Group 8
Furkan Keskin – 2018400150
Sinem Koçoğlu - 2020400339

1)

Following statistics are deduced from the data given.

mean:	243,03797468
std dev:	345,06453578
median:	125,55437069
variance:	119069,53385031
range:	3011,20487915
min:	0,28051238
max:	3011,48539153
sum:	172799,99999997
count:	711

2)

H0: Normally distributed with mean 200, std dev 50.

H1: Not normally distributed with mean 200, std dev 50.

max D+: 0,435907775

max D-: 0,230207768

max {D+, D-}: 0,435907775

D (0.05, 711): 0,051003985

As 0,051003985 < 0,43907775 , we reject H0.

3)

The shapes of frequency histograms resemble the exponential distribution. As the time interval increases, the fluctuations in the histogram decrease, so that it more closely resembles an exponential probability distribution function.

4)

H0: Exponentially distributed.
H1: Not exponentially distributed.

$X_0^2 = 184{,}447$

$X_{\alpha,\, k\text{-}s\text{-}1}^2 = 355.051$ ($k=302$, $s=1$, $\alpha=0{,}05$)

As $355.051 > 184{,}447$, we do not reject the hypothesis.

Note: We decided to ignore the outliers at the right end for a healthier test result.

5)

We think that the QQ-plot is quite similar to the linear function. Therefore, data likely comes from an exponential distribution.

6)

When the observation time is between 60000 and 80000 there is an increase in interarrival times. Similarly, between 140000 and 180000 there is an increase in interarrival times. Although we think that this is not an obvious pattern, as observation times and interarrival times might be correlated due to the above explanation; data is non-stationary.

7)

Autocorrelation1	Autocorrelation2	Autocorrelation3
0,290699481	0,242539989	0,143768389

As autocorrelation values approach to 0, data becomes less correlated.
As it can be seen from our autocorrelation values which are quite close to 0, our data is not autocorrelated.
Additionally, since autocorrelation3 is the closest one to 0, it is the less correlated one.
This can also be seen from the plots.

8)

```
random.seed(123456)

mean = 243
limit = 10 * 24 * 60 * 60
current_time = 0

interarrival_times = []
interarrival_times.append("Generated Interarival Times")

while current_time < limit:
    interarrival_time = -mean * math.log(1 - random.uniform(0, 1))
    interarrival_times.append(interarrival_time)
    current_time += interarrival_time
```

We generated the interarrival times for 10 consecutive days with the above code. An interarrival time is calculated by the formula $-(1/\lambda)\ \ln(1-Ri)$. We generated just enough for 10 days.