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

ИНСТИТУТ КИБЕРНЕТИКИ

КАФЕДРА ВЫСШЕЙ МАТЕМАТИКИ

# Лабораторная работа 3

по дисциплине «**Системы массового обслуживания**»

Тема: “**Многоканальные системы массового обслуживания с бесконечной очередью”**

Выполнил:
Студент 4-го курса

Скорописцев М.М.

Группа: КМБО-04-18

МОСКВА 2021

## Оглавление

# 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. 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 $\mathrm{FIFO}$.

**I.** Система массового обслуживания $(\mathrm{D}|\mathrm{M}|\mathrm{n})$.

**Дано:**

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

− параметр $\mu$ показательного распределения времени обслуживания заявки каждым прибором.

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

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 поступит первая заявка (произойдет событие с номером 1). Момент наступления первого события (типа 1) равен $t_{соб}(1) = \Delta T_з$, в этот момент определяется время обслуживания $t_{обсл}(1)$ заявки 1 в соответствии с показательным законом распределения с параметром $\mu$. После события 1 система находится в состоянии 1.

**II.** Система массового обслуживания $(M|D|n)$.

**Дано:**

− среднее число заявок $\lambda$, поступающих за единицу времени (время между приходом заявок имеет показательное распределение с параметром $\lambda$);

− время обслуживания заявки прибором $T_{об}$ (заданная постоянная величина).

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 $\lambda$.

**III.** Система массового обслуживания $(M|M|n)$.

**Дано:**

− среднее число заявок $\lambda$, поступающих за единицу времени (время между приходом заявок имеет показательное распределение с параметром $\lambda$);

− параметр $\mu$ 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 $\lambda$, 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 $\mu$.

**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 $l$;

– минимальное оставшееся время $t_{осмин}(l)$ обслуживания приборами заявок после события $l$ (если после события все приборы свободны, 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 $l$.

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 $q(j)=0$);

– время пребывания заявки в очереди $t_{оч}(j)$;

– момент начала обслуживания заявки $t_{ноб}(j)$;

– время обслуживания заявки $t_{обсл}(j)$;

– момент $t_{коб}(j)$ окончания обслуживания заявки $j$ и выхода её из СМО.

4. Составить таблицу 3 с данными о приборах:

– номер прибора $k$;

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

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

– коэффициент простоя прибора на интервале $[0,t_{соб}(100)]$ (отношение времени простоя прибора на интервале $[0,t_{соб}(100)]$ к $t_{соб}(100)$);

5. 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 $\overline{z}(100)=\frac{1}{100}\sum_{l=1}^{100} z(l)$, где $z(l)$ – число заявок в СМО после события $l$;

– среднее время пребывания заявок в очереди на интервале $[0,t_{соб}(100)]$, 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 $\overline{t}_{CMO}(100) = \frac{1}{JF(100)} \sum_{j=1}^{JF(100)} [t_{коб}(j) - t_з(j)]$;

Для СМО (D|M|n) и (M|D|n) составить таблицу относительных частот пребывания СМО в состояниях следующего вида:

| $i$ | $\nu_i(100)$ |
|---|---|
| 0 | $\nu_0(100)$ |
| 1 | $\nu_1(100)$ |
| … | … |

где $i$ – состояние СМО, $\nu_i(100)$ – отношение числа попаданий СМО в состояние $i$ за 100 событий к 100.

