Duvall Pinkney 13128408 2/8/2021 Spring 2021 Week 2 Homework Build a linear model (height = m * year + b) to escribe the trend of height increase for United Kingdom between 1900 and 1980. Compute the mean square error of your model, and display the model line together with the data points.

In [1]:

```
import numpy as np  # scientific computation
import pandas as pd  # data handling
import matplotlib.pyplot as plt  # plotting
# The following "magic command" allows figures to be displayed automatically in notebook
%matplotlib inline
```

In [2]:

```
height_data = pd.read_csv("average-height-of-men-for-selected-countries.csv")
```

In [15]:

```
height_data.head()
```

Out[15]:

	Entity	Code	Year	Human Height (University of Tuebingen (2015))
0	Afghanistan	AFG	1870	168.4
1	Afghanistan	AFG	1880	165.7
2	Afghanistan	AFG	1930	166.8
3	Albania	ALB	1880	170.1
4	Albania	ALB	1890	169.8

In [81]:

```
region_filter = (height_data["Entity"] == "United Kingdom") & (height_data['Year'] >= 1900) & (height_data['Year
print(region_filter)
data = height_data[region_filter]
data.head()
data
```

```
0       False
1       False
2       False
3       False
4       False
        ...
1245    False
1246    False
1247    False
1248    False
1249    False
Length: 1250, dtype: bool
```

Out[81]:

	Entity	Code	Year	Human Height (University of Tuebingen (2015))
1185	United Kingdom	GBR	1900	169.4
1186	United Kingdom	GBR	1910	170.9
1187	United Kingdom	GBR	1920	171.0
1188	United Kingdom	GBR	1930	173.9
1189	United Kingdom	GBR	1940	174.9
1190	United Kingdom	GBR	1950	176.0
1191	United Kingdom	GBR	1960	176.9
1192	United Kingdom	GBR	1970	177.1
1193	United Kingdom	GBR	1980	176.8

In [38]:

```
def height_model_algorithm(m, year, height):

    #height = m * year + b
    b = height - m * year
    return b
```

In [23]:

```
def slope_algo(year1, year2, height1, height2):
    m = (year2 - year1)/(height2 - height1)
    return m
```

In [24]:

```
def mse_algo():
    mse =
    return mse
```

```
  File "<ipython-input-24-f37f06342756>", line 2
    mse =
         ^
SyntaxError: invalid syntax
```

In [25]:

```
first_slope = slope_algo(1900,1910,169.4,170.9)
print(first_slope)
```

```
6.666666666666667
```

In [26]:

```
second_slope = slope_algo(1910,1920,170.9,171.1)
print(second_slope)
```

```
50.00000000000284
```

In [27]:

```
# Calculate the average increase per year between 1900 and 1980
# Two points: (1900, 170.0) and (1980, 179.0)
# What is the slope of the line connecting these two points?

slope = slope_algo(1900,1980,169.4,176.8)
print(slope)
```

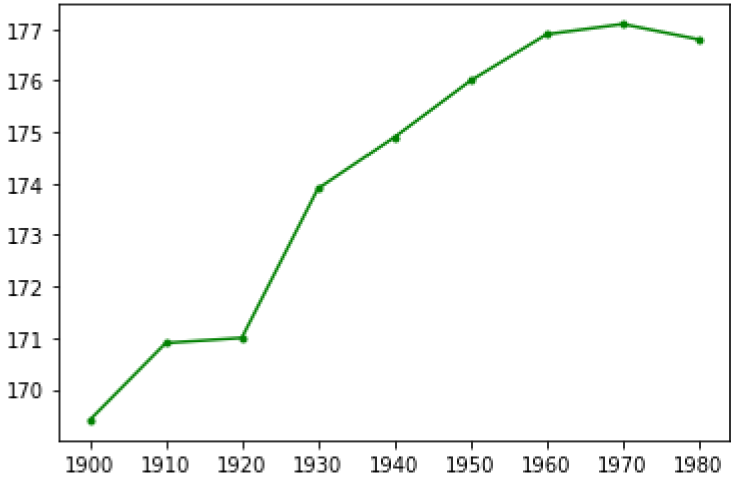
```
10.810810810810802
```

In [34]:

```
height = data['Human Height (University of Tuebingen (2015))']
year = data['Year']
plt.plot(year, height, 'g.-')
```

Out[34]:

```
[<matplotlib.lines.Line2D at 0x20c10301bc8>]
```



In [40]:

```
# Let the slope be the average rate of increase.
# If the model is height = m * year + b,
# find the value of b so that point (1900, 170.0) lies on the line.
height = 170
year = 1900
m = slope_algo(year, 1910, height, 170.9)
b = height_model_algorithm(m, year, height)
print("Y-Intercept:", b)
```

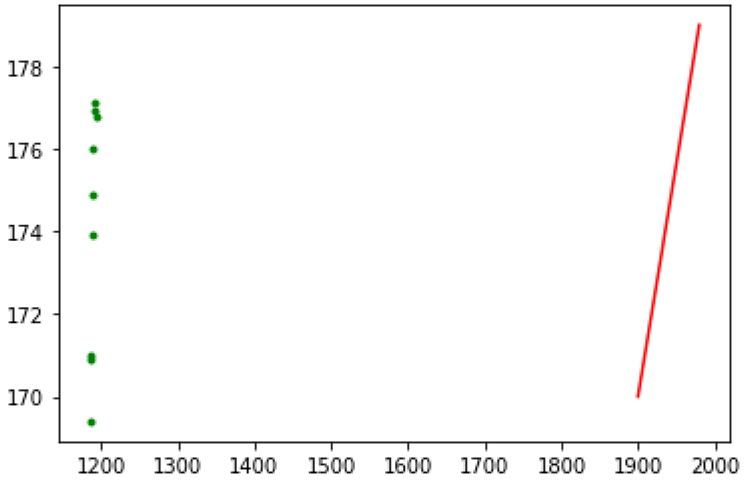
```
Y-Intercept: -20941.11111111098
```

In [45]:

```
# Plot the model line.
years = np.array([1900, 1940, 1980])
heights = 0.1125 * years - 43.75

plt.plot(years, heights, 'r-')

# show the data points on this plot
plt.plot(data.index, data['Human Height (University of Tuebingen (2015))'], 'g.')
```

Out[45]:

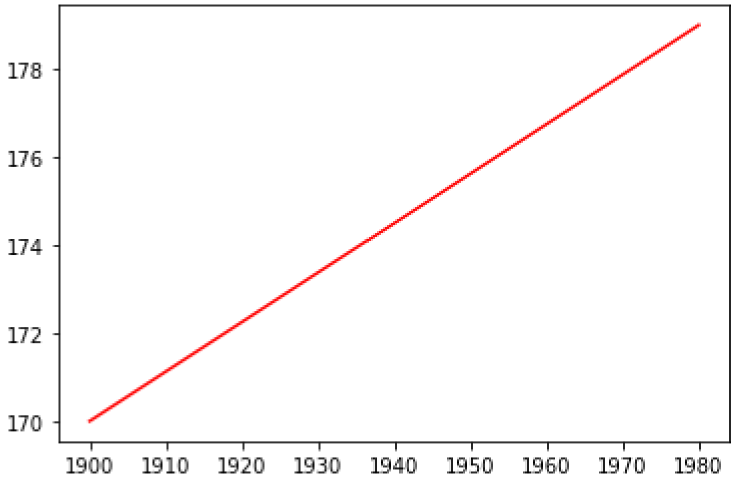
```
[<matplotlib.lines.Line2D at 0x20c1065ca08>]
```



In [44]:
```
m2 = 0.112
b2 = -42.2

plt.plot(years, heights, 'r-')
plt.plot(data.index, data['Height(cm)'], 'g.')

# plot the second line on this graph
heights2 = m2 * years + b2
plt.plot(years, heights2, 'b-')
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\Anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, method, tolerance)
   2896             try:
-> 2897                 return self._engine.get_loc(key)
   2898             except KeyError:

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'Height(cm)'

During handling of the above exception, another exception occurred:

KeyError                                  Traceback (most recent call last)
<ipython-input-44-271a60255721> in <module>
      3 
      4 plt.plot(years, heights, 'r-')
----> 5 plt.plot(data.index, data['Height(cm)'], 'g.')
      6 
      7 # plot the second line on this graph

~\Anaconda3\lib\site-packages\pandas\core\frame.py in __getitem__(self, key)
   2993             if self.columns.nlevels > 1:
   2994                 return self._getitem_multilevel(key)
-> 2995             indexer = self.columns.get_loc(key)
   2996             if is_integer(indexer):
```

```
   2997                     indexer = [indexer]

~\Anaconda3\lib\site-packages\pandas\core\indexes\base.py in get_loc(self, key, method, tolerance)
   2897                     return self._engine.get_loc(key)
   2898                 except KeyError:
-> 2899                     return self._engine.get_loc(self._maybe_cast_indexer(key))
   2900             indexer = self.get_indexer([key], method=method, tolerance=tolerance)
   2901             if indexer.ndim > 1 or indexer.size > 1:

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: 'Height(cm)'
```



$$MSE = \frac{1}{numberofdata} \sum_{\text{(x, y) in dataset}} (mx + b - y)^2$$

In [63]:

```
# Calculate errors for each year
# Example: Calculate the error for 1900
m = slope_algo(1900,1910,169.4,170.9)
b = height_model_algorithm(m, 1900, 169.4)

# For the data point about 1900:
x = 1900
y = 169.4

# Prediction of the model for 1930:
prediction = m * x + b
print("Predicton for 1900:", prediction)

# Squared Error for 1900:
error = (prediction - y) ** 2
print("Error for 1900:", error)
errors = error
```

```
Predicton for 1900: 169.39999999999964
Error for 1900: 1.3651711281392742e-25
```

In [64]:

```
# Example: Calculate the error for 1910
m = slope_algo(1910,1920,170.9,171.0)
b = height_model_algorithm(m, 1910, 170.9)

# For the data point about 1910:
x = 1910
y = 171.0

# Prediction of the model for 1910:
prediction = m * x + b
print("Predicton for 1910:", prediction)

# Squared Error for 1910:
error = (prediction - y) ** 2
print("Error for 1910:", error)
errors = error
```

```
Predicton for 1910: 170.89999999999418
Error for 1910: 0.010000000001164154
```

