МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»

ИНСТИТУТ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

КАФЕДРА ВЫСШЕЙ МАТЕМАТИКИ

Лабораторная работа 2

по дисциплине «Системы массового обслуживания»

ВАРИАНТ 6

Тема: **Многоканальные системы массового обслуживания**

Выполнил:
Студент 4-го курса
Едренников Д.А.

Группа: КМБО-01-20

МОСКВА – 2023

Задание

В рассматриваемых системах массового обслуживания (СМО) состояние в любой момент времени t характеризуется числом заявок, находящихся в системе. Для всех СМО задано количество приборов n , все приборы пронумерованы.

Событием в развитии СМО является переход из одного состояния в другое.

В СМО (D|M|n) и (M|M|n) события могут быть двух типов: 1 – появление в системе новой заявки, 2 – завершение обслуживания заявки прибором (при этом данный прибор освобождается, и, если есть заявки в очереди, то первая из них поступает сразу же на обслуживание в этот прибор). Если при появлении в системе новой заявки есть свободные приборы, то она сразу же принимается на обслуживание свободным прибором с наименьшим номером, в противном случае заявка становится в очередь типа FIFO.

В СМО (M|M|n |m) события могут быть двух типов: 1 – появление в СМО новой заявки, которая принимается на обслуживание свободным прибором или становится в очередь; 2 – появление в СМО новой заявки, которая получает отказ в обслуживании (все приборы и места в очереди заняты), 3 – завершение обслуживания заявки прибором (при этом данный прибор освобождается, и, если есть заявки в очереди, то первая из них поступает сразу же на обслуживание в этот прибор). Если при появлении в системе новой заявки есть свободные приборы, то она сразу же принимается на обслуживание свободным прибором с наименьшим номером и одновременно определяется время ее обслуживания. Если при появлении в системе новой заявки все приборы заняты и есть свободные места в очереди, то заявка становится в очередь типа FIFO.

1. Система массового обслуживания (D|M|n)

   Дано:

   - время между приходом заявок $\Delta T_З$ (заданная постоянная величина);
   - параметр μ показательного распределения времени обслуживания заявки каждым прибором.

Предполагается, что в начальный момент времени t = 0 в СМО нет заявок, т.е. состояние системы 0, и через заданное время $\Delta T_З$ в СМО поступает первая заявка (произойдет событие с номером 1). Момент наступления первого события (типа 1) равен $t_{соб}(1) = \Delta T_З$. После события 1 СМО находится в состоянии 1, в котором она будет оставаться время $t_{обсл}(1)$, определяемое в соответствии с показательным законом распределения с параметром μ. После события 1 система находится в состоянии 1.

2. Система массового обслуживания (M|M|n)

   Дано:

- среднее число заявок λ, поступающих за единицу времени (время между приходом заявок имеет показательное распределение с параметром λ);
- параметр μ показательного распределения времени обслуживания заявки каждым прибором.

Предполагается, что в начальный момент времени t = 0 СМО находится в состоянии 0 и в этот момент определяется время поступления в СМО первой заявки $t_З(1)$ в соответствии с показательным законом распределения с параметром λ, а в момент поступления каждой заявки на обслуживание в прибор определяется $t_{обсл}(1)$ в соответствии с показательным законом распределения с параметром μ.

3. Система массового обслуживания (M|M|n|m)

Дано:

- среднее число заявок λ, поступающих за единицу времени (время между приходом заявок имеет показательное распределение с параметром λ);
- параметр μ показательного распределения времени обслуживания заявки каждым прибором.

Предполагается, что в начальный момент времени t = 0 система находится в состоянии 0 и в этот момент определяется время поступления в систему первой заявки $t_З(1)$ в соответствии с показательным законом распределения с параметром λ, а в момент поступления каждой заявки на обслуживание в прибор определяется время её обслуживания $t_{обсл}(1)$ в соответствии с показательным законом распределения с параметром μ.

Требуется:

1. Провести моделирование первых 100 событий в развитии каждой системы.
2. Составить таблицу 1 с данными о событиях:
   - номер события l;
   - момент наступления события $t_{соб}(l)$;
   - тип события Type (l);
   - состояние СМО C(l) после события l;
   - оставшееся время $t_{ост}(l)$ обслуживания прибором заявки после события l (если после события все приборы свободны, то $t_{ост}(l) = -1$);
   - время ожидания $t_{ожз}(l)$, через которое после события l в СМО появится новая заявка;
   - номер заявки j(l), участвующей в событии l.
   - номер прибор k (l), участвующем в событии l (если заявка встала в очередь или получила отказ в обслуживании, то $k(l) = -1$);
3. Составить таблицу 2 с данными о всех поступивших заявках:
   - номер заявки j;

- момент $t_з(j)$ появления заявки j в СМО;
- номер места в очереди q(j), на которое попала заявка j (если заявка сразу начала обслуживаться, то номер места в очереди q (j) = 0, если заявка получила отказ в обслуживании, то q(j) = -1);
- время пребывания заявки в очереди $t_{оч}(j)$ (если заявка получила отказ в обслуживании, то $t_{оч}(j)=0$;
- момент начала обслуживания заявки $t_{ноб}(j)$ (если заявка получила отказ в обслуживании, то $t_{ноб}(j) =-1$);
- время $t_{обсл}(j)$ обслуживания прибором заявки j (если заявка получила отказ в обслуживании, то $t_{обсл}(j)=0$);
- момент $t_{коб}(j)$ окончания обслуживания заявки j и выхода её из СМО (если заявка получила отказ в обслуживании, то $t_{коб}(j) =t_з(j)$);
- номер прибора k (j), который обслуживал заявку j (если заявка получила отказ в обслуживании, то k(j)=-2).

4. Составить таблицу 3 с данными о приборах вида:

k	N(k)	$t_{зна}(k)$	$t_{пр}(k)$	$\Delta_{пр}(k)$
1	N(1)	$t_{зна}(1)$	$t_{пр}(1)$	$\Delta_{пр}(1)$
…	…	…	…	…
n	N(n)	$t_{зна}(n)$	$t_{пр}(n)$	$\Delta_{пр}(n)$
	$\sum_{k=0}^{n} N(k)$	$\frac{1}{n}\sum_{k=0}^{n} t_{зна}(k)$	$\frac{1}{n}\sum_{k=0}^{n} t_{пр}(k)$	$\frac{1}{n}\sum_{k=0}^{n} \Delta_{пр}(k)$

Где

k – номер прибора;

N(k) − общее число заявок, поступивших на обслуживание в прибор k на интервале $[0, t_{соб}(100)]$;

$t_{зан}(k)$ - общее время занятости прибора k на интервале $[0, t_{соб}(100)]$;

$t_{пр}(k)$ − общее время простоя прибора k на интервале $[0, t_{соб}(100)]$;

$\Delta_{пр}(k) = \frac{t_{пр}(k)}{t_{соб}(100)}$ − коэффициент простоя прибора k на интервале $[0, t_{соб}(100)]$;

5. Для СМО (D|M|n) составить таблицу 4 с данными о состояниях вида:

Состояние	$R_i(100)$	$v_i(100)$	$T_i(100)$	$\Delta_i(100)$
0	$R_0(100)$	$v_0(100)$	$T_0(100)$	$\Delta_0(100)$
1	$R_1(100)$	$v_1(100)$	$T_1(100)$	$\Delta_1(100)$
2	$R_2(100)$	$v_2(100)$	$T_2(100)$	$\Delta_2(100)$
….	….	….	….	….
	$\sum_i R_i(100)$	$\sum_i v_i(100)$	$\sum_i T_i(100)$	$\sum_i \Delta_i(100)$

Для СМО (M|M|n) и (M|M|n|m) составить таблицу 4 с данными о состояниях вида:

Состояние	r_i	$R_i(100)$	$v_i(100)$	$T_i(100)$	$\Delta_i(100)$
0	r_o	$R_0(100)$	$v_0(100)$	$T_0(100)$	$\Delta_0(100)$
1	r_1	$R_1(100)$	$v_1(100)$	$T_1(100)$	$\Delta_1(100)$
2	r_2	$R_2(100)$	$v_2(100)$	$T_2(100)$	$\Delta_2(100)$
….	…	….	….	….	….
	$\sum_i r_i(100)$	$\sum_i R_i(100)$	$\sum_i v_i(100)$	$\sum_i T_i(100)$	$\sum_i \Delta_i(100)$

Где:

$R_i(100)$ – число попаданий СМО в состояние i в событиях с 1-го по 100 ;

$v_i(100) = \frac{R_i(100)}{100}$ – относительная частота попадания СМО в состояние i в событиях с 1-го по 100 ;

$T_i(100)$ – общее время пребывания СМО в состоянии i на интервале $[0, t_{соб}(100)]$;

$\Delta_i(100) = \frac{T_i(100)}{t_{соб}(100)}$ – доля времени пребывания СМО в состоянии i на интервале $[0, t_{соб}(100)]$;

r_i – теоретическое значение стационарной вероятности для состояния i.

6. Найти:
   - число заявок J(100) , поступивших в СМО на интервале $[0, t_{соб}(100)]$;
   - число JF(100) полностью обслуженных заявок на интервале $[0, t_{соб}(100)]$;
   - среднее число заявок, находившихся в СМО, на интервале $[0, t_{соб}(100)]$, которое находится по формуле $\overline{z} = \frac{1}{100} \sum_{l=1}^{100} z(l)$, где z (l) − число заявок в СМО после события l ;

- среднее время пребывания заявок в очереди на интервале [0, $t_{соб}(100)$], которое находится по формуле $\overline{t_{оч}}$ (100) = $\frac{1}{JF(100)}\sum_{j=1}^{JF(100)} t_{оч}(j)$;
- среднее время пребывания заявок в СМО на интервале [0, $t_{соб}(100)$], которое находится по формуле $\overline{t_{СМО}}(100)$ = $\frac{1}{JF(100)}\sum_{j=1}^{JF(100)}[t_{коб}(j) - t_з(j)]$;

Для СМО (M|M|n) и (M|M|n|m) найти дополнительно теоретические значения:

$\overline{k}$ – среднее число занятых приборов;

$\overline{r}$ – средняя длина очереди;

$\overline{z}$ – среднее число заявок в СМО;

$\overline{t}_{оч}$ – среднее время пребывания заявок в очереди;

$\overline{t}_{СМО}$ – среднее время пребывания заявок в СМО.

