EE325 HW1

Aditya Anand, Rishab Rao, Aakash Gore

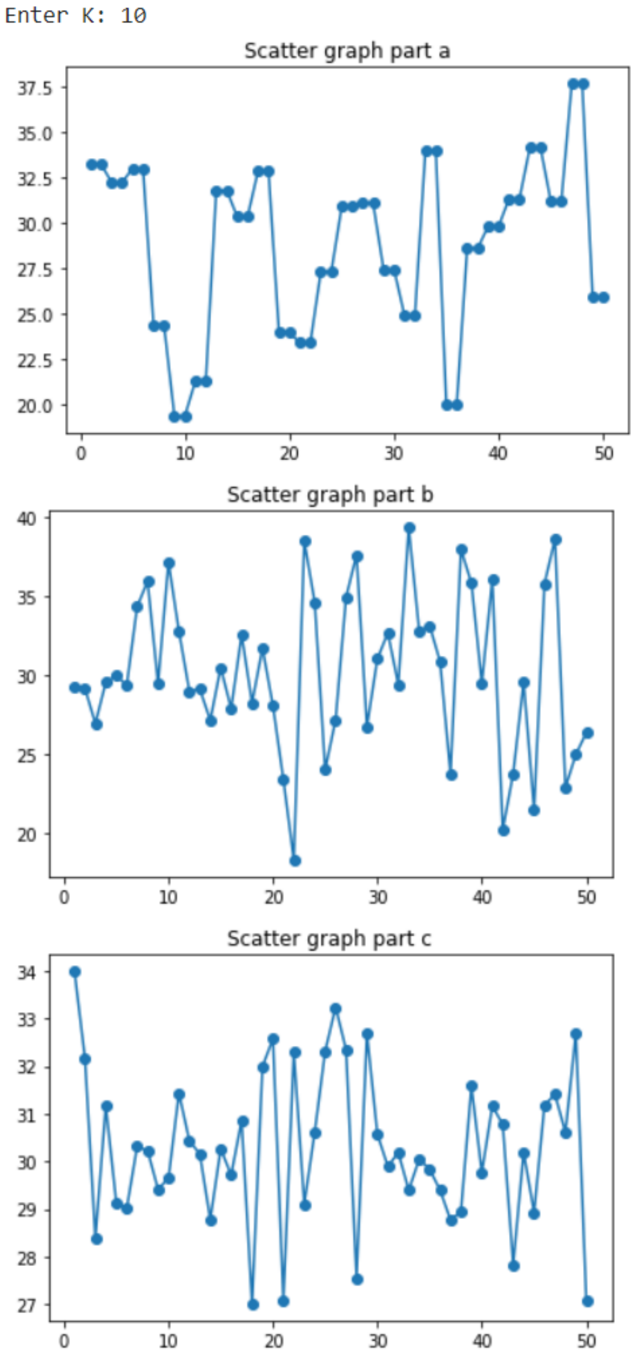
August 11, 2022

1 Question 1

1.1 Python code

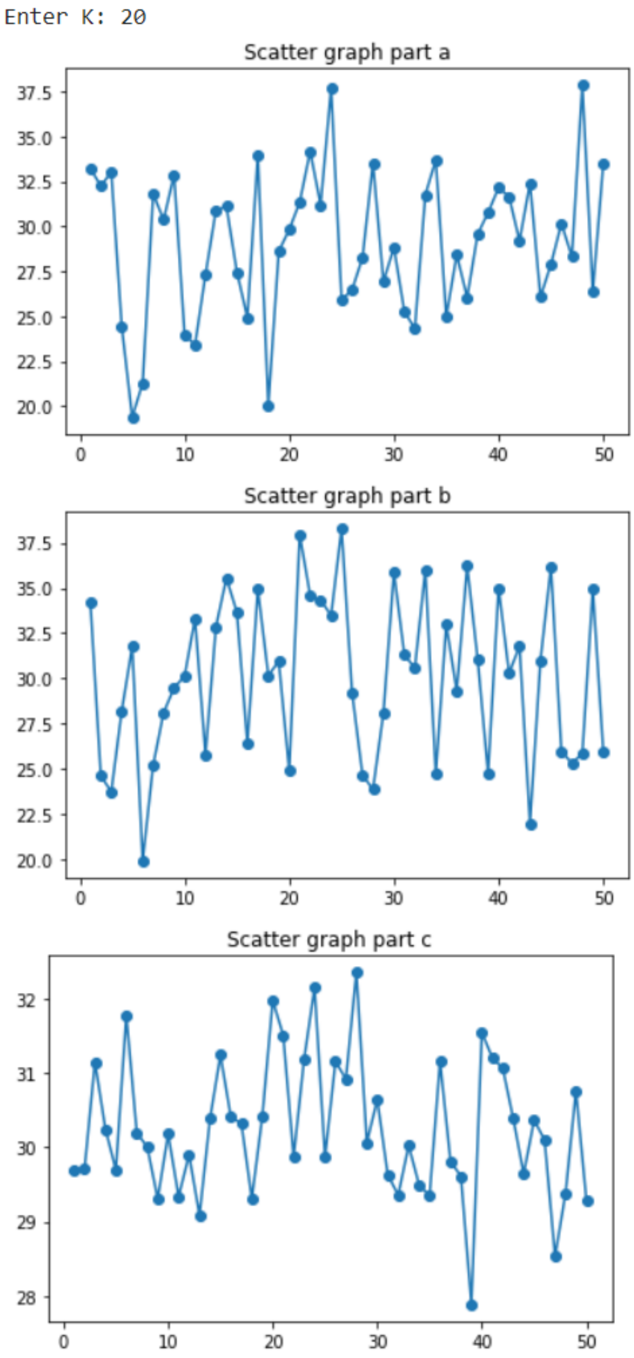
```
from numpy.lib.arraysetops import setxor1d
import numpy as np
import random
import statistics
from numpy.lib.shape_base import expand_dims
from matplotlib import pyplot as plt
import math

data = np.loadtxt("A.txt")

k = int(input("Enter K: "))
i = 0
j = 0
sumavg = [0]*50
while j < 50:
  while i < k:
    sumavg[j] = sumavg[j] + data[i + k*j]/k
    i += 1
  i = 0
  j += 1
x = [None]*50
while i < 50:
  x[i] = i + 1
  i += 1

plt.title("Scatter graph part a")
plt.scatter(x,sumavg)
plt.plot(x,sumavg)
plt.show()
```

```
# b part
i = 0
sumavgb = [0]*50
xb = [None]*10000

while i < 10000:
  xb[i] = i
  i += 1

i = 0
j = 0
while j < 50:
  r = random.choice(xb)                     #r=selecting a random number from 0-9999
  while i < k:
    sumavgb[j] = sumavgb[j] + data[abs(r - i)]/k  #taking k data set below the rth number an
    i += 1
  i = 0
  j += 1

plt.title("Scatter graph part b")
plt.scatter(x,sumavgb)
plt.plot(x,sumavgb)
plt.show()


#part c
j = 0
i = 0
sumavgc = [0]*50
while j < 50:
  while i < k:
    r = random.choice(data)
    sumavgc[j] = sumavgc[j] + r/k
    i += 1
  i = 0
  j += 1

plt.title("Scatter graph part c")
plt.scatter(x,sumavgc)
plt.plot(x,sumavgc)
plt.show()
```



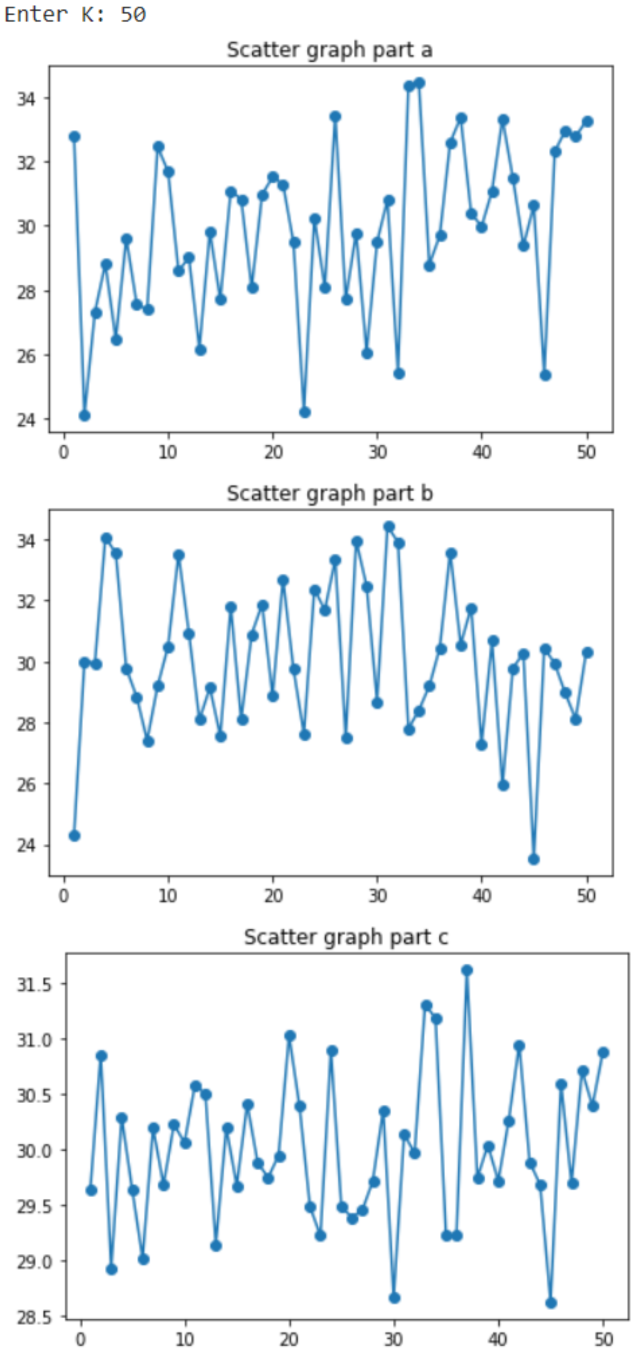
```
Enter K: 10
```

Figure 1: Scatter plot when k=10



```
Enter K: 20
```

Figure 2: Scatter plot when k=50



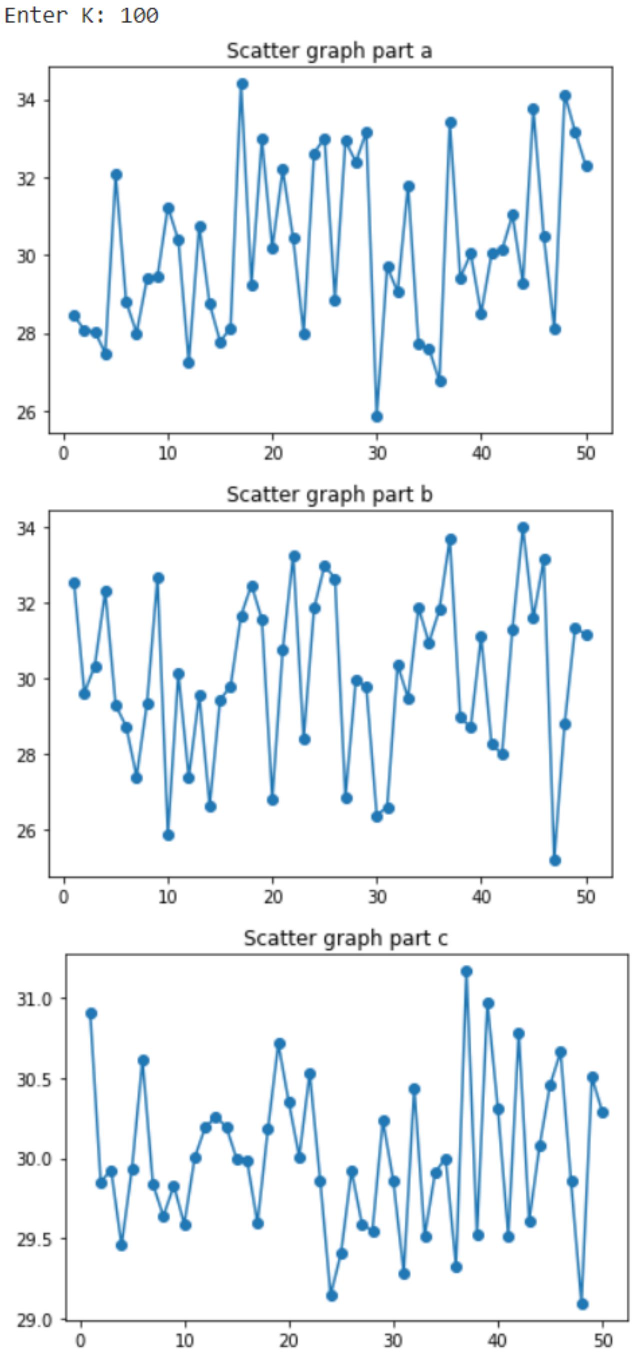
```
Enter K: 50
```

Scatter graph part a

Scatter graph part b

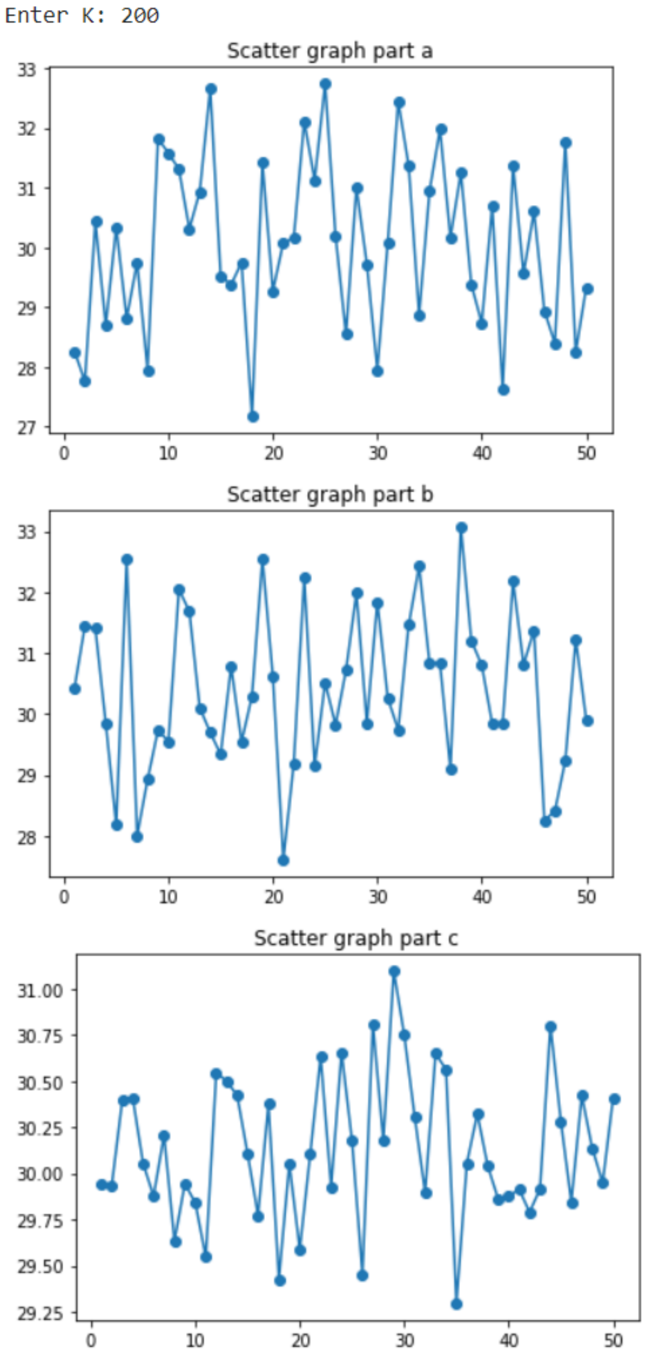
Scatter graph part c

Figure 3: Scatter plot when k=50



```
Enter K: 100
```

Figure 4: Scatter plot when k=100



```
Enter K: 200
```

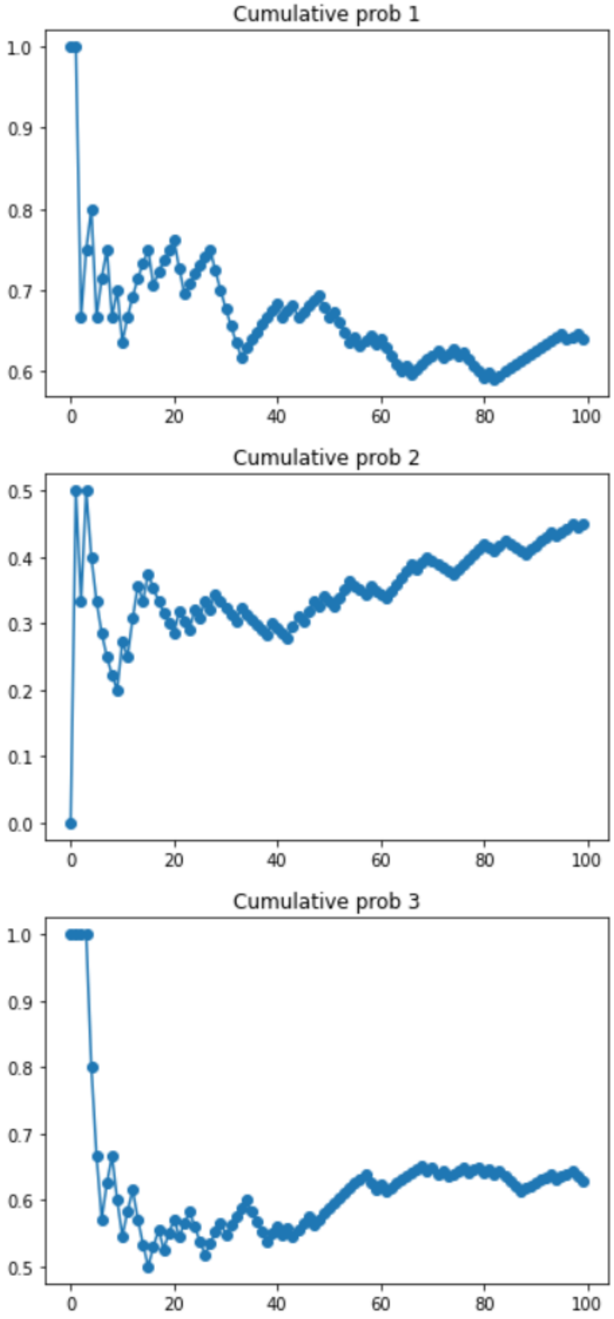
Figure 5: Scatter plot when k=200

2 Question 2

2.1 Python code

```
from numpy.lib.arraysetops import setxor1d
import numpy as np
import random
import statistics
from numpy.lib.shape_base import expand_dims
from matplotlib import pyplot as plt
import math

data1 = np.loadtxt("hw1b1.txt")
data2 = np.loadtxt("hw1b2.txt")
data3 = np.loadtxt("hw1b3.txt")
data4 = np.loadtxt("hw1b4.txt")
cp1 = [None]*100         #commulative probability
cp2 = [None]*100
cp3 = [None]*100
cp4 = [None]*100
x = [None]*100
i = 0
while i < 100:
  x[i] = i
  i += 1

countofheads = 0
nooftoss = 1
i = 0
while i < 100 :
  if data1[i] == 1:
    countofheads += 1
    cp1[i] = countofheads/nooftoss
  else : cp1[i] = countofheads/nooftoss
  nooftoss += 1
  i += 1
i = 0
countofheads = 0
nooftoss = 1
while i < 100 :
  if data2[i] == 1:
    countofheads += 1
    cp2[i] = countofheads/nooftoss
  else : cp2[i] = countofheads/nooftoss
  nooftoss += 1
  i += 1
```

```
i = 0
countofheads = 0
nooftoss = 1
while i < 100 :
  if data3[i] == 1:
    countofheads += 1
    cp3[i] = countofheads/nooftoss
  else : cp3[i] = countofheads/nooftoss
  nooftoss += 1
  i += 1
i = 0
countofheads = 0
nooftoss = 1
while i < 100 :
  if data4[i] == 1:
    countofheads += 1
    cp4[i] = countofheads/nooftoss
  else : cp4[i] = countofheads/nooftoss
  nooftoss += 1
  i += 1

plt.title("Cumulative prob 1")
plt.scatter(x,cp1)
plt.plot(x,cp1)
plt.show()

plt.title("Cumulative prob 2")
plt.scatter(x,cp2)
plt.plot(x,cp2)
plt.show()

plt.title("Cumulative prob 3")
plt.scatter(x,cp3)
plt.plot(x,cp3)
plt.show()

plt.title("Cumulative prob 4")
plt.scatter(x,cp4)
plt.plot(x,cp4)
plt.show()
```


Cumulative prob 1

Cumulative prob 2

Cumulative prob 3

Figure 6: (No. Of heads till now/total coin tosses till now) vs No. of tosses

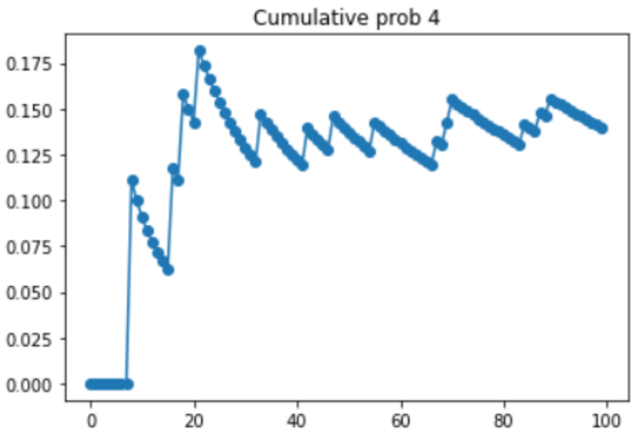


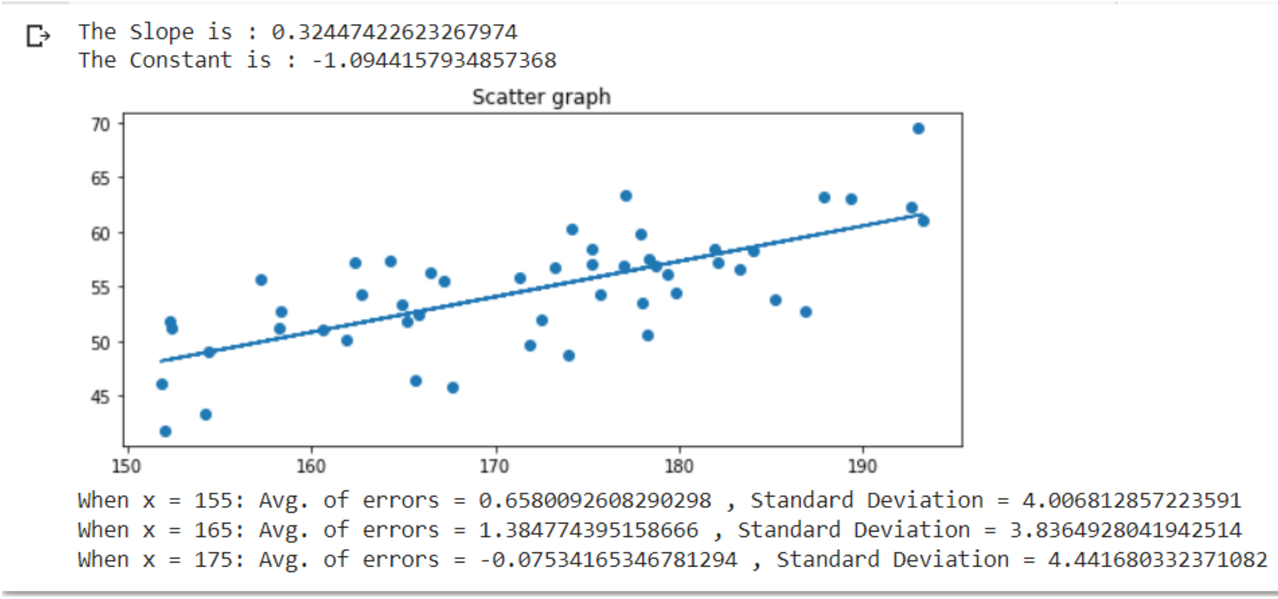
Figure 7: No. Of heads till now/total coin tosses till now) vs No. of tosses

3 Question 3

3.1 Python code

```
    import numpy as np
import statistics
from numpy.lib.shape_base import expand_dims
from matplotlib import pyplot as plt
import math
dt = np.array([
[161.8917578714209,50.06513915417443],
[175.20557024887373,57.08997118968728],
[164.2676553766112,57.34038829783626],
[178.7222340341774,56.88540143502565],
[178.00073820784604,53.478696908960934],
[173.2772419681515,56.768169670936345],
[160.59676770218624,50.98936280545692],
[192.65051786762572,62.35249986686582],
[165.16827896723976,51.88319410967615],
[186.83322519192234,52.674703814115304],
[158.3181312404598,52.69023951205429],
[152.40314131522246,51.27488019363186],
[176.96604318719778,56.878634760999624],
[174.1476319971853,60.283919884274155],
[165.64381551861507,46.438146326369136],
[178.33112793512723,57.504487839354574],
[167.63588343343207,45.744829489560345],
[158.2518544667988,51.17557752289022],
[162.39631147040325,57.227062735277805],
[164.93377167203346,53.41123363598972],
[185.23966651457155,53.7655228010482],
[174.00884801974885,48.77457857823918],
[178.29163623554024,50.57295828987031],
[154.42607294132182,49.109738459110076],
[152.02373388539078,41.84216598976299],
[171.35451719883235,55.871446827859636],
[172.46880217666438,52.03636592100854],
[187.8259234866104,63.228012473952845],
[165.8348672259168,52.497642699508894],
[154.21715210274448,43.42309786298475],
[183.99341334642781,58.2807014841803],
[192.96899615398104,69.4769468120694],
[193.24767230032984,61.140432492527324],
[152.34085565439042,51.89160705026108],
[177.86128609011809,59.88218450922603],
[167.1786624519332,55.475816112146205],
[183.32867104686477,56.65606435614058],
[189.32630987297262,62.99451248585423],
[171.8239506780645,49.64889204896306],
[162.7533361998475,54.28403891707874],
[181.8937717199068,58.38375250582365],
[151.81355565372678,46.20229138646405],
[175.70637064056112,54.28590662636416],
[179.82951362312016,54.51621196857059],
[179.39069884595247,56.063543839297886],
[177.08397660941904,63.33081820162695],
[182.1282175100166,57.229870486518834],
[175.2020315276609,58.40650863480111],
[166.4430523185654,56.285176888103265],
[157.22110851705304,55.73018000987019]
])

x = dt[:,0]                                #x has 50 elements
y = dt[:,1]                                #y has 50 elements
y1 = np.array([50.84083481424745,          #y1,y1,y3 are array for weights of 25,25,25 peopl
50.014150611241,                           #having height 155,165,175 resp.
47.434433994546275,
48.18509369914934,
49.47940437427491,
51.46395679542809,
50.92745554133917,
50.05313719206625,
44.32619446516681,
58.43082902641991,
51.78016307917405,
44.02198762936742,
50.72446115642888,
50.705517805027014,
57.59462421345118,
54.470110375852684,
53.22931789122948,
51.774125788427334,
54.474225400616,
48.6041391956155,
45.39272640839777,
