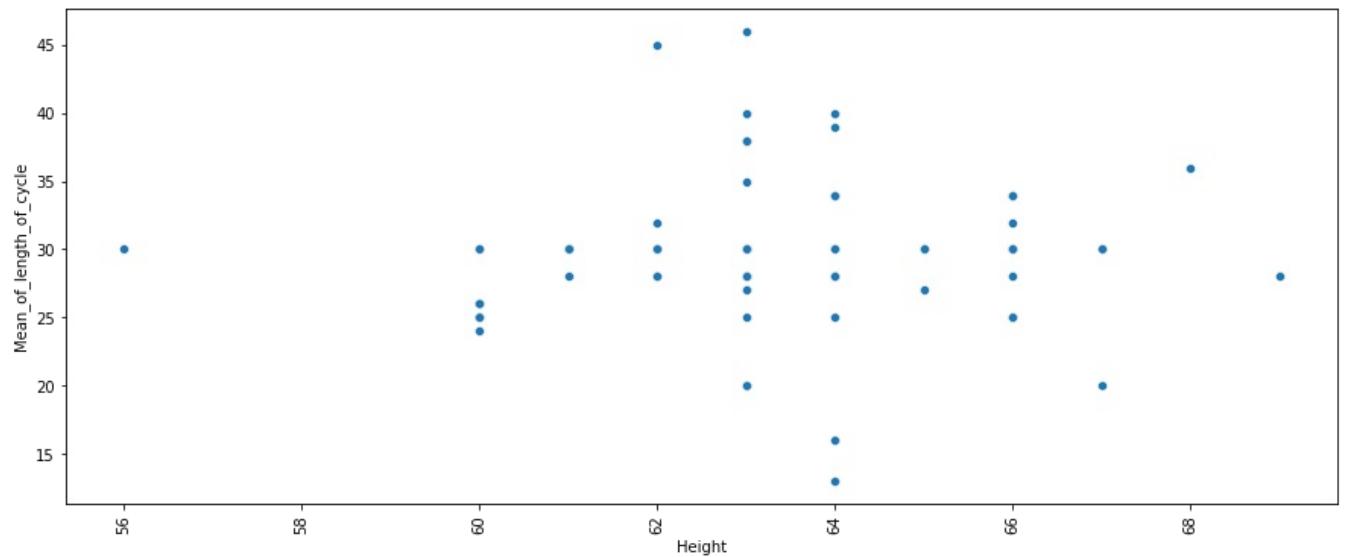


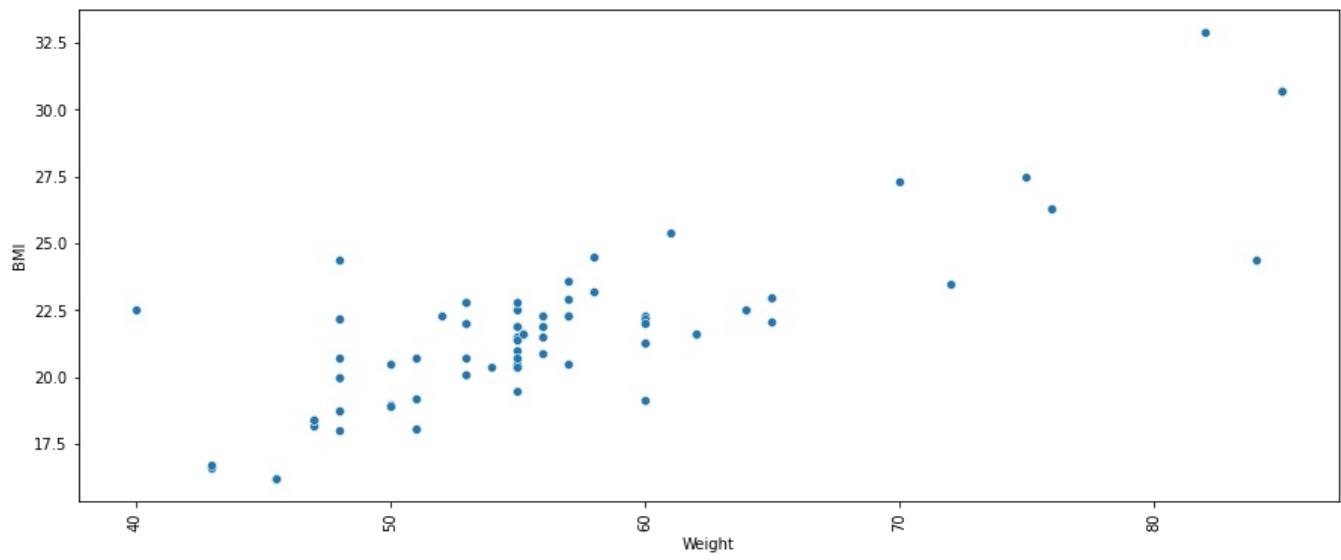
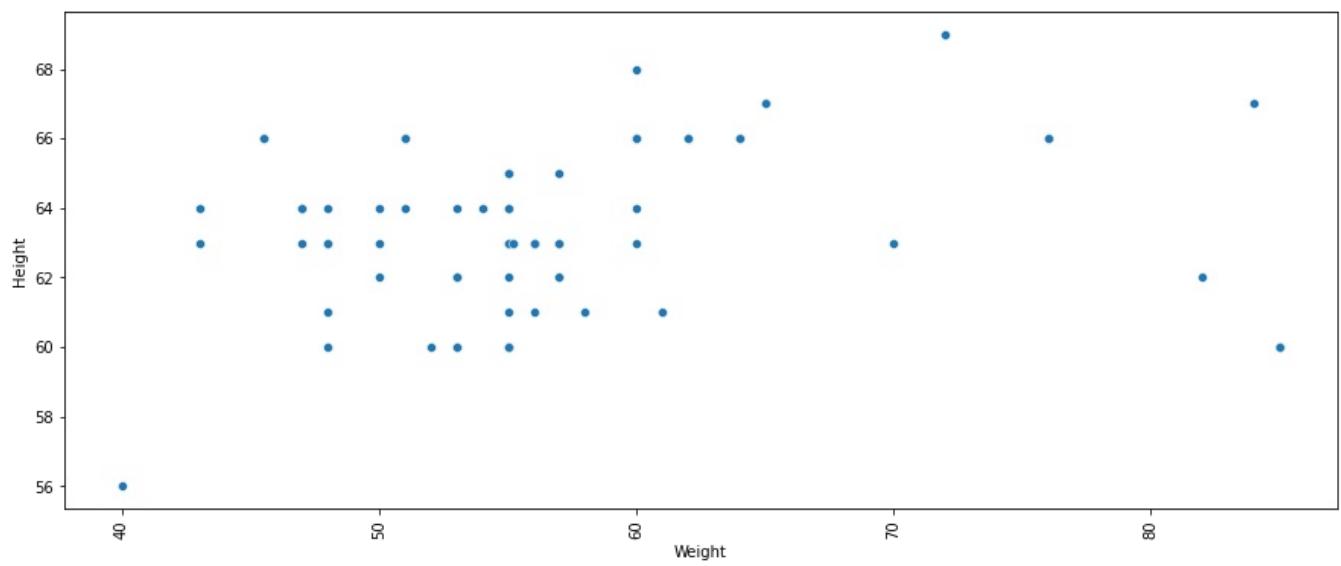
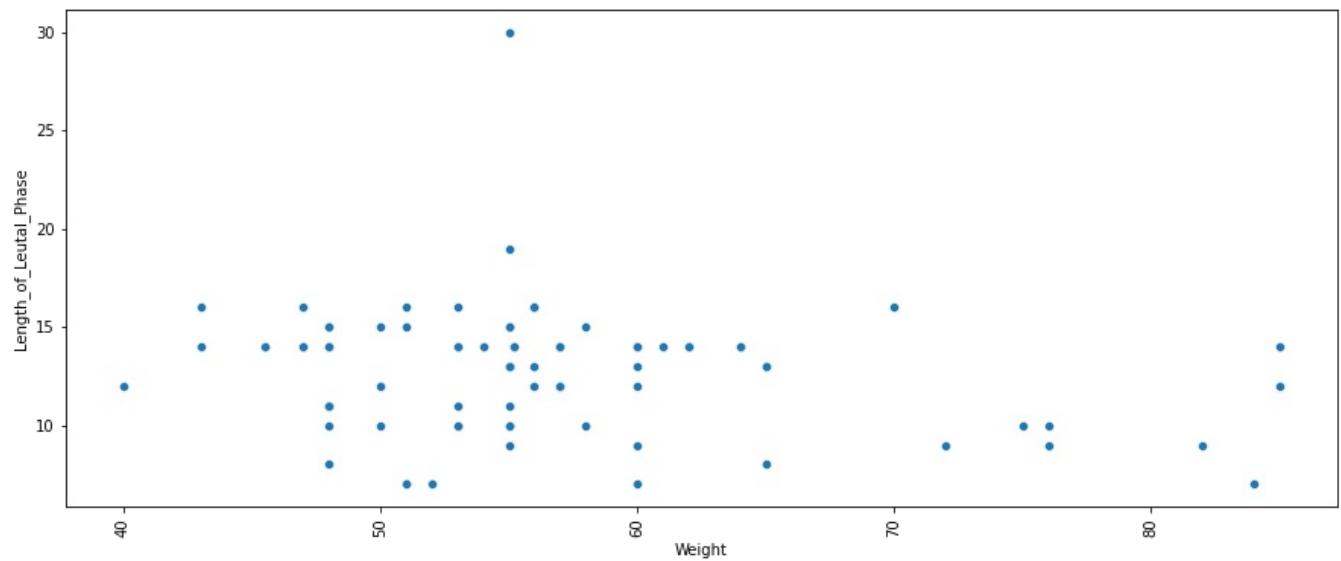


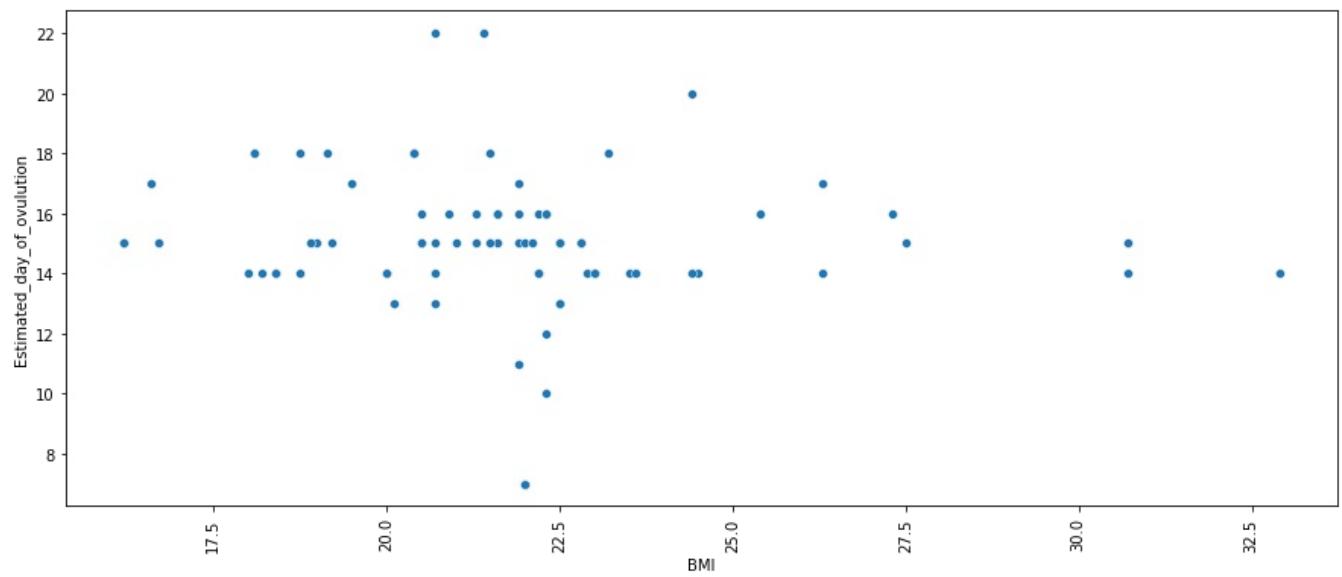
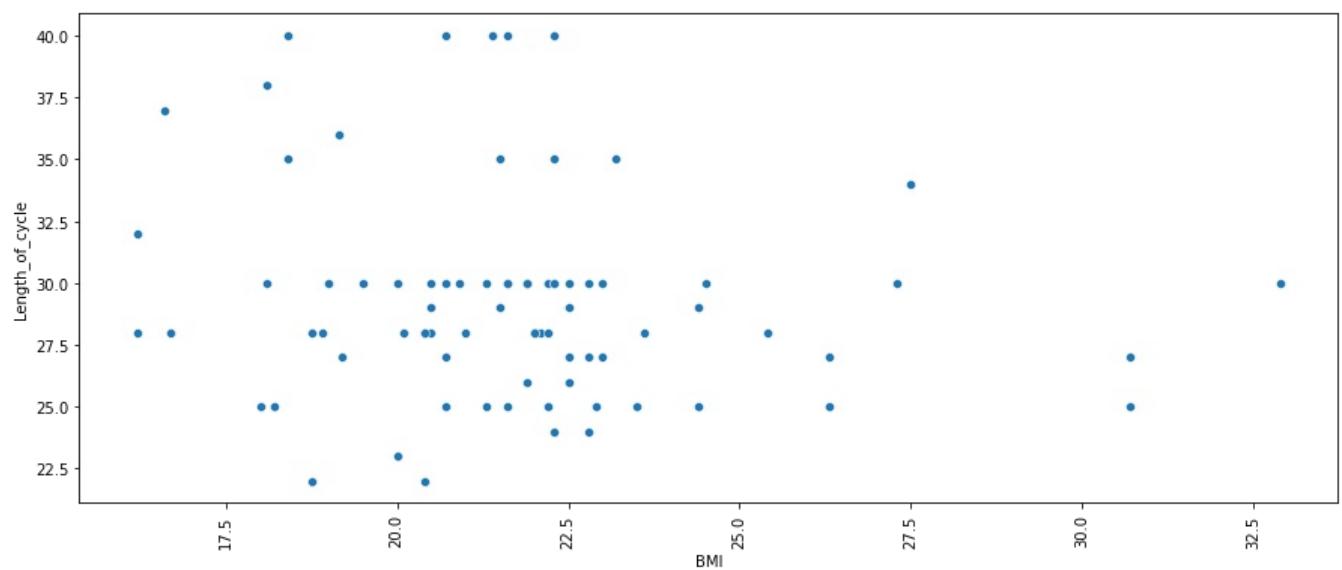
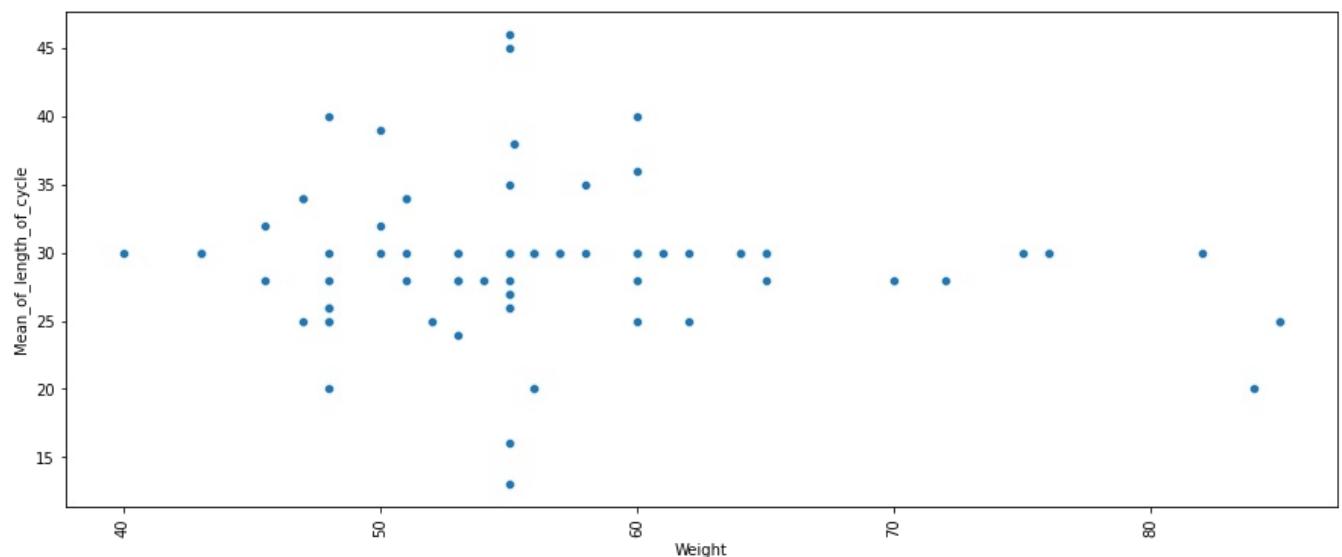


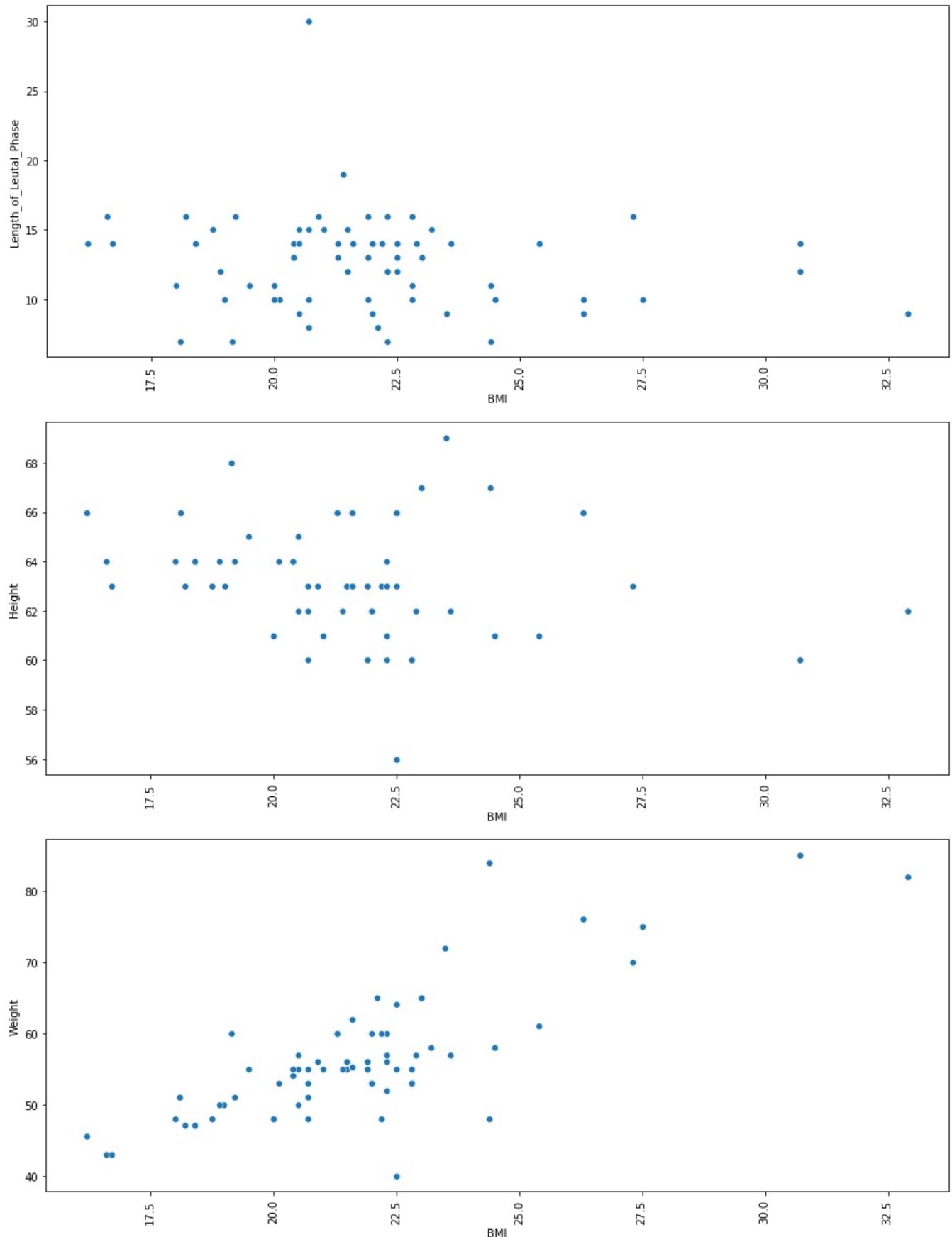


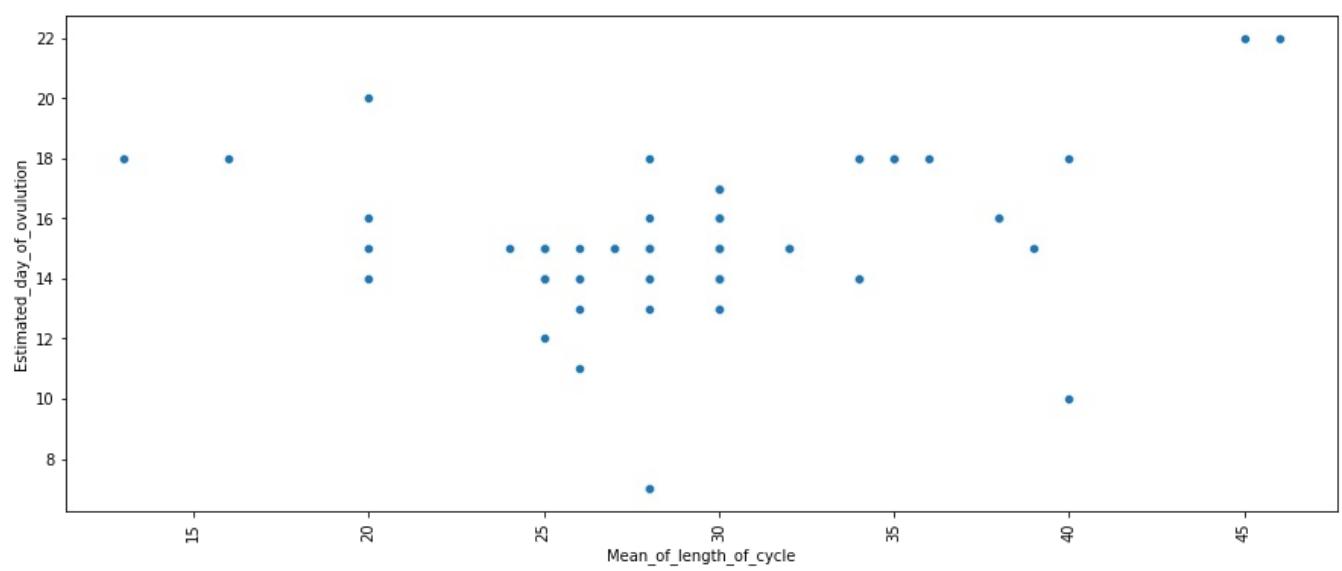
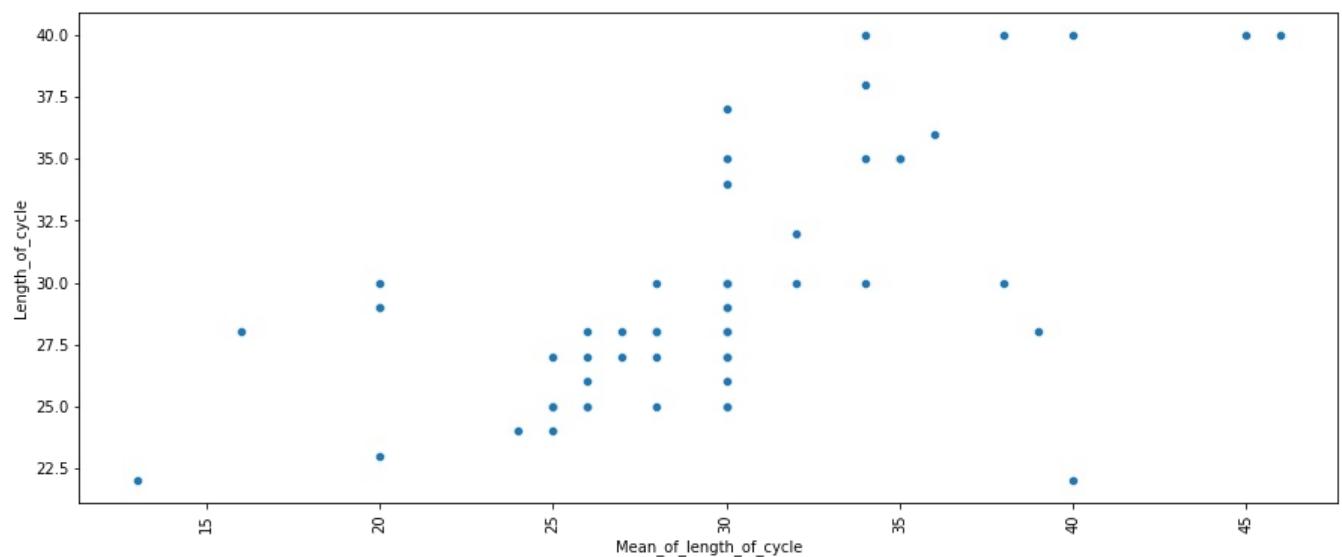
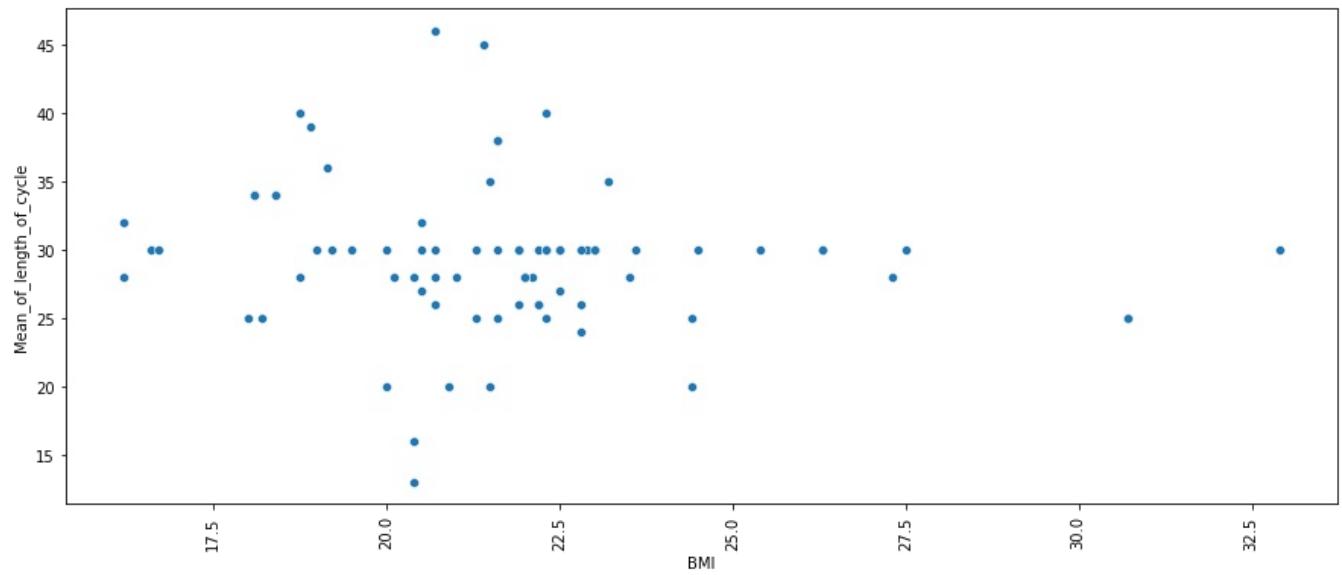


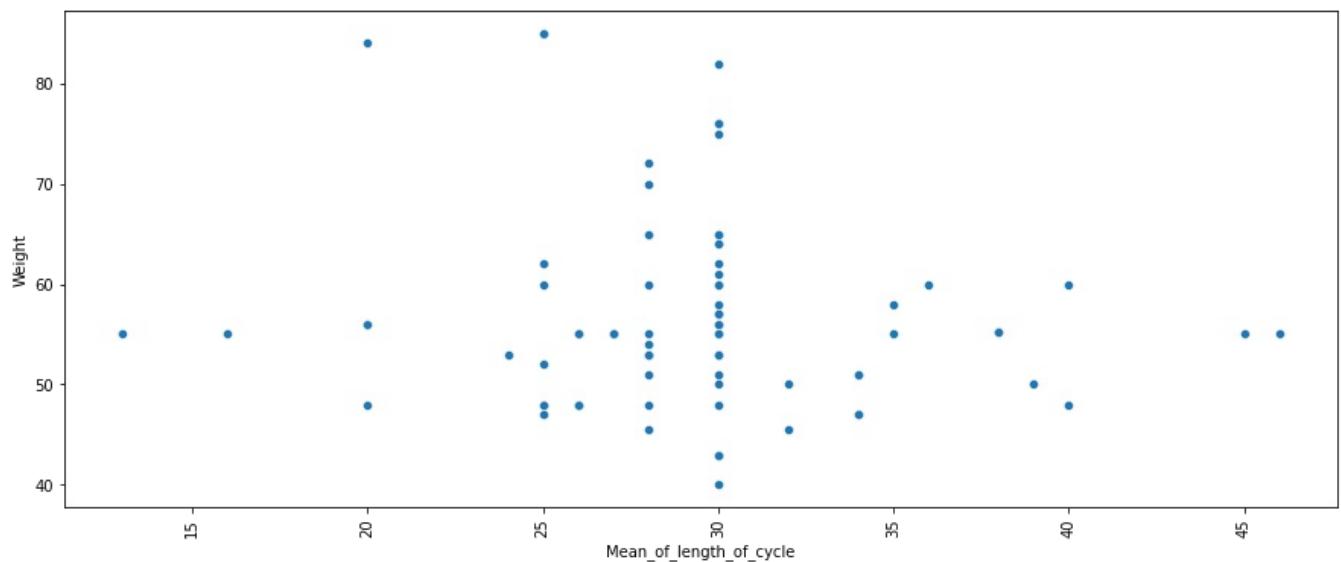
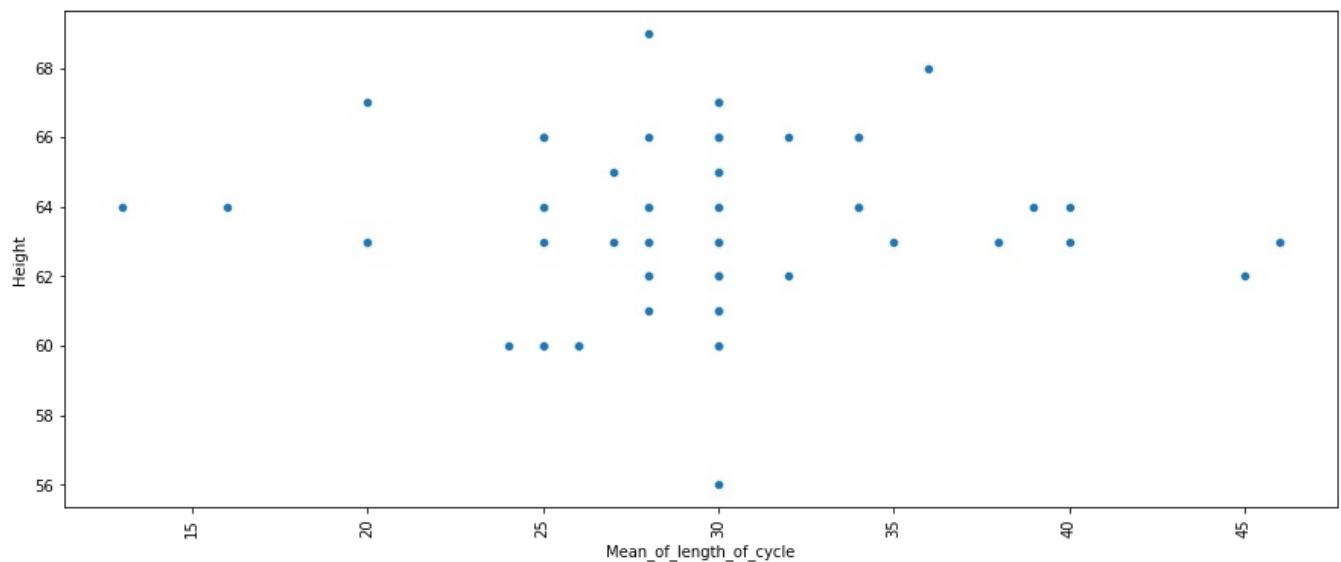
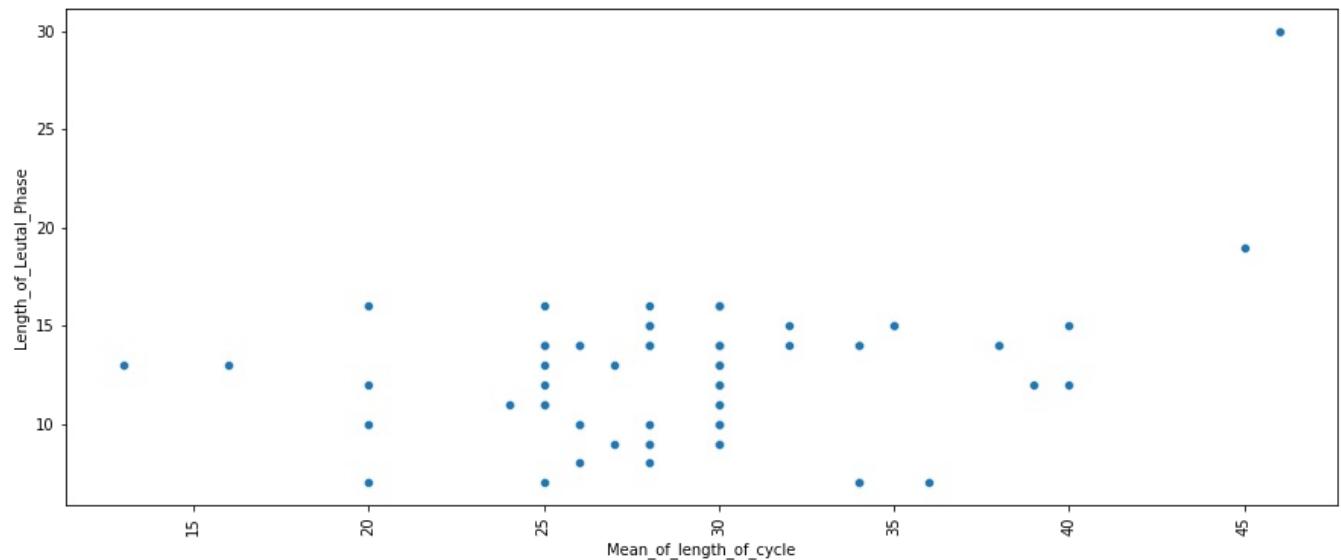


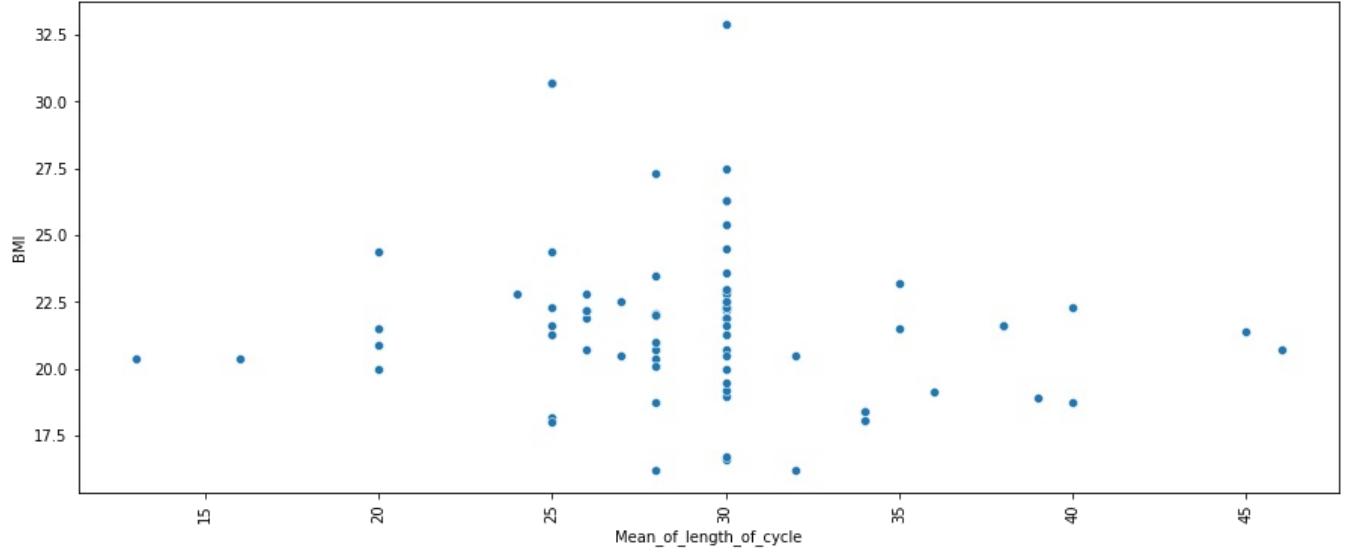




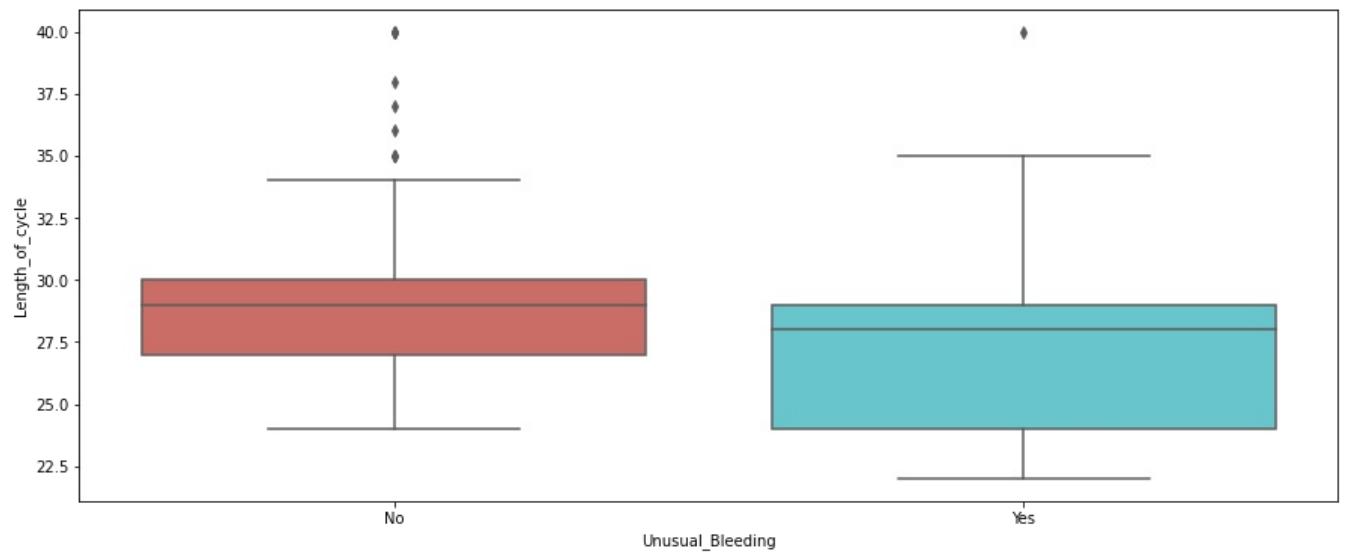


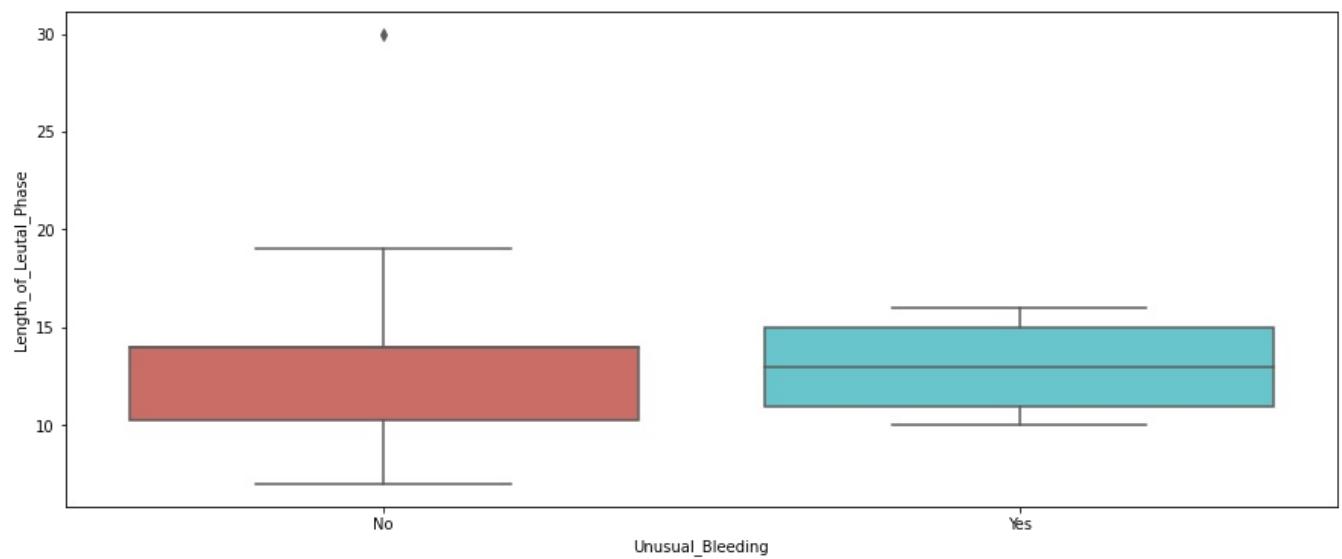
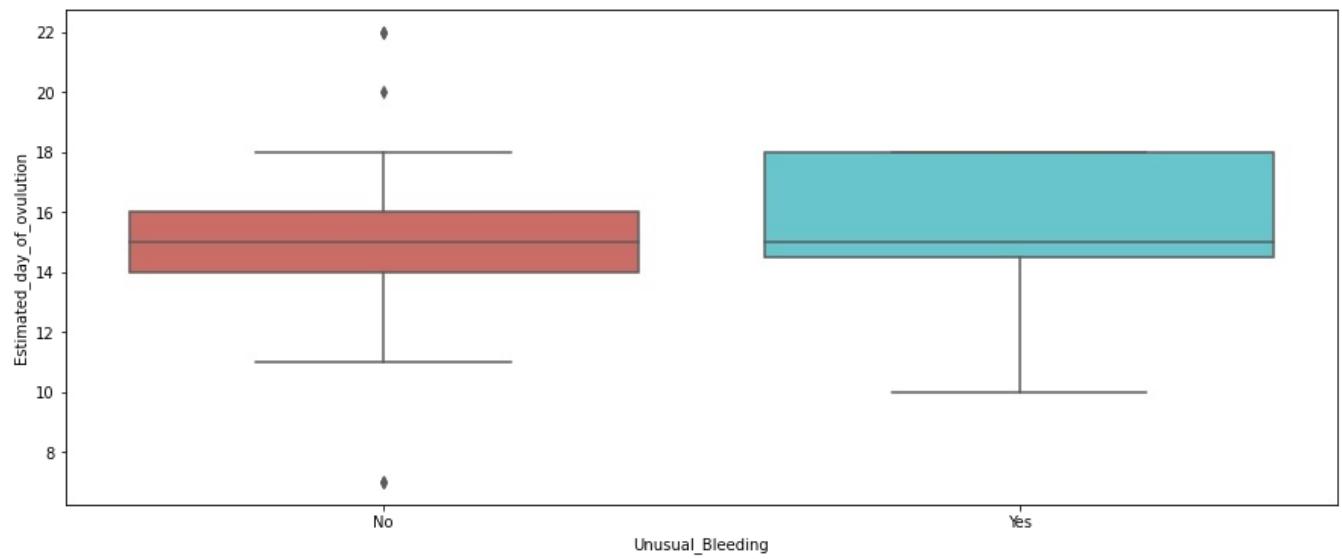