Для СМО (M|M|n) найти первые значения стационарных вероятностей состояний $(r_0, r_1, r_2, ..., r_M)$, где $M = \max\{C(l),\ l = 1, ..., 100\}$ и составить таблицу относительных частот пребывания СМО в состояниях следующего вида:

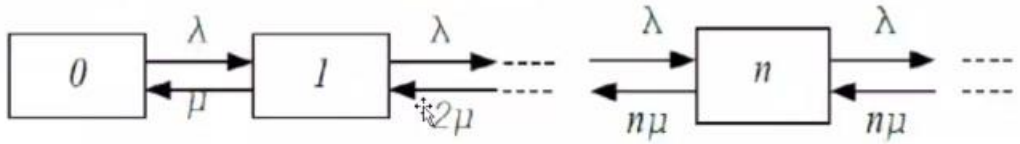
| $i$ | $r_i$ | $\nu_i(100)$ | $\lvert\nu_i(100) - r_i\rvert$ |
|---|---|---|---|
| 0 | $r_0$ | $\nu_0(100)$ | $\lvert\nu_0(100) - r_0\rvert$ |
| 1 | $r_1$ | $\nu_1(100)$ | $\lvert\nu_1(100) - r_1\rvert$ |
| … | … | … | … |
| $M$ | $r_M$ | $\nu_M(100)$ | $\lvert\nu_M(100) - r_M\rvert$ |
| | $\sum_{i=0}^{M} r_i$ | $\sum_{i=0}^{M} \nu_i(100)$ | $\max\{\lvert\nu_i(100) - r_i\rvert\}$ |

**Вывод результатов проводить с округлением до 0,00001.**

## Краткие теоретические сведения

### СМО $(M|M|n)$

Состояние СМО $(M|M|n)$ в любой момент времени $t$ характеризуется числом заявок $Z_t$, находящихся в СМО. СМО может находится в состояниях $k = 0, 1, 2,\ldots$:
в состоянии 0 прибор свободен, в состоянии $k \geq n+1$ прибор занят и $k-n$ заявка находится в очереди. Вероятности состояний $P(Z_t = k) = p_k(t)$, $k = 0, 1, 2,\ldots$.
Граф СМО $(M|M|n)$ имеет вид




Система дифференциальных уравнений Колмогорова для вероятностей состояний СМО $(M|M|n)$ имеет вид

$$\begin{cases} p_0'(t) = -\lambda p_0(t) + \mu p_1(t); \\ p_k'(t) = \lambda p_{k-1}(t) - (\lambda + k\mu) p_k(t) + (k+1)\mu p_{k+1}(t), \ 1 \leq k < n, \\ p_n'(t) = \lambda p_{n-1}(t) - (\lambda + n\mu) p_n(t) + n\mu p_{n+1}(t); \\ p_k'(t) = \lambda p_{k-1}(t) - (\lambda + n\mu) p_k(t) + n\mu p_{k+1}(t), \ k > n. \end{cases}$$

Функция распределения времени $\tau$ между двумя последовательно приходящими заявками

$$F_\tau(x)=\begin{cases}0, x\le 0;\\ 1-e^{-\lambda x}, x>0.\end{cases}$$

Плотность распределения времени $\tau$ между двумя последовательно приходящими заявками

$$f_\tau(x)=\begin{cases}0, x<0;\\ \lambda e^{-\lambda x}, x\ge 0.\end{cases}$$

Среднее время между двумя последовательно приходящими заявками $\overline{\tau}=\frac{1}{\lambda}$.

**Стационарные вероятности состояний**

Стационарные вероятности состояний $r_0, r_1, r_2, \ldots$ удовлетворяют системе линейных алгебраических уравнений

$$\begin{cases}0=-\lambda r_0+\mu r_1;\\ 0=\lambda r_{k-1}(t)-(\lambda+k\mu)r_k(t)+(k+1)\mu r_{k+1}(t),\ 1\le k<n;\\ 0=\lambda r_{n-1}(t)-(\lambda+n\mu)r_n(t)+n\mu r_{n+1}(t);\\ 0=\lambda r_{k-1}(t)-(\lambda+n\mu)r_k(t)+n\mu r_{k+1}(t),\ k>n.\end{cases}$$

а также уравнению нормировки

$$\sum_{k=0}^{\infty} r_k=1.$$

Следствия из СЛАУ для стационарных вероятностей состояний:

$r_1=\rho r_0$, $\lambda r_{k-1}=k\mu r_k$ при $1\le k\le n$, $\lambda r_{k-1}=n\mu r_k$ при $k\ge n+1$,

где $\rho=\frac{\lambda}{\mu}$.

Поэтому $r_k=\frac{\rho^k}{k!}r_0$. при $1\le k\le n$, $r_{n+l}=\nu^l r_n$ при $k=n+l$,

$l\ge 1$, где $\nu=\frac{\lambda}{n\mu}=\frac{\rho}{n}$. Из уравнения нормировки получаем, что при $\nu<1$

$$r_0=\{1+\rho+\frac{\rho^2}{2!}+\ldots+\frac{\rho^{n-1}}{(n-1)!}+\frac{\rho^n}{n!}\frac{1}{1-\nu}\}^{-1}.$$

**Вероятность отказа**

$$P_{\text{отк}} = 0.$$

**Относительная пропускная способность**

$$Q = 1.$$

**Абмолютная пропускная способность**

$$A = \lambda Q = \lambda.$$

**Среднее число занятых приборов**

$$\overline{k} = \rho.$$

**Средняя длина очереди**

$$\overline{q} = \frac{\nu r_n}{(1-\nu)^2}.$$

**Среднее число заявок, находящихся в СМО**

$$\overline{z} = \overline{k} + \overline{q} = \rho + \frac{\nu r_n}{(1-\nu)^2}.$$

**Среднего времени пребывания заявок в очереди**

$$\overline{t}_{оч} = \frac{\overline{q}}{\lambda} = \frac{r_n}{n\mu(1-\nu)^2} = \frac{n\mu r_n}{(n\mu-\lambda)^2}.$$

**Среднего времени пребывания заявок в СМО**

$$\overline{t}_{смо} = \frac{\overline{z}}{\lambda} = \frac{\overline{k}}{\lambda} + \frac{\overline{q}}{\lambda} = \frac{1}{\mu} + \frac{r_n}{n\mu(1-\nu)^2}.$$

**Средства языка программирования**

np.random.exponential(1/ λ , size) - возвращает случайное вещественное число из экспоненциального (показательного)

# Результаты расчетов

Вариант №72, n = 11 $T_{об} = 0.178$, $\Delta T_з = 1.741$ , $\lambda = 6.178$, $\mu = 0.571$

**Задание 1**

Система массового обслуживания (D|M|n).