In [65]:

```
# Example: Calculate the error for 1920
m = slope_algo(1920,1930,171.0,173.9)
b = height_model_algorithm(m, 1920,171.0)

# For the data point about 1920:
x = 1920
y = 171.0

# Prediction of the model for 1920:
prediction = m * x + b
print("Predicton for 1920:", prediction)

# Squared Error for 1920:
error = (prediction - y) ** 2
print("Error for 1920:", error)
errors = error
```

```
Predicton for 1920: 171.0
Error for 1920: 0.0
```

In [66]:

```
# Example: Calculate the error for 1930
m = slope_algo(1930,1940,173.9,174.9)
b = height_model_algorithm(m, 1930,173.9)

# For the data point about 1930:
x = 1930
y = 173.9

# Prediction of the model for 1930:
prediction = m * x + b
print("Predicton for 1930:", prediction)

# Squared Error for 1930:
error = (prediction - y) ** 2
print("Error for 1930:", error)
errors = error
```

```
Predicton for 1930: 173.90000000000146
Error for 1930: 2.101071067627368e-24
```

In [67]:

```
# Example: Calculate the error for 1940
m = slope_algo(1940,1950,174.9,176.0)
b = height_model_algorithm(m, 1940,174.9)

# For the data point about 1940:
x = 1940
y = 174.9

# Prediction of the model for 1940:
prediction = m * x + b
print("Predicton for 1940:", prediction)

# Squared Error for 1940:
error = (prediction - y) ** 2
print("Error for 1940:", error)
errors = error
```

```
Predicton for 1940: 174.90000000000146
Error for 1940: 2.101071067627368e-24
```

In [68]:

```
# Example: Calculate the error for 1950
m = slope_algo(1950,1960,176.0,176.9)
b = height_model_algorithm(m, 1950,176.0)

# For the data point about 1950:
x = 1950
y = 176.0

# Prediction of the model for 1950:
prediction = m * x + b
print("Predicton for 1950:", prediction)

# Squared Error for 1950:
error = (prediction - y) ** 2
print("Error for 1950:", error)
errors = error
```

```
Predicton for 1950: 176.0
Error for 1950: 0.0
```

In [69]:

```
# Example: Calculate the error for 1960
m = slope_algo(1960,1970,176.9,177.1)
b = height_model_algorithm(m, 1960,176.9)

# For the data point about 1960:
x = 1960
y = 176.9

# Prediction of the model for 1960:
prediction6 = m * x + b
print("Predicton for 1960:", prediction6)

# Squared Error for 1960:
error6 = (prediction6 - y) ** 2
print("Error for 1960:", error6)
errors = error6
```

```
Predicton for 1960: 176.89999999999418
Error for 1960: 3.3947524650918934e-23
```

In [70]:

```
# Example: Calculate the error for 1970
m = slope_algo(1970,1980,177.1,176.8)
b = height_model_algorithm(m, 1970,177.1)

# For the data point about 1970:
x = 1970
y = 177.1

# Prediction of the model for 1970:
prediction7 = m * x + b
print("Predicton for 1970:", prediction7)

# Squared Error for 1970:
error7 = (prediction7 - y) ** 2
print("Error for 1970:", error7)
errors = error7
```

```
Predicton for 1970: 177.10000000000582
Error for 1970: 3.3947524650918934e-23
```

In [82]:

```
# Example: Calculate the error for 1980
m = slope_algo(1980,1990,176.8,177.1)
b = height_model_algorithm(m, 1980,176.8)

# For the data point about 1980:
x = 1980
y = 176.8

# Prediction of the model for 1980:
prediction8 = m * x + b
print("Predicton for 1980:", prediction8)

# Squared Error for 1980:
error8 = (prediction8 - y) ** 2
print("Error for 1980:", error8)
errors = error8
```

```
Predicton for 1980: 176.8000000000029
Error for 1980: 8.404284270509473e-24
```

In [89]:

```


data.index

errors = [1.3651711281392742e-25,0.010000000001164154,0.0,2.101071067627368e-24,2.101071067627368e-24,0.0,3.3947
year = 0
for year in range(len(data.index)):
    # Calculate the squared error for that year
    # error = ???
    error = errors[year] ** 2
    
    
    # append the error to the errors list
    errors.append(error)

# Now you should have a list of errors.
print(errors)

# Calculate the mean squared error, use np.mean() function
print(np.mean(errors))

```

```
[1.3651711281392742e-25, 0.010000000001164154, 0.0, 2.101071067627368e-24, 2.101071067627368e-24, 0.0, 3.39475
24650918934e-23, 8.404284270509473e-24, 1.8636922091050586e-50, 0.00010000000002328308, 0.0, 4.414499631220808
6e-48, 4.4144996312208086e-48, 0.0, 1.1524344299247487e-45, 7.063199409953294e-47, 3.4733486502788934e-100]
0.0005941176471286727
```

In [ ]: