QUESTION 1

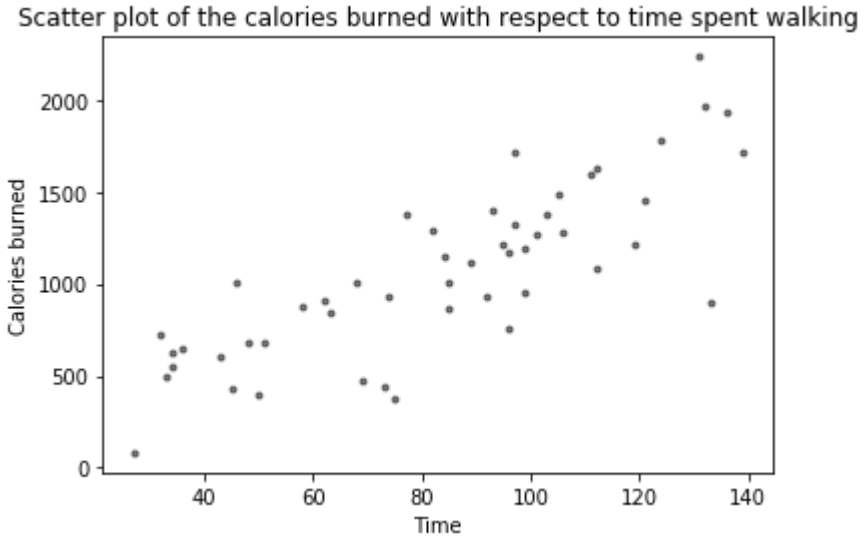
In [ ]:
```
import numpy as np
import os
import matplotlib.pyplot as plt
from decimal import Decimal
```

In [117]:
```
#question letter a

#finding the standard deviation and mean without using std() or mean(), respec
tively

time = [101,96,133,45,105,48,34,92,74,75,99,112,62,99,69,121,119,50,93,89,84,1
24,51,58,34,132,97,77,96,85,97,85,106,82,139,46,32,33,27,68,73,63,111,95,112,1
31,136,43,36,103]

calories = [1267,759,899,434,1489,681,544,929,934,376,949,1087,910,1188,469,14
49,1218,395,1399,1114,1146,1775,678,874,630,1967,1321,1382,1167,1009,1716,860,
1277,1286,1718,1004,723,491,82,1003,443,845,1598,1212,1633,2235,1936,601,645,1
378]

number_elements_time = len(time)
number_elements_cal = len(calories)

mean_time = (1/number_elements_time)*sum(time)
mean_cal = (1/number_elements_cal)*sum(calories)

mean_time_sigfig = round(mean_time,1)
mean_cal_sigfig = round(mean_cal,-1)

print('Mean time measured without appropriate significant figures')
print(mean_time)

print('Mean time measured')
print(mean_time_sigfig)

print('Mean calories burned without appropriate significant figures')
print(mean_cal)

print('Mean calories burned')
print(mean_cal_sigfig)

sum_time = sum(np.power([x - mean_time for x in time],2))
std_time = np.sqrt(1/number_elements_time)*np.sqrt(sum_time)

#we round to the first decimal place to get 3 significant figures
std_time_sigfig = round(std_time,1)

sum_cal = sum(np.power([x - mean_cal for x in calories],2))
std_cal = np.sqrt(1/number_elements_cal)*np.sqrt(sum_cal)


#we round to no decimal place to get 3 significant figures
std_cal_sigfig = int(std_cal)

print('Standard deviation of times measured without appropriate significant fi
gures')
print(std_time)

print('Standard deviation of times measured')
print(std_time_sigfig)

print('Standard deviation of times measured without appropriate significant fi
```

```
gures')
print(std_cal)

print('Standard deviation of burned calories measured')
print(std_cal_sigfig)
```

```
Mean time measured without appropriate significant figures
83.44
Mean time measured
83.4
Mean calories burned without appropriate significant figures
1062.5
Mean calories burned
1060.0
Standard deviation of times measured without appropriate significant figures
31.15905004970466
Standard deviation of times measured
31.2
Standard deviation of times measured without appropriate significant figures
464.27057843460204
Standard deviation of burned calories measured
464
```

In [69]:
```
#question 1 letter b

N = 100
x = [101,96,133,45,105,48,34,92,74,75,99,112,62,99,69,121,119,50,93,89,84,124,
51,58,34,132,97,77,96,85,97,85,106,82,139,46,32,33,27,68,73,63,111,95,112,131,
136,43,36,103]
y = [1267,759,899,434,1489,681,544,929,934,376,949,1087,910,1188,469,1449,1218
,395,1399,1114,1146,1775,678,874,630,1967,1321,1382,1167,1009,1716,860,1277,12
86,1718,1004,723,491,82,1003,443,845,1598,1212,1633,2235,1936,601,645,1378]
colors = (0,0,0)
area = np.pi*3

# Plot
plt.scatter(x, y, s=area, c=colors, alpha=0.5)
plt.title('Scatter plot of the calories burned with respect to time spent walk
ing')
plt.xlabel('Time')
plt.ylabel('Calories burned')
plt.show()
```



QUESTION 1, C)

First of all, we would expect the correlation coefficient to be positive given that the relationship is linearly increasing. Moreover, we would expect the correlation coefficient to be closer to 1 than 0 given that there is a visible trend.

In [119]:
```
#question 1 letter d

time = [101,96,133,45,105,48,34,92,74,75,99,112,62,99,69,121,119,50,93,89,84,1
24,51,58,34,132,97,77,96,85,97,85,106,82,139,46,32,33,27,68,73,63,111,95,112,1
31,136,43,36,103]
calories = [1267,759,899,434,1489,681,544,929,934,376,949,1087,910,1188,469,14
49,1218,395,1399,1114,1146,1775,678,874,630,1967,1321,1382,1167,1009,1716,860,
1277,1286,1718,1004,723,491,82,1003,443,845,1598,1212,1633,2235,1936,601,645,1
378]

#we find the correlation coefficient

Pearson = np.corrcoef(time, calories)[0,1]
Pearson_sigfig = round(Pearson,3)

print('Pearson coefficient of time spent walking and calories burned')
print(Pearson_sigfig)

#we find the covariance. We could use the Pearson coefficient or np.cov. We sh
ow that both give the same result.

cov_np = np.cov(time, calories)[0,1]
cov_np_sigfig = round(cov_np,-2)

cov_p = (Pearson)*(np.std(time, ddof=1)*np.std(calories, ddof=1))
cov_p_sigfig = round(cov_p,-2)

print('Covariance found using np.cov without appropriate number of significant
 figures')
print(cov_np)
print('Covariance found using np.cov')
print(cov_np_sigfig)
print('Covariance found using rho without appropriate number of significant fi
gures')
print(cov_p)
print('Covariance found using rho')
print(cov_p_sigfig)
```

```
Pearson coefficient of time spent walking and calories burned
0.812
Covariance found using np.cov without appropriate number of significant figur
es
11981.102040816326
Covariance found using np.cov
12000.0
Covariance found using rho without appropriate number of significant figures
11981.102040816328
Covariance found using rho
12000.0
```

In [ ]: