In [1]:
```
# This Python 3 environment comes with many helpful analytics libraries instal
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/d
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will lis

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that ge
# You can also write temporary files to /kaggle/temp/, but they won't be saved
```

```
/kaggle/input/weight-height/weight-height.csv
```

In [8]:
```
import seaborn as sn
```

```
/opt/conda/lib/python3.10/site-packages/scipy/__init__.py:146: UserWarning: A
NumPy version >=1.16.5 and <1.23.0 is required for this version of SciPy (det
ected version 1.23.5
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [3]:
```
#reading dataset from csv file and storing it
df=pd.read_csv("//kaggle//input//weight-height//weight-height.csv")
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Gender  10000 non-null  object
 1   Height  10000 non-null  float64
 2   Weight  10000 non-null  float64
dtypes: float64(2), object(1)
memory usage: 234.5+ KB
```

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

Out[5]:

	Height	Weight
count	10000.000000	10000.000000
mean	66.367560	161.440357
std	3.847528	32.108439
min	54.263133	64.700127
25%	63.505620	135.818051
50%	66.318070	161.212928
75%	69.174262	187.169525
max	78.998742	269.989699

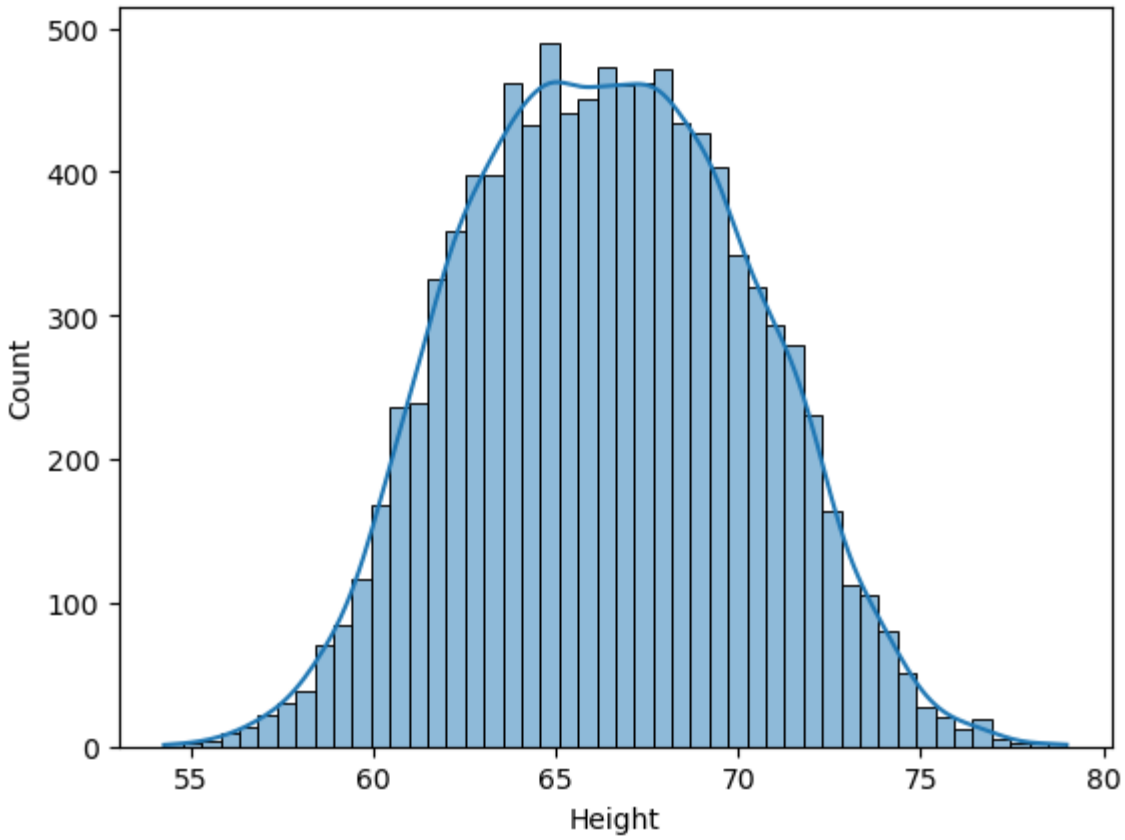
In [7]:
```
df.head()
```

Out[7]:

	Gender	Height	Weight
0	Male	73.847017	241.893563
1	Male	68.781904	162.310473
2	Male	74.110105	212.740856
3	Male	71.730978	220.042470
4	Male	69.881796	206.349801

In [10]:
```
sn.histplot(df["Height"],kde=True)
```

Out[10]:
```
<Axes: xlabel='Height', ylabel='Count'>
```



In [12]:
```
mean_height=df["Height"].mean()
mean_height
```

Out[12]:
```
66.36755975482124
```

In [14]:
```
standard_dev_height=df["Height"].std()
standard_dev_height
```

Out[14]:
```
3.8475281207732293
```

In [20]:
```
upper_limit_height=mean_height+3*standard_dev_height #upper limit is equal to
lower_limit_height=mean_height-3*standard_dev_height #Lower limit is equal to
print("upper_limit_height",upper_limit,"lower_limit_height",lower_limit_height
```

```
upper_limit_height 77.91014411714093 lower_limit_height 54.82497539250156
```

In [21]:
```
#getting rows for height lower than lower lower_limit_height
df[df["Height"]<lower_limit_height]
```

Out[21]:

	Gender	Height	Weight
6624	Female	54.616858	71.393749
9285	Female	54.263133	64.700127

In [22]:
```
#getting rows for height greater than upper_limit_height
df[df["Height"]>upper_limit_height]
```

Out[22]:

	Gender	Height	Weight
994	Male	78.095867	255.690835
1317	Male	78.462053	227.342565
2014	Male	78.998742	269.989699
3285	Male	78.528210	253.889004
3757	Male	78.621374	245.733783

In [24]:
```
df[(df["Height"]>upper_limit_height) | (df["Height"]<lower_limit_height)] #get
```

Out[24]:

	Gender	Height	Weight
994	Male	78.095867	255.690835
1317	Male	78.462053	227.342565
2014	Male	78.998742	269.989699
3285	Male	78.528210	253.889004
3757	Male	78.621374	245.733783
6624	Female	54.616858	71.393749
9285	Female	54.263133	64.700127

In [27]:
```
df_no_outliers=df[(df["Height"]<upper_limit_height) & (df["Height"]>lower_limi
```

Out[27]:

	Gender	Height	Weight
0	Male	73.847017	241.893563
1	Male	68.781904	162.310473
2	Male	74.110105	212.740856
3	Male	71.730978	220.042470
4	Male	69.881796	206.349801
...	...	...	...
9995	Female	66.172652	136.777454
9996	Female	67.067155	170.867906
9997	Female	63.867992	128.475319
9998	Female	69.034243	163.852461
9999	Female	61.944246	113.649103

9993 rows × 3 columns

```
Conclusion :
there were 7 rows which were outliers based on Height Column.
```