44.650940250705666,
42.69887479467695,
47.003806810494126,
48.146952021872984])
y2 = np.array([56.09178233874335,
53.45028077511345,
53.921575413701156,
48.85934258973829,
56.95307675637719,
58.34536050529028,
58.38396662842817,
59.53843135500536,
63.86284924117021,
50.37873251772672,
53.542546379774144,
55.65272988547241,
56.19834658523187,
49.931028628110056,
49.067210771486366,
57.081250056851644,
52.941790056077416,
50.11083820970033,
52.196245895572396,
52.86016951388828,
52.997389807842836,
50.77474626914866,
47.91793610117185,
51.317062516525056,
53.34045945347973])
y3=np.array([58.7729713699485,
57.52174253372618,
46.63896463839615,
60.08699337124898,
54.97048946487911,
50.89015721919731,
51.16430051196914,
60.44417409273054,
47.330206543798724,
58.62280875417217,
51.94075563649851,
53.85545926849789,
55.19225332108357,
57.316954222269885,
61.96906592544718,
64.45515173613902,
58.145794819284355,
52.00153353928704,
53.687784624881914,
54.467373628898336,
54.83437188336155,
58.39119923096617,
52.72810010849628,
53.5992234950328,
61.30297365392382])
plt.title("Scatter graph")
m,c=np.polyfit(x,y,1)              #.polyfit is used to find the best fit polynomial
print (’The Slope is :’ ,m)        # m slope c constant
print (’The Constant is :’ ,c)
plt.scatter(x,y)
plt.plot(x,m*x+c)                  # y=mx+c
plt.show()
y_hat = [None]*25
error1 = [None]*25
error2 = [None]*25
error3 = [None]*25                 #error1,2,3 is an array of size 25
sumoferrors1 = 0                   #initallty the sum of errors is zero
sumoferrors2 = 0
sumoferrors3 = 0
y1hat = m*155+c                    #guessed weights
y2hat = m*165+c
y3hat = m*175+c

i = 0
while i < 25:
  error1[i] = y1[i]-y1hat          # finding error1[0],error1[1]...
  error2[i] = y2[i]-y2hat
  error3[i] = y3[i]-y3hat

  sumoferrors1 = sumoferrors1 + error1[i]
  sumoferrors2 = sumoferrors2 + error2[i]
  sumoferrors3 = sumoferrors3 + error3[i]
  i += 1
SD1 = statistics.stdev(error1)     #using the function to find the standard
SD2 = statistics.stdev(error2)     #deviation of the value stored in an array
SD3 = statistics.stdev(error3)
print('When x = 155: Avg. of errors =',sumoferrors1/25,', Standard Deviation =',SD1)
print('When x = 165: Avg. of errors =',sumoferrors2/25,', Standard Deviation =',SD2)
print('When x = 175: Avg. of errors =',sumoferrors3/25,', Standard Deviation =',SD3)
```

3.2 Scatter plot, Value of Slope, Constant, Average and Standard Deviation.



```
The Slope is : 0.32447422623267974
The Constant is : -1.0944157934857368
```

```
When x = 155: Avg. of errors = 0.6580092608290298 , Standard Deviation = 4.006812857223591
When x = 165: Avg. of errors = 1.384774395158666 , Standard Deviation = 3.8364928041942514
When x = 175: Avg. of errors = -0.07534165346781294 , Standard Deviation = 4.441680332371082
```

Figure 8: Scatter plot