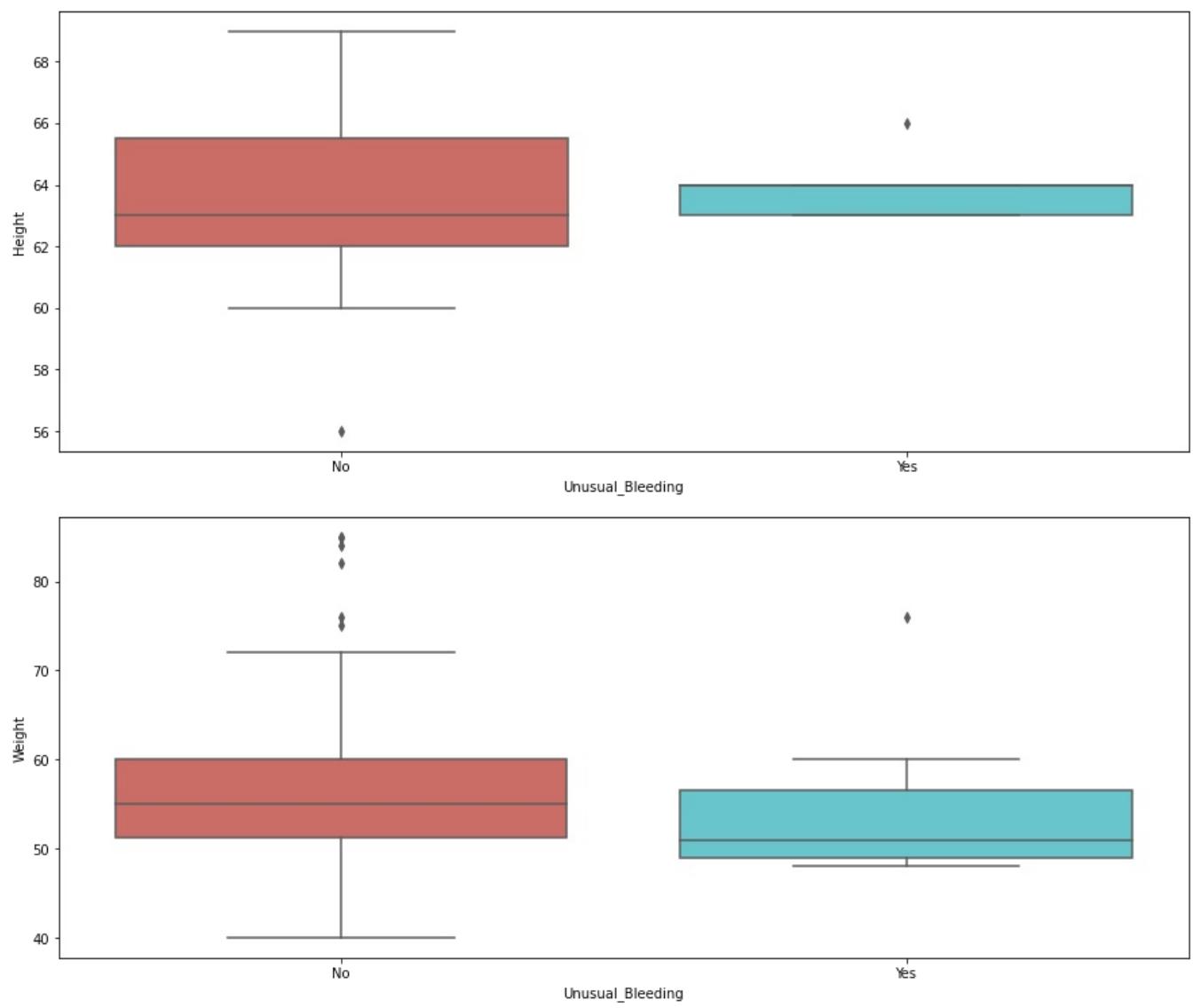


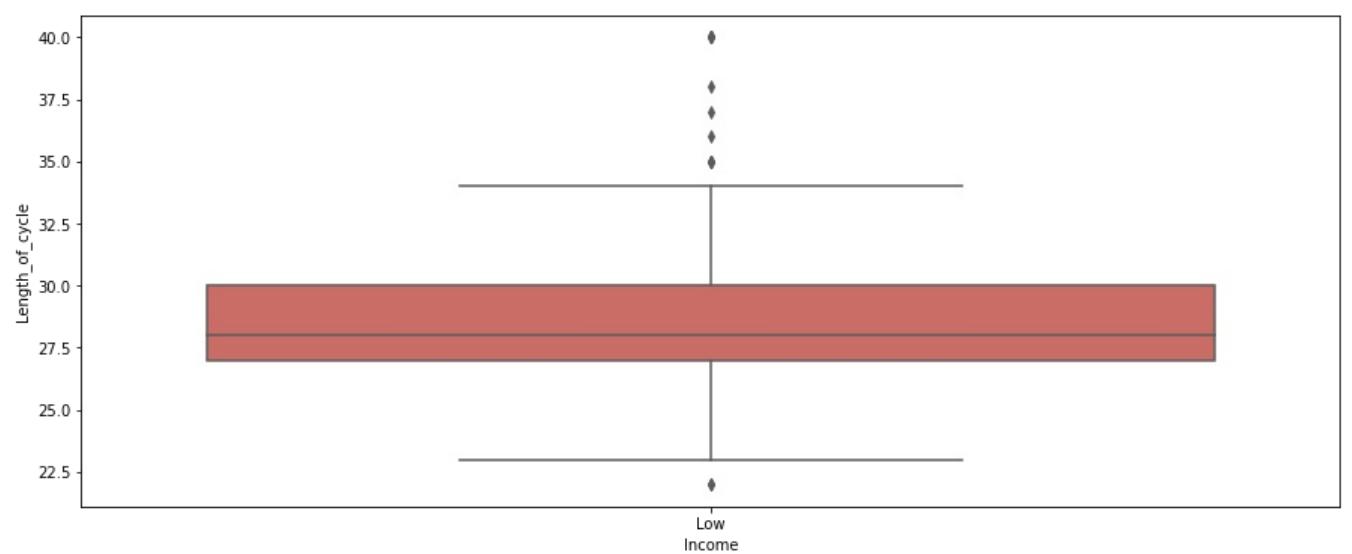
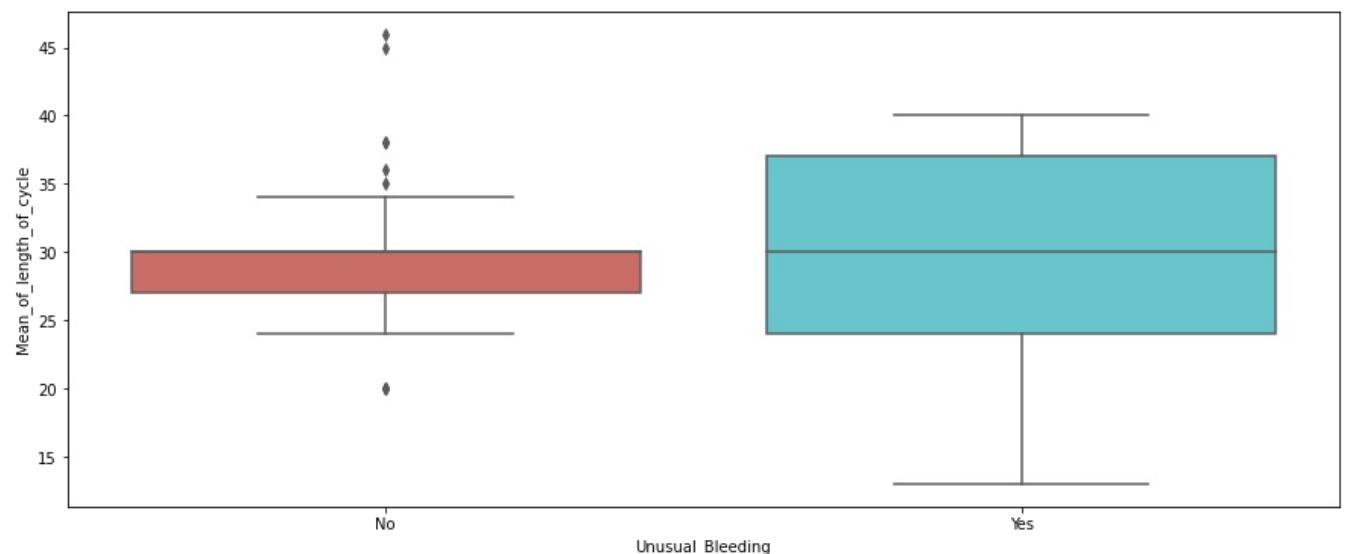
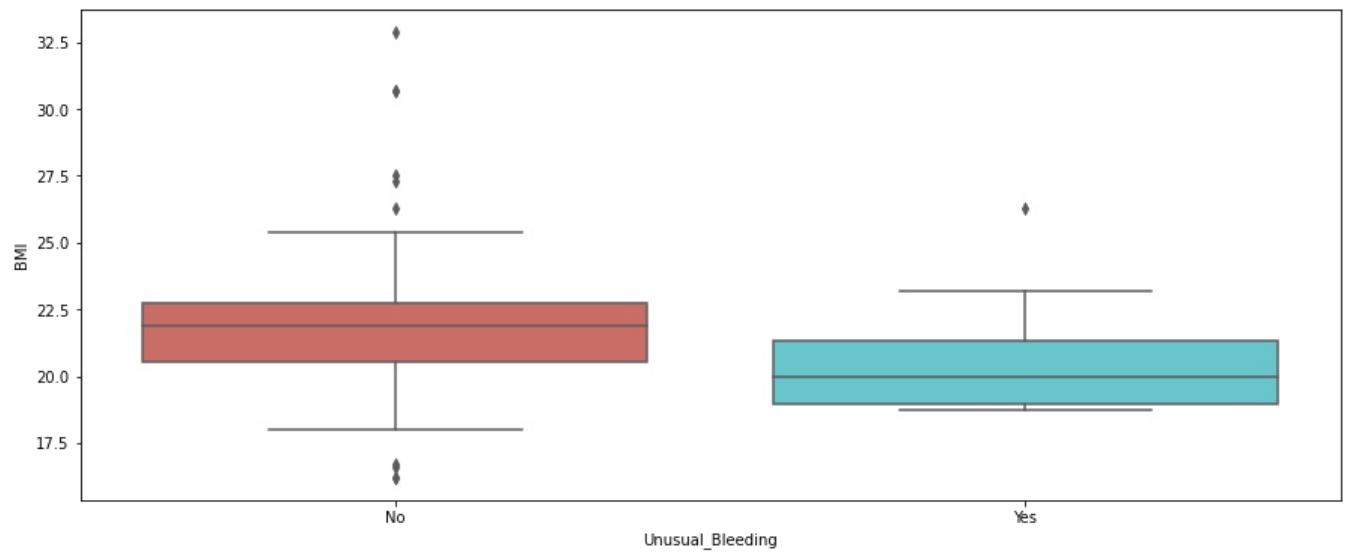


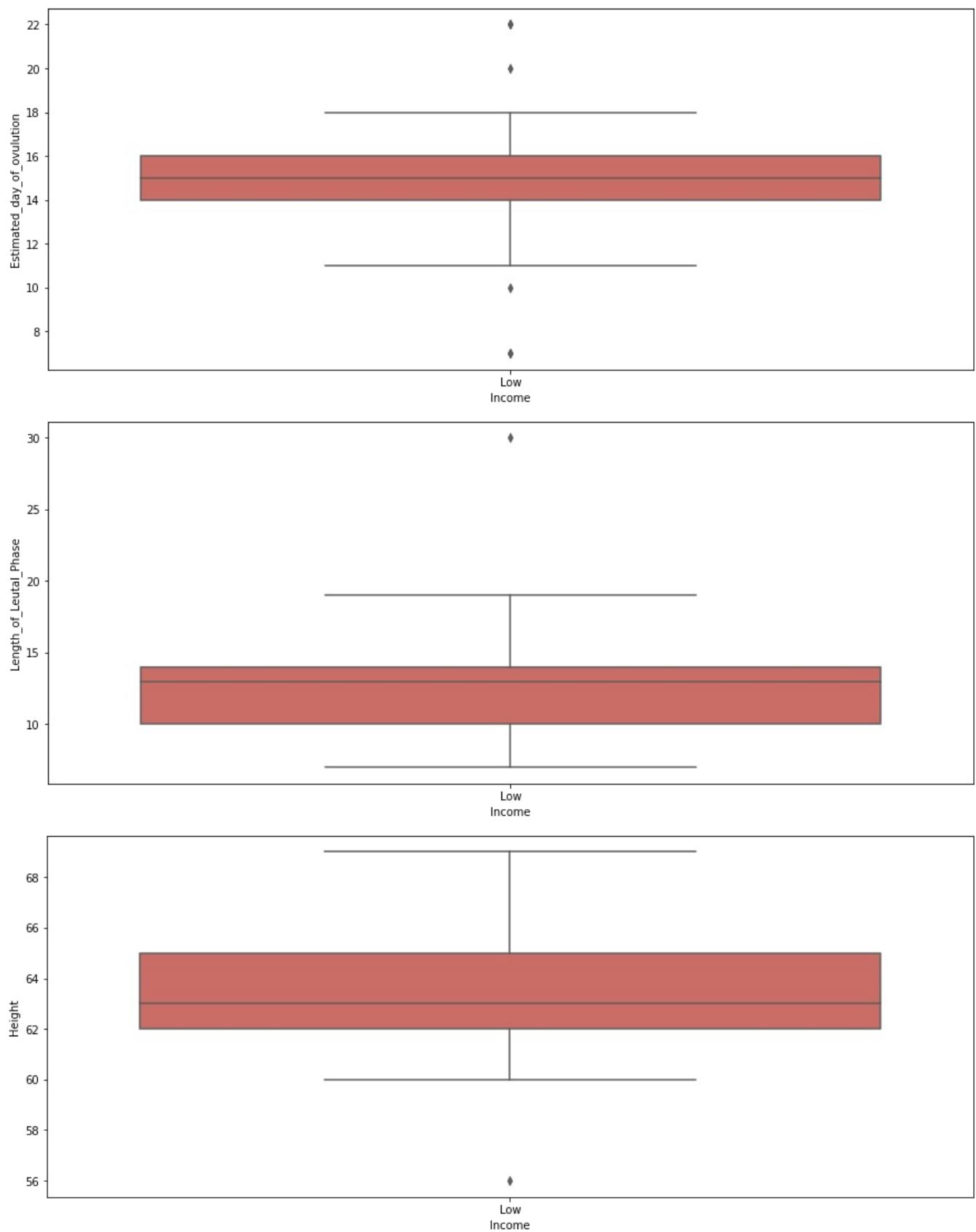
```
In [38]: for i in categorical:
    for j in continuous:
        plt.figure(figsize=(15,6))
        sns.boxplot(x = df[i], y = df[j], data = df, palette = 'hls')
        plt.show()
```

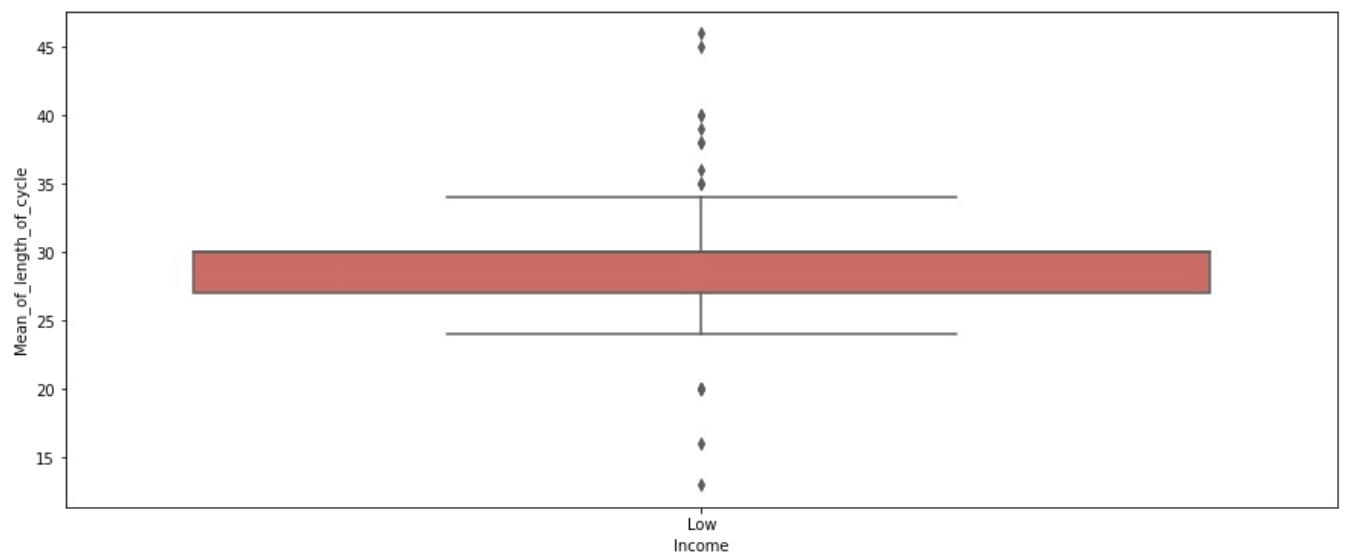
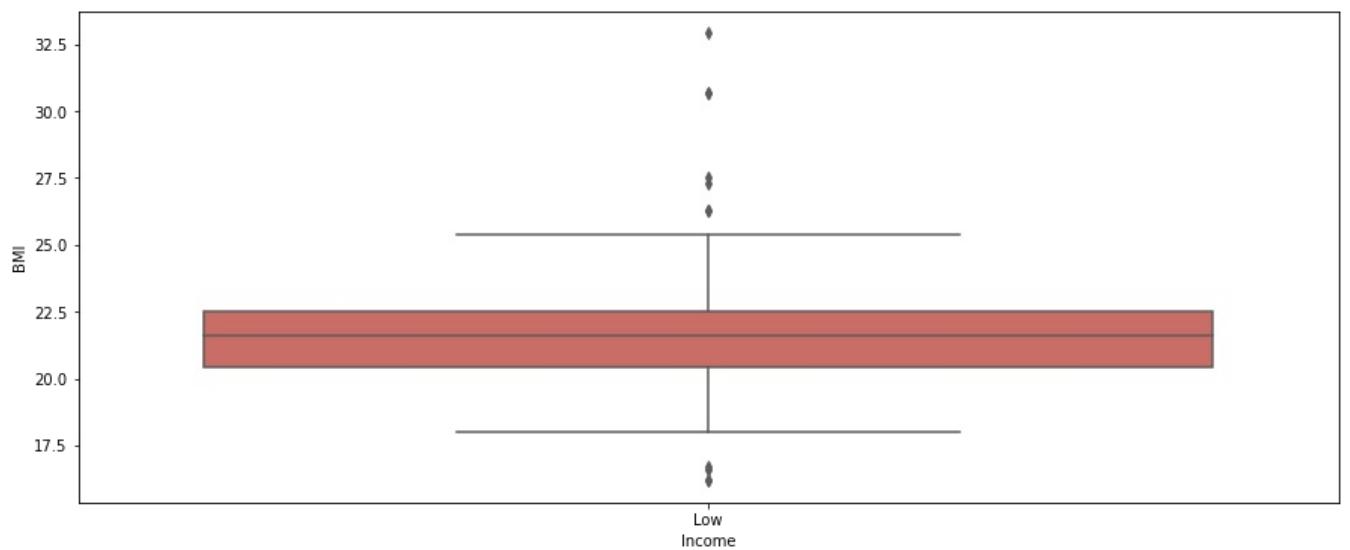
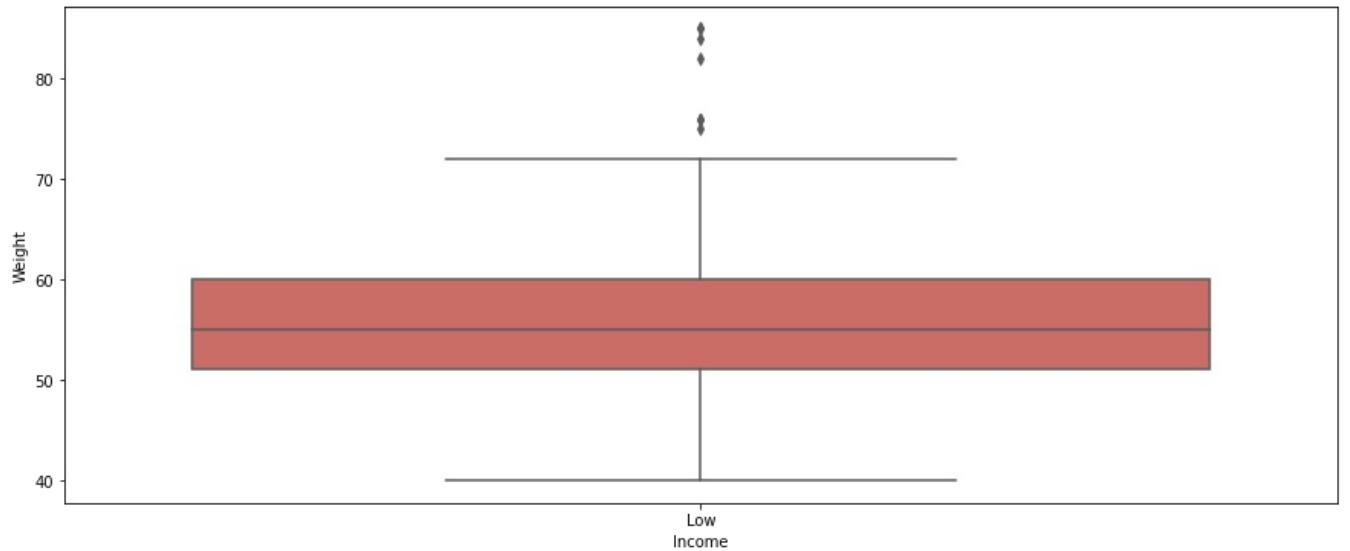


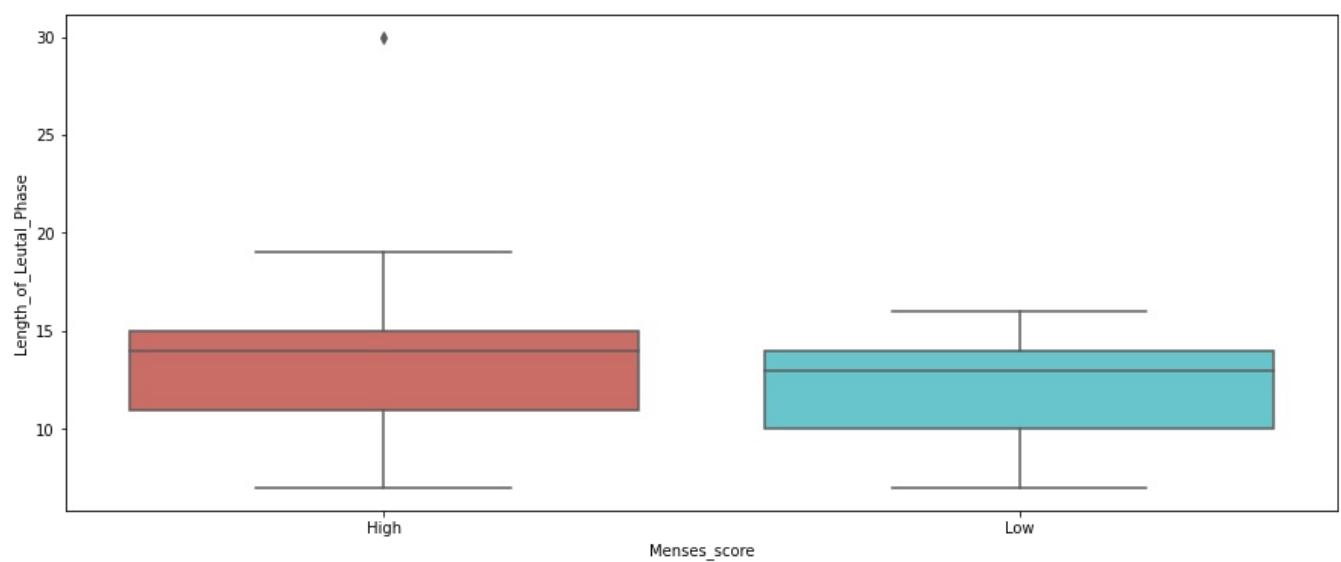
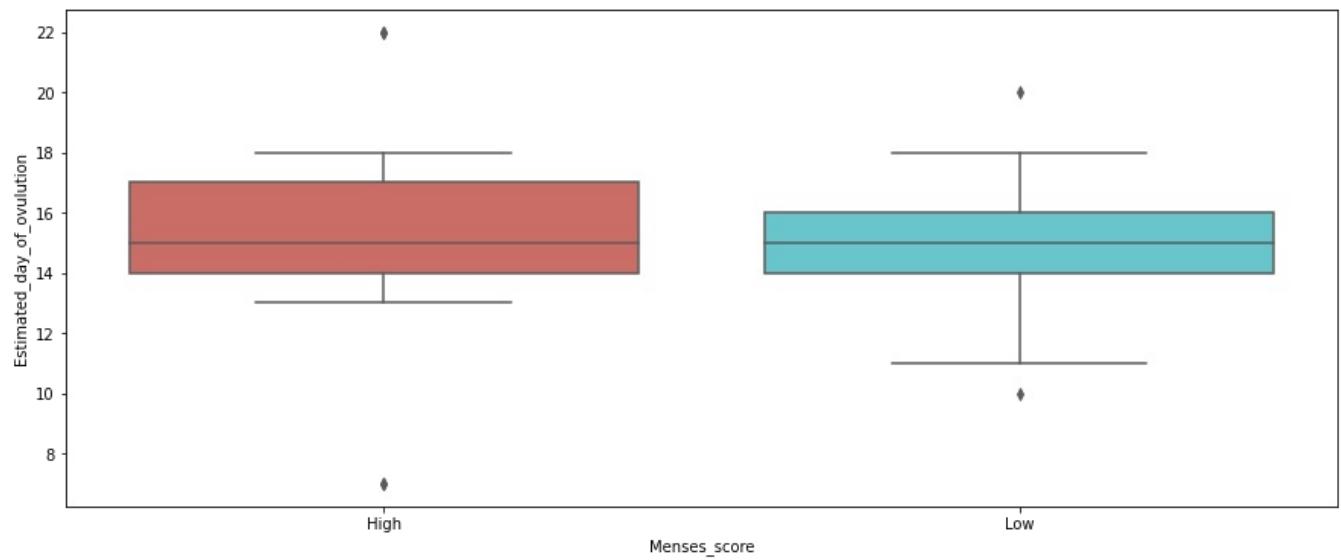
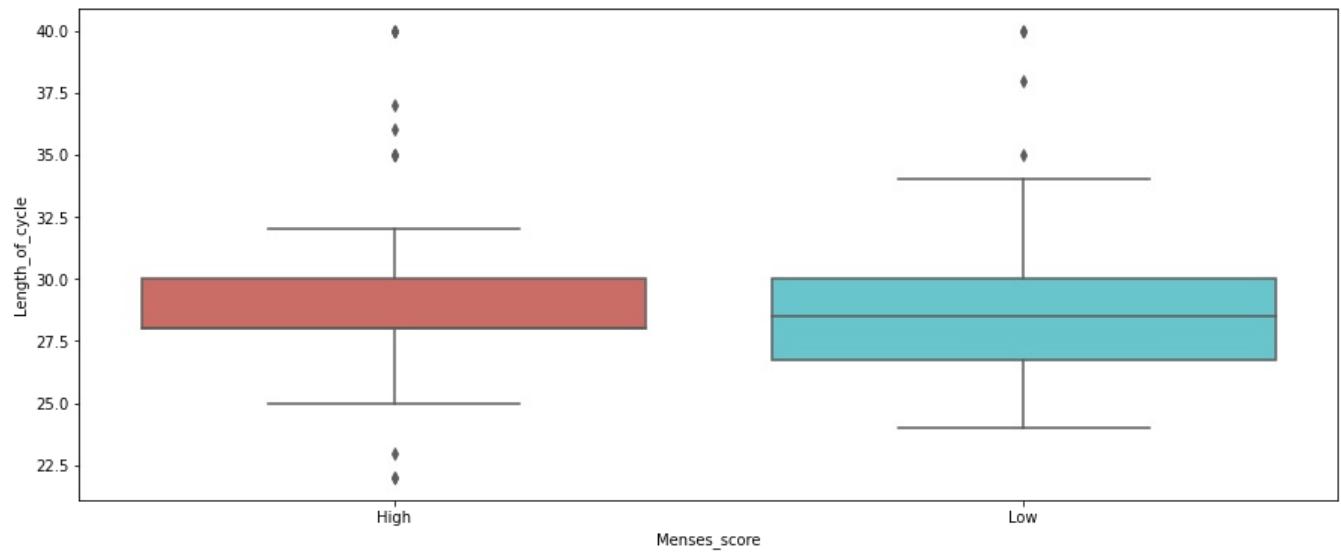


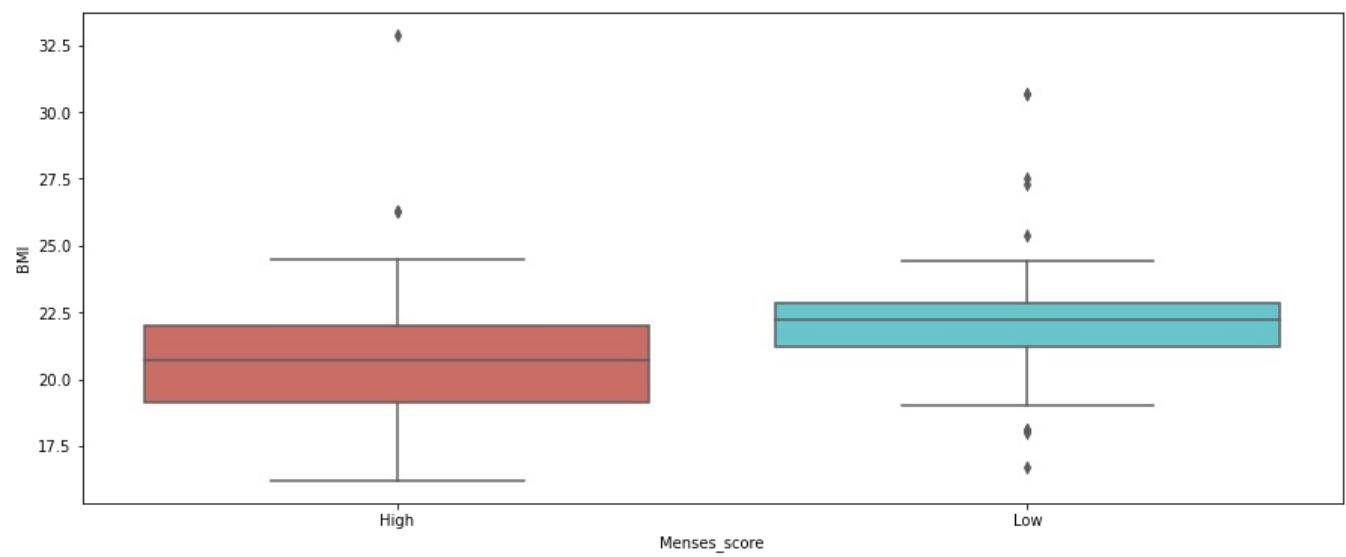
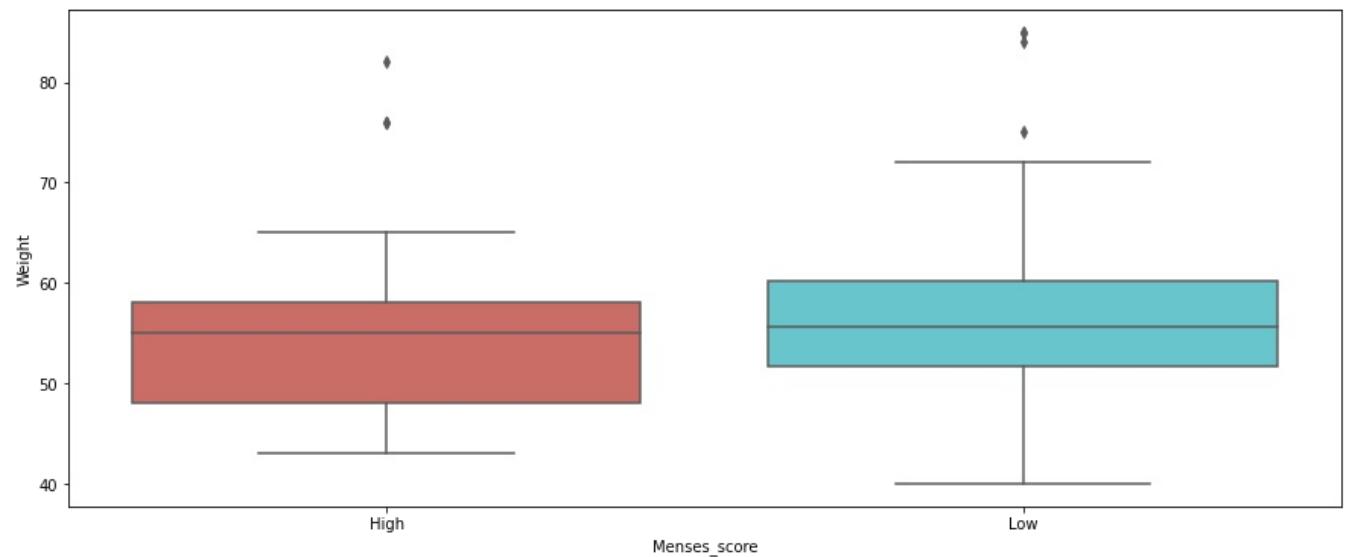
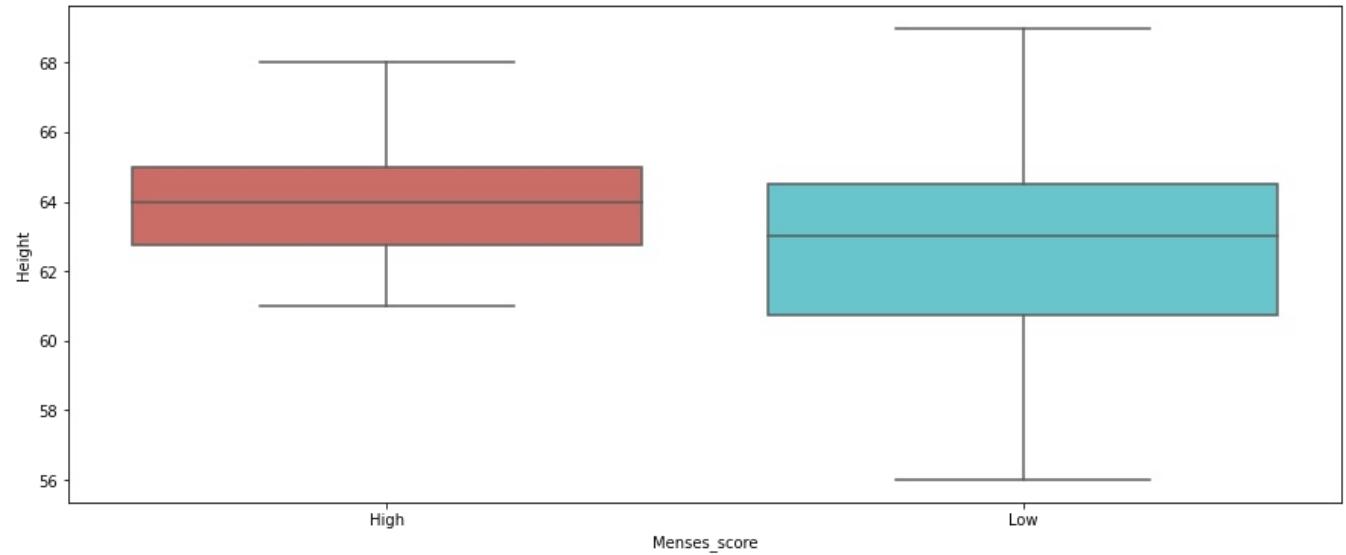


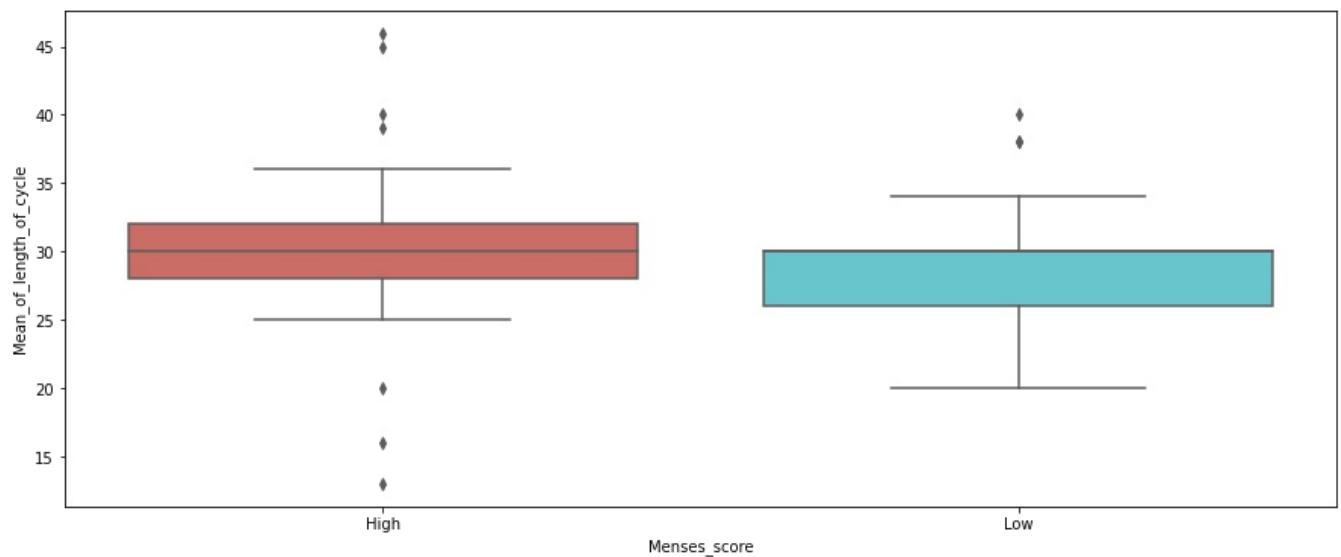




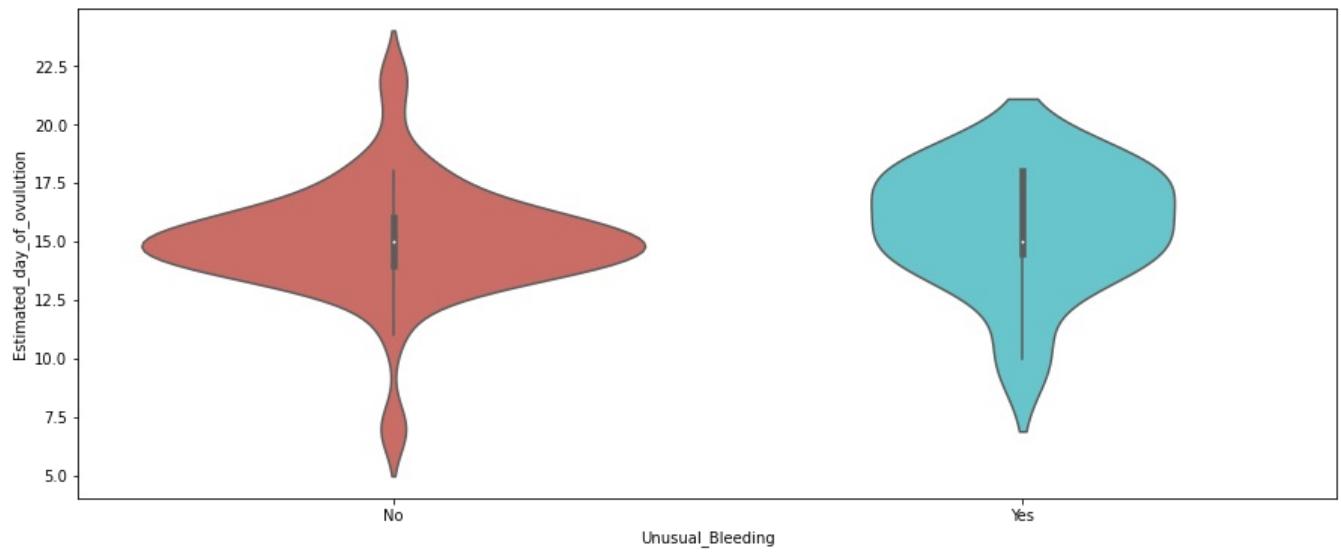
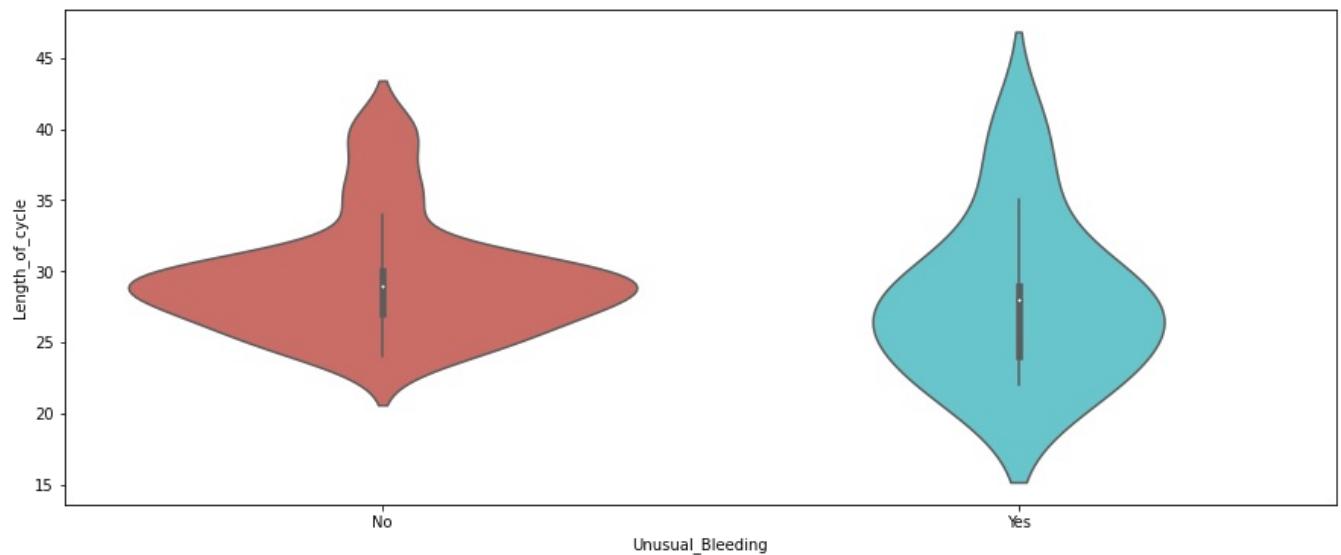


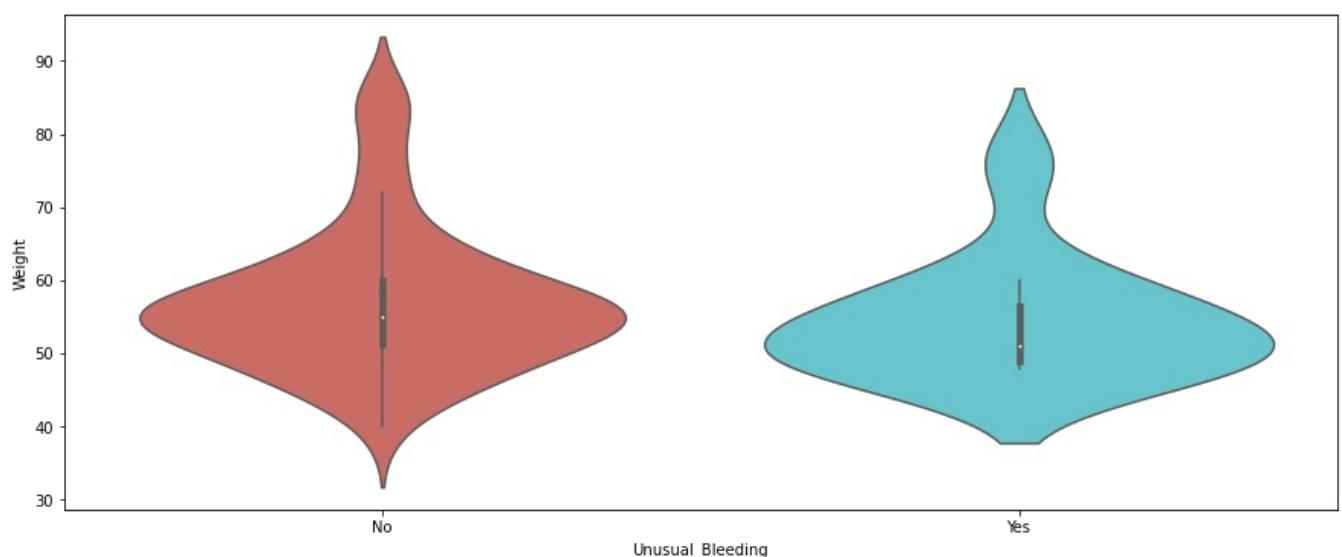
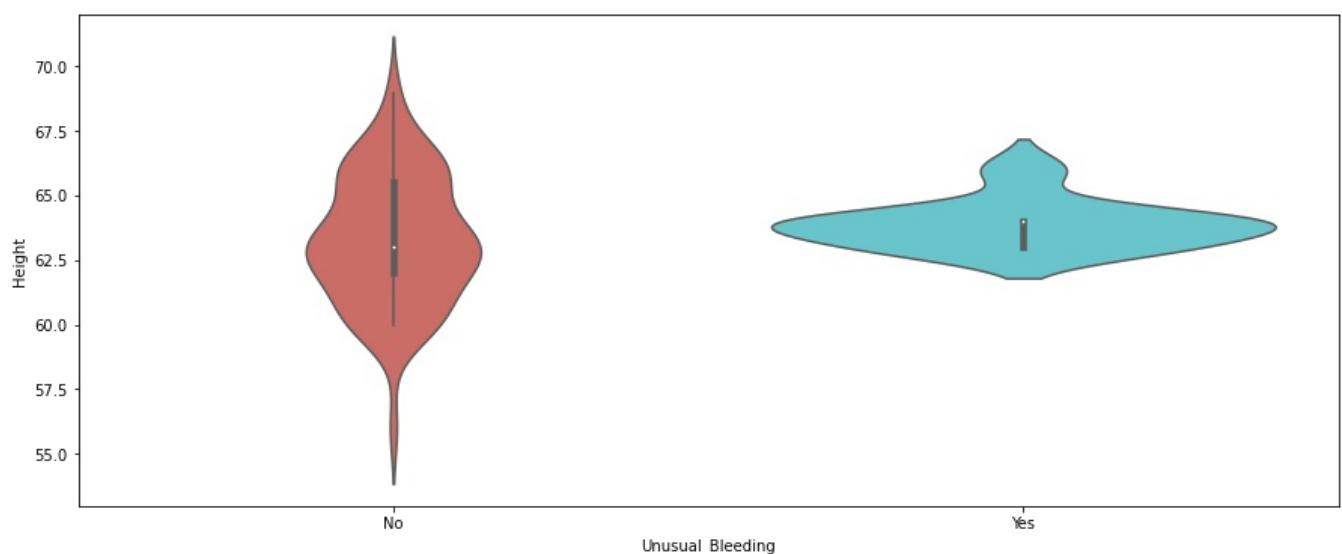
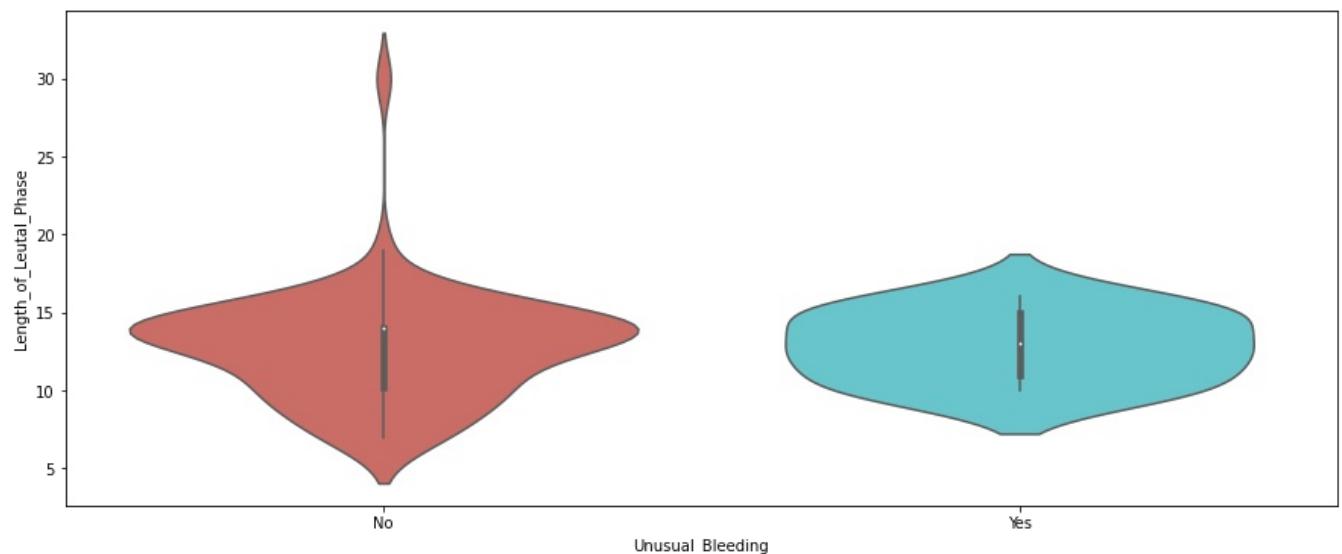


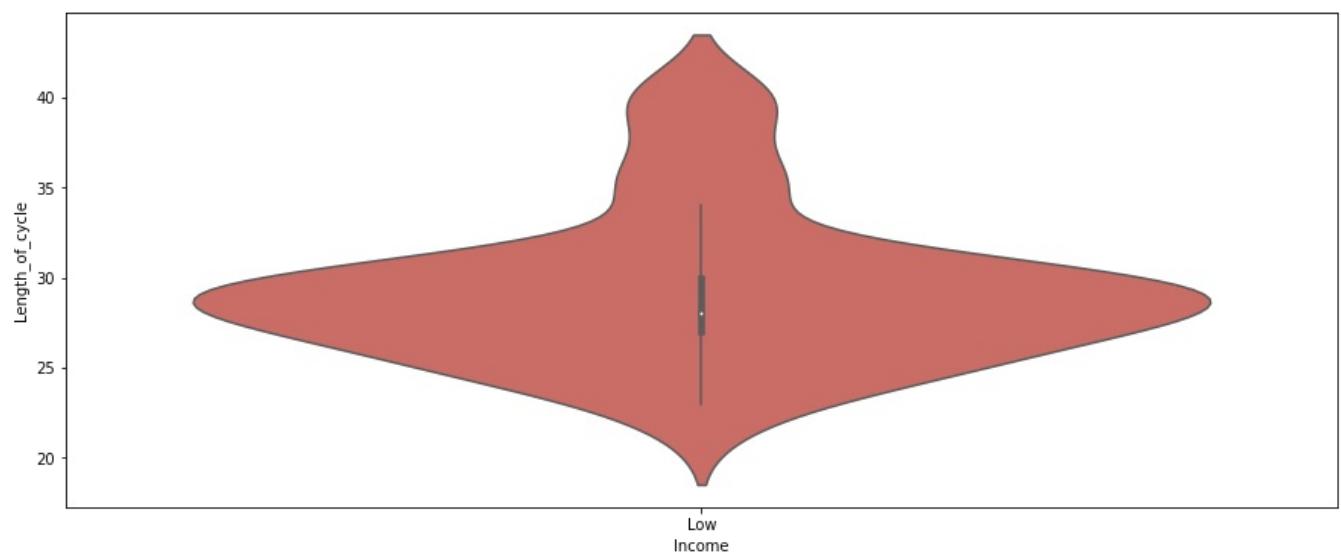
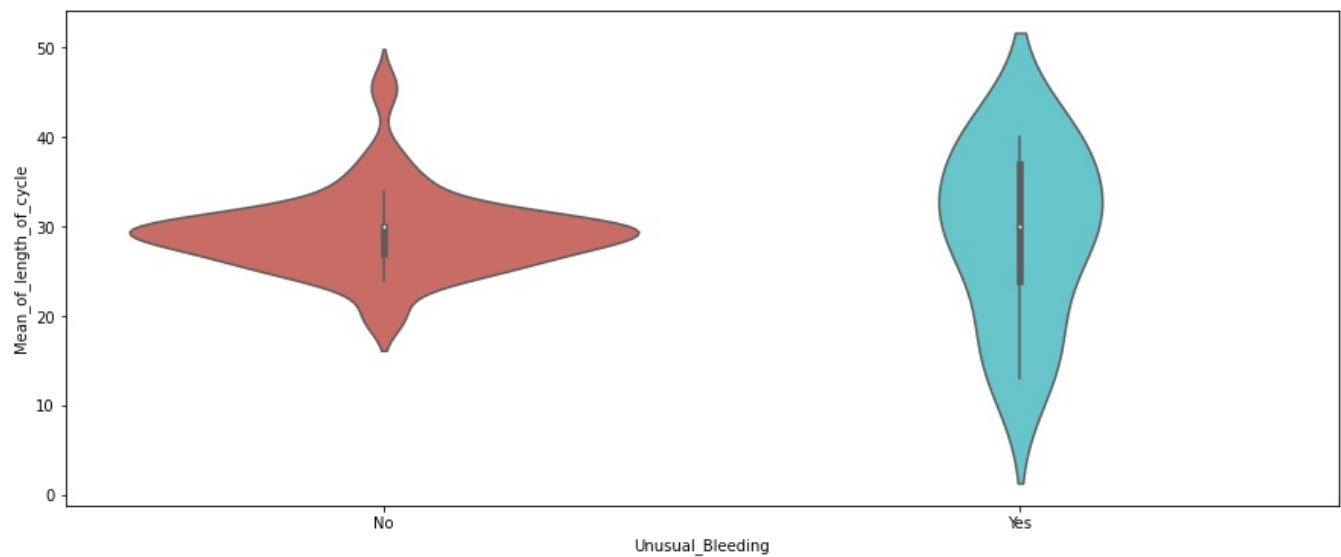
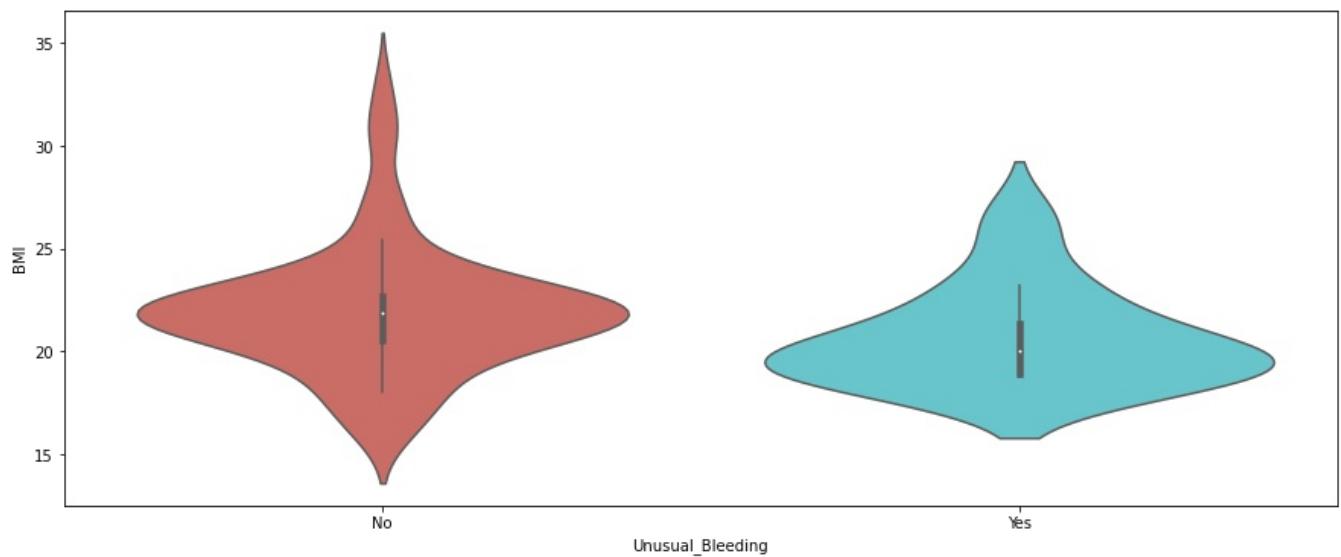


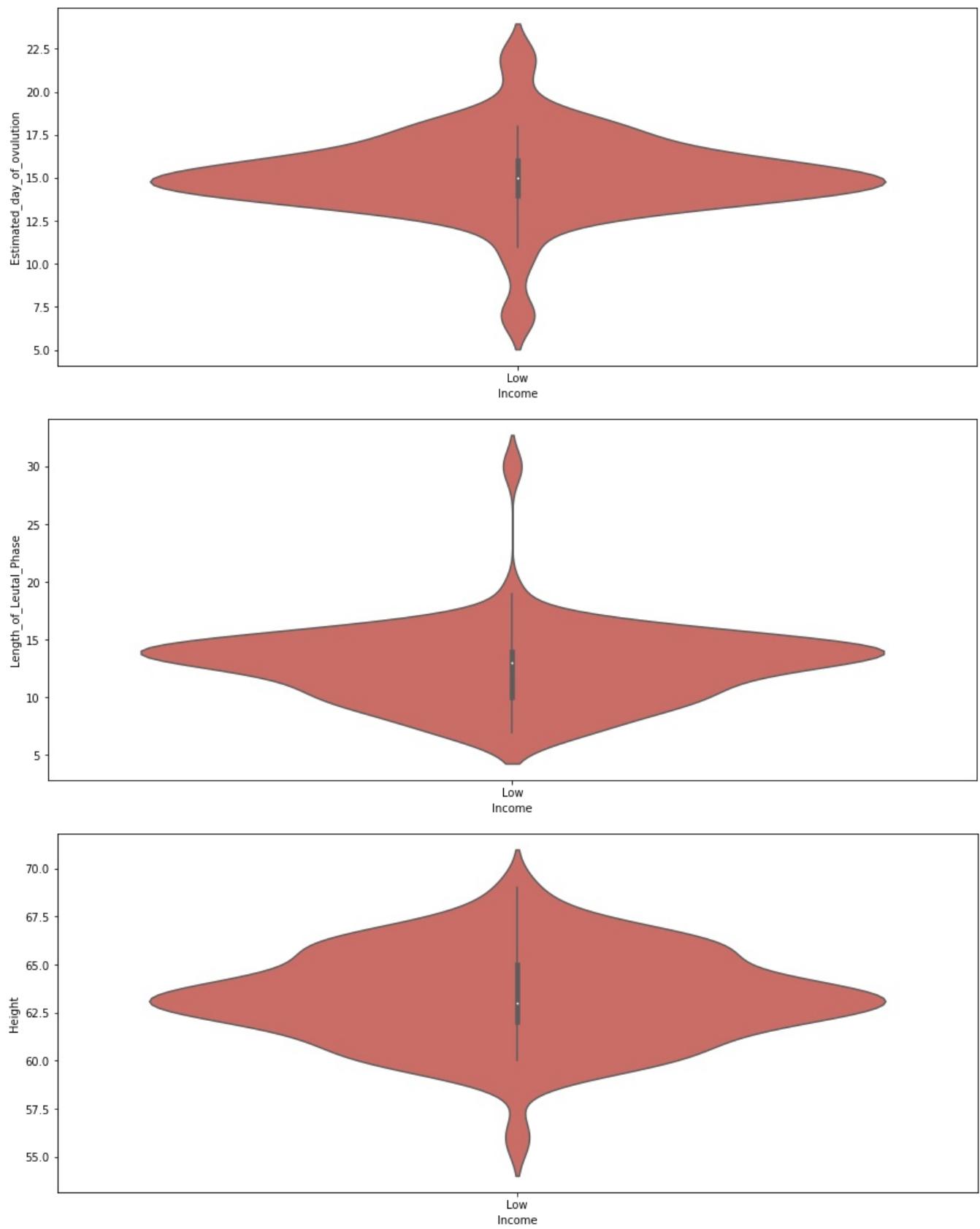


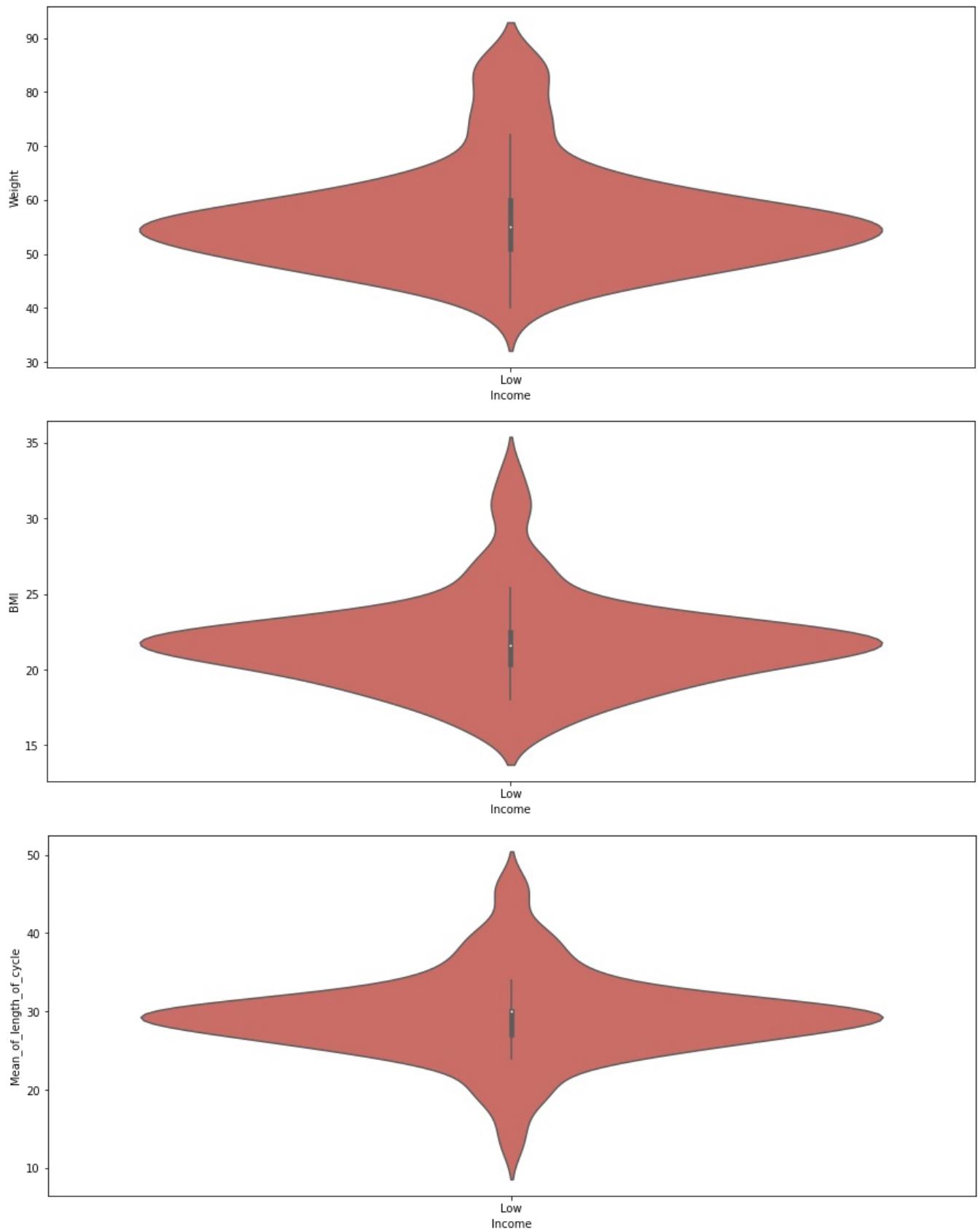
```
In [39]: for i in categorical:
    for j in continuous:
        plt.figure(figsize=(15,6))
        sns.violinplot(x = df[i], y = df[j], data = df, palette = 'hls')
        plt.show()
```

