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 14

Тема: **Многоканальные системы массового обслуживания**

Выполнил:
Студент 4-го курса
Малов И.М.

Группа: КМБО-01-20

МОСКВА 2023

Задание

В рассматриваемых системах массового обслуживания (СМО) состояние в любой момент времени t характеризуется числом заявок, находящихся в СМО. 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.

I. Система массового обслуживания $(D|M|n)$.

Дано:

— время между приходом заявок $\Delta T_з$ (заданная постоянная величина);

— параметр μ показательного распределения времени обслуживания заявки прибором.

В момент поступлении каждой заявки на обслуживание в прибор определяется время её обслуживания $t_{\text{обсл}}$ в соответствии с показательным законом распределения с заданным параметром μ.

Предполагается, что в начальный момент времени $t = 0$ в СМО нет заявок, т.е. состояние системы 0, и через заданное время $\Delta T_{\text{з}}$ в СМО поступит первая заявка (произойдет событие с номером 1). Момент наступления первого события (типа 1) равен $t_{\text{соб}}(1) = \Delta T_{\text{з}}$, в этот момент определяется время обслуживания $t_{\text{обсл}}(1)$ заявки 1 в соответствии с показательным законом распределения с параметром μ. После события 1 система находится в состоянии 1.

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

III. 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 соответствии заявки $t_з(1)$ в с показательным законом распределения с параметром λ, а в момент поступления каждой заявки на обслуживание в прибор определяется время сё 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 события:

— номер события l;

— момент наступления события $t_{соб}(l)$;

— тип события $Type(l)$;

— состояние СМО $C(l)$ после события l;

— минимальное ставшееся время $t_{осмин}(l)$ обслуживания прибором заявки после события l (если после события прибор свободен, то $t_{осмин}(l) = -1$);

— время ожидания $t_{ожз}(l)$, через которое после события l в СМО появится новая заявка;

— номер заявки $j(l)$, участвующей в событии l;

— номер прибора $k(l)$, участвующего 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 $t_{\text{оч}}(j)$ (если заявка получила отказ в обслуживании, то $t_{\text{оч}}(j) = 0$);

— момент начала обслуживания заявки $t_{\text{ноб}}(j)$ если заявка получила отказ в обслуживании, то $t_{\text{ноб}}(j) = -1$);

— время обслуживания заявки $t_{\text{обсл}}(j)$ (если заявка получила отказ в обслуживании, то $t_{\text{обсл}}(j) = 0$);

— момент $t_{\text{коб}}(j)$ окончания обслуживания заявки j и выхода её из СМО (если заявка получила отказ в обслуживании, то $t_{\text{коб}}(j) = t_{\text{з}}(j)$);

— 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	$N(k)$	$t_{\text{зан}}(k)$	$t_{np}(k)$	$\Delta_{np}(k)$
1	$N(1)$	$t_{\text{зан}}(1)$	$t_{np}(1)$	$\Delta_{np}(1)$
…	…	…	…	…
n	$N(n)$	$t_{\text{зан}}(n)$	$t_{np}(n)$	$\Delta_{np}(n)$
	$\sum_{k=0}^{n} N(k)$	$\frac{1}{n}\sum_{k=0}^{n} t_{\text{зан}}(k)$	$\frac{1}{n}\sum_{k=0}^{n} t_{np}(k)$	$\frac{1}{n}\sum_{k=0}^{n} \Delta_{np}(k)$

где

$k -$ номер прибора;

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

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

$t_{np}(k) -$ общее время простоя прибора k на интервале $[0, t_{\text{соб}}(100)]$;

$\Delta_{np}(k) = \frac{t_{np}(k)}{t_{\text{соб}}(100)} -$ коэффициент простоя прибора k на интервале $[0, t_{\text{соб}}(100)]$.

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	$R_1(100)$	$v_1(100)$	$T_1(100)$	$\Delta_1(100)$
…	…	…	…	…
	$\sum_i R_i(100)$	$\sum_i v_i(100)$	$\sum_i T_i(100)$	$\sum_i \Delta_i(100)$

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_0	$R_0(100)$	$v_0(100)$	$T_0(100)$	$\Delta_0(100)$
1	r_1	$R_1(100)$	$v_1(100)$	$T_1(100)$	$\Delta_1(100)$
…	…	…	…	…	…
	$\sum_i r_i$	$\sum_i R_i(100)$	$\sum_i v_i(100)$	$\sum_i T_i(100)$	$\sum_i \Delta_i(100)$

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 из интервале $[0, t_{\text{соб}}(100)]$;

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

r_i- теоретическое значение стационарной вероятности для состояния i.

Число строк в таблице 4 определяется значением $M = \max\{C(l), l = 1, \dots, 100\}$.

6. Найти:

— число заявок $J(100)$, поступивших в СМО на интервале $[0, t_{\text{соб}}(100)]$;

— число $JF(100)$ полностью обслуженных заявок на интервале $[0, t_{\text{соб}}(100)]$;

— среднее число заявок, находившихся в СМО, на интервале $[0, t_{\text{соб}}(100)]$, которое находится по формуле $\bar{z}(100) = \frac{1}{100}\sum_{l=1}^{100} z(l)$, где $z(l)$ - число заявок в СМО после события l;

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

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

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

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

$\bar{r}$ – средняя длина очереди;

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

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

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

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

Краткие теоретические сведения

Основные понятия

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.

Характеристики стационарного режима:

1.1. Для СМО ($M|M|n$):

- стационарные вероятности состояний:

  При $\nu < 1$

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

- среднее число занятых приборов:

$$\bar{k} = \rho$$

- средняя длина очереди:

$$\bar{r} = \frac{\nu r_k}{(1-\nu)^2}$$

- среднее число заявок:

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

- среднее время пребывание заявок в очереди:

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

- среднее время пребывания заявок в СМО:

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

- вероятность отказа в обслуживании:

$$P_{\text{отк}} = 0$$

1.2. Для СМО $(M|M|n|m)$:

- стационарные вероятности состояний:

  При $v < 1$:

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

- среднее число занятых приборов:

$$\bar{k} = \rho(1 - r_{n+m}) = \rho\left(1 - v^m \frac{\rho^n}{n!} r_0\right)$$

- средняя длина очереди:

$$\bar{r} = v r_n \frac{1 - (m+1)v^m + m v^{m+1}}{(1-v)^2}$$

- среднее число заявок:

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

- среднее время пребывания заявок в очереди:

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

- среднее время пребывания заявок в СМО:

$$\bar{t}_{\text{СМО}} = \frac{\bar{k} + \bar{r}}{\lambda}$$

- вероятность отказа в обслуживании:

$$P_{\text{отк}} = r_{n+m} = v^m \frac{\rho^n}{n!} r_0$$

2.1. Характеристики СМО $(M|M|16)$:

Дано: $\lambda = 6.403, \mu = 0.412$

Вычислим:

- среднее число занятых приборов:

$$\bar{k} = \rho = \frac{\lambda}{\mu} = \frac{6.403}{0.412} \approx 15.541262$$

- средняя длина очереди:

$$\bar{r} = \frac{\nu r_{16}}{(1-\nu)^2} \approx 29.480208$$

- среднее число заявок:

$$\bar{z} = \bar{k} + \bar{r} = 45.021470$$

- среднее время пребывание заявок в очереди:

$$\bar{t}_{\text{оч}} = \frac{\bar{r}}{\lambda} = 4.604124$$

- среднее время пребывания заявок в СМО:

$$\bar{t}_{\text{СМО}} = \frac{\bar{z}}{\lambda} = 7.031309$$

2.2. Характеристики СМО ($M|M|16|3$):

Дано: $\lambda = 6.403, \mu = 0.412$

Вычислим:

- среднее число занятых приборов:

$$\bar{k} = \rho(1 - P_{\text{отк}}) = 13{,}964838$$

- средняя длина очереди:

$$\bar{r} = \nu r_{16} \frac{1 - 4\nu^3 + 3\nu^4}{(1-\nu)^2} = 0.620673$$

- среднее число заявок:

$$\bar{z} = \bar{k} + \bar{r} = 14.585511$$

- среднее время пребывания заявок в очереди:

$$\bar{t}_{\text{оч}} = \frac{\bar{r}}{\lambda} = 0.096935$$

- среднее время пребывания заявок в СМО:

$$\bar{t}_{\text{СМО}} = \frac{\bar{z}}{\lambda} = 2.277918$$

- вероятность отказа в обслуживании:

$$P_{\text{отк}} = \nu^3 \frac{\rho^{16}}{16!} r_0 = 0.101435$$

Средства языка программирования

В программе расчета был использован язык программирования Python.

Используемые методы модуля Numpy:

- $random.exponential\left(\frac{1}{\lambda}\right)$ - возвращает массив случайных выборок из экспоненциального распределения, где λ – параметр экспоненциального распределения ($\lambda > 0$)

Используемые методы модуля Pandas:

- $DataFrame.from\_dict(dict)$ - создается фрейм данных из словаря типа dict или массива. Используя столбцы или индексы словаря и учитывая объявление Dtype, он создает объект DataFrame.

Результаты расчетов

Задание 1

$$n = 16, m = 3, \Delta T_з = 0.164, \lambda = 6.403, \mu = 0.412$$

l	$t_{соб}(l)$	$Type(l)$	$C(l)$	$t_{осмин}(l)$	$t_{ожз}(l)$	$j(l)$	$k(l)$
1	0.164	1	1	1.595693	0.164	1	1
2	0.328	1	2	1.431693	0.164	2	2
3	0.492	1	3	1.267693	0.164	3	3
4	0.656	1	4	0.144755	0.164	4	4
5	0.800755	2	3	0.958938	0.019245	4	4
6	0.82	1	4	0.939693	0.164	5	4
7	0.984	1	5	0.775693	0.164	6	5
8	1.148	1	6	0.611693	0.164	7	6
9	1.312	1	7	0.447693	0.164	8	7
10	1.476	1	8	0.283693	0.164	9	8
11	1.640	1	9	0.119693	0.164	10	9
12	1.759693	2	8	0.389074	0.044307	1	1
13	1.804	1	9	0.220657	0.164	11	1
14	1.968	1	10	0.056657	0.164	12	10
15	2.024657	2	9	0.12411	0.107343	11	1
16	2.132	1	10	0.016767	0.164	13	1
17	2.148767	2	9	1.138673	0.147233	10	9
18	2.296	1	10	0.297473	0.164	14	9
19	2.1946	1	11	0.133473	0.164	15	11
20	2.593473	2	10	0.693967	0.030527	14	9
21	2.624	1	11	0.66344	0.164	16	9
22	2.788	1	12	0.49944	0.164	17	12
23	2.952	1	13	0.33544	0.164	18	13
24	3.116	1	14	0.17144	0.164	19	14
25	3.280	1	15	0.00744	0.164	20	15
26	3.287440	2	14	0.364404	0.15656	3	3
27	3.444	1	15	0.207844	0.164	21	3
28	3.608	1	16	0.043844	0.164	22	16
29	3.651844	2	15	0.063730	0.120156	5	4
30	3.715574	2	14	0.228588	0.056426	20	15
31	3.772	1	15	0.172162	0.164	23	4
32	3.936	1	16	0.008162	0.164	24	15
33	3.944162	2	15	0.220451	0.155838	18	13
34	4.100	1	16	0.064613	0.164	25	13
35	4.164613	2	15	0.158924	0.099387	7	6
36	4.264	1	16	0.059537	0.164	26	6

37	4.323537	2	15	0.052376	0.104463	15	11
38	4.375913	2	14	0.030323	0.052087	22	16
39	4.406236	2	13	0.059605	0.021764	16	9
40	4.428	1	14	0.037841	0.164	27	9
41	4.465841	2	13	0.207505	0.126159	24	15
42	4.592	1	14	0.081346	0.164	28	11
43	4.673346	2	13	0.085987	0.082654	21	3
44	4.756	1	14	0.003333	0.164	29	3
45	4.759333	2	13	0.084382	0.160667	8	7
46	4.843715	2	12	0.103339	0.076285	13	1
47	4.920	1	13	0.027054	0.164	30	1
48	4.947054	2	12	0.637641	0.136946	26	6
49	5.084	1	13	0.500695	0.164	31	6
50	5.248	1	14	0.336695	0.164	32	7
51	5.412	1	15	0.172695	0.164	33	15
52	5.576	1	16	0.008695	0.164	34	16
53	5.584695	2	15	0.107606	0.155305	9	8
54	5.692301	2	14	0.451781	0.047699	29	3
55	5.1974	1	15	0.404082	0.164	35	3
56	5.904	1	16	0.240082	0.164	36	8
57	6.068	1	17	0.076082	0.164	37	-1
58	6.144082	2	16	0.110764	0.087918	6	5
59	6.232	1	17	0.022846	0.164	38	-1
60	6.254846	2	16	0.081623	0.141154	32	7
61	6.336469	2	15	0.252895	0.059531	38	7
62	6.396	1	16	0.193364	0.164	39	7
63	6.560	1	17	0.029364	0.164	40	-1
64	6.589364	2	16	0.303721	0.134636	31	6
65	6.724	1	17	0.169085	0.164	41	-1
66	6.888	1	18	0.005085	0.164	42	-1
67	6.893085	2	17	0.118661	0.158915	27	9
68	7.011746	2	16	0.056113	0.040254	28	11
69	7.052	1	17	0.015859	0.164	43	-1
70	7.067859	2	16	0.209063	0.148141	36	8
71	7.216	1	17	0.060922	0.164	44	-1
72	7.276922	2	16	0.07403	0.103078	30	1
73	7.350952	2	15	0.364902	0.029048	25	13
74	7.1938	1	16	0.290984	0.164	45	13
75	7.544	1	17	0.126984	0.164	46	-1
76	7.670984	2	16	0.044870	0.037016	45	13
77	7.708	1	17	0.007854	0.164	47	-1
78	7.715854	2	16	0.074730	0.156146	19	14
79	7.790584	2	15	0.340575	0.081416	34	16

80	7.872	1	16	0.145552	0.164	48	16
81	8.017552	2	15	0.113607	0.018448	48	16
82	8.036	1	16	0.095159	0.164	49	16
83	8.131159	2	15	0.120606	0.068841	33	15
84	8.200	1	16	0.051765	0.164	50	15
85	8.251765	2	15	0.079123	0.112235	47	14
86	8.330888	2	14	0.078762	0.033112	43	8
87	8.364	1	15	0.045650	0.164	51	8
88	8.40965	2	14	0.369571	0.11835	12	10
89	8.528	1	15	0.251221	0.164	52	10
90	8.692	1	16	0.087221	0.164	53	14
91	8.779221	2	15	0.094230	0.076779	51	8
92	8.856	1	16	0.017451	0.164	54	8
93	8.873451	2	15	0.018051	0.146549	49	16
94	8.891502	2	14	0.231037	0.128498	52	10
95	9.020	1	15	0.102539	0.164	55	10
96	9.122539	2	14	0.019039	0.061461	46	13
97	9.141578	2	13	0.153804	0.042422	39	7
98	9.184	1	14	0.111382	0.164	56	7
99	9.295382	2	13	0.178108	0.052618	40	6
100	9.348	1	14	0.125490	0.164	57	6

Таблица 1.1

j	$t_з(j)$	$q(j)$	$t_{оч}(j)$	$t_{ноб}(j)$	$t_{обсл}(j)$	$t_{коб}(j)$	$k(j)$
1	0.164	0	0.0	0.164	1.595693	1.759693	1
2	0.328	0	0.0	0.328	9.479823	9.807823	2
3	0.492	0	0.0	0.492	2.79544	3.28744	3
4	0.656	0	0.0	0.656	0.144755	0.800755	4
5	0.82	0	0.0	0.82	2.831844	3.651844	4
6	0.984	0	0.0	0.984	5.160082	6.144082	5
7	1.148	0	0.0	1.148	3.016613	4.164613	6
8	1.312	0	0.0	1.312	3.447333	4.759333	7
9	1.476	0	0.0	1.476	4.108695	5.584695	8
10	1.640	0	0.0	1.640	0.508767	2.148767	9
11	1.804	0	0.0	1.804	0.220657	2.024657	1
12	1.968	0	0.0	1.968	6.44165	8.40965	10
13	2.132	0	0.0	2.132	2.711715	4.843715	1
14	2.296	0	0.0	2.296	0.297473	2.593473	9
15	2.46	0	0.0	2.460	1.863537	4.323537	11
16	2.624	0	0.0	2.624	1.782236	4.406236	9
17	2.788	0	-0.0	2.788	7.455583	10.243583	12
18	2.952	0	-0.0	2.952	0.992162	3.944162	13

19	3.116	0	-0.0	3.116	4.599854	7.715854	14
20	3.280	0	-0.0	3.280	0.435574	3.715574	15
21	3.444	0	-0.0	3.444	1.229346	4.673346	3
22	3.608	0	-0.0	3.608	0.767913	4.375913	16
23	3.772	0	-0.0	3.772	8.156535	11.928535	4
24	3.936	0	-0.0	3.936	0.529841	4.465841	15
25	4.100	0	-0.0	4.100	3.250952	7.350952	13
26	4.264	0	0.0	4.264	0.683054	4.947054	6
27	4.428	0	-0.0	4.428	2.465085	6.893085	9
28	4.592	0	-0.0	4.592	2.419746	7.011746	11
29	4.756	0	-0.0	4.756	0.936301	5.692301	3
30	4.920	0	-0.0	4.920	2.356922	7.276922	1
31	5.084	0	-0.0	5.084	1.505364	6.589364	6
32	5.248	0	-0.0	5.248	1.006846	6.254846	7
33	5.412	0	-0.0	5.412	2.719159	8.131159	15
34	5.576	0	-0.0	5.576	2.214584	7.790584	16
35	5.740	0	0.0	5.740	6.38138	12.12138	3
36	5.904	0	0.0	5.904	1.163859	7.067859	8
37	6.068	1	3.329408	9.397408	0.076082	9.47349	5
38	6.232	1	0.081623	6.313623	0.022846	6.336469	7
39	6.396	0	0.0	6.396	2.745578	9.141578	7
40	6.560	1	2.706018	9.266018	0.029364	9.295382	6
41	6.724	1	3.781033	10.505033	0.169085	10.674118	9
42	6.888	2	3.023546	9.911546	0.005085	9.916631	11
43	7.052	1	1.263029	8.315029	0.015859	8.330888	8
44	7.216	1	9.050537	16.266537	0.060922	16.327459	1
45	7.380	0	0.0	7.380	0.290984	7.670984	13
46	7.544	1	1.451555	8.995555	0.126984	9.122539	13
47	7.708	1	0.535911	8.243911	0.007854	8.251765	14
48	7.872	0	0.0	7.872	0.145552	8.017552	16
49	8.036	0	0.0	8.036	0.837451	8.873451	16
50	8.200	0	0.0	8.200	3.20045	11.40045	15
51	8.364	0	0.0	8.364	0.415221	8.779221	8
52	8.528	0	0.0	8.528	0.363502	8.891502	10
53	8.692	0	0.0	8.692	1.886747	10.578747	14
54	8.856	0	0.0	8.856	1.160325	10.016325	8
55	9.020	0	0.0	9.200	7.838954	16.858954	10
56	9.184	0	0.0	9.184	3.003538	12.187538	7
57	9.348	0	0.0	9.348	9.499601	18.847601	6

Таблица 1.2

k	$N(k)$	$t_{зан}(k)$	$t_{пр}(k)$	$\Delta_{пр}(k)$
1	5	8.956065	0.391935	0.041927
2	1	9.020	0.328	0.035088
3	4	8.569087	0.778913	0.083324
4	3	8.552599	0.795401	0.085088
5	2	8.364	0.984	0.105263
6	5	7.911049	1.436951	0.153717
7	5	7.44538	1.90262	0.203532
8	5	7.442804	1.905196	0.203808
9	5	7.508476	1.839524	0.196783
10	3	7.133152	2.214848	0.236933
11	3	6.619537	2.728463	0.291877
12	1	6.560	2.788	0.298246
13	4	5.985653	3.362347	0.359686
14	3	5.791765	3.556235	0.380427
15	4	4.832574	4.515426	0.483037
16	4	3.9655	5.3825	0.575792
	57	7.166103	2.181897	0.233408

Таблица 1.3

Состояние	$R_i(100)$	$v_i(100)$	$T_i(100)$	$\Delta_i(100)$
0	1	0.01	0.164	0.017544
1	1	0.01	0.164	0.017544
2	1	0.01	0.164	0.017544
3	2	0.02	0.183245	0.019603
4	2	0.02	0.308755	0.033029
5	1	0.01	0.164	0.017544
6	1	0.01	0.164	0.017544
7	1	0.01	0.164	0.017544

8	2	0.02	0.208307	0.022284
9	4	0.04	0.538269	0.057581
10	4	0.04	0.267951	0.028664
11	2	0.02	0.297473	0.031822
12	3	0.03	0.377231	0.040354
13	9	0.09	0.765053	0.081841
14	14	0.14	1.151909	0.123225
15	21	0.21	1.885803	0.201733
16	21	0.21	1.752347	0.187457
17	9	0.09	0.622572	0.066599
18	1	0.01	0.005085	0.000544
	100	1.000000	9.348	1.000000

Таблица 1.4

Число заявок: 57

Число полностью обслуженных заявок: 43

Среднее число заявок, находившихся в СМО: 13.340000

Среднее время пребывания заявок в очереди: 0.140422

Среднее время пребывания заявок в СМО: 1.796334

Задание 2

$$n = 16, m = 3, \Delta T_з = 0.164, \lambda = 6.403, \mu = 0.412$$

l	$t_{соб}(l)$	$Type(l)$	$C(l)$	$t_{осмин}(l)$	$t_{ожз}(l)$	$j(l)$	$k(l)$
1	0.003894	1	1	0.74961	0.003894	1	1
2	0.007788	1	2	0.745716	0.019773	2	2
3	0.027561	1	3	0.725943	0.288446	3	3
4	0.316007	1	4	0.311529	0.098027	4	4
5	0.414034	1	5	0.213502	0.110122	5	5
6	0.524156	1	6	0.10338	0.141554	6	6
7	0.627536	2	5	0.125968	0.038174	4	4
8	0.66571	1	6	0.087794	0.086765	7	4
9	0.752475	1	7	0.001029	0.021261	8	7
10	0.753504	2	6	0.236484	0.020232	1	1
11	0.773736	1	7	0.216252	0.005151	9	1
12	0.778887	1	8	0.211101	0.03711	10	8
13	0.815997	1	9	0.173991	0.075696	11	9
14	0.891693	1	10	0.098295	0.02703	12	10
15	0.918723	1	11	0.071265	0.012755	13	11
16	0.931478	1	12	0.058510	0.169974	14	12
17	0.989988	2	11	0.043775	0.111464	2	2
18	1.033763	2	10	0.250961	0.067689	13	11
19	1.101452	1	11	0.183272	0.008354	15	2
20	1.109806	1	12	0.174918	0.537937	16	11
21	1.284724	2	11	0.038172	0.363019	5	5
22	1.322896	2	10	0.272386	0.324847	7	4
23	1.595282	2	9	0.081402	0.052461	15	2
24	1.647743	1	10	0.028941	0.012237	17	2
25	1.65998	1	11	0.016704	0.011815	18	4
26	1.671795	1	12	0.004889	0.305937	19	5
27	1.676684	2	11	0.080433	0.301048	11	9
28	1.757117	2	10	0.025956	0.220615	9	1
29	1.783073	2	9	0.109404	0.194659	17	2

30	1.892477	2	8	0.066553	0.085255	14	12
31	1.95903	2	7	0.105550	0.018702	19	5
32	1.977732	1	8	0.086848	0.262501	20	1
33	2.06458	2	7	0.440856	0.175653	18	4
34	2.240233	1	8	0.224867	0.099848	21	2
35	2.340081	1	9	0.125019	0.01872	22	4
36	2.358801	1	10	0.106299	0.111323	23	5
37	2.465100	2	9	0.040336	0.005024	21	2
38	2.470124	1	10	0.035312	0.258167	24	2
39	2.505436	2	9	0.281756	0.222855	6	6
40	2.728291	1	10	0.058901	0.02883	25	6
41	2.757121	1	11	0.030071	0.241236	26	9
42	2.787192	2	10	0.036948	0.211165	3	3
43	2.824140	2	9	0.410835	0.174217	26	9
44	2.998357	1	10	0.236618	0.04756	27	3
45	3.045917	1	11	0.125519	0.16134	28	9
46	3.171436	2	10	0.063539	0.035821	28	9
47	3.207257	1	11	0.027718	0.017337	29	9
48	3.224594	1	12	0.010381	1.094098	30	12
49	3.234975	2	11	0.520478	1.083717	10	8
50	3.755453	2	10	0.137986	0.563239	27	3
51	3.893439	2	9	0.291254	0.425253	12	10
52	4.184693	2	8	0.107246	0.133999	29	9
53	4.291939	2	7	0.004561	0.026753	23	5
54	4.296500	2	6	0.302607	0.022192	25	6
55	4.318692	1	7	0.280415	0.210284	31	3
56	4.528976	1	8	0.070131	0.061294	32	5
57	4.59027	1	9	0.008837	0.938797	33	6
58	4.599107	2	8	1.056570	0.92996	8	7
59	5.529067	1	9	0.126610	0.002909	34	7
60	5.531976	1	10	0.123701	0.384359	35	8
61	5.655677	2	9	0.179323	0.260658	24	2
62	5.835000	2	8	0.530279	0.081335	35	8
63	5.916335	1	9	0.448944	0.052144	36	2

64	5.968479	1	10	0.396800	0.051486	37	8
65	6.019965	1	11	0.345314	0.209302	38	9
66	6.229267	1	12	0.136012	0.072585	39	10
67	6.301852	1	13	0.027003	0.204013	40	13
68	6.328855	2	12	0.036424	0.17701	40	13
69	6.365279	2	11	0.052797	0.140586	33	6
70	6.418076	2	10	0.224924	0.087789	34	7
71	6.505865	1	11	0.137135	0.017226	41	6
72	6.523091	1	12	0.119909	0.055145	42	7
73	6.578236	1	13	0.064764	0.143169	43	13
74	6.643000	2	12	0.014715	0.078405	38	9
75	6.657715	2	11	0.392410	0.06369	41	6
76	6.721405	1	12	0.328720	0.144903	44	6
77	6.866308	1	13	0.183817	0.315537	45	9
78	7.050125	2	12	0.013293	0.13172	32	5
79	7.063418	2	11	0.159510	0.118427	44	6
80	7.181845	1	12	0.041083	0.182975	46	5
81	7.222928	2	11	0.241179	0.141892	37	8
82	7.364820	1	12	0.099287	0.055239	47	6
83	7.420059	1	13	0.044048	0.095928	48	8
84	7.464107	2	12	0.201626	0.05188	39	10
85	7.515987	1	13	0.149746	0.35037	49	10
86	7.665733	2	12	0.001310	0.200624	47	6
87	7.667043	2	11	0.042368	0.199314	43	13
88	7.709411	2	10	0.316406	0.156946	45	9
89	7.866357	1	11	0.159460	0.036839	50	6
90	7.903196	1	12	0.122621	0.124826	51	9
91	8.025817	2	11	0.015875	0.002205	22	4
92	8.028022	1	12	0.013670	0.169816	52	4
93	8.041692	2	11	0.821023	0.156146	46	5
94	8.197838	1	12	0.664877	0.322329	53	5
95	8.520167	1	13	0.342548	0.072146	54	13
96	8.592313	1	14	0.270402	0.055917	55	14
97	8.648230	1	15	0.214485	0.01601	56	15

98	8.664240	1	16	0.198475	0.027501	57	16
99	8.691741	1	17	0.170974	0.225137	58	-1
100	8.862715	2	16	0.048338	0.054163	30	12

Таблица 2.1

j	$t_з(j)$	$q(j)$	$t_{оч}(j)$	$t_{ноб}(j)$	$t_{обсл}(j)$	$t_{коб}(j)$	$k(j)$
1	0.003894	0	0.0	0.003894	0.74961	0.753504	1
2	0.007788	0	0.0	0.007788	0.9822	0.989988	2
3	0.027561	0	0.0	0.027561	2.759631	2.787192	3
4	0.316007	0	0.0	0.316007	0.311529	0.627536	4
5	0.414034	0	0.0	0.414034	0.87069	1.284724	5
6	0.524156	0	-0.0	0.524156	1.98128	2.505436	6
7	0.665710	0	-0.0	0.66571	0.657186	1.322896	4
8	0.752475	0	-0.0	0.752475	3.846632	4.599107	7
9	0.773736	0	-0.0	0.773736	0.983381	1.757117	1
10	0.778887	0	-0.0	0.778887	2.456088	3.234975	8
11	0.815997	0	-0.0	0.815997	0.860687	1.676684	9
12	0.891693	0	-0.0	0.891693	3.001746	3.893439	10
13	0.918723	0	-0.0	0.918723	0.11504	1.033763	11
14	0.931478	0	-0.0	0.931478	0.960999	1.892477	12
15	1.101452	0	0.0	1.101452	0.49383	1.595282	2
16	1.109806	0	0.0	1.109806	11.880366	12.990172	11
17	1.647743	0	-0.0	1.647743	0.13533	1.783073	2
18	1.659980	0	-0.0	1.65998	0.4046	2.06458	4
19	1.671795	0	-0.0	1.671795	0.287235	1.95903	5
20	1.977732	0	0.0	1.977732	10.415945	12.393677	1
21	2.240233	0	0.0	2.240233	0.224867	2.465100	2
22	2.340081	0	0.0	2.340081	5.685736	8.025817	4
23	2.358801	0	0.0	2.358801	1.933138	4.291939	5
24	2.470124	0	0.0	2.470124	3.185553	5.655677	2
25	2.728291	0	-0.0	2.728291	1.568209	4.296500	6
26	2.757121	0	-0.0	2.757121	0.067019	2.82414	9
27	2.998357	0	-0.0	2.998357	0.757096	3.755453	3

28	3.045917	0	-0.0	3.045917	0.125519	3.171436	9
29	3.207257	0	-0.0	3.207257	0.977436	4.184693	9
30	3.224594	0	-0.0	3.224594	5.638121	8.862715	12
31	4.318692	0	0.0	4.318692	9.286894	13.605586	3
32	4.528976	0	0.0	4.528976	2.521149	7.050125	5
33	4.59027	0	0.0	4.59027	1.775009	6.365279	6
34	5.529067	0	0.0	5.529067	0.889009	6.418076	7
35	5.531976	0	0.0	5.531976	0.303024	5.835	8
36	5.916335	0	0.0	5.916335	3.925758	9.842093	2
37	5.968479	0	0.0	5.968479	1.254449	7.222928	8
38	6.019965	0	0.0	6.019965	0.623035	6.643	9
39	6.229267	0	0.0	6.229267	1.23484	7.464107	10
40	6.301852	0	0.0	6.301852	0.027003	6.328855	13
41	6.505865	0	0.0	6.505865	0.15185	6.657715	6
42	6.523091	0	0.0	6.523091	3.437787	9.960878	7
43	6.578236	0	0.0	6.578236	1.088807	7.667043	13
44	6.721405	0	0.0	6.721405	0.342013	7.063418	6
45	6.866308	0	0.0	6.866308	0.843103	7.709411	9
46	7.181845	0	0.0	7.181845	0.859847	8.041692	5
47	7.364820	0	0.0	7.36482	0.300913	7.665733	6
48	7.420059	0	0.0	7.420059	6.353983	13.774042	8
49	7.515987	0	0.0	7.515987	2.215534	9.731521	10
50	7.866357	0	0.0	7.866357	2.255404	10.121761	6
51	7.903196	0	0.0	7.903196	5.651654	13.55485	9
52	8.028022	0	0.0	8.028022	0.883031	8.911053	4
53	8.197838	0	0.0	8.197838	2.186918	10.384756	5
54	8.520167	0	0.0	8.520167	4.874768	13.394935	13
55	8.592313	0	0.0	8.592313	3.848517	12.44083	14
56	8.648230	0	0.0	8.64823	1.187927	9.836157	15
57	8.664240	0	0.0	8.66424	0.284517	8.948757	16
58	8.691741	1	1.571344	10.263085	0.170974	10.434059	12

Таблица 2.2

k	$N(k)$	$t_{\text{зан}}(k)$	$t_{np}(k)$	$\Delta_{np}(k)$
1	3	8.617974	0.244741	0.027615
2	6	7.96816	0.894555	0.100935
3	3	8.060750	0.801965	0.090488
4	5	7.893744	0.968971	0.109331
5	6	7.136936	1.725779	0.194724
6	7	7.115632	1.747083	0.197127
7	3	7.075265	1.787450	0.201682
8	4	5.456217	3.406498	0.384363
9	7	4.456318	4.406397	0.497184
10	3	5.583314	3.279401	0.370022
11	2	7.867949	0.994766	0.112242
12	3	6.59912	2.263595	0.255406
13	3	1.458358	7.404357	0.835450
14	1	0.270402	8.592313	0.969490
15	1	0.214485	8.64823	0.975799
16	1	0.198475	8.66424	0.977606
	58	5.373319	3.489396	0.393716

Таблица 2.3

Состояние	r_i	$R_i(100)$	$v_i(100)$	$T_i(100)$	$\Delta_i(100)$
0	0.000000	1	0.01	0.010945	0.001234
1	0.000001	2	0.02	0,082784	0.009335
2	0.000005	2	0.02	0.214880	0.024231
3	0.000028	2	0.02	0.290154	0.032720
4	0.000110	2	0.02	0.469004	0.052888
5	0.000341	1	0.01	0.143635	0.016197
6	0.000882	1	0.01	0.019167	0.002161
7	0.001958	1	0.01	0.038787	0.004374

8	0.003804	3	0.03	0.649593	0.073253
9	0.006569	5	0.05	0.433336	0.048866
10	0.010209	4	0.04	0.548416	0.061843
11	0.014424	7	0.07	0.474144	0.053468
12	0.018606	10	0.1	0.970529	0.109444
13	0.022332	13	0.13	0.959335	0.108182
14	0.024791	10	0.1	0.881826	0.099441
15	0.025685	3	0.03	0.248885	0.028066
16	0.024949	4	0.04	0.203896	0.022993
17	0.024234	3	0.03	0.143677	0.016202
	0,178928	100	1.0	8.867813	1.0

Таблица 2.4

Число заявок: 58

Число полностью обслуженных заявок: 42

Среднее число заявок, находившихся в СМО: 9.980000

Среднее время пребывания заявок в очереди: 0.0

Среднее время пребывания заявок в СМО: 1.291296

Задание 3

$$n = 16, m = 3, \Delta T_з = 0.164, \lambda = 6.403, \mu = 0.412$$

l	$t_{соб}(l)$	$Type(l)$	$C(l)$	$t_{осмин}(l)$	$t_{ожз}(l)$	$j(l)$	$k(l)$
1	0.11049	1	1	0.55738	0.11049	1	1
2	0.22098	1	2	0.44689	0.063424	2	2
3	0.284404	1	3	0.383466	0.049289	3	3
4	0.333693	1	4	0.334177	0.004917	4	4
5	0.33861	1	5	0.329260	0.127566	5	5
6	0.466176	1	6	0.201694	0.29078	6	6
7	0.667870	3	5	0.614919	0.089086	1	1
8	0.756956	1	6	0.085066	0.035788	7	1
9	0.792744	1	7	0.049278	0.032333	8	7
10	0.825077	1	8	0.016945	0.269739	9	8
11	0.842022	3	7	0.398125	0.252794	7	1
12	1.094816	1	8	0.145331	0.015875	10	1
13	1.110691	1	9	0.129456	0.621128	11	9
14	1.240147	3	8	0.042642	0.491672	9	8
15	1.282789	3	7	0.254020	0.44903	3	3
16	1.536809	3	6	0.504761	0.19501	10	1
17	1.731819	1	7	0.08537	0.006021	12	1
18	1.73784	1	8	0.079349	0.179762	13	3
19	1.817189	3	7	0.224381	0.100413	12	1
20	1.917602	1	8	0.123968	0.053403	14	1
21	1.971005	1	9	0.070565	0.035913	15	8
22	2.006918	1	10	0.034652	0.023529	16	10
23	2.030447	1	11	0.011123	0.219465	17	11
24	2.04157	3	10	0.258424	0.208342	2	2
25	2.249912	1	11	0.050082	0.127822	18	2

26	2.299994	3	10	0.359749	0.07774	5	5
27	2.377734	1	11	0.282009	0.054629	19	5
28	2.432363	1	12	0.227380	0.165993	20	12
29	2.598356	1	13	0.061387	0.019627	21	13
30	2.617983	1	14	0.041760	0.075611	22	14
31	2.659743	3	13	0.295802	0.033851	18	2
32	2.693594	1	14	0.261951	0.107526	23	2
33	2.801120	1	15	0.154425	0.082364	24	15
34	2.883484	1	16	0.072061	0.815664	25	16
35	2.955545	3	15	0.259925	0.743603	20	12
36	3.215470	3	14	0.036882	0.483678	6	6
37	3.252352	3	13	0.091562	0.446796	16	10
38	3.343914	3	12	0.160554	0.355234	22	14
39	3.504468	3	11	0.006360	0.19468	25	16
40	3.510828	3	10	0.178520	0.18832	19	5
41	3.689348	3	9	0.256596	0.0098	21	13
42	3.699148	1	10	0.246796	0.008029	26	5
43	3.707177	1	11	0.238767	0.288259	27	6
44	3.945944	3	10	0.048895	0.049492	24	15
45	3.994839	3	9	0.134991	0.000597	26	5
46	3.995436	1	10	0.134394	0.097732	28	5
47	4.093168	1	11	0.036662	0.021031	29	10
48	4.114199	1	12	0.015631	0.045418	30	12
49	4.129830	3	11	0.294226	0.029787	4	4
50	4.159617	1	12	0.264439	0.084778	31	4
51	4.244395	1	13	0.179661	0.264426	32	13
52	4.424056	3	12	0.146554	0.084765	29	10
53	4.508821	1	13	0.061789	0.215837	33	10
54	4.570610	3	12	0.070505	0.154048	8	7

55	4.641115	3	11	0.092379	0.083543	11	9
56	4.724658	1	12	0.008836	0.053402	34	7
57	4.733494	3	11	0.110072	0.044566	27	6
58	4.778060	1	12	0.065506	0.033323	35	6
59	4.811383	1	13	0.032183	0.006332	36	9
60	4.817715	1	14	0.025851	0.042384	37	14
61	4.843566	3	13	0.114775	0.016533	31	4
62	4.860099	1	14	0.098242	0.046421	38	4
63	4.906520	1	15	0.051821	0.123444	39	15
64	4.958341	3	14	0.042086	0.071623	33	10
65	5.000427	3	13	0.173608	0.029537	15	8
66	5.029964	1	14	0.144071	0.048596	40	8
67	5.078560	1	15	0.095475	0.006633	41	10
68	5.085193	1	16	0.088842	0.235978	42	16
69	5.174035	3	15	0.639018	0.147136	14	1
70	5.321171	1	16	0.491882	0.093551	43	1
71	5.414722	1	17	0.398331	0.089974	44	-1
72	5.504696	1	18	0.308357	0.151854	45	-1
73	5.656550	1	19	0.156503	0.027676	46	-1
74	5.684226	2	19	0.128827	0.010506	47	-1
75	5.694732	2	19	0.118321	0.454931	48	-1
76	5.813053	3	18	0.315375	0.33661	30	12
77	6.128428	3	17	0.034404	0.021235	34	7
78	6.149663	1	18	0.013169	0.170478	49	-1
79	6.162832	3	17	0.064571	0.157309	37	14
80	6.227403	3	16	0.083420	0.092738	17	11
81	6.310823	3	15	0.136025	0.009318	44	12
82	6.320141	1	16	0.126707	0.332705	50	12
83	6.446848	3	15	0.280725	0.205998	45	7

84	6.652846	1	16	0.074727	0.293646	51	7
85	6.727573	3	15	0.271552	0.218919	50	12
86	6.946492	1	16	0.025745	0.270066	52	12
87	6.972237	3	15	0.026888	0.244321	52	12
88	6.999125	3	14	0.074242	0.217433	36	9
89	7.073367	3	13	0.051948	0.143191	28	5
90	7.125315	3	12	0.003024	0.091243	38	4
91	7.128339	3	11	0.000316	0.088219	32	13
92	7.128655	3	10	0.393964	0.087903	23	2
93	7.216558	1	11	0.306061	0.191849	53	2
94	7.408407	1	12	0.114212	0.271008	54	4
95	7.522619	3	11	0.043555	0.156796	35	6
96	7.566174	3	10	0.079423	0.113241	41	10
97	7.645597	3	9	0.025288	0.033818	43	1
98	7.670885	3	8	0.188999	0.00853	39	15
99	7.679415	1	9	0.180469	0.050473	55	1
100	7.729888	1	10	0.129996	0.497414	56	5

Таблица 3.1

j	$t_з(j)$	$q(j)$	$t_{оч}(j)$	$t_{ноб}(j)$	$t_{обсл}(j)$	$t_{коб}(j)$	$k(j)$
1	0.11049	0	0.0	0.11049	0.55738	0.66787	1
2	0.22098	0	0.0	0.22098	1.82059	2.04157	2
3	0.284404	0	0.0	0.284404	0.998385	1.282789	3
4	0.333693	0	0.0	0.333693	3.796137	4.12983	4
5	0.33861	0	0.0	0.33861	1.961384	2.299994	5
6	0.466176	0	-0.0	0.466176	2.749294	3.21547	6
7	0.756956	0	-0.0	0.756956	0.085066	0.842022	1
8	0.792744	0	-0.0	0.792744	3.777866	4.57061	7
9	0.825077	0	-0.0	0.825077	0.41507	1.240147	8

10	1.094816	0	0.0	1.094816	0.441993	1.536809	1
11	1.110691	0	0.0	1.110691	3.530424	4.641115	9
12	1.731819	0	0.0	1.731819	0.08537	1.817189	1
13	1.73784	0	0.0	1.73784	14.485843	16.223683	3
14	1.917602	0	0.0	1.917602	3.256433	5.174035	1
15	1.971005	0	-0.0	1.971005	3.029422	5.000427	8
16	2.006918	0	0.0	2.006918	1.245434	3.252352	10
17	2.030447	0	0.0	2.030447	4.196956	6.227403	11
18	2.249912	0	0.0	2.249912	0.409831	2.659743	2
19	2.377734	0	0.0	2.377734	1.133094	3.510828	5
20	2.432363	0	0.0	2.432363	0.523182	2.955545	12
21	2.598356	0	0.0	2.598356	1.090992	3.689348	13
22	2.617983	0	0.0	2.617983	0.725931	3.343914	14
23	2.693594	0	0.0	2.693594	4.435061	7.128655	2
24	2.801120	0	0.0	2.80112	1.144824	3.945944	15
25	2.883484	0	0.0	2.883484	0.620984	3.504468	16
26	3.699148	0	0.0	3.699148	0.295691	3.994839	5
27	3.707177	0	0.0	3.707177	1.026317	4.733494	6
28	3.995436	0	0.0	3.995436	3.077931	7.073367	5
29	4.093168	0	0.0	4.093168	0.330888	4.424056	10
30	4.114199	0	0.0	4.114199	1.698854	5.813053	12
31	4.159617	0	0.0	4.159617	0.683949	4.843566	4
32	4.244395	0	0.0	4.244395	2.883944	7.128339	13
33	4.508821	0	0.0	4.508821	0.44952	4.958341	10
34	4.724658	0	0.0	4.724658	1.40377	6.128428	7
35	4.778060	0	0.0	4.77806	2.744559	7.522619	6
36	4.811383	0	0.0	4.811383	2.187742	6.999125	9
37	4.817715	0	0.0	4.817715	1.345117	6.162832	14
38	4.860099	0	0.0	4.860099	2.265216	7.125315	4

39	4.906520	0	0.0	4.90652	2.764365	7.670885	15
40	5.029964	0	0.0	5.029964	2.95908	7.989044	8
41	5.078560	0	0.0	5.07856	2.487614	7.566174	10
42	5.082465	0	0.0	5.082465	2.712373	7.794838	7
43	5.083878	0	0.0	5.083878	2.975391	8.059269	11
44	5.085193	0	0.0	5.085193	2.79247	7.877663	16
45	5.321171	0	0.0	5.321171	2.324426	7.645597	1
46	5.414722	1	0.49777	5.912492	0.398331	6.310823	12
47	5.504696	2	0.633795	6.138491	0.308357	6.446848	7
48	5.656550	3	3.376807	9.033357	0.156503	9.18986	14
49	6.149663	2	3.166746	9.316409	0.013169	9.329578	11
50	6.320141	0	0.0	6.320141	0.407432	6.727573	12
51	6.652846	0	0.0	6.652846	1.334411	7.987257	7
52	6.946492	0	0.0	6.946492	0.025745	6.972237	12
53	7.216558	0	0.0	7.216558	0.809246	8.025804	2
54	7.408407	0	0.0	7.408407	0.451477	7.859884	4
55	7.679415	0	0.0	7.679415	1.91585	9.595265	1
56	7.729888	0	0.0	7.729888	4.130384	11.860272	5

Таблица 3.2

k	$N(k)$	$t_{\text{зан}}(k)$	$t_{np}(k)$	$\Delta_{np}(k)$
1	7	6.801141	0.928747	0.120150
2	4	7.178812	0.551076	0.071292
3	2	6.990433	0.739455	0.095662
4	4	7.066783	0.663105	0.085785
5	5	6.468100	1.261788	0.163235
6	3	6.52017	1.209718	0.156499
7	5	6.577098	1.152790	0.149134
8	3	6.144416	1.585472	0.205109

9	2	5.718166	2.011722	0.260252
10	4	4.513456	3.216432	0.416103
11	3	5.699441	2.030447	0.262675
12	5	3.152983	4.576905	0.592105
13	2	3.974936	3.754952	0.485771
14	3	3.638104	4.091784	0.529346
15	2	3.909189	3.820699	0.494276
16	2	3.265679	4.464209	0.577526
	56	5.476182	2.253706	0.291557

Таблица 3.3

Состояние	r_i	$R_i(100)$	$v_i(100)$	$T_i(100)$	$\Delta_i(100)$
0	0.000000	1	0.01	0.110490	0.014294
1	0.000001	1	0.01	0.110490	0.014294
2	0.000007	1	0.01	0.063424	0.008205
3	0.000037	1	0.01	0.049289	0.006376
4	0.000144	1	0.01	0.004917	0.000636
5	0.000449	2	0.02	0.216652	0.028028
6	0.001162	3	0.03	0.432492	0.055951
7	0.002580	5	0.05	0.645581	0.083518
8	0.005012	6	0.06	0.216744	0.028040
9	0.008654	6	0.06	0.251527	0.032540
10	0.013449	9	0.09	0.810113	0.104803
11	0.019002	12	0.12	0.775608	0.100339
12	0.024609	10	0.1	0.741621	0.095942
13	0.029420	9	0.09	0.490840	0.063499
14	0.032659	8	0.08	0.423364	0.054770
15	0.033837	9	0.09	1.009002	0.130533
16	0.032867	7	0.07	0.565053	0.073100

17	0.030047	3	0.03	0.175780	0.022740
18	0.025943	3	0.03	0.480398	0.062148
19	0.021220	3	0.03	0.156503	0.020246
	0.281099	100	1.0	7,729888	1.000002

Таблица 3.4

Число заявок: 56

Число полностью обслуженных заявок: 44

Среднее число заявок, находившихся в СМО: 11.650000

Среднее время пребывания заявок в очереди: 0.025717

Среднее время пребывания заявок в СМО: : 1.642555

Список литературы

1. Системы массового обслуживания [Электронный ресурс]: методические указания / А.А. Лобузов. - М.: РТУ МИРЭА, 2023. - 45 с.
2. Аникина О., Гущина О. М. Табличное моделирование динамики работы одноканальной системы массового обслуживания с ограниченной очередью. – Тольятти, 2017. – 12 с.
3. Плескунов М. А. Теория массового обслуживания. – Екатеринбург: Уральский энергетический институт, 2022. – 268 с.

Приложение

```
import numpy as np
import pandas as pd
from copy import copy
```

```
n = 16
m = 3
lambd = 1/6.403
mu = 1/0.412
```

```
sample = {'l' : 0, 't_sob' : 0, 'type' : 0, 'c' : 0, 't_min' : 0, 't_ozh'
: 0, 'j' : 0, 'k' : 0}
device = {'j' : 0, 'k' : 0, 't_ost' : 0}
```

Задание 1

```
dT = 0.164
table = [{'l' : 0, 't_sob' : 0, 'type' : 0, 'c' : 0, 't_min' : 0, 't_ozh'
: 0, 'j' : 0, 'k' : 0} for i in range(100)]
devices = [{'j' : 0, 'k' : i + 1, 't_ost' : 0} for i in range(n)]
table2 = []
table3 = [{'N' : i + 1, 't_zan' : 0, 't_pr' : 0, 'delta_pr' : 0} for i in
range(n)]
table4 = [{'sost' : i, 'R_100' : 0, 'nu_100' : 0 , 'T_100' :
0,  'delta_100' : 0 } for i in range(101)]

for i in range(n):
    devices[i]['k'] = i + 1
queue = []
current_device = -1
time_cummulative = 0
min_time = 0
total_j = 0
```

```
for i in range(100):

    if current_device == -1:

        current_device = 0
        total_j += 1
        min_time = np.round(np.random.exponential(mu), 6)
        if i == 0:
            time_cummulative += dT
            table4[0]['T_100'] += dT
        else:
            time_cummulative += table[i-1]['t_ozh']
            table4[0]['T_100'] += table[i-1]['t_ozh']
```

```
        table4[0]['R_100'] += 1

        table[i]['l'] = i + 1
        table[i]['j'] = total_j
        table[i]['t_sob'] = time_cummulative
        table[i]['type'] = 1
        table[i]['c'] = 1
        table[i]['t_min'] = min_time
        table[i]['t_ozh'] = dT
        table[i]['k'] = 1

        devices[0]['j'] = total_j
        devices[0]['t_ost'] = min_time

        table2.append({'j' : total_j, 't_z' : time_cummulative, 'q_j' : 0,
't_och' : 0, 't_nob' : 0, 't_obs' : min_time,
                       't_kob' : 0, 'k_j' : 1})


    elif min_time < table[i-1]['t_ozh']:

        time_cummulative += min_time

        for j in range(n):
            if devices[j]['j'] != 0:
                devices[j]['t_ost'] -= min_time
                table3[j]['t_zan'] += min_time
        table4[table[i-1]['c']]['T_100'] += min_time
        table4[table[i-1]['c']]['R_100'] += 1

        table[i]['l'] = i + 1
        table[i]['j'] = devices[current_device]['j']
        table[i]['t_sob'] = time_cummulative
        table[i]['t_ozh'] = np.round(table[i-1]['t_ozh'] - min_time, 6)
        table[i]['type'] = 2
        table[i]['c'] = table[i-1]['c'] - 1
        table[i]['k'] = current_device + 1

        table2[table[i]['j'] - 1]['t_kob'] = np.round(time_cummulative, 6)
        table2[table[i]['j'] - 1]['t_nob'] = np.round(time_cummulative -
table2[table[i]['j']-1]['t_obs'], 6)
        table2[table[i]['j']- 1]['t_och'] = np.round(table2[ table[i]['j']
- 1]['t_nob']  - table2[table[i]['j']-1]['t_z'], 6)


        devices[current_device]['j'] = 0
        devices[current_device]['t_ost'] = 0

        if len(queue) > 0:
            tmp = queue.pop(0)
```

```
            devices[current_device]['j'] = tmp[0]
            devices[current_device]['t_ost'] = tmp[1]
            table2[tmp[0] - 1]['k_j'] = current_device + 1

        current_device = -1
        min_time = np.inf
        for j in range(n):
            if devices[j]['j'] != 0 and min_time > devices[j]['t_ost']:
                min_time = devices[j]['t_ost']
                current_device = j
        if current_device == -1:
            table[i]['t_min'] = -1
        else:
            table[i]['t_min'] = min_time

    else:
        min_time -= table[i-1]['t_ozh']
        time_cummulative += table[i - 1]['t_ozh']
        total_j += 1
        for j in range(n):
            if devices[j]['j'] != 0:
                devices[j]['t_ost'] -= table[i - 1]['t_ozh']
                table3[j]['t_zan'] += table[i - 1]['t_ozh']

        table4[table[i-1]['c']]['T_100'] += table[i-1]['t_ozh']
        table4[table[i-1]['c']]['R_100'] += 1


        table[i]['l'] = i + 1
        table[i]['j'] = total_j
        table[i]['t_sob'] = time_cummulative
        table[i]['t_ozh'] = dT
        table[i]['type'] = 1
        table[i]['c'] = table[i-1]['c'] + 1

        counter = -1
        for j in range(n):
            if devices[j]['j'] == 0:
                counter = j
                break


        if counter == -1:
            queue.append([total_j, np.round(np.random.exponential(mu),
6)])
            table[i]['k'] = -1
            table2.append({'j' : total_j, 't_z' : time_cummulative, 'q_j'
: len(queue), 't_och' : 0, 't_nob' : 0, 't_obs' : min_time,
                           't_kob' : 0, 'k_j' : -1})
        else:
```

```
            devices[counter]['j'] = total_j
            devices[counter]['t_ost'] =
np.round(np.random.exponential(mu), 6)
            if devices[counter]['t_ost'] < min_time:
                min_time = devices[counter]['t_ost']
                current_device = counter
            table[i]['k'] = counter + 1
            table2.append({'j' : total_j, 't_z' : time_cummulative, 'q_j'
: 0, 't_och' : 0, 't_nob' : 0, 't_obs' : devices[counter]['t_ost'],
                        't_kob' : 0, 'k_j' : counter + 1})

        table[i]['t_min'] = min_time


end_service_flg = False
i = 100

while True:

    if len(queue) == 0:
        end_service_flg = True
        for j in range(n):
            if devices[j]['j'] != 0:
                end_service_flg = False
                break
        if end_service_flg:
            break


    time_cummulative += min_time
    table.append({'l' : 0, 't_sob' : 0, 'type' : 0, 'c' : 0, 't_min' : 0,
't_ozh' : 0, 'j' : 0, 'k' : 0})

    for j in range(n):
        if devices[j]['j'] != 0:
            devices[j]['t_ost'] -= min_time

    table[i]['l'] = i + 1
    table[i]['j'] = devices[current_device]['j']
    table[i]['t_sob'] = time_cummulative
    table[i]['t_ozh'] = np.round(table[i-1]['t_ozh'] - min_time, 6)
    table[i]['type'] = 2
    table[i]['c'] = table[i-1]['c'] - 1
    table[i]['k'] = current_device + 1

    table2[table[i]['j'] - 1]['t_kob'] = np.round(time_cummulative, 6)
    table2[table[i]['j'] - 1]['t_nob'] = np.round(time_cummulative -
table2[table[i]['j']-1]['t_obs'], 6)
    table2[table[i]['j']- 1]['t_och'] = np.round(table2[ table[i]['j'] -
1]['t_nob']  - table2[table[i]['j']-1]['t_z'], 6)
```

```
    devices[current_device]['j'] = 0
    devices[current_device]['t_ost'] = 0

    if len(queue) > 0:
        tmp = queue.pop(0)
        devices[current_device]['j'] = tmp[0]
        devices[current_device]['t_ost'] = tmp[1]
        table2[tmp[0] - 1]['k_j'] = current_device + 1

    current_device = -1
    min_time = np.inf
    for j in range(n):
        if devices[j]['j'] != 0 and min_time > devices[j]['t_ost']:
            min_time = devices[j]['t_ost']
            current_device = j
    if current_device == -1:
        table[i]['t_min'] = -1
    else:
        table[i]['t_min'] = min_time

    i += 1


t100 = table[99]['t_sob']
for i in range(n):
    table3[i]['t_pr'] = t100 - table3[i]['t_zan']
    table3[i]['delta_pr'] = table3[i]['t_pr']/t100

notnullindex = 0
for i in range(101):
    if table4[i]['R_100'] != 0:
        notnullindex = i

table4 = table4[:notnullindex + 1]

for i in range(len(table4)):
    table4[i]['nu_100'] = table4[i]['R_100']/100
    table4[i]['delta_100'] = table4[i]['T_100']/t100

table3.append({'N' : '', 't_zan' : 1/n * np.sum([table3[i]['t_zan'] for i
in range(n)]),
               't_pr' : 1/n * np.sum([table3[i]['t_pr'] for i in
range(n)]),
               'delta_pr' : 1/n * np.sum([table3[i]['delta_pr'] for i in
range(n)])})

table4.append({'sost' : '', 'R_100' : np.sum([table4[i]['R_100'] for i in
range(len(table4))]),
```

```
                'nu_100' : np.sum([table4[i]['nu_100'] for i in
range(len(table4))]),
                'T_100' : np.sum([table4[i]['T_100'] for i in
range(len(table4))]),
                'delta_100' : np.sum([table4[i]['delta_100'] for i in
range(len(table4))])})
```

```
table = table[:100]
table_df = pd.DataFrame.from_dict(table)
```

```
table_df.head()
table_df.to_excel('table1_1.xlsx', index = False)
```

```
table_df.tail()
```

```
table_2_df = pd.DataFrame.from_dict(table2)
```

```
table_2_df.to_excel('table1_2.xlsx', index = False)
```

```
table_3_df = pd.DataFrame.from_dict(table3)
table_3_df.to_excel('table1_3.xlsx', index = False)
table_4_df = pd.DataFrame.from_dict(table4)
table_4_df.to_excel('table1_4.xlsx', index = False)
```

```
table_3_df
```

```
table_4_df
```

```
serv_demands = 46
print(f"Count of served demands: {serv_demands}")
z = 0
for i in range(100):
    cnt = 0
    t_event = table[i]['t_sob']
    for j in range(len(table1)):
        if table1[j]['t_z'] >= t_event:
            cnt += 1
    z += cnt
print(f"z(100): {z/100}")

sum_q_time = 0
sum_stay_time = 0
for i in range(serv_demands):
    sum_q_time += table1[i]['t_och']
```

```
    sum_stay_time += (table1[i]['t_kob'] - table1[i]['t_z'])
q_time_mean = sum_q_time / serv_demands
print(f"q_time_mean: {round(q_time_mean, 5)}")
stay_time_mean = sum_stay_time / serv_demands
print(f"stay_time_mean: {round(stay_time_mean, 5)}")
```

Задание 2

```
table = [{'l' : 0, 't_sob' : 0, 'type' : 0, 'c' : 0, 't_min' : 0, 't_ozh'
: 0, 'j' : 0, 'k' : 0} for i in range(100)]
devices = [{'j' : 0, 'k' : i + 1, 't_ost' : 0} for i in range(n)]
table2 = []
table3 = [{'N' : i + 1, 't_zan' : 0, 't_pr' : 0, 'delta_pr' : 0} for i in
range(n)]
table4 = [{'sost' : i, 'R_100' : 0, 'nu_100' : 0 , 'T_100' :
0,  'delta_100' : 0 } for i in range(101)]

for i in range(n):
    devices[i]['k'] = i + 1
queue = []
current_device = -1
time_cummulative = 0
min_time = 0
total_j = 0
```

```
for i in range(100):

    if current_device == -1:

        dT =  np.round(np.random.exponential(lambd), 6)
        current_device = 0
        total_j += 1
        min_time = np.round(np.random.exponential(mu), 6)
        if i == 0:
            time_cummulative += dT
            table4[0]['T_100'] += dT
        else:
            time_cummulative += table[i-1]['t_ozh']
            table4[0]['T_100'] += table[i-1]['t_ozh']

        table4[0]['R_100'] += 1

        table[i]['l'] = i + 1
        table[i]['j'] = total_j
        table[i]['t_sob'] = time_cummulative
        table[i]['type'] = 1
        table[i]['c'] = 1
        table[i]['t_min'] = min_time
        table[i]['t_ozh'] = dT
```

```
        table[i]['k'] = 1

        devices[0]['j'] = total_j
        devices[0]['t_ost'] = min_time

        table2.append({'j' : total_j, 't_z' : time_cummulative, 'q_j' : 0,
't_och' : 0, 't_nob' : 0, 't_obs' : min_time,
                       't_kob' : 0, 'k_j' : 1})


    elif min_time < table[i-1]['t_ozh']:

        time_cummulative += min_time

        for j in range(n):
            if devices[j]['j'] != 0:
                devices[j]['t_ost'] -= min_time
                table3[j]['t_zan'] += min_time
        table4[table[i-1]['c']]['T_100'] += min_time
        table4[table[i-1]['c']]['R_100'] += 1

        table[i]['l'] = i + 1
        table[i]['j'] = devices[current_device]['j']
        table[i]['t_sob'] = time_cummulative
        table[i]['t_ozh'] = np.round(table[i-1]['t_ozh'] - min_time, 6)
        table[i]['type'] = 2
        table[i]['c'] = table[i-1]['c'] - 1
        table[i]['k'] = current_device + 1

        table2[table[i]['j'] - 1]['t_kob'] = np.round(time_cummulative, 6)
        table2[table[i]['j'] - 1]['t_nob'] = np.round(time_cummulative -
table2[table[i]['j']-1]['t_obs'], 6)
        table2[table[i]['j']- 1]['t_och'] = np.round(table2[ table[i]['j']
- 1]['t_nob']  - table2[table[i]['j']-1]['t_z'], 6)


        devices[current_device]['j'] = 0
        devices[current_device]['t_ost'] = 0

        if len(queue) > 0:
            tmp = queue.pop(0)
            devices[current_device]['j'] = tmp[0]
            devices[current_device]['t_ost'] = tmp[1]
            table2[tmp[0] - 1]['k_j'] = current_device + 1

        current_device = -1
        min_time = np.inf
        for j in range(n):
            if devices[j]['j'] != 0 and min_time > devices[j]['t_ost']:
                min_time = devices[j]['t_ost']
```

```
                current_device = j
        if current_device == -1:
            table[i]['t_min'] = -1
        else:
            table[i]['t_min'] = min_time

    else:
        dT =  np.round(np.random.exponential(lambd), 6)
        min_time -= table[i-1]['t_ozh']
        time_cummulative += table[i - 1]['t_ozh']
        total_j += 1
        for j in range(n):
            if devices[j]['j'] != 0:
                devices[j]['t_ost'] -= table[i - 1]['t_ozh']
                table3[j]['t_zan'] += table[i - 1]['t_ozh']

        table4[table[i-1]['c']]['T_100'] += table[i-1]['t_ozh']
        table4[table[i-1]['c']]['R_100'] += 1


        table[i]['l'] = i + 1
        table[i]['j'] = total_j
        table[i]['t_sob'] = time_cummulative
        table[i]['t_ozh'] = dT
        table[i]['type'] = 1
        table[i]['c'] = table[i-1]['c'] + 1

        counter = -1
        for j in range(n):
            if devices[j]['j'] == 0:
                counter = j
                break


        if counter == -1:
            queue.append([total_j, np.round(np.random.exponential(mu),
6)])
            table[i]['k'] = -1
            table2.append({'j' : total_j, 't_z' : time_cummulative, 'q_j'
: len(queue), 't_och' : 0, 't_nob' : 0, 't_obs' : min_time,
                           't_kob' : 0, 'k_j' : -1})
        else:
            devices[counter]['j'] = total_j
            devices[counter]['t_ost'] =
np.round(np.random.exponential(mu), 6)
            if devices[counter]['t_ost'] < min_time:
                min_time = devices[counter]['t_ost']
                current_device = counter
            table[i]['k'] = counter + 1
```

```
            table2.append({'j' : total_j, 't_z' : time_cummulative, 'q_j'
: 0, 't_och' : 0, 't_nob' : 0, 't_obs' : devices[counter]['t_ost'],
                           't_kob' : 0, 'k_j' : counter + 1})

        table[i]['t_min'] = min_time


end_service_flg = False
i = 100

while True:

    if len(queue) == 0:
        end_service_flg = True
        for j in range(n):
            if devices[j]['j'] != 0:
                end_service_flg = False
                break
        if end_service_flg:
            break


    time_cummulative += min_time
    table.append({'l' : 0, 't_sob' : 0, 'type' : 0, 'c' : 0, 't_min' : 0,
't_ozh' : 0, 'j' : 0, 'k' : 0})

    for j in range(n):
        if devices[j]['j'] != 0:
            devices[j]['t_ost'] -= min_time

    table[i]['l'] = i + 1
    table[i]['j'] = devices[current_device]['j']
    table[i]['t_sob'] = time_cummulative
    table[i]['t_ozh'] = np.round(table[i-1]['t_ozh'] - min_time, 6)
    table[i]['type'] = 2
    table[i]['c'] = table[i-1]['c'] - 1
    table[i]['k'] = current_device + 1

    table2[table[i]['j'] - 1]['t_kob'] = np.round(time_cummulative, 6)
    table2[table[i]['j'] - 1]['t_nob'] = np.round(time_cummulative -
table2[table[i]['j']-1]['t_obs'], 6)
    table2[table[i]['j']- 1]['t_och'] = np.round(table2[ table[i]['j'] -
1]['t_nob']  - table2[table[i]['j']-1]['t_z'], 6)


    devices[current_device]['j'] = 0
    devices[current_device]['t_ost'] = 0

    if len(queue) > 0:
        tmp = queue.pop(0)
        devices[current_device]['j'] = tmp[0]
```

```
        devices[current_device]['t_ost'] = tmp[1]
        table2[tmp[0] - 1]['k_j'] = current_device + 1

    current_device = -1
    min_time = np.inf
    for j in range(n):
        if devices[j]['j'] != 0 and min_time > devices[j]['t_ost']:
            min_time = devices[j]['t_ost']
            current_device = j
    if current_device == -1:
        table[i]['t_min'] = -1
    else:
        table[i]['t_min'] = min_time

    i += 1


t100 = table[99]['t_sob']
for i in range(n):
    table3[i]['t_pr'] = t100 - table3[i]['t_zan']
    table3[i]['delta_pr'] = table3[i]['t_pr']/t100

notnullindex = 0
for i in range(101):
    if table4[i]['R_100'] != 0:
        notnullindex = i

table4 = table4[:notnullindex + 1]

for i in range(len(table4)):
    table4[i]['nu_100'] = table4[i]['R_100']/100
    table4[i]['delta_100'] = table4[i]['T_100']/t100

table3.append({'N' : '', 't_zan' : 1/n * np.sum([table3[i]['t_zan'] for i
in range(n)]),
               't_pr' : 1/n * np.sum([table3[i]['t_pr'] for i in
range(n)]),
               'delta_pr' : 1/n * np.sum([table3[i]['delta_pr'] for i in
range(n)])})

table4.append({'sost' : '', 'R_100' : np.sum([table4[i]['R_100'] for i in
range(len(table4))]),
               'nu_100' : np.sum([table4[i]['nu_100'] for i in
range(len(table4))]),
               'T_100' : np.sum([table4[i]['T_100'] for i in
range(len(table4))]),
               'delta_100' : np.sum([table4[i]['delta_100'] for i in
range(len(table4))])})


table = table[:100]
```

```
table_df = pd.DataFrame.from_dict(table)
table2_df = pd.DataFrame.from_dict(table2)
table3_df = pd.DataFrame.from_dict(table3)
table4_df = pd.DataFrame.from_dict(table4)
table_df.to_excel('table2_1.xlsx', index = False)
table2_df.to_excel('table2_2.xlsx', index = False)
table3_df.to_excel('table2_3.xlsx', index = False)
table4_df.to_excel('table2_4.xlsx', index = False)
```

```
serv_demands = 48
print(f"Count of served demands: {serv_demands}")
z = 0
for i in range(100):
    cnt = 0
    t_event = table[i]['t_sob']
    for j in range(len(table2)):
        if table2[j]['t_z'] >= t_event:
            cnt += 1
    z += cnt
print(f"z(100): {z/100}")

sum_q_time = 0
sum_stay_time = 0
for i in range(serv_demands):
    sum_q_time += table2[i]['t_och']
    sum_stay_time += (table2[i]['t_kob'] - table2[i]['t_z'])
q_time_mean = sum_q_time / serv_demands
print(f"q_time_mean: {round(q_time_mean, 5)}")
stay_time_mean = sum_stay_time / serv_demands
print(f"stay_time_mean: {round(stay_time_mean, 5)}")
```

Задача 3

```
table = [{'l' : 0, 't_sob' : 0, 'type' : 0, 'c' : 0, 't_min' : 0, 't_ozh'
: 0, 'j' : 0, 'k' : 0} for i in range(100)]
devices = [{'j' : 0, 'k' : i + 1, 't_ost' : 0} for i in range(n)]
table2 = []
table3 = [{'N' : i + 1, 't_zan' : 0, 't_pr' : 0, 'delta_pr' : 0} for i in
range(n)]
table4 = [{'sost' : i, 'R_100' : 0, 'nu_100' : 0 , 'T_100' :
0,  'delta_100' : 0 } for i in range(101)]

for i in range(n):
    devices[i]['k'] = i + 1
queue = []
current_device = -1
time_cummulative = 0
min_time = 0
total_j = 0
```

```
for i in range(100):

    if current_device == -1:

        dT =  np.round(np.random.exponential(lambd), 6)
        current_device = 0
        total_j += 1
        min_time = np.round(np.random.exponential(mu), 6)
        if i == 0:
            time_cummulative += dT
            table4[0]['T_100'] += dT
        else:
            time_cummulative += table[i-1]['t_ozh']
            table4[0]['T_100'] += table[i-1]['t_ozh']

        table4[0]['R_100'] += 1

        table[i]['l'] = i + 1
        table[i]['j'] = total_j
        table[i]['t_sob'] = time_cummulative
        table[i]['type'] = 1
        table[i]['c'] = 1
        table[i]['t_min'] = min_time
        table[i]['t_ozh'] = dT
        table[i]['k'] = 1

        devices[0]['j'] = total_j
        devices[0]['t_ost'] = min_time

        table2.append({'j' : total_j, 't_z' : time_cummulative, 'q_j' : 0,
't_och' : 0, 't_nob' : 0, 't_obs' : min_time,
                       't_kob' : 0, 'k_j' : 1})


    elif min_time < table[i-1]['t_ozh']:

        time_cummulative += min_time

        for j in range(n):
            if devices[j]['j'] != 0:
                devices[j]['t_ost'] -= min_time
                table3[j]['t_zan'] += min_time
        table4[table[i-1]['c']]['T_100'] += min_time
        table4[table[i-1]['c']]['R_100'] += 1

        table[i]['l'] = i + 1
        table[i]['j'] = devices[current_device]['j']
        table[i]['t_sob'] = time_cummulative
        table[i]['t_ozh'] = np.round(table[i-1]['t_ozh'] - min_time, 6)
```

```
        table[i]['type'] = 3
        table[i]['c'] = table[i-1]['c'] - 1
        table[i]['k'] = current_device + 1

        table2[table[i]['j'] - 1]['t_kob'] = np.round(time_cummulative, 6)
        table2[table[i]['j'] - 1]['t_nob'] = np.round(time_cummulative -
table2[table[i]['j']-1]['t_obs'], 6)
        table2[table[i]['j']- 1]['t_och'] = np.round(table2[ table[i]['j']
- 1]['t_nob']  - table2[table[i]['j']-1]['t_z'], 6)


        devices[current_device]['j'] = 0
        devices[current_device]['t_ost'] = 0

        if len(queue) > 0:
            tmp = queue.pop(0)
            devices[current_device]['j'] = tmp[0]
            devices[current_device]['t_ost'] = tmp[1]
            table2[tmp[0] - 1]['k_j'] = current_device + 1

        current_device = -1
        min_time = np.inf
        for j in range(n):
            if devices[j]['j'] != 0 and min_time > devices[j]['t_ost']:
                min_time = devices[j]['t_ost']
                current_device = j
        if current_device == -1:
            table[i]['t_min'] = -1
        else:
            table[i]['t_min'] = min_time

    else:
        dT =  np.round(np.random.exponential(lambd), 6)
        min_time -= table[i-1]['t_ozh']
        time_cummulative += table[i - 1]['t_ozh']
        total_j += 1
        for j in range(n):
            if devices[j]['j'] != 0:
                devices[j]['t_ost'] -= table[i - 1]['t_ozh']
                table3[j]['t_zan'] += table[i - 1]['t_ozh']

        table4[table[i-1]['c']]['T_100'] += table[i-1]['t_ozh']
        table4[table[i-1]['c']]['R_100'] += 1


        table[i]['l'] = i + 1
        table[i]['j'] = total_j
        table[i]['t_sob'] = time_cummulative
        table[i]['t_ozh'] = dT
        table[i]['type'] = 1
```

```
        table[i]['c'] = table[i-1]['c'] + 1

        counter = -1
        for j in range(n):
            if devices[j]['j'] == 0:
                counter = j
                break


        if counter == -1:
            if len(queue) < m:
                queue.append([total_j, np.round(np.random.exponential(mu),
6)])
                table2.append({'j' : total_j, 't_z' : time_cummulative,
'q_j' : len(queue), 't_och' : 0, 't_nob' : 0, 't_obs' : min_time,
                        't_kob' : 0, 'k_j' : -1})
            else:
                table[i]['type'] = 2
                table[i]['c'] = table[i-1]['c']
                table2.append({'j' : total_j, 't_z' : time_cummulative,
'q_j' : -1, 't_och' : 0, 't_nob' : -1, 't_obs' : 0,
                        't_kob' : time_cummulative, 'k_j' : -2})
            table[i]['k'] = -1
        else:
            devices[counter]['j'] = total_j
            devices[counter]['t_ost'] =
np.round(np.random.exponential(mu), 6)
            if devices[counter]['t_ost'] < min_time:
                min_time = devices[counter]['t_ost']
                current_device = counter
            table[i]['k'] = counter + 1
            table2.append({'j' : total_j, 't_z' : time_cummulative, 'q_j'
: 0, 't_och' : 0, 't_nob' : 0, 't_obs' : devices[counter]['t_ost'],
                        't_kob' : 0, 'k_j' : counter + 1})

        table[i]['t_min'] = min_time


end_service_flg = False
i = 100

while True:

    if len(queue) == 0:
        end_service_flg = True
        for j in range(n):
            if devices[j]['j'] != 0:
                end_service_flg = False
                break
        if end_service_flg:
```

```
            break


    time_cummulative += min_time
    table.append({'l' : 0, 't_sob' : 0, 'type' : 0, 'c' : 0, 't_min' : 0,
't_ozh' : 0, 'j' : 0, 'k' : 0})

    for j in range(n):
        if devices[j]['j'] != 0:
            devices[j]['t_ost'] -= min_time

    table[i]['l'] = i + 1
    table[i]['j'] = devices[current_device]['j']
    table[i]['t_sob'] = time_cummulative
    table[i]['t_ozh'] = np.round(table[i-1]['t_ozh'] - min_time, 6)
    table[i]['type'] = 2
    table[i]['c'] = table[i-1]['c'] - 1
    table[i]['k'] = current_device + 1

    table2[table[i]['j'] - 1]['t_kob'] = np.round(time_cummulative, 6)
    table2[table[i]['j'] - 1]['t_nob'] = np.round(time_cummulative -
table2[table[i]['j']-1]['t_obs'], 6)
    table2[table[i]['j']- 1]['t_och'] = np.round(table2[ table[i]['j'] -
1]['t_nob']  - table2[table[i]['j']-1]['t_z'], 6)


    devices[current_device]['j'] = 0
    devices[current_device]['t_ost'] = 0

    if len(queue) > 0:
        tmp = queue.pop(0)
        devices[current_device]['j'] = tmp[0]
        devices[current_device]['t_ost'] = tmp[1]
        table2[tmp[0] - 1]['k_j'] = current_device + 1

    current_device = -1
    min_time = np.inf
    for j in range(n):
        if devices[j]['j'] != 0 and min_time > devices[j]['t_ost']:
            min_time = devices[j]['t_ost']
            current_device = j
    if current_device == -1:
        table[i]['t_min'] = -1
    else:
        table[i]['t_min'] = min_time

    i += 1


t100 = table[99]['t_sob']
for i in range(n):
```

```
    table3[i]['t_pr'] = t100 - table3[i]['t_zan']
    table3[i]['delta_pr'] = table3[i]['t_pr']/t100

notnullindex = 0
for i in range(101):
    if table4[i]['R_100'] != 0:
        notnullindex = i

table4 = table4[:notnullindex + 1]

for i in range(len(table4)):
    table4[i]['nu_100'] = table4[i]['R_100']/100
    table4[i]['delta_100'] = table4[i]['T_100']/t100

table3.append({'N' : '', 't_zan' : 1/n * np.sum([table3[i]['t_zan'] for i
in range(n)]),
               't_pr' : 1/n * np.sum([table3[i]['t_pr'] for i in
range(n)]),
               'delta_pr' : 1/n * np.sum([table3[i]['delta_pr'] for i in
range(n)])})

table4.append({'sost' : '', 'R_100' : np.sum([table4[i]['R_100'] for i in
range(len(table4))]),
               'nu_100' : np.sum([table4[i]['nu_100'] for i in
range(len(table4))]),
               'T_100' : np.sum([table4[i]['T_100'] for i in
range(len(table4))]),
               'delta_100' : np.sum([table4[i]['delta_100'] for i in
range(len(table4))])})
```

```
table = table[:100]
table_df = pd.DataFrame.from_dict(table)
table2_df = pd.DataFrame.from_dict(table2)
table3_df = pd.DataFrame.from_dict(table3)
table4_df = pd.DataFrame.from_dict(table4)
table_df.to_excel('table3_1.xlsx', index = False)
table2_df.to_excel('table3_2.xlsx', index = False)
table3_df.to_excel('table3_3.xlsx', index = False)
table4_df.to_excel('table3_4.xlsx', index = False)
```