Для СМО (M|M|n|m) найти теоретическую вероятность отказа в обслуживании.

Вывод результатов проводить с округлением до 0,00001.

Краткие теоретические сведения

Система массового обслуживания (СМО) - это математическая модель систем, предназначенных для обслуживания заявок (требований, запросов, клиентов, заказчиков...), поступающих в нее, как правило, в случайные моменты времени.

Для $(M|M|n)$:

— стационарные вероятности состояний (при $\nu < 1$):

$$\begin{cases} r_0 = \left(1 + \rho + \frac{\rho^2}{2!} + \cdots + \frac{\rho^{n-1}}{(n-1)!} + \frac{\rho^n}{n!} * \frac{1}{\nu - 1}\right)^{-1} \\ r_k = \frac{\rho^k}{k!} r_0, 1 \le k \le n \\ \rho = \frac{\lambda}{\mu} \\ \nu = \frac{\lambda}{n\mu} = \frac{\rho}{n} \end{cases}$$

— среднее число занятых приборов:

$\overline{k} = \rho$

— средняя длина очереди:

$\overline{r} = \frac{\nu\, r_n}{(1-\nu)^2}$

— среднее время пребывания в очереди:

$\overline{t}_{\text{оч}} = \frac{\overline{r}}{\lambda}$

— среднее время пребывания заявок в СМО:

$\overline{t}_{\text{СМО}} = \frac{\overline{z}}{\lambda}$

Для $(M|M|n|m)$:

— стационарные вероятности состояний (при $\nu < 1$):

$$\begin{cases} r_0 = \left(1 + \rho + \frac{\rho^2}{2!} + \cdots + \frac{\rho^{n-1}}{(n-1)!} + \frac{\rho^n}{n!} * \frac{1}{\nu - 1}\right)^{-1} \\ r_k = \frac{\rho^k}{k!} r_0, 1 \le k \le n \\ r_{n+l} = \nu^l r_n, l = 1, \ldots \\ \rho = \frac{\lambda}{\mu} \\ \nu = \frac{\lambda}{n\mu} = \frac{\rho}{n} \end{cases}$$

— вероятность отказа:

$P_{\text{отк}} = r_{n+m}$

— среднее число занятых приборов:

$\overline{k} = \rho(1 - P_{\text{отк}})$

— средняя длина очереди:

$\overline{r} = \nu r_n * \frac{1-(m+1)\nu^m + m\nu^{m+1}}{(1-\nu)^2}$

— среднее число заявок:

$\overline{z} = \overline{r} + \overline{k}$

— среднее время пребывания в очереди:

$\overline{t}_{\text{оч}} = \frac{\overline{r}}{\lambda}$

— среднее время пребывания заявок в СМО:

$\overline{t}_{\text{СМО}} = \frac{\overline{z}}{\lambda}$

Используемые функции из языка python:

expon.rvs(scale=1 / u, size=100) – генерации случайных значений из экспоненциального распределения.

Результаты расчетов

СМО (D|M5):

Вариант 6. $\Delta T_З$ = 0,17; μ = 1,214

Таблица №1

l	$t_{соб}$(l)	Type(l)	состояние СМО C(l) после события l	$t_{ост}$(l)	$t_{ожз}$(l)	j(l)	k(l)
1	0.17	1	1	0.20894	0.17	1	1
2	0.34	1	2	0.03894	0.17	2	2
3	0.37894	2	1	0.20148	0.13106	1	2
4	0.51	1	2	0.07042	0.17	3	1
5	0.58042	2	1	0.10361	0.09958	2	1
6	0.68	1	2	0.00403	0.17	4	2
7	0.68403	2	1	0.28348	0.16597	3	2
8	0.85	1	2	0.11751	0.17	5	1
9	0.96751	2	1	0.46103	0.05249	4	1
10	1.02	1	2	0.32818	0.17	6	2
11	1.19	1	3	0.15818	0.17	7	3
12	1.34818	2	2	0.08036	0.01182	6	1
13	1.36	1	3	0.05176	0.17	8	2
14	1.41176	2	2	0.01678	0.11824	8	1
15	1.42854	2	1	0.40366	0.10146	5	3
16	1.53	1	2	0.3022	0.17	9	1
17	1.7	1	3	0.12154	0.17	10	2
18	1.82154	2	2	0.01066	0.04846	10	3
19	1.8322	2	1	0.21578	0.0378	7	1
20	1.87	1	2	0.08447	0.17	11	2
21	1.95447	2	1	0.09352	0.08553	11	1
22	2.04	1	2	0.00799	0.17	12	2
23	2.04799	2	1	0.5086	0.16201	9	2
24	2.21	1	2	0.07558	0.17	13	1
25	2.28558	2	1	0.271	0.09442	13	2
26	2.38	1	2	0.17658	0.17	14	1
27	2.55	1	3	0.00658	0.17	15	3
28	2.55658	2	2	0.78794	0.16342	12	1
29	2.72	1	3	0.16168	0.17	16	2
30	2.88168	2	2	0.46285	0.00832	16	1

31	2.89	1	3	0.44373	0.17	17	2
32	3.06	1	4	0.02876	0.17	18	4
33	3.08876	2	3	0.24497	0.14124	18	2
34	3.23	1	4	0.10373	0.17	19	4
35	3.33373	2	3	0.0024	0.06627	17	4
36	3.33614	2	2	0.00839	0.06386	19	1
37	3.34452	2	1	0.44583	0.05548	14	3
38	3.4	1	2	0.39036	0.17	20	1
39	3.57	1	3	0.22036	0.17	21	2
40	3.74	1	4	0.05036	0.17	22	4
41	3.79036	2	3	0.79964	0.11964	15	2
42	3.91	1	4	0.02906	0.17	23	3
43	3.93906	2	3	0.65093	0.14094	23	2
44	4.08	1	4	0.50999	0.17	24	3
45	4.25	1	5	0.33999	0.17	25	5
46	4.42	1	6	0.16999	0.17	26	-1
47	4.58999	2	5	0.14903	0.00001	21	1
48	4.59	1	6	0.14902	0.17	27	-1
49	4.73902	2	5	0.17888	0.02098	20	4
50	4.76	1	6	0.1579	0.17	28	-1
51	4.9179	2	5	0.01487	0.0121	22	5
52	4.93	1	6	0.00277	0.17	29	-1
53	4.93277	2	5	0.03655	0.16723	25	5
54	4.96933	2	4	0.10145	0.13067	29	3
55	5.07077	2	3	0.09448	0.02923	24	4
56	5.1	1	4	0.06525	0.17	30	3
57	5.16525	2	3	0.57365	0.10475	28	3
58	5.27	1	4	0.04985	0.17	31	4
59	5.31985	2	3	0.41906	0.12015	31	3
60	5.44	1	4	0.29891	0.17	32	4
61	5.61	1	5	0.12891	0.17	33	5
62	5.73891	2	4	0.16277	0.04109	30	1
63	5.78	1	5	0.12168	0.17	34	3
64	5.90168	2	4	1.53772	0.04832	27	3
65	5.95	1	5	0.37415	0.17	35	1
66	6.12	1	6	0.20415	0.17	36	-1
67	6.29	1	7	0.03415	0.17	37	-1
68	6.32415	2	6	0.43366	0.13585	35	1
69	6.46	1	7	0.29781	0.17	38	-1
70	6.63	1	8	0.12781	0.17	39	-1
71	6.75781	2	7	0.10008	0.04219	36	1
72	6.8	1	8	0.05789	0.17	40	-1
73	6.85789	2	7	0.58151	0.11211	37	3

74	6.97	1	8	0.4694	0.17	41	-1
75	7.14	1	9	0.2994	0.17	42	-1
76	7.31	1	10	0.1294	0.17	43	-1
77	7.4394	2	9	0.0993	0.0406	34	1
78	7.48	1	10	0.0587	0.17	44	-1
79	7.5387	2	9	0.01495	0.1113	38	3
80	7.55365	2	8	0.03297	0.09635	39	2
81	7.58662	2	7	0.26934	0.06338	26	3
82	7.65	1	8	0.20596	0.17	45	-1
83	7.82	1	9	0.03596	0.17	46	-1
84	7.85596	2	8	0.23878	0.13404	41	1
85	7.99	1	9	0.10474	0.17	47	-1
86	8.09474	2	8	0.03636	0.06526	40	1
87	8.1311	2	7	0.17168	0.0289	44	2
88	8.16	1	8	0.14277	0.17	48	-1
89	8.30277	2	7	0.43516	0.02723	42	2
90	8.33	1	8	0.40793	0.17	49	-1
91	8.5	1	9	0.23793	0.17	50	-1
92	8.67	1	10	0.06793	0.17	51	-1
93	8.73793	2	9	0.04635	0.10207	46	4
94	8.78428	2	8	0.06107	0.05572	32	5
95	8.84	1	9	0.00535	0.17	52	-1
96	8.84535	2	8	0.01544	0.16465	33	4
97	8.86079	2	7	0.03259	0.14921	48	4
98	8.89338	2	6	0.1513	0.11662	50	1
99	9.01	1	7	0.03468	0.17	53	-1
100	9.04468	2	6	0.0514	0.13532	45	2

Таблица №2

j	$t_з(j)$	q(j)	$t_{оч}(j)$	$t_{ноб}(j)$	$t_{обсл}(j)$	$t_{коб(j)}$	k(j)
1	0.17	0	0.	0.17	0.20894	0.37894	1
2	0.34	0	0.	0.34	0.24042	0.58042	2
3	0.51	0	0.	0.51	0.17403	0.68403	1
4	0.68	0	0.	0.68	0.28751	0.96751	2
5	0.85	0	0.	0.85	0.57854	1.42854	1
6	1.02	0	0.	1.02	0.32818	1.34818	2
7	1.19	0	0.	1.19	0.6422	1.8322	3
8	1.36	0	0.	1.36	0.05176	1.41176	2
9	1.53	0	0.	1.53	0.51799	2.04799	1

10	1.7	0	0.	1.7	0.12154	1.82154	2
11	1.87	0	0.	1.87	0.08447	1.95447	2
12	2.04	0	0.	2.04	0.51658	2.55658	2
13	2.21	0	0.	2.21	0.07558	2.28558	1
14	2.38	0	0.	2.38	0.96452	3.34452	1
15	2.55	0	0.	2.55	1.24036	3.79036	3
16	2.72	0	0.	2.72	0.16168	2.88168	2
17	2.89	0	0.	2.89	0.44373	3.33373	2
18	3.06	0	0.	3.06	0.02876	3.08876	4
19	3.23	0	0.	3.23	0.10614	3.33614	4
20	3.4	0	0.	3.4	1.33902	4.73902	1
21	3.57	0	0.	3.57	1.01999	4.58999	2
22	3.74	0	0.	3.74	1.1779	4.9179	4
23	3.91	0	0.	3.91	0.02906	3.93906	3
24	4.08	0	0.	4.08	0.99077	5.07077	3
25	4.25	0	0.	4.25	0.68277	4.93277	5
26	4.42	1	2.18046	4.58999	2.99662	7.58662	2
27	4.59	1	2.15948	4.73902	1.16266	5.90168	1
28	4.76	1	2.16836	4.9179	0.24735	5.16525	4
29	4.93	1	2.01323	4.93277	0.03655	4.96933	5
30	5.1	0	0.	5.1	0.63891	5.73891	3
31	5.27	0	0.	5.27	0.04985	5.31985	4
32	5.44	0	0.	5.44	3.34428	8.78428	4
33	5.61	0	0.	5.61	3.23535	8.84535	5
34	5.78	0	0.	5.78	1.6594	7.4394	3
35	5.95	0	0.	5.95	0.37415	6.32415	1
36	6.12	1	2.21461	6.32415	0.43366	6.75781	1
37	6.29	2	2.09637	6.75781	0.10008	6.85789	1
38	6.46	2	2.36003	6.85789	0.68081	7.5387	1
39	6.63	3	2.70801	7.4394	0.11425	7.55365	3
40	6.8	3	2.6381	7.5387	0.55604	8.09474	1
41	6.97	3	3.04961	7.55365	0.30231	7.85596	3
42	7.14	4	3.00115	7.58662	0.71616	8.30277	2
43	7.31	5	2.91561	7.85596	1.31168	9.16764	3
44	7.48	5	2.84491	8.09474	0.03636	8.1311	1
45	7.65	3	2.78617	8.1311	0.91358	9.04468	1
46	7.82	4	2.73771	8.30277	0.43516	8.73793	2
47	7.99	4	2.80649	8.73793	0.35815	9.09608	2
48	8.16	3	2.72298	8.78428	0.07651	8.86079	4
49	8.33	3	2.98814	8.84535	0.85657	9.70192	5
50	8.5	4	2.93968	8.86079	0.03259	8.89338	4
51	8.67	5	2.85414	8.89338	0.41607	9.30945	4
52	8.84	4	2.7071	9.04468	2.12136	11.16604	1

53	9.01	2	-1	-1	-1	-1	-1

Таблица №3

k	N(K)	$t_{зан}$(k)	$t_{пр}$(k)	$\Delta_{пр}$(k)
1	16	8.11596	0.92872	0.10268
2	14	7.71055	1.33413	0.1475
3	9	6.80598	2.2387	0.24752
4	9	5.21468	3.83	0.42345
5	4	4.15401	4.89067	0.54072
	52	6.40024	2.64444	0.29237

Таблица №4

Состояние	R_i(100)	v_i(100)	T_i(100)	Δ_i(100)
0	1	0.01	0.17	0.0188
1	10	0.1	1.1558	0.12779
2	17	0.17	1.29833	0.14355
3	14	0.14	1.49808	0.16563
4	11	0.11	0.85788	0.09485
5	8	0.08	0.66022	0.073
6	8	0.08	0.90216	0.09975
7	9	0.09	0.54524	0.06028
8	11	0.11	1.11299	0.12305
9	8	0.08	0.58794	0.065
10	3	0.03	0.25603	0.02831
	100	1	9.04468	1

Число заявок J(100), поступивших в СМО на интервале [0, 9.04468] = 53.
Число JF(100) полностью обслуженных заявок на интервале [0, 9.04468] = 47.
Среднее число заявок, находившихся в СМО, на интервале [0, 9.04468] = 4.71.
Среднее время пребывания заявок в очереди на интервале [0, 9.04468] = 1.21254.
Среднее время пребывания заявок в СМО на интервале [0, 9.04468] = 0.6416.

СМО (M|M|5):
Вариант 6; λ = 5,865; μ = 1,214;

Таблица №1

l	$t_{соб}$(l)	Type(l)	состояние СМО C(l) после события l	$t_{ост}$(l)	$t_{ожз}$(l)	j(l)	k(j)
1	0.13453	1	1	0.14096	0.1552	1	1
2	0.27549	2	0	-1.	0.01423	1	1
3	0.28972	1	1	1.526	0.02833	2	1
4	0.31805	1	2	1.36619	0.04747	3	2
5	0.36552	1	3	1.31872	0.05784	4	3
6	0.42336	1	4	0.01551	0.09628	5	4
7	0.43887	2	3	1.24538	0.08077	5	2
8	0.51964	1	4	0.50005	0.16774	6	4
9	0.68739	1	5	0.31878	0.04122	7	5
10	0.72861	1	6	0.27756	0.26759	8	-1
11	0.9962	1	7	0.00997	0.02089	9	-1
12	1.00617	2	6	0.01353	0.01092	7	4
13	1.01709	1	7	0.00261	0.00621	10	-1
14	1.0197	2	6	0.4722	0.00361	6	4
15	1.0233	1	7	0.46859	0.12036	11	-1
16	1.14366	1	8	0.34823	0.01673	12	-1
17	1.1604	1	9	0.3315	0.49141	13	-1
18	1.4919	2	8	0.19235	0.15991	9	2
19	1.65181	1	9	0.03244	0.13555	14	-1
20	1.68425	2	8	0.13148	0.10312	3	1
21	1.78736	1	9	0.02836	0.24225	15	-1
22	1.81572	2	8	0.02245	0.21389	2	4
23	1.83817	2	7	0.28873	0.19144	10	1
24	2.02961	1	8	0.09729	0.03841	16	-1
25	2.06802	1	9	0.05888	0.01128	17	-1
26	2.0793	1	10	0.04761	0.20049	18	-1
27	2.1269	2	9	0.18899	0.15288	12	2
28	2.27979	1	10	0.03611	0.21556	19	-1
29	2.31589	2	9	0.38736	0.17945	11	1
30	2.49534	1	10	0.20791	0.33838	20	-1
31	2.70325	2	9	0.02513	0.13047	14	4
32	2.72838	2	8	0.00023	0.10534	13	2
33	2.72861	2	7	0.32237	0.10511	15	1
34	2.83372	1	8	0.21726	0.04806	21	-1
35	2.88178	1	9	0.1692	0.08154	22	-1
36	2.96333	1	10	0.08765	0.03342	23	-1

37	2.99674	1	11	0.05424	0.01886	24	-1
38	3.0156	1	12	0.03538	0.58151	25	-1
39	3.05098	2	11	0.08192	0.54613	16	2
40	3.1329	2	10	0.0145	0.46421	18	3
41	3.1474	2	9	0.0143	0.44971	4	3
42	3.1617	2	8	0.04209	0.43541	21	1
43	3.20379	2	7	0.18538	0.39332	19	3
44	3.38918	2	6	0.04088	0.20794	22	5
45	3.43006	2	5	0.05239	0.16706	8	4
46	3.48244	2	4	0.06946	0.11467	17	5
47	3.5519	2	3	0.07959	0.04521	25	1
48	3.59711	1	4	0.03438	0.18704	26	4
49	3.63149	2	3	0.46643	0.15266	23	2
50	3.78415	1	4	0.2595	0.03363	27	1
51	3.81778	1	5	0.22587	0.16355	28	5
52	3.98133	1	6	0.06232	1.05524	29	-1
53	4.04365	2	5	0.05426	0.99292	27	2
54	4.09791	2	4	0.7335	0.93865	20	3
55	4.83141	2	3	0.12833	0.20515	24	5
56	4.95974	2	2	0.19146	0.07683	28	4
57	5.03657	1	3	0.11463	0.20401	30	2
58	5.1512	2	2	0.09644	0.08938	26	2
59	5.24058	1	3	0.00706	0.08977	31	3
60	5.24764	2	2	0.38857	0.08271	30	1
61	5.33035	1	3	0.03299	0.07839	32	2
62	5.36334	2	2	0.27286	0.0454	32	1
63	5.40874	1	3	0.16091	0.00064	33	2
64	5.40938	1	4	0.16026	0.11498	34	4
65	5.52437	1	5	0.04528	0.09689	35	5
66	5.56965	2	4	0.06656	0.05161	33	1
67	5.62125	1	5	0.01495	0.18284	36	2
68	5.63621	2	4	0.62664	0.16788	29	4
69	5.80409	1	5	0.45875	0.28195	37	1
70	6.08604	1	6	0.1768	0.09634	38	-1
71	6.18238	1	7	0.08046	0.05746	39	-1
72	6.23984	1	8	0.023	0.2999	40	-1
73	6.26284	2	7	0.23946	0.2769	34	1
74	6.50231	2	6	0.10134	0.03744	37	5
75	6.53974	1	7	0.0639	0.31262	41	-1
76	6.60365	2	6	0.13056	0.24872	35	4
77	6.73421	2	5	0.01687	0.11816	38	5
78	6.75107	2	4	0.1121	0.10129	40	2
79	6.85237	1	5	0.01081	0.52179	42	5

80	6.86317	2	4	0.11609	0.51098	36	5
81	6.97927	2	3	0.33219	0.39489	42	4
82	7.31146	2	2	0.04217	0.0627	41	1
83	7.35363	2	1	0.10657	0.02052	39	3
84	7.37415	1	2	0.08605	0.07495	43	1
85	7.4491	1	3	0.01109	0.32229	44	2
86	7.4602	2	2	0.07442	0.3112	31	2
87	7.53462	2	1	0.0886	0.23678	44	1
88	7.62322	2	0	-1.	0.14818	43	1
89	7.7714	1	1	0.86465	0.69479	45	1
90	8.46619	1	2	0.08699	0.34853	46	2
91	8.55318	2	1	0.08287	0.26154	46	1
92	8.63605	2	0	-1.	0.17867	45	1
93	8.81472	1	1	0.45011	0.00979	47	1
94	8.8245	1	2	0.44032	0.34456	48	2
95	9.16906	1	3	0.09576	0.1441	49	3
96	9.26482	2	2	0.81427	0.04833	47	3
97	9.31316	1	3	0.76593	0.00159	50	1
98	9.31474	1	4	0.76435	0.24096	51	4
99	9.5557	1	5	0.52339	0.20725	52	5
100	9.76295	1	6	0.31614	0.14154	53	-1

Таблица №2

j	$t_з(j)$	q(j)	$t_{оч}(j)$	$t_{ноб}(j)$	$t_{обсл}(j)$	$t_{коб(j)}$	k(j)
1	0.13453	0	0.	0.13453	0.14096	0.27549	1.
2	0.28972	0	0.	0.28972	1.526	1.81572	1.
3	0.31805	0	0.	0.31805	1.36619	1.68425	2.
4	0.36552	0	0.	0.36552	2.78188	3.1474	3.

5	0.42336	0	0.	0.42336	0.01551	0.43887	4.
6	0.51964	0	0.	0.51964	0.50005	1.0197	4.
7	0.68739	0	0.	0.68739	0.31878	1.00617	5.
8	0.72861	1	6.31767	1.00617	2.42389	3.43006	5.
9	0.9962	2	6.52228	1.0197	0.4722	1.4919	4.
10	1.01709	2	4.07931	1.4919	0.34627	1.83817	4.
11	1.0233	2	5.16143	1.68425	0.63165	2.31589	2.
12	1.14366	3	5.35226	1.81572	0.31118	2.1269	1.
13	1.1604	4	6.22574	1.83817	0.89021	2.72838	4.
14	1.65181	4	6.00297	2.1269	0.57635	2.70325	1.
15	1.78736	4	6.09283	2.31589	0.41272	2.72861	2.
16	2.02961	3	3.6134	2.70325	0.34773	3.05098	1.
17	2.06802	4	4.32905	2.72838	0.75406	3.48244	4.
18	2.0793	5	4.72207	2.72861	0.40429	3.1329	2.
19	2.27979	5	4.5171	3.05098	0.15281	3.20379	1.

20	2.49534	5	5.02227	3.1329	0.96501	4.09791	2.
21	2.83372	3	3.26818	3.1474	0.0143	3.1617	3.
22	2.88178	4	3.44759	3.1617	0.22747	3.38918	3.
23	2.96333	5	3.79375	3.20379	0.4277	3.63149	1.
24	2.99674	6	5.20257	3.38918	1.44224	4.83141	3.
25	3.0156	7	5.30556	3.43006	0.12185	3.5519	5.
26	3.59711	0	0.	3.59711	1.55409	5.1512	4.
27	3.78415	0	0.	3.78415	0.2595	4.04365	1.
28	3.81778	0	0.	3.81778	1.14196	4.95974	5.
29	3.98133	1	4.73227	4.04365	1.59255	5.63621	1.
30	5.03657	0	0.	5.03657	0.21107	5.24764	2.
31	5.24058	0	0.	5.24058	2.21962	7.4602	3.
32	5.33035	0	0.	5.33035	0.03299	5.36334	2.
33	5.40874	0	0.	5.40874	0.16091	5.56965	2.
34	5.40938	0	0.	5.40938	0.85346	6.26284	4.

35	5.52437	0	0.	5.52437	1.07928	6.60365	5.
36	5.62125	0	0.	5.62125	1.24192	6.86317	2.
37	5.80409	0	0.	5.80409	0.69822	6.50231	1.
38	6.08604	1	4.52104	6.26284	0.47136	6.73421	4.
39	6.18238	2	5.27602	6.50231	0.85132	7.35363	1.
40	6.23984	3	5.36599	6.60365	0.14743	6.75107	5.
41	6.53974	2	4.05141	6.73421	0.57725	7.31146	4.
42	6.85237	0	0.	6.85237	0.1269	6.97927	5.
43	7.37415	0	0.	7.37415	0.24906	7.62322	1.
44	7.4491	0	0.	7.4491	0.08552	7.53462	2.
45	7.7714	0	0.	7.7714	0.86465	8.63605	1.
46	8.46619	0	0.	8.46619	0.08699	8.55318	2.
47	8.81472	0	0.	8.81472	0.45011	9.26482	1.
48	8.8245	0	0.	8.8245	1.72124	10.54574	2
49	9.16906	0	0.	9.16906	0.91003	10.07909	3

50	9.31316	0	0.	9.31316	0.86918	10.18233	1
51	9.31474	0	0.	9.31474	0.8197	10.13444	4
52	9.5557	0	0.	9.5557	1.00405	10.55975	5
53	9.76295	1	-1	-1	-1	-1	-1

Таблица №3

k	N(K)	$t_{зан}$(k)	$t_{пр}$(k)	$\Delta_{пр}$(k)
1	15	8.02876	1.73419	0.17763
2	12	4.81646	4.94649	0.50666
3	6	6.36936	3.39358	0.3476
4	11	6.06297	3.69998	0.37898
5	8	4.56328	5.19967	0.53259
	52	5.96816	3.79478	0,38869

Таблица №4

Состояние	r_i	R_i(100)	v_i(100)	T_i(100)	Δ_i(100)
0	0.00141	4	0.04	0.47561	0.04872
1	0.00682	6	0.06	1.06586	0.10917
2	0.06178	11	0.11	1.01321	0.10378
3	0.02653	13	0.13	1.06077	0.10865
4	0.03204	12	0.12	1.84703	0.18919
5	0.03096	10	0.1	0.88852	0.09101
6	0.02991	9	0.09	0.64966	0.06654
7	0.0289	9	0.09	0.9757	0.09994
8	0.02798	9	0.09	0.45401	0.0465
9	0.02698	9	0.09	0.85688	0.08777
10	0.02607	5	0.05	0.33954	0.03478
11	0.02519	2	0.02	0.10078	0.01032
12	0.02434	1	0.01	0.03538	0.00362
	0.30358	100	1	9.76295	1

Число заявок J(100), поступивших в СМО на интервале [0, 9.76295] = 53.

Число JF(100) полностью обслуженных заявок на интервале [0, 9.76295] = 47.
Среднее число заявок, находившихся в СМО, на интервале [0, 9.76295] = 5.19.
Среднее время пребывания заявок в очереди на интервале [0, 9.76295] = 2.49354.
Среднее время пребывания заявок в СМО на интервале [0, 9.76295] = 0.69207.

Теоритические значения:
$\overline{k}$ = 4.83114
$\overline{r}$ = 0.09342
$\overline{z}$ = 5.02193
$\overline{t_{\text{оч}}}$ = 0.01593
$\overline{t_{\text{СМО}}}$ = 0.83965

СМО (M|M|5|14):
Вариант 6. λ = 1,052. μ = 1,254.

Вариант 6; λ = 5,865; μ = 1,214;

Таблица №1

l	$t_{соб}(l)$	Type(l)	состояние СМО C(l) после события l	$t_{ост}(l)$	$t_{ожз}(l)$	j(l)	k(j)
1	0.07727	1	1	0.44527	0.20885	1	1
2	0.28612	1	2	0.23642	0.48474	2	2
3	0.52254	3	1	2.19402	0.24832	1	2
4	0.77086	1	2	0.71822	0.04441	3	1
5	0.81527	1	3	0.29004	0.0266	4	3
6	0.84187	1	4	0.26344	0.15149	5	4
7	0.99336	1	5	0.11196	0.00845	6	5
8	1.0018	1	6	0.10351	0.55378	7	-1
9	1.10531	3	5	0.38377	0.45027	4	1
10	1.48908	3	4	0.10876	0.0665	3	3
11	1.55558	1	5	0.04226	0.09504	8	1
12	1.59784	3	4	0.00915	0.05279	7	5
13	1.60699	3	3	0.26393	0.04363	6	1
14	1.65063	1	4	0.16131	0.03592	9	3
15	1.68654	1	5	0.1254	0.07718	10	5
16	1.76373	1	6	0.04822	0.47999	11	-1

17	1.81194	3	5	0.05898	0.43177	9	1
18	1.87092	3	4	0.06856	0.3728	8	5
19	1.93948	3	3	0.07515	0.30423	10	3
20	2.01463	3	2	0.61968	0.22908	11	4
21	2.24371	1	3	0.3906	0.03268	12	1
22	2.27639	1	4	0.35792	0.31399	13	3
23	2.59038	1	5	0.04393	0.67474	14	5
24	2.63431	3	4	0.03072	0.63081	5	3
25	2.66503	3	3	0.05152	0.60009	13	2
26	2.71656	3	2	0.2447	0.54856	2	5
27	2.96126	3	1	4.54992	0.30386	14	1
28	3.26512	1	2	1.1203	0.03334	15	2
29	3.29846	1	3	0.96927	0.00621	16	3
30	3.30467	1	4	0.33544	0.00575	17	4
31	3.31043	1	5	0.10027	0.02085	18	5
32	3.33128	1	6	0.07941	0.45789	19	-1
33	3.41069	3	5	0.22943	0.37848	18	4
34	3.64012	3	4	0.04526	0.14905	17	5
35	3.68538	3	3	0.58235	0.10379	19	3
36	3.78917	1	4	0.47857	0.06942	20	4
37	3.85859	1	5	0.40915	0.16502	21	5
38	4.0236	1	6	0.24413	0.14355	22	-1
39	4.16716	1	7	0.10058	0.24903	23	-1
40	4.26774	3	6	0.07034	0.14845	16	4
41	4.33808	3	5	0.04366	0,07811	20	3
42	4.38174	3	4	0.00369	0.03445	22	2
43	4.38543	3	3	0.08287	0.03076	15	4
44	4.41619	1	4	0.05211	1.34814	24	2
45	4.4683	3	3	0.03147	1.29604	23	2
46	4.49976	3	2	0.09034	1.26457	24	5
47	4.5901	3	1	2.92107	1.17423	21	1
48	5.76433	1	2	1.74684	0.01314	25	2
49	5.77748	1	3	0.21863	0.05132	26	3
50	5.8288	1	4	0.16731	0.20123	27	4
51	5.99611	3	3	0.09051	0.03392	26	4
52	6.03002	1	4	0.05659	0.01974	28	3
53	6.04977	1	5	0.03685	0.19044	29	5
54	6.08662	3	4	0.59844	0.15359	27	3
55	6.24021	1	5	0.44485	0.25026	30	4
56	6.49046	1	6	0.19459	0.32338	31	-1
57	6.68506	3	5	0.16751	0.12879	28	5
58	6.81384	1	6	0.03872	0.20151	32	-1
59	6.85256	3	5	0.65861	0.16279	29	1

60	7.01535	1	6	0.49583	0.38725	33	-1
61	7.40259	1	7	0.10858	0.31276	34	-1
62	7.51118	3	6	0.03024	0.20418	12	3
63	7.54141	3	5	0.09898	0,1052	31	2
64	7.6404	3	4	0.0926	0.07496	25	1
65	7.71535	1	5	0.01764	0.05456	35	2
66	7.733	3	4	0.03066	0.03692	33	4
67	7.76366	3	3	0.09991	0.00626	30	2
68	7.76992	1	4	0.09365	0.40824	36	1
69	7.86357	3	3	0.05168	0.31459	35	1
70	7.91524	3	2	0.56318	0.26291	36	5
71	8.17815	1	3	0.30027	0.42762	37	1
72	8.47842	3	2	0.05794	0.12735	32	3
73	8.53636	3	1	0.02857	0.06941	34	1
74	8.56493	3	0	-1.	0.04085	37	1
75	8.60578	1	1	0.29642	0.18346	38	1
76	8.78924	1	2	0.11295	0.33519	39	2
77	8.90219	3	1	0.58254	0.22224	38	2
78	9.12443	1	2	0.04392	0.08839	40	1
79	9.16835	3	1	0.31638	0.04447	40	2
80	9.21282	1	2	0.27191	0.03096	41	1
81	9.24378	1	3	0.24095	0.01959	42	3
82	9.26338	1	4	0.22135	0.09482	43	4
83	9.3582	1	5	0.12653	0.04993	44	5
84	9.40812	1	6	0.07661	0.04386	45	-1
85	9.45199	1	7	0.03274	0.24445	46	-1
86	9.48473	3	6	0.10073	0.21171	39	5
87	9.58546	3	5	0.00955	0.11098	44	1
88	9.59501	3	4	0.10624	0.10143	41	2
89	9.69644	1	5	0.00481	0.03495	47	1
90	9.70124	3	4	0.01649	0.03015	45	3
91	9.71773	3	3	0.45793	0.01366	42	5
92	9.73139	1	4	0.0814	0.37298	48	2
93	9.81279	3	3	0.36287	0.29158	48	5
94	10.10437	1	4	0.07129	0.20646	49	2
95	10.17566	3	3	0.0707	0.13517	46	4
96	10.24637	3	2	0.01337	0.06446	43	1
97	10.25974	3	1	0.30842	0.05109	47	2
98	10.31083	1	2	0.1098	0.16688	50	1
99	10.42064	3	1	0.14753	0.05707	50	2
100	10.47771	1	2	0.09045	0.44025	51	1

Таблица №2

j	$t_з(j)$	q(j)	$t_{оч}(j)$	$t_{ноб}(j)$	$t_{обсл}(j)$	$t_{коб(j)}$	k(j)
1	0.07727	0.	0.	0.07727	0.44527	0.52254	1.
2	0.28612	0.	0.	0.28612	2.43044	2.71656	2.
3	0.77086	0.	0.	0.77086	0.71822	1.48908	1.
4	0.81527	0.	0.	0.81527	0.29004	1.10531	3.
5	0.84187	0.	0.	0.84187	1.79244	2.63431	4.
6	0.99336	0.	0.	0.99336	0.61364	1.60699	5.
7	1.0018	1.	4.01038	1.10531	0.49253	1.59784	3.
8	1.55558	0.	0.	1.55558	0.31534	1.87092	1.
9	1.65063	0.	0.	1.65063	0.16131	1.81194	3.
10	1.68654	0.	0.	1.68654	0.25294	1.93948	5.
11	1.76373	1.	1.47303	1.81194	0.20269	2.01463	3.
12	2.24371	0.	0.	2.24371	5.26746	7.51118	1.
13	2.27639	0.	0.	2.27639	0.38864	2.66503	3.

14	2.59038	0.	0.	2.59038	0.37088	2.96126	5.
15	3.26512	0.	0.	3.26512	1.1203	4.38543	2.
16	3.29846	0.	0.	3.29846	0.96927	4.26774	3.
17	3.30467	0.	0.	3.30467	0.33544	3.64012	4.
18	3.31043	0.	0.	3.31043	0.10027	3.41069	5.
19	3.33128	1.	2.87939	3.41069	0.27469	3.68538	5.
20	3.78917	0.	0.	3.78917	0.54891	4.33808	4.
21	3.85859	0.	0.	3.85859	0.73152	4.5901	5.
22	4.0236	1.	2.01351	4.26774	0.114	4.38174	3.
23	4.16716	2.	2.00018	4.33808	0.13022	4.4683	4.
24	4.41619	0.	0.	4.41619	0.08357	4.49976	2.
25	5.76433	0.	0.	5.76433	1.87606	7.6404	2.
26	5.77748	0.	0.	5.77748	0.21863	5.99611	3.
27	5.8288	0.	0.	5.8288	0.25782	6.08662	4.
28	6.03002	0.	0.	6.03002	0.65503	6.68506	3.

29	6.04977	0.	0.	6.04977	0.8028	6.85256	5.
30	6.24021	0.	0.	6.24021	1.52345	7.76366	4.
31	6.49046	1.	4.29006	6.68506	0.85636	7.54141	3.
32	6.81384	1.	5.50222	6.85256	1.62586	8.47842	5.
33	7.01535	1.	5.52611	7.51118	0.22182	7.733	1.
34	7.40259	2.	6.13381	7.54141	0.99495	8.53636	3.
35	7.71535	0.	0.	7.71535	0.14821	7.86357	2.
36	7.76992	0.	0.	7.76992	0.14533	7.91524	1.
37	8.17815	0.	0.	8.17815	0.38677	8.56493	1.
38	8.60578	0.	0.	8.60578	0.29642	8.90219	1.
39	8.78924	0.	0.	8.78924	0.69549	9.48473	2.
40	9.12443	0.	0.	9.12443	0.04392	9.16835	1.
41	9.21282	0.	0.	9.21282	0.38219	9.59501	1.
42	9.24378	0.	0.	9.24378	0.47395	9.71773	3.
43	9.26338	0.	0.	9.26338	0.98299	10.24637	4.

44	9.3582	0.	0.	9.3582	0.22726	9.58546	5.
45	9.40812	1.	2.35951	9.48473	0.21651	9.70124	2.
46	9.45199	2.	2.90585	9.58546	0.59021	10.17566	5.
47	9.69644	0.	0.	9.69644	0.5633	10.25974	1.
48	9.73139	0.	0.	9.73139	0.0814	9.81279	2.
49	10.10437	0.	0.	10.10437	0.46379	10.56816	2
50	10.31083	0.	0.	10.31083	0.1098	10.42064	1.
51	10.47771	0.	0.	10.47771	0.91358	11.39129	1

Таблица №3

k	N(K)	$t_{зан}$(k)	$t_{пр}$(k)	$\Delta_{пр}$(k)
1	13	4.42374	6.05396	0.57779
2	9	6.56154	3.91617	0.37376
3	12	5.81742	4.66029	0.44478
4	7	5.57128	4.90643	0.46827
5	10	5.59005	4.88766	0.46648
	51	5,59281	4,8849	0,46622

Таблица №4

Состояние	r_i	R_i(100)	v_i(100)	T_i(100)	Δ_i(100)
0	0.01081	2	0.02	0.11812	0.01127
1	0.05224	9	0.09	2.52217	0.24072
2	0.12618	15	0.15	1.52329	0.14538
3	0.20321	18	0.18	1.24079	0.11842
4	0.24543	23	0.23	1.7579	0.16778

5	0.23714	19	0.19	1.8331	0.17495
6	0.08183	11	0.11	1.24042	0.11839
7	0.02824	3	0.03	0.2419	0.02309
	0.98512	100	1	10.47771	1

Число заявок J(100), поступивших в СМО на интервале [0, 10.47771] = 51.
Число JF(100) полностью обслуженных заявок на интервале [0, 10.47771] = 49.
Среднее число заявок, находившихся в СМО, на интервале [0, 10.47771] = 3.67.
Среднее время пребывания заявок в очереди на интервале [0, 10.47771] = 0.79784.
Среднее время пребывания заявок в СМО на интервале [0, 10.47771] = 0.66381.

Теоритические значения:
$\overline{k}$ = 4.83114
$\overline{r}$ = 0.19079
$\overline{z}$ = 5.02193
$\overline{t_{\text{оч}}}$ = 0.03253
$\overline{t_{\text{СМО}}}$ = 0.85625
Теоритическая вероятность отказа в обслуживании – 0.0

Список литературы

1. Кирпичников А.П. Методы прикладной теории массового обслуживания. – М.: URSS, 2018 – 224 с.
2. Ивченко Г.И., Каштанов В.А., Коваленко И.Н. Теория массового обслуживания. – М.: URSS, 2012 – 304 с
3. Введение в теорию массового обслуживания [Электронный ресурс]: учебное пособие для студентов, обучающихся по направлению «Информационные системы и технологии» / Е. К. Белый. --- Петрозаводск: Издательство ПетрГУ, 2014 – 76 с.
4. Лобузов А.А., Гумляева С.Д., Норин Н.В. Задачи по теории случайных процессов. – М.: МИРЭА, 1993 – 68 с.

Приложение

```
import math
import numpy as np
from scipy.stats import expon
from decimal import Decimal

f = open('answer.txt', 'r+')

np.set_printoptions(suppress=True)
u = 1.214
lambd = 5.865
Tt = 0.17
n = 5
m = 14
# Время между заявками

L = list(range(1, 101))
k = [float('inf'), float('inf'), float('inf'), float('inf'), float('inf')]
kj = [-1, -1, -1, -1, -1]
service_time = expon.rvs(scale=1 / u, size=100)
service_time = [0.2089425, 0.24042362, 0.17402998, 0.28750913,
0.57853876, 0.3281805, 0.64220276, 0.05176051, 0.51798679,
                0.12153939, 0.08446545, 0.51658193, 0.0755792, 0.96452253,
1.24035621, 0.1616758, 0.44373353, 0.02876248,
                0.10613567, 1.33902265, 1.01999454, 1.1779021, 0.02906335,
0.99077037, 0.68277089, 2.99662216, 1.16265632,
                0.24735046, 0.03655438, 0.63890711, 0.04985124, 3.34427941,
3.2353478, 1.65940145, 0.37414538, 0.43366022,
                0.10008147, 0.68081293, 0.11424763, 0.55604128, 0.30231069,
0.71615587, 1.31168242, 0.0363553, 0.91358345,
                0.43515835, 0.35815254, 0.07650709, 0.85657331, 0.03259491,
0.41606914, 2.12135799, 1.19049445, 0.11147616,
                0.64662056, 0.14681455, 0.65388675, 1.46099198, 2.28533155,
0.63652264, 0.03639166, 1.06086346, 0.20777427,
                0.64148374, 0.42320609, 0.55148203, 0.65518179, 0.46765486,
0.70194748, 0.1279088, 1.17681095, 0.3871591,
                1.23335323, 0.76315801, 0.90589213, 2.12009595, 0.25639081,
0.68965077, 1.14325731, 0.64423279, 1.22593882,
                0.48603955, 0.36128709, 0.71295673, 1.04207984, 0.7153235,
0.33775967, 0.66214228, 0.32824737, 2.90752264,
                1.37187689, 1.41407363, 0.82795116, 0.35135893, 0.03515003,
0.23388852, 0.55634401, 0.04169105, 0.07508141,
                0.24626046]
```

```
Ttime = []
Ttype = []
condition = []
Tremained = []
Tnew = []
numj = []
numk = []

jN = np.zeros(100)
jP = np.zeros(100)
jQ = np.zeros(100)
jQt = np.zeros(100)
jS = np.zeros(100)
jD = np.zeros(100)
jF = np.zeros(100)
jk = np.zeros(100)

R = np.zeros(100)
V = np.zeros(100)

k41 = np.zeros(n)
k42 = np.zeros(n)
k43 = np.zeros(n)
k44 = np.zeros(n)

i = 0
it = 0

S_con = 0
trimen = 0

R[0] += 1
V[0] += Tt

J5 = 0
JF5 = 0
Z5 = 0
Tq = 0
Tl = 0

Ttime.append(Tt)

jN[it] = i + 1
jP[it] = Tt
```

```
jQ[it] = 0
jQt[it] = 0
jS[it] = Tt
jD[it] = service_time[0]
jF[it] = Tt + service_time[0]
jk[it] = it + 1

Ttype.append(1)
condition.append(1)

# T1 += service_time[i]

Tremained.append(service_time[0])
Tnew.append(Tt)
numj.append(it + 1)
numk.append(it + 1)
trimen = Tt - service_time[0]
S_con = 1
k[0] = service_time[it]
kj[0] = it+1

i += 1
it += 1
J5 += 1


while len(Ttime) != 100:
    if S_con == 0:

        jN[i] = i + 1
        jP[i] = Ttime[-1] + Tnew[-1]
        jQ[i] = 0
        jQt[i] = 0
        jS[i] = Ttime[-1] + Tnew[-1]
        jk[it] = S_con + 1

        V[S_con] += Tnew[-1]
        Ttime.append(Ttime[-1] + Tnew[-1])
        Ttype.append(1)
        condition.append(S_con + 1)

        # T1 += service_time[i]

        k[S_con] = service_time[i]
```

```
                Tremained.append(service_time[i])
                Tnew.append(Tt)
                numj.append(i + 1)
                numk.append(S_con + 1)
                trimen = Tt - service_time[i]
                S_con += 1
                R[S_con] += 1
                k[0] = service_time[i]
                kj[0] = i+1
                k41[0] += 1
                k42[0] += service_time[i]
                # V[S_con] += min(Ts, arrive_time[it + 1])
                it += 1
                i += 1
                J5 += 1
                Z5 += S_con
            elif trimen > 0:
                jS[kj[np.argmin(k)]-1] = Ttime[-1] + k[np.argmin(k)] -
service_time[kj[np.argmin(k)]-1]
                jD[kj[np.argmin(k)]-1] = service_time[kj[np.argmin(k)]-1]
                jF[kj[np.argmin(k)]-1] = Ttime[-1] + Tremained[-1]
                jk[kj[np.argmin(k)]-1] = np.argmin(k) + 1

                V[S_con] += k[np.argmin(k)]
                numj.append(kj[np.argmin(k)])
                Ttime.append(Ttime[-1] + k[np.argmin(k)])
                k = np.subtract(k, k[np.argmin(k)])
                Ttype.append(2)
                condition.append(S_con - 1)
                if S_con <= n:
                    kj[np.argmin(k)] = -1
                    k[np.argmin(k)] = float('inf')
                else:
                    k[np.argmin(k)] = float('inf')
                    iop = np.argmax(k)
                    jS[j] = Ttime[-1]
                    jD[j] = service_time[j]
                    jF[j] = Ttime[-1] + service_time[j]
                    jk[j] = np.argmax(k) + 1
                    k41[iop] += 1
                    k42[iop] += service_time[kj[iop] - 1]
                    k[iop] = service_time[j]
                    kj[iop] = j+1
                    j += 1
```

```
            if j>i:
                i+=1
        if S_con - 1 == 0:
            Tremained.append(-1)
            Tnew.append(trimen)
            Tl += trimen
        else:
            Tremained.append(min(k))
            Tnew.append(trimen)

        trimen = trimen - min(k)
        numk.append(np.argmin(k) + 1)
        S_con -= 1
        R[S_con] += 1
        JF5 += 1
        Z5 += S_con
    elif trimen < 0:
        tp = trimen + min(k)
        k = np.subtract(k, tp)
        jN[i] = i + 1
        jP[i] = Ttime[-1] + tp
        jQ[i] = S_con
        if S_con < n:
            jQt[i] = 0
        else:
            for d in range(S_con):
                jQt[i] += service_time[d + it + 1]
            jQt[i] -= trimen

        if max(k) == float('inf'):
            numk.append(np.argmax(k) + 1)
            jS[it] = (i + 1) * Tt + jQt[i]
            jD[it] = service_time[i]
            jF[it] = (i + 1) * Tt + jQt[i] + service_time[i]
            jk[i] = np.argmax(k) + 1
            k41[np.argmax(k)] += 1
            k42[np.argmax(k)] += service_time[kj[np.argmax(k)] - 1]
            kj[np.argmax(k)] = i+1
            k[np.argmax(k)] = service_time[i]
            j = i+1
        else:
            numk.append(-1)
        V[S_con] += tp
        Ttime.append((i + 1) * Tt)
```

```
        Ttype.append(1)
        condition.append(S_con + 1)
        Tremained.append(min(k))
        Tnew.append(Tt)
        i += 1
        S_con += 1
        R[S_con] += 1
        numj.append(i)
        trimen = Tt - min(k)
        J5 += 1
        Z5 += S_con

Rot = R / 100
Vot = V / Ttime[-1]
Z5 = Z5 / 100
wer = 0
for d3 in range(len(jF)):
    if jF[d3] > 0:
        wer += (jF[d3] - jS[d3])
Tq5 = sum(jQt) / JF5
Tm5 = wer / JF5

for d1 in range(len(jQ)):
    if jQ[d1] < 5:
        jQ[d1] = 0
    else:
        jQ[d1] -= 4

for d2 in range(n):
    if k[d2] != float('inf'):
        k42[d2] -= k[d2]
    k43[d2] = Ttime[-1] - k42[d2]
    k44[d2] = k43[d2] / Ttime[-1]

f.write(str(np.around(L, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Ttime, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Ttype, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
```

```
f.write(str(np.around(condition, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Tremained, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Tnew, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(numj, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(numk, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(jN, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jP, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jQ, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jQt, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jS, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jD, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jF, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(jk, 5)))
f.write('\n')
```

```
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(R, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(V, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Rot, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Vot, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(sum(R), 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(sum(V), 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(sum(Rot), 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(sum(Vot), 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(J5, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(JF5, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Z5, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Tq5, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(Tm5, 5)))
f.write('\n')
f.write('\n')
```

```
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(k41, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(k42, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(k43, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(k44, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')

k = [float('inf'), float('inf'), float('inf'), float('inf'), float('inf')]
kj = [-1, -1, -1, -1, -1]
Ttime = []
Ttype = []
condition = []
Tremained = []
Tnew = []
numj = []
numk = []

jN = np.zeros(100)
jP = np.zeros(100)
jQ = np.zeros(100)
jQt = np.zeros(100)
jS = np.zeros(100)
jD = np.zeros(100)
jF = np.zeros(100)
jk = np.zeros(100)

R = np.zeros(100)
V = np.zeros(100)

k41 = np.zeros(n)
```

```
k42 = np.zeros(n)
k43 = np.zeros(n)
k44 = np.zeros(n)

T0 = Tt
T1 = 0

arrive_time = expon.rvs(scale=1 / lambd, size=105)
arrive_time = [0.13452634, 0.15519843, 0.0283299, 0.04746884,
0.05783817, 0.09628136, 0.16774346, 0.04122172, 0.26758845,
        0.02089449, 0.0062113, 0.1203606, 0.01673303, 0.49141359,
0.13555423, 0.24224803, 0.0384094, 0.01127594,
        0.20048834, 0.2155568, 0.33837971, 0.04805961, 0.08154375,
0.03341779, 0.01885529, 0.58151262, 0.18703896,
        0.03363032, 0.16354833, 1.05523772, 0.20401247, 0.0897726,
