```
import numpy as np
import matplotlib.pyplot as plt
from collections import Counter
from scipy.stats import chisquare
```

KM estimate

```
# Returns matrix leading to calculation of all probabilities of survival and death usin
def KM(death,censor,tot_pop):
    death_events=dict(Counter(death))
    censor_events=dict(Counter(censor))
    event_days=np.unique(np.append(death,censor))    # Days at which any event happen

    # Consider days at which death or censor happened
    healthy_at_start=[]
    subjects_dead_at_end=[]
    subjects_censored_at_end=[]
    pr_death=[]
    pr_survival=[]
    cum_pr_survival=[1]
    cum_pr_death=[]

    healthy_at_start.append(tot_pop)
    for i in event_days:
        subjects_dead_at_end.append(int(0 if i not in death_events else death_events[i]
        subjects_censored_at_end.append(int(0 if i not in censor_events else censor_eve
        pr_death.append(subjects_dead_at_end[-1]/healthy_at_start[-1])
        pr_survival.append(1-pr_death[-1])
        cum_pr_survival.append(pr_survival[-1]*cum_pr_survival[-1])
        cum_pr_death.append(1-cum_pr_survival[-1])

        if i!=event_days[-1]:   # Calculate next number of subjects
            healthy_at_start.append(healthy_at_start[-1]-subjects_dead_at_end[-1]-subje

    return [0]+event_days.tolist(), cum_pr_survival
```

Log Rank Test

```
def log_rank(death1,censor1,death2,censor2,tot_pop):
    death_events=np.sort(np.unique(np.append(death1,death2)))
    death1_events=dict(Counter(death1))
    death2_events=dict(Counter(death2))
    censor1_events=dict(Counter(censor1))
    censor2_events=dict(Counter(censor2))
    all_events=np.unique(np.concatenate((death1,censor1,death2,censor2)))
    death_events2idx={key:value for value,key in enumerate(death_events)}

    n1t, n2t=[], []   # Number of people at risk
```

```
    nt=[]             # total people at risk
    o1t, o2t=np.zeros_like(death_events), np.zeros_like(death_events)   # Number of dea
    ot=[]             # total deaths
    e1t, e2t=[],[]    # Expected number of events
    pop1,pop2=tot_pop,tot_pop

    c=0
    for i in all_events:
        if i in death_events:
            n1t.append(pop1)
            n2t.append(pop2)
            if i in death1_events:
                o1t[death_events2idx[i]]+=death1_events[i]
                pop1-=death1_events[i]
            if i in death2_events:
                o2t[death_events2idx[i]]+=death2_events[i]
                pop2-=death2_events[i]
        if i in censor1_events:
            pop1-=censor1_events[i]
        if i in censor2_events:
            pop2-=censor2_events[i]

    nt=np.array(n1t)+np.array(n2t)
    ot=np.array(o1t)+np.array(o2t)
    e1t=np.array(n1t)*ot/nt
    e2t=np.array(n2t)*ot/nt

    return np.sum(o1t),np.sum(o2t),np.sum(e1t),np.sum(e2t)
```

Ans-1

```
c_before_surgery_death=np.array([8,12,26,14,21,27])
c_before_surgery_censor=np.array([8,32,20,40])

c_after_surgery_death=np.array([33,28,41])
c_after_surgery_censor=np.array([48,48,25,37,48,25,43])
```

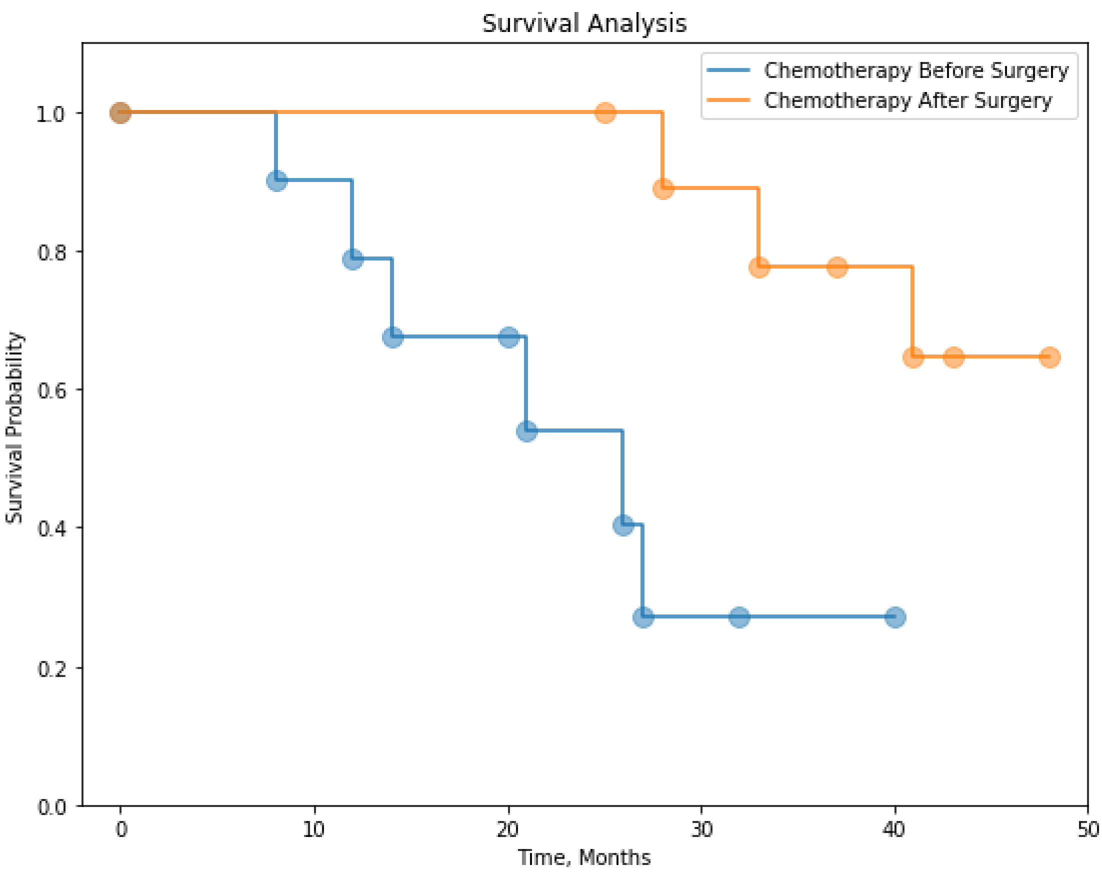
```
event_days_before,cum_pr_survival_before=KM(c_before_surgery_death,c_before_surgery_cen
event_days_after,cum_pr_survival_after=KM(c_after_surgery_death,c_after_surgery_censor,
```

```
print(event_days_before)
print(cum_pr_survival_before)
```

```
[0, 8, 12, 14, 20, 21, 26, 27, 32, 40]
[1, 0.9, 0.7875, 0.675, 0.675, 0.54, 0.405, 0.2700000000000001, 0.2700000000000001, 0
```

```
plt.figure(figsize=(9,7))
plt.step(event_days_before,cum_pr_survival_before, where='post', label='Chemotherapy Be
plt.scatter(event_days_before, cum_pr_survival_before, alpha=0.5,s=100)

plt.step(event_days_after,cum_pr_survival_after, where='post', label='Chemotherapy Afte
plt.scatter(event_days_after, cum_pr_survival_after, alpha=0.5,s=100)
```

```
plt.scatter(event_days_after, cum_pr_survival_after, alpha=0.5,s=100)

plt.ylim([0,1.1])
plt.xlim([-2,50])
plt.xlabel('Time, Months')
plt.ylabel('Survival Probability')
plt.title('Survival Analysis')
plt.legend()
plt.show()
```



1) Median survival for group 'Chemotherapy Before Surgery' has median survival rate of 26 months

2) Median survival for group 'Chemotherapy After Surgery' has median survival rate of infinite mon

```
o1t,o2t,e1t,e2t=log_rank(c_before_surgery_death,c_before_surgery_censor,c_after_surgery_
p_value=chisquare([o1t,o2t],[e1t,e2t])[1]

p_value
```

```
0.013155428547470005
```

As p-value is < 0.05, we reject the null-hypothesis that both of the survival plots are statisctically si