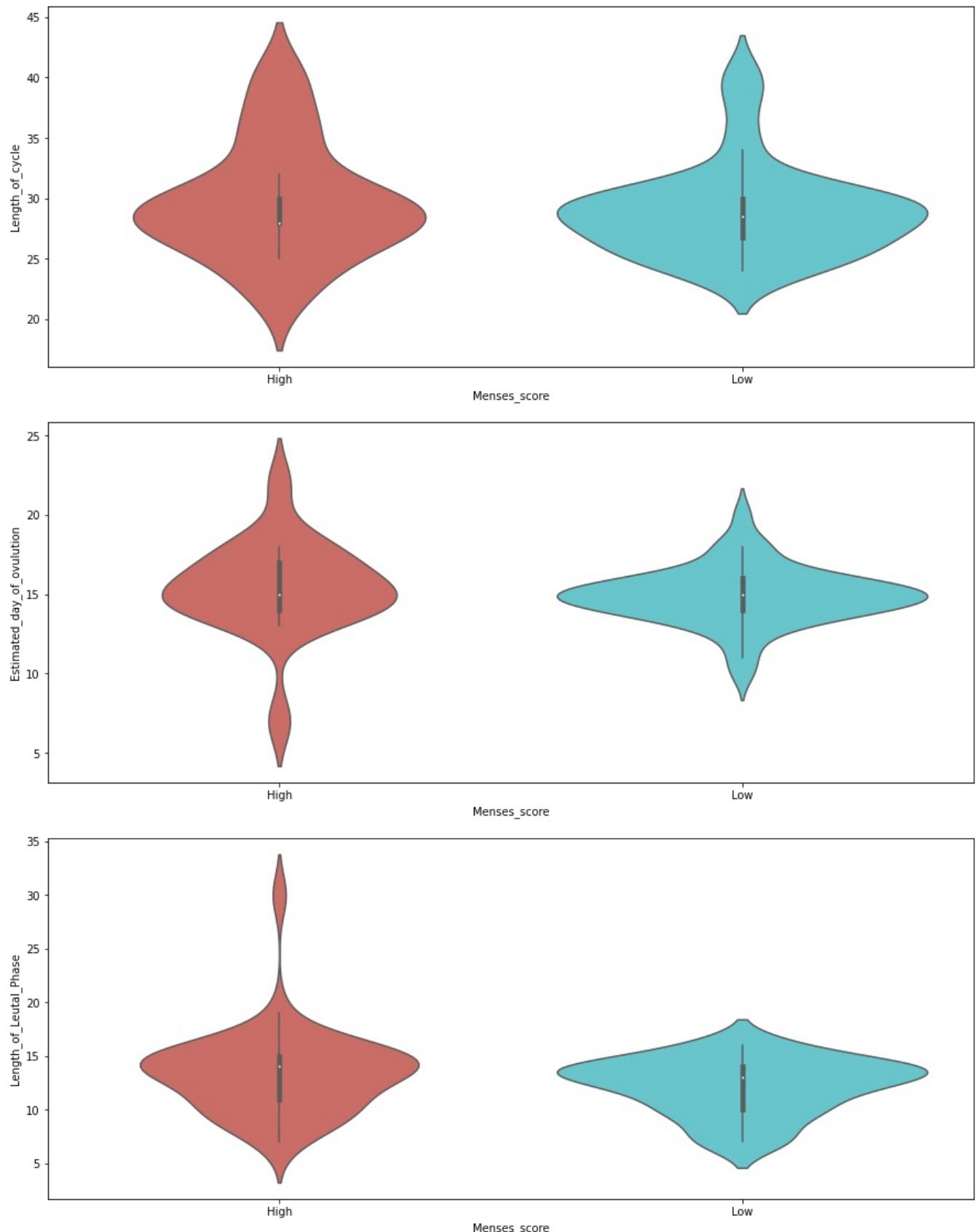


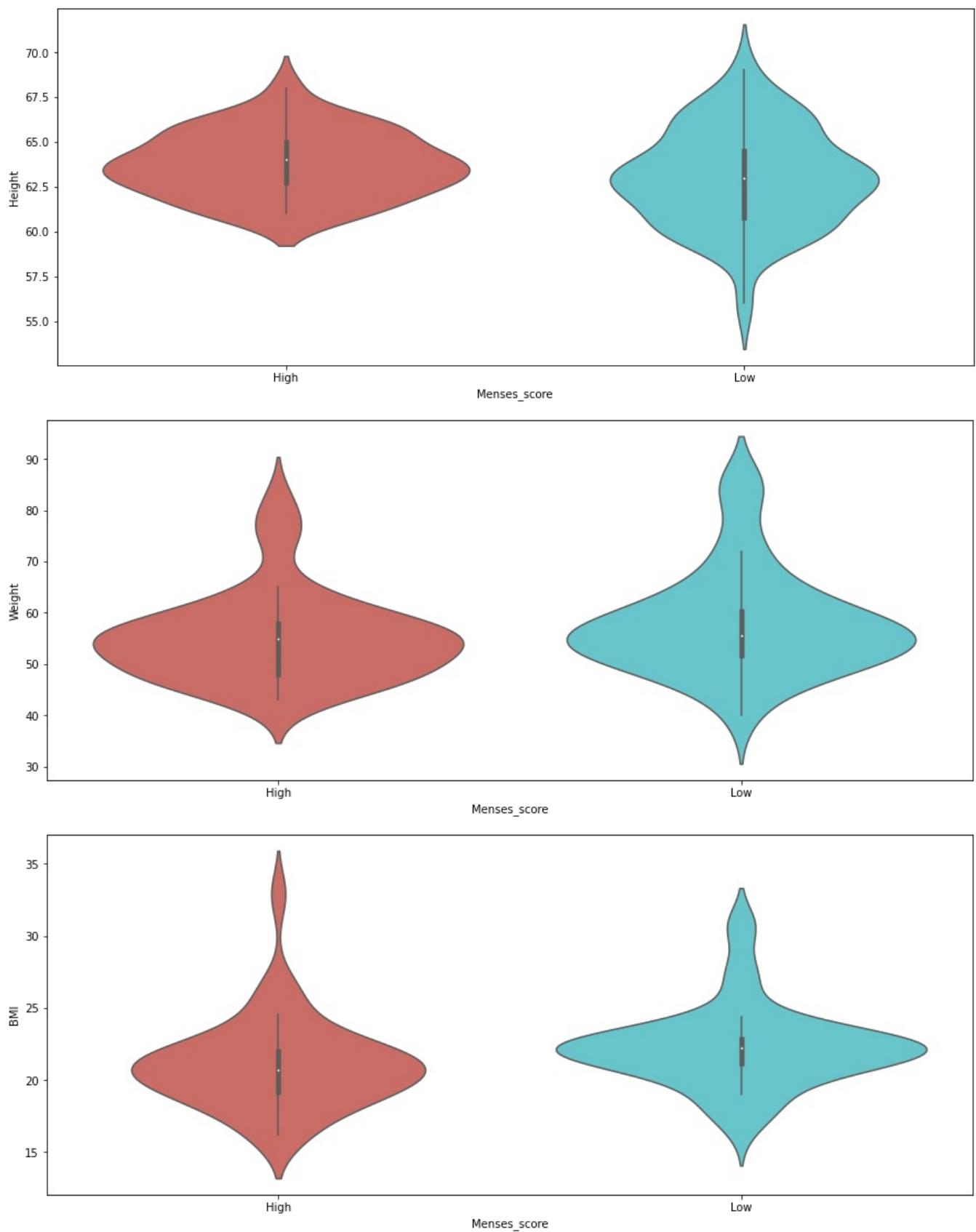


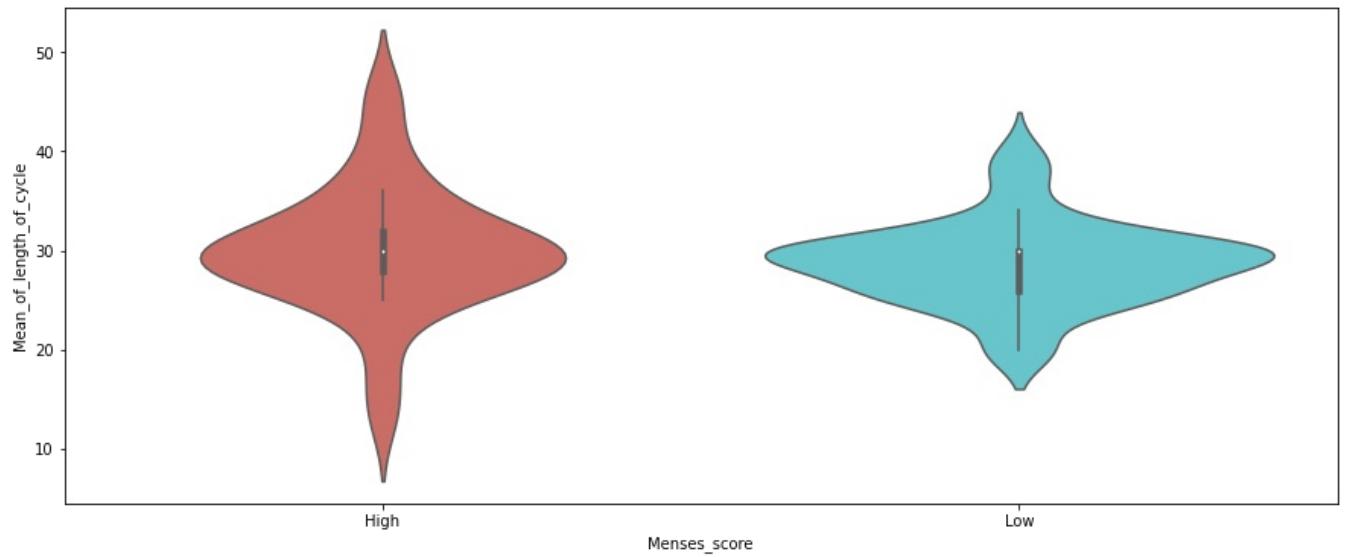












```
In [40]: pivot_categorical = pd.pivot_table(df, values=continuous, index=categorical,
                                         aggfunc='mean')
```

```
In [41]: pivot_categorical
```

```
Out[41]:
```

	BMI	Estimated_day_of_ovulation	Height	Length_of_Leutal_Phase	Length_of_cycle	Mean	
Unusual_Bleeding	Income	Menses_score					
No	Low	High	21.132759	15.103448	63.692308	13.551724	30.655172
		Low	22.458537	15.024390	62.945946	12.170732	28.658537
Yes	Low	High	20.837500	16.500000	64.000000	12.875000	26.375000
		Low	20.166667	13.333333	63.666667	12.666667	32.333333

```
In [42]: cross_tab_discrete = pd.crosstab(index=df['Unusual_Bleeding'],
                                         columns=df['number_of_peak'])
```

```
In [43]: cross_tab_discrete
```

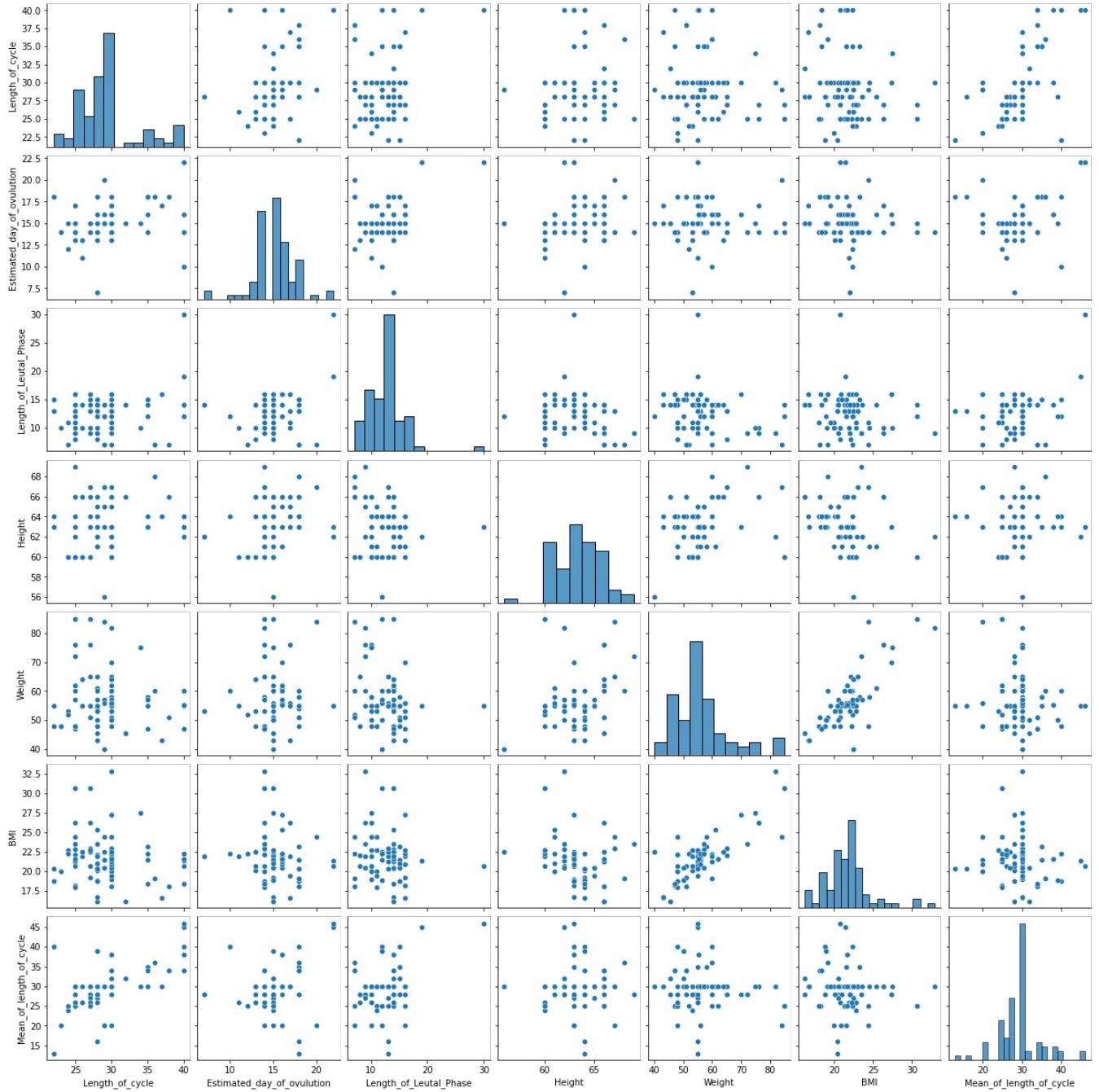
```
Out[43]:
```

number_of_peak	1	2	3	4	5
Unusual_Bleeding					

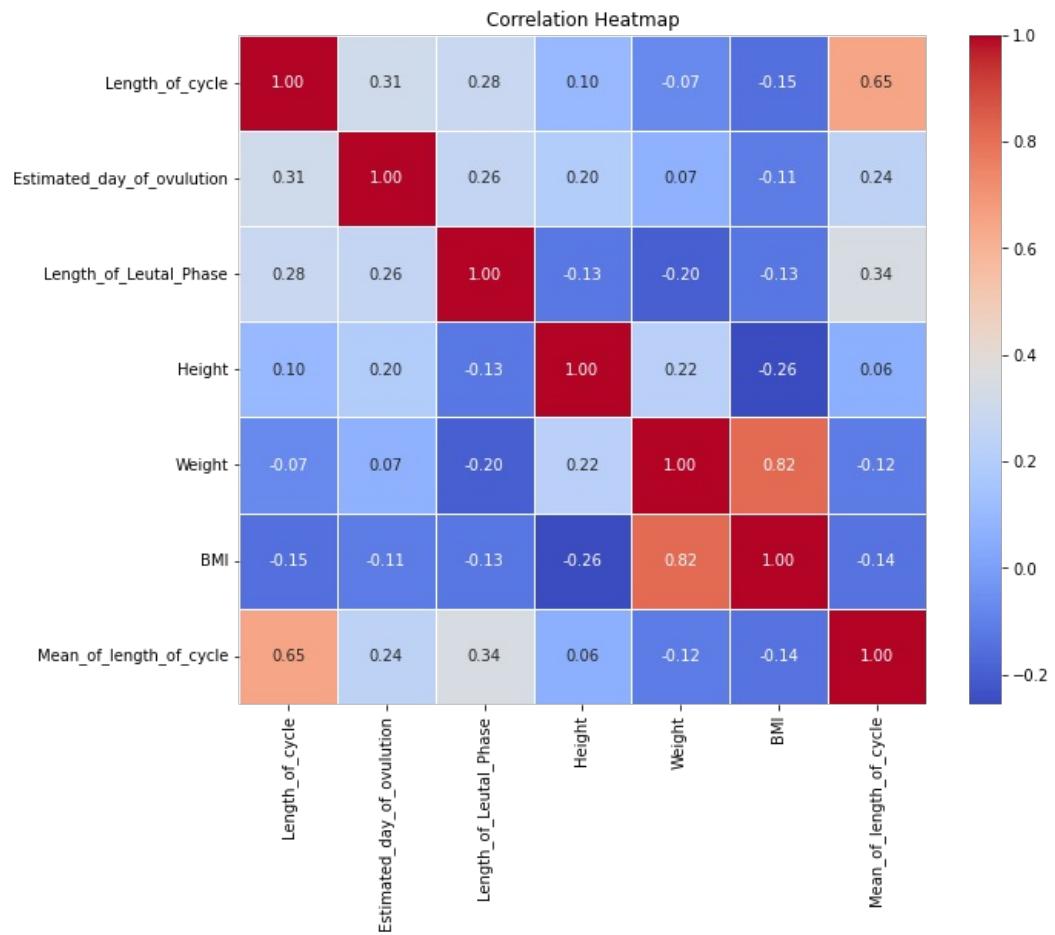
No	3	35	30	2	0
Yes	0	4	4	2	1

```
In [44]: plt.figure(figsize=(15,6))
sns.pairplot(df[continuous])
plt.show()
```

```
<Figure size 1080x432 with 0 Axes>
```



```
In [45]: plt.figure(figsize=(10, 8))
sns.heatmap(df[continuous].corr(), annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)
plt.title('Correlation Heatmap')
plt.show()
```



Thanks !!!

Copyright@ Prof. Nirmal Gaud

Loading [MathJax]/extensions/Safe.js