0.07839006, 0.00064297, 0.11498224, 0.0968871,
        0.18283561, 0.28195189, 0.09633881, 0.05745809, 0.29990406,
0.31262433, 0.52178662, 0.07495145, 0.32229107,
        0.69479259, 0.3485287, 0.00978547, 0.34455846, 0.14409691,
0.0015866, 0.24095841, 0.20724527, 0.14153678, 0.21186249,
        0.11534158, 0.27668605, 0.23990952, 0.12584108, 0.02072617,
0.18680627, 0.10086884, 0.23286964, 1.29150504,
        0.19749515, 0.14165763, 0.01949458, 0.07271728, 0.06531828,
0.16581318, 0.08882034, 0.26548143, 0.17186677,
        0.04648546, 0.06211913, 0.06518982, 0.12855165, 0.35920409,
0.0315275, 0.0637701, 0.23644445, 0.0669599,
        0.13345968, 0.09329895, 0.041594, 0.19948896, 0.20128085,
0.0413679, 0.05342318, 0.20120517, 0.3631961,
        0.38923137, 0.09698655, 0.04347043, 0.01412543, 0.38276278,
0.20002275, 0.07304276, 0.07767907, 0.20347489,
        0.0347942, 0.16591234, 0.12607382, 0.00059887, 0.18488854]
service_time = expon.rvs(scale=1 / u, size=105)
service_time = [0.14096427, 1.52599583, 1.36619083, 2.78187587,
0.01550708, 0.50005387, 0.31878136, 2.42388767, 0.47219829,
        0.34627476, 0.63164567, 0.31118265, 0.89021229, 0.57634907,
0.41271947, 0.34772789, 0.75406021, 0.40428931,
        0.15281013, 0.96501346, 0.01430193, 0.22747386, 0.42769759,
1.44223716, 0.1218457, 1.55408956, 0.25950125,
        1.14195904, 1.59255417, 0.2110721, 2.21962061, 0.03299088,
0.16090538, 0.85345717, 1.07927905, 1.24191927,
        0.69821569, 0.47136381, 0.85132492, 0.14742841, 0.57725097,
0.12689934, 0.24906341, 0.08551547, 0.86465192,
        0.08699023, 0.45010711, 1.72123805, 0.91003177, 0.86917588,
0.81969871, 1.00404901, 0.35450958, 0.13491099,
```

```
                0.61593932, 0.08734727, 1.32834542, 2.77345115, 2.84737523,
0.11407401, 2.18020131, 0.58461787, 1.20795147,
                0.17302712, 0.62418966, 0.29549693, 0.19365026, 0.12110518,
0.0660802, 0.73917226, 0.69735931, 0.37485889,
                0.5456252, 1.83985775, 2.70396212, 0.07995836, 0.06215885,
1.99436774, 0.77589616, 0.05003577, 2.45220238,
                1.14352085, 0.91552118, 0.43143425, 0.91820335, 0.33512254,
1.63188178, 0.45073285, 0.11618706, 0.98611168,
                0.6218516, 0.13076004, 0.49089057, 0.4967578, 0.63775862,
1.16009522, 0.29896259, 0.58792701, 1.153215,
                1.56923019, 0.54900448, 0.7352573, 0.87693621, 1.33628879,
0.44052683]
        i = 0
        j = 0
        it = i

        S_con = 0
        trimen = 0

        R[0] += 1
        V[0] += arrive_time[i]

        J5 = 0
        JF5 = 0
        Z5 = 0
        Tl = 0

        Ttime.append(arrive_time[i])

        jN[it] = i + 1
        jP[it] = arrive_time[i]
        jQ[it] = 0
        jQt[it] = 0
        jS[it] = arrive_time[i]
        jD[it] = service_time[0]
        jF[it] = arrive_time[i] + service_time[0]
        jk[it] = it + 1

        Ttype.append(1)
        condition.append(1)

        # T1 += service_time[i]

        Tremained.append(service_time[j])
```

```
Tnew.append(arrive_time[i + 1])
k[0] = service_time[j]
kj[0] = it+1
numj.append(it + 1)
numk.append(it + 1)
trimen = arrive_time[it + 1] - service_time[j]
S_con = 1
# V[S_con] += min(service_time[j],arrive_time[i + 1])
i += 1
it += 1
J5 += 1

while len(Ttime) != 100:
    if S_con == 0:

        jN[i] = i + 1
        jP[i] = Ttime[-1] + Tnew[-1]
        jQ[i] = 0
        jQt[i] = 0
        jS[i] = Ttime[-1] + Tnew[-1]
        jk[i] = 1

        V[S_con] += Tnew[-1]
        Ttime.append(Ttime[-1] + Tnew[-1])
        Ttype.append(1)
        condition.append(S_con + 1)

        # T1 += service_time[i]

        Tremained.append(service_time[i])
        Tnew.append(arrive_time[i + 1])
        numj.append(i + 1)
        numk.append(1)
        trimen = arrive_time[i + 1] - service_time[j]
        S_con += 1
        R[S_con] += 1
        k[0] = service_time[j]
        kj[0] = i+1
        # V[S_con] += min(service_time[j], arrive_time[i + 1])
        it += 1
        i += 1
        J5 += 1
        Z5 += S_con
    elif trimen > 0:
```

```
            jS[kj[np.argmin(k)] - 1] = Ttime[-1] + k[np.argmin(k)] -
service_time[kj[np.argmin(k)] - 1]
            jD[kj[np.argmin(k)] - 1] = service_time[kj[np.argmin(k)] - 1]
            jF[kj[np.argmin(k)] - 1] = Ttime[-1] + Tremained[-1]
            jk[kj[np.argmin(k)] - 1] = np.argmin(k) + 1

            V[S_con] += k[np.argmin(k)]
            numj.append(kj[np.argmin(k)])
            Ttime.append(Ttime[-1] + k[np.argmin(k)])
            k = np.subtract(k, k[np.argmin(k)])
            Ttype.append(2)
            condition.append(S_con - 1)
            j += 1
            if S_con <= n:
                kj[np.argmin(k)] = -1
                k[np.argmin(k)] = float('inf')
            else:
                k[np.argmin(k)] = float('inf')
                iop = np.argmax(k)
                jS[it] = Ttime[-1]
                jD[it] = service_time[it]
                jF[it] = Ttime[-1] + service_time[it]
                jk[it] = np.argmax(k) + 1
                k41[iop] += 1
                k42[iop] += service_time[kj[iop] - 1]
                k[iop] = service_time[it]
                kj[iop] = it + 1
                it+=1
                if it>i:
                    i+=1
            if S_con - 1 == 0:
                Tremained.append(-1)
                Tnew.append(trimen)
                Tl += trimen
            else:
                Tremained.append(min(k))
                Tnew.append(trimen)

            trimen = trimen - min(k)
            numk.append(np.argmin(k) + 1)
            S_con -= 1
            R[S_con] += 1
            JF5 += 1
```

```
            Z5 += S_con
        elif trimen < 0:
            tp = trimen + min(k)
            k = np.subtract(k, tp)
            V[S_con] += tp
            jN[i] = i + 1
            jP[i] = Ttime[-1] + tp
            jQ[i] = S_con
            if S_con < n:
                jQt[i] = 0
            else:
                for d in range(S_con):
                    jQt[i] += service_time[d + j + 1]
                jQt[i] -= trimen

            if max(k) == float('inf'):
                numk.append(np.argmax(k) + 1)
                jk[i] = np.argmax(k) + 1
                jS[it] = Ttime[-1] + tp + jQt[i]
                jD[it] = service_time[i]
                jF[it] = Ttime[-1] + tp + service_time[i]
                k41[np.argmax(k)] += 1
                k42[np.argmax(k)] += service_time[kj[np.argmax(k)] - 1]
                kj[np.argmax(k)] = i+1
                k[np.argmax(k)] = service_time[i]
                it = i+1
            else:
                numk.append(-1)

            Ttime.append(Ttime[-1] + tp)
            Ttype.append(1)
            condition.append(S_con + 1)
            Tremained.append(min(k))
            Tnew.append(arrive_time[i + 1])
            i += 1
            S_con += 1
            R[S_con] += 1
            numj.append(i)
            trimen = arrive_time[i] - min(k)
            J5 += 1
            Z5 += S_con

    Rot = R / 100
    print(sum(V))
```

```
print(Ttime[-1])
Vot = V / Ttime[-1]
Z5 = Z5 / 100
wer = 0
for d3 in range(len(jF)):
    if jF[d3] > 0:
        wer += (jF[d3] - jS[d3])
Tq5 = sum(jQt) / JF5
Tm5 = wer / JF5

for d1 in range(len(jQ)):
    if jQ[d1] < 5:
        jQ[d1] = 0
    else:
        jQ[d1] -= 4

for d2 in range(n):
    if k[d2] != float('inf'):
        k42[d2] -= k[d2]
    k43[d2] = Ttime[-1] - k42[d2]
    k44[d2] = k43[d2] / Ttime[-1]

f.write(str(np.around(L, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Ttime, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Ttype, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(condition, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Tremained, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Tnew, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(numj, 5)))
f.write('\n')
f.write('\n')
```

```
f.write('\n')
f.write('\n')
f.write(str(np.around(numk, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(jN, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jP, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jQ, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jQt, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jS, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jD, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jF, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(jk, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(R, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(V, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Rot, 5)))
f.write('\n')
f.write('\n')
```

```
f.write(str(np.around(Vot, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(sum(R), 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(sum(V), 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(sum(Rot), 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(sum(Vot), 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(J5, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(JF5, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Z5, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Tq5, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(Tm5, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(k41, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(k42, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(k43, 5)))
f.write('\n')
```

```
f.write('\n')
f.write(str(np.around(k44, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')

k = [float('inf'), float('inf'), float('inf'), float('inf'), float('inf')]
kj = [-1, -1, -1, -1, -1]
m = 14
baned = []
Ttime = []
Ttype = []
condition = []
Tremained = []
Tnew = []
numj = []
numk = []

jN = np.zeros(100)
jP = np.zeros(100)
jQ = np.zeros(100)
jQt = np.zeros(100)
jS = np.zeros(100)
jD = np.zeros(100)
jF = np.zeros(100)
jk = np.zeros(100)

R = np.zeros(100)
V = np.zeros(100)

k41 = np.zeros(n)