Ans-2

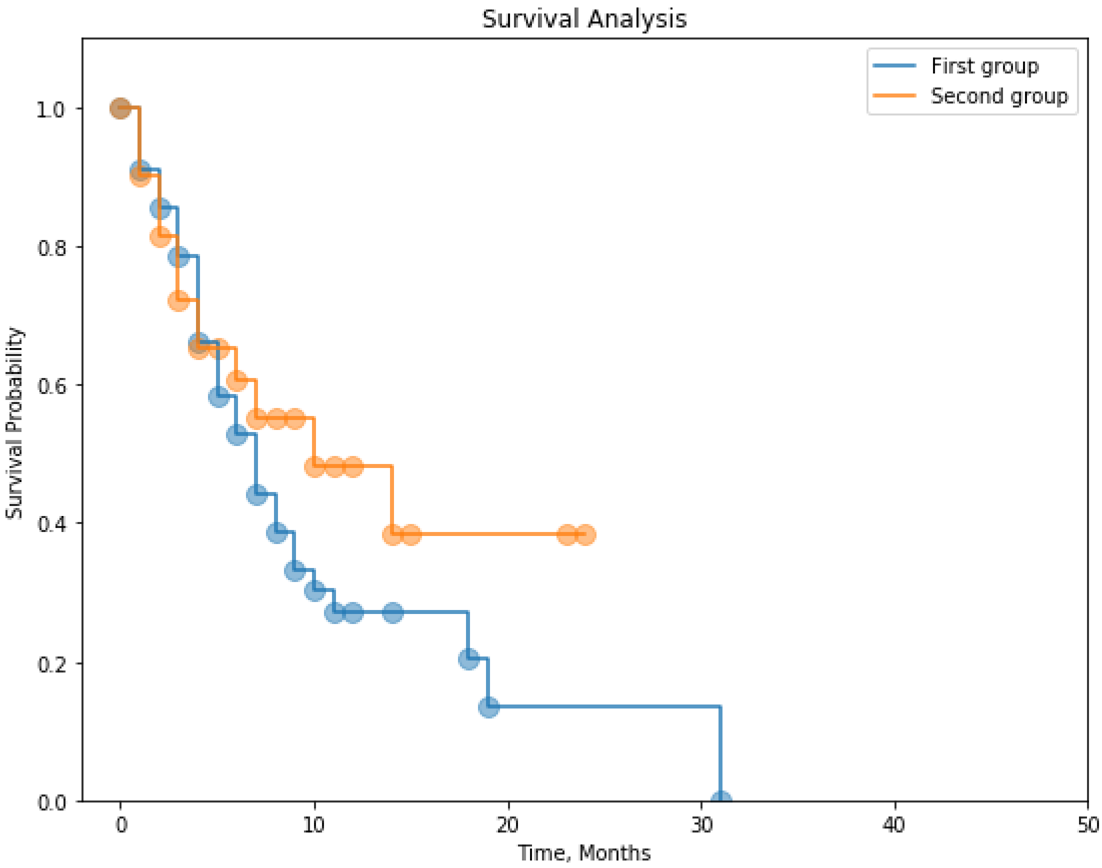
```
# Generate parameters of exponential distribution for both groups from different unifor
death1=np.random.randint(8,12,100)
death2=np.random.randint(4,16,100)
```

```
death2=np.random.randint(4,16,100)

death_time1=np.empty(100)
death_time2=np.empty(100)

# Sample death times from exponential distibution and take ceiling of them
for i in range(100):
    death_time1[i]=np.random.exponential(death1[i],1)[0]
    death_time2[i]=np.random.exponential(death2[i],1)[0]

death_time1=np.ceil(death_time1)
death_time2=np.ceil(death_time2)

# Censoring time- use geometric distibution with success prob. as 0.1
censor1=np.random.geometric(0.1,100)
censor2=np.random.geometric(0.1,100)
```

```
# Find whether death occurs before censoring
death_times1=death_time1[np.where(death_time1<=censor1)]
censor_times1=censor1[np.where(death_time1>censor1)]

death_times2=death_time2[np.where(death_time2<=censor2)]
censor_times2=censor2[np.where(death_time2>censor2)]
```

```
event_days_1,cum_pr_survival_1=KM(death_times1, censor_times1, 100)
event_days_2,cum_pr_survival_2=KM(death_times2, censor_times2, 100)
```

```
plt.figure(figsize=(9,7))
plt.step(event_days_1, cum_pr_survival_1, where='post', label='First group')
plt.scatter(event_days_1, cum_pr_survival_1, alpha=0.5,s=100)

plt.step(event_days_2,cum_pr_survival_2, where='post', label='Second group')
plt.scatter(event_days_2, cum_pr_survival_2, alpha=0.5,s=100)

plt.ylim([0,1.1])
plt.xlim([-2,50])
plt.xlabel('Time, Months')
plt.ylabel('Survival Probability')
plt.title('Survival Analysis')
plt.legend()
plt.show()
```




```
# Perform log rank test and get chi-square value
o1t,o2t,e1t,e2t=log_rank(death_times1, censor_times1, death_times2, censor_times2, 100)
p_value=chisquare([o1t,o2t],[e1t,e2t])[1]

p_value
```

```
0.2779532948855028
```

As the above value is greater than 0.05, this means that both the groups are statistically similar.