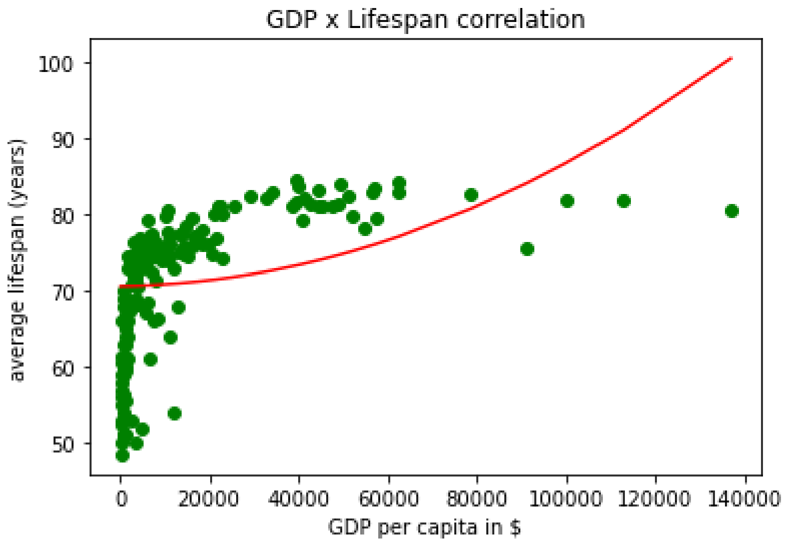
In [46]:

```
import numpy as np
import matplotlib.pyplot as plt
import random as rnd

# ---- Essentials ----

plt.title("GDP x Lifespan correlation")
plt.ylabel("average lifespan (years)")
plt.xlabel("GDP per capita in $")

nations = np.genfromtxt("nations.csv", delimiter=",", skip_header=True)
perCapita = nations[:,3]/nations[:,6]*1000000

plt.plot(perCapita, nations[:,4], "og")


# ---- Approximation function ----

def rndApprox():

    x = np.sort(perCapita)

    x = np.expand_dims(perCapita, axis = 0)
    p = np.expand_dims(np.arange(0,3), axis = 1)
    x2 = np.power(x, p)

    coeffs = np.array([rnd.uniform(0,100),rnd.uniform(-0.00001,0.00001),rnd.uniform(-0.0000
    coeffs = np.expand_dims(coeffs, axis = 1)

    y = np.matmul(x2.transpose(), coeffs)

    return coeffs, y


# ---- RMSE calculation function ----

def getRMSE(y, y1):
    y1 = y1.transpose() # <-- somehow this seems to be nessecary... :/

    rmse = np.sqrt(np.mean((y1-y)**2))

    return rmse




# ---- MAIN ----

bestResult = rndApprox()

#print(bestResult[1].shape) <-- (163, 1)
#print(nations[:,4].shape) <-- (163,) vector vs 2D array, but seems to be fine

for i in range(10000):
    result = rndApprox()
    if getRMSE(nations[:,4], result[1]) <  getRMSE(nations[:,4], bestResult[1]):
        bestResult = result
```

```
x, y = zip(*sorted(zip(perCapita, bestResult[1]))) # <-- little trick to sort x-values
                                # Source: https://stackoverflow.com/questions/37414916/pythons-mat

plt.plot(x, y, "r")
plt.show()

print("RMSE: " , getRMSE(nations[:,4], bestResult[1]), "\n\nk1: ", bestResult[0][0], "\nk2:
```



```
RMSE:  8.430400789068099

k1:  [70.61077232]
k2:  [7.93612535e-06]
k3:  [1.54117597e-09]
```