k42 = np.zeros(n)
k43 = np.zeros(n)
k44 = np.zeros(n)

T0 = Tt
T1 = 0

arrive_time = expon.rvs(scale=1 / lambd, size=105)
print(arrive_time)
service_time = expon.rvs(scale=1 / u, size=105)
```

```
print(service_time)
i = 0
j = 0
it = i

S_con = 0
trimen = 0

R[0] += 1
V[0] += arrive_time[i]

J5 = 0
JF5 = 0
Z5 = 0
Tl = 0

Ttime.append(arrive_time[i])

jN[it] = i + 1
jP[it] = arrive_time[i]
jQ[it] = 0
jQt[it] = 0
jS[it] = arrive_time[i]
jD[it] = service_time[0]
jF[it] = arrive_time[i] + service_time[0]
jk[it] = it + 1

Ttype.append(1)
condition.append(1)

# T1 += service_time[i]

Tremained.append(service_time[j])
Tnew.append(arrive_time[i + 1])
k[0] = service_time[j]
kj[0] = it+1
numj.append(it + 1)
numk.append(it + 1)
trimen = arrive_time[it + 1] - service_time[j]
S_con = 1
# V[S_con] += min(service_time[j],arrive_time[i + 1])
i += 1
it += 1
J5 += 1
```

```
        while len(Ttime) != 100:
            if S_con == 0:

                jN[i] = i + 1
                jP[i] = Ttime[-1] + Tnew[-1]
                jQ[i] = 0
                jQt[i] = 0
                jS[i] = Ttime[-1] + Tnew[-1]
                jk[i] = 1

                V[S_con] += Tnew[-1]
                Ttime.append(Ttime[-1] + Tnew[-1])
                Ttype.append(1)
                condition.append(S_con + 1)

                # T1 += service_time[i]

                Tremained.append(service_time[j])
                Tnew.append(arrive_time[i + 1])
                numj.append(i + 1)
                numk.append(1)
                trimen = arrive_time[i + 1] - service_time[j]
                S_con += 1
                R[S_con] += 1
                k[0] = service_time[j]
                kj[0] = i+1
                it += 1
                while it in baned:
                    it += 1
                i += 1
                J5 += 1
                Z5 += S_con
            elif trimen > 0:

                jS[kj[np.argmin(k)] - 1] = Ttime[-1] + k[np.argmin(k)] -
service_time[kj[np.argmin(k)] - 1]
                jD[kj[np.argmin(k)] - 1] = service_time[kj[np.argmin(k)] - 1]
                jF[kj[np.argmin(k)] - 1] = Ttime[-1] + Tremained[-1]
                jk[kj[np.argmin(k)] - 1] = np.argmin(k) + 1

                V[S_con] += k[np.argmin(k)]
                numj.append(kj[np.argmin(k)])
                Ttime.append(Ttime[-1] + k[np.argmin(k)])
```

```
        k = np.subtract(k, k[np.argmin(k)])
        Ttype.append(3)
        condition.append(S_con - 1)
        j += 1
        if S_con <= n:
            kj[np.argmin(k)] = -1
            k[np.argmin(k)] = float('inf')
        else:
            k[np.argmin(k)] = float('inf')
            iop = np.argmax(k)
            k41[iop] += 1
            k42[iop] += service_time[kj[iop] - 1]
            k[iop] = service_time[it]
            kj[iop] = it + 1
            it += 1
            while it in baned:
                it += 1
            if it > i:
                i += 1
        if S_con - 1 == 0:
            Tremained.append(-1)
            Tnew.append(trimen)
            Tl += trimen
        else:
            Tremained.append(min(k))
            Tnew.append(trimen)

        trimen = trimen - min(k)
        numk.append(np.argmin(k) + 1)
        S_con -= 1
        R[S_con] += 1
        JF5 += 1
        Z5 += S_con
    elif trimen < 0:
        if S_con < n + m:
            tp = trimen + min(k)
            k = np.subtract(k, tp)
            V[S_con] += tp
            jN[i] = i + 1
            jP[i] = Ttime[-1] + tp
            jQ[i] = S_con
            if S_con < n:
                jQt[i] = 0
            else:
```

```
            for d in range(S_con):
                jQt[i] += service_time[d + j + 1]
            jQt[i] -= trimen

        if max(k) == float('inf'):
            numk.append(np.argmax(k) + 1)
            jk[i] = np.argmax(k) + 1
            k41[np.argmax(k)] += 1
            k42[np.argmax(k)] += service_time[kj[np.argmax(k)] - 1]
            kj[np.argmax(k)] = i + 1
            k[np.argmax(k)] = service_time[i]
            it = i+1
        else:
            numk.append(-1)

        Ttime.append(Ttime[-1] + tp)
        Ttype.append(1)
        condition.append(S_con + 1)
        Tremained.append(min(k))
        Tnew.append(arrive_time[i + 1])
        i += 1
        S_con += 1
        R[S_con] += 1
        numj.append(i)
        trimen = arrive_time[i] - min(k)
        J5 += 1
        Z5 += S_con
    else:
        tp = trimen + min(k)
        k = np.subtract(k, tp)
        V[S_con] += tp
        jN[i] = i + 1
        jP[i] = Ttime[-1] + tp
        jQ[i] = -1
        jQt[i] = 0
        jS[it] = -1
        jD[it] = 0
        jF[it] = jP[i]
        jk[it] = -2

        Ttime.append(Ttime[-1] + tp)
        Ttype.append(2)
        numk.append(-1)
        condition.append(S_con)
```

```
            Tremained.append(min(k))
            Tnew.append(arrive_time[i + 1])
            baned.append(i)
            i += 1
            R[S_con] += 1
            # V[S_con] += min(-trimen, arrive_time[i])
            numj.append(i)
            trimen = arrive_time[i] - min(k)
            J5 += 1
            Z5 += S_con

Rot = R / 100
print(sum(V))
print(Ttime[-1])
Vot = V / Ttime[-1]
Z5 = Z5 / 100
wer = 0
for d3 in range(len(jF)):
    if jF[d3] > 0:
        wer += (jF[d3] - jS[d3])
Tq5 = sum(jQt) / JF5
Tm5 = wer / JF5

for d1 in range(len(jQ)):
    if jQ[d1] < 5:
        jQ[d1] = 0
    else:
        jQ[d1] -= 4

for d2 in range(n):
    if k[d2] != float('inf'):
        k42[d2] -= k[d2]
    k43[d2] = Ttime[-1] - k42[d2]
    k44[d2] = k43[d2] / Ttime[-1]

f.write(str(np.around(L, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Ttime, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Ttype, 5)))
f.write('\n')
f.write('\n')
```

```
f.write('\n')
f.write(str(np.around(condition, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Tremained, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Tnew, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(numj, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(numk, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(jN, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jP, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jQ, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jQt, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jS, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jD, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(jF, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(jk, 5)))
```

```
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(R, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(V, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Rot, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Vot, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(sum(R), 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(sum(V), 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(sum(Rot), 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(sum(Vot), 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(J5, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(JF5, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Z5, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Tq5, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(Tm5, 5)))
f.write('\n')
```

```
        f.write('\n')
        f.write('\n')
        f.write('\n')
        f.write('\n')
        f.write('\n')
        f.write(str(np.around(k41, 5)))
        f.write('\n')
        f.write('\n')
        f.write(str(np.around(k42, 5)))
        f.write('\n')
        f.write('\n')
        f.write(str(np.around(k43, 5)))
        f.write('\n')
        f.write('\n')
        f.write(str(np.around(k44, 5)))
        f.write('\n')
        f.write('\n')
        f.write('\n')
        f.write('\n')
        f.write('\n')
        f.write('\n')

        ro = lambd / u
        v = ro / m
        Ro1 = 1/(1 + ro + (ro * ro) / 2 + (ro * ro * ro) / 6 + (ro * ro * ro * ro) / 24 +
(ro * ro * ro * ro * ro / 120)/(n-ro))
        Ro2 = 1/(1 + ro + ro * ro / 2 + ro * ro * ro / 6 + ro * ro * ro * ro / 24 + (ro
* ro * ro * ro * ro / 120) * (
                1 + v + pow(v, 2) + pow(v, 3) + pow(v, 4) + pow(v, 5)
                + pow(v, 6) + pow(v, 7) + pow(v, 8) + pow(v, 9)
                + pow(v, 10) + pow(v, 11) + pow(v, 12) + pow(v, 13)
                + pow(v, 14)))
        K1 = ro
        K2 = ro * (1 - ((pow(ro, 5) / 120) * pow(v, m-n) * Ro2))
        print(K2)
        Q1 = v * pow(ro, n) / 120 * Ro1 / ((1 - v) * (1 - v))
        Q2 = v * pow(ro, n) / 120 * Ro2 * (1 - (m + 1) * pow(v, m) + m * pow(v,
m + 1)) / ((1 - v) * (1 - v))
        Z1 = K1 + Q1
        Z2 = K2 + Q2
        tq1 = Q1 / lambd
        tq2 = Q2 / lambd
        tin1 = Z1 / lambd
        tin2 = Z2 / lambd
```

```
f.write(str(np.around(K1, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Q1, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Z1, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(tq1, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(tin1, 5)))
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
f.write(str(np.around(K2, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Q2, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(Z2, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(tq2, 5)))
f.write('\n')
f.write('\n')
f.write(str(np.around(tin2, 5)))
f.write('\n')
f.write('\n')
t = Ro2 * pow(ro, n) * pow(v, m) / 120
print(t)
f.write(str(np.around(t, 16)))
f.write('\n')
f.write(str(np.around(t, 5)))
f.write('\n')
f.write('\n')
```

```
f.write('\n')
f.write('\n')
f.write('\n')
f.write('\n')
r5 = Ro1*pow(ro,5)/math.factorial(5)
l = 0
for i in range(6):
    print(i)
    print(Ro2*pow(ro,i)/math.factorial(i))
    l += (Ro2*pow(ro,i)/math.factorial(i))

for i in range(6,8):
    print(i)
    print(r5*pow(ro/m,i-n))
    l += (r5*pow(ro/m,i-n))
print(l)
```