$$\Delta T_з = 1.741 , \mu = 0.571$$

| $l$ | $t_{соб}(l)$ | $Type(l)$ | $C(l)$ | $t_{ост}(l)$ | $t_{ожз}(l)$ | $j(l)$ |
|---|---|---|---|---|---|---|
| 1 | 1.741 | 1 | 1 | 0.65043 | 1.741 | 1 |
| 2 | 2.39143 | 2 | 0 | -1 | 1.09057 | 1 |
| 3 | 3.482 | 1 | 1 | 1.17694 | 1.741 | 2 |
| 4 | 4.65894 | 2 | 0 | -1 | 0.56406 | 2 |
| 5 | 5.223 | 1 | 1 | 3.03249 | 1.741 | 3 |
| 6 | 6.964 | 1 | 2 | 1.29149 | 1.741 | 4 |
| 7 | 8.25549 | 2 | 1 | 1.17034 | 0.44951 | 3 |
| 8 | 8.705 | 1 | 2 | 0.23552 | 1.741 | 5 |
| 9 | 8.94052 | 2 | 1 | 0.48531 | 1.50548 | 5 |
| 10 | 9.42583 | 2 | 0 | -1 | 1.02017 | 4 |
| 11 | 10.446 | 1 | 1 | 2.75215 | 1.741 | 6 |
| 12 | 12.187 | 1 | 2 | 0.13082 | 1.741 | 7 |
| 13 | 12.31782 | 2 | 1 | 0.88033 | 1.61018 | 7 |
| 14 | 13.19815 | 2 | 0 | -1 | 0.72985 | 6 |
| 15 | 13.928 | 1 | 1 | 0.32926 | 1.741 | 8 |
| 16 | 14.25726 | 2 | 0 | -1 | 1.41174 | 8 |
| 17 | 15.669 | 1 | 1 | 3.53433 | 1.741 | 9 |
| 18 | 17.41 | 1 | 2 | 1.79333 | 1.741 | 10 |
| 19 | 19.151 | 1 | 3 | 0.05233 | 1.741 | 11 |
| 20 | 19.20333 | 2 | 2 | 0.33646 | 1.68867 | 9 |
| 21 | 19.53979 | 2 | 1 | 3.05432 | 1.35221 | 10 |
| 22 | 20.892 | 1 | 2 | 1.70211 | 1.741 | 12 |
| 23 | 22.59411 | 2 | 1 | 1.23881 | 0.03889 | 11 |
| 24 | 22.633 | 1 | 2 | 1.19992 | 1.741 | 13 |
| 25 | 23.83292 | 2 | 1 | 0.30814 | 0.54108 | 12 |
| 26 | 24.14106 | 2 | 0 | -1 | 0.23294 | 13 |
| 27 | 24.374 | 1 | 1 | 1.37363 | 1.741 | 14 |
| 28 | 25.74763 | 2 | 0 | -1 | 0.36737 | 14 |
| 29 | 26.115 | 1 | 1 | 0.62574 | 1.741 | 15 |
| 30 | 26.74074 | 2 | 0 | -1 | 1.11526 | 15 |
| 31 | 27.856 | 1 | 1 | 1.60045 | 1.741 | 16 |
| 32 | 29.45645 | 2 | 0 | -1 | 0.14055 | 16 |
| 33 | 29.597 | 1 | 1 | 0.33362 | 1.741 | 17 |
| 34 | 29.93062 | 2 | 0 | -1 | 1.40738 | 17 |
| 35 | 31.338 | 1 | 1 | 2.44824 | 1.741 | 18 |
| 36 | 33.079 | 1 | 2 | 0.70724 | 1.741 | 19 |
| 37 | 33.78624 | 2 | 1 | 6.15657 | 1.03376 | 18 |
| 38 | 34.82 | 1 | 2 | 2.01229 | 1.741 | 20 |

| 39 | 36.561 | 1 | 3 | 0.06646 | 1.741 | 21 |
|---|---|---|---|---|---|---|
| 40 | 36.62746 | 2 | 2 | 0.20483 | 1.67454 | 21 |
| 41 | 36.83229 | 2 | 1 | 3.11052 | 1.46971 | 20 |
| 42 | 38.302 | 1 | 2 | 0.38676 | 1.741 | 22 |
| 43 | 38.68876 | 2 | 1 | 1.25404 | 1.35424 | 22 |
| 44 | 39.94281 | 2 | 0 | -1 | 0.10019 | 19 |
| 45 | 40.043 | 1 | 1 | 3.17694 | 1.741 | 23 |
| 46 | 41.784 | 1 | 2 | 1.41864 | 1.741 | 24 |
| 47 | 43.20264 | 2 | 1 | 0.01731 | 0.32236 | 24 |
| 48 | 43.21994 | 2 | 0 | -1 | 0.30506 | 23 |
| 49 | 43.525 | 1 | 1 | 4.53263 | 1.741 | 25 |
| 50 | 45.266 | 1 | 2 | 2.79163 | 1.741 | 26 |
| 51 | 47.007 | 1 | 3 | 1.05063 | 1.741 | 27 |
| 52 | 48.05763 | 2 | 2 | 0.03354 | 0.69037 | 25 |
| 53 | 48.09117 | 2 | 1 | 2.10041 | 0.65683 | 27 |
| 54 | 48.748 | 1 | 2 | 1.44358 | 1.741 | 28 |
| 55 | 50.19158 | 2 | 1 | 0.03127 | 0.29742 | 26 |
| 56 | 50.22285 | 2 | 0 | -1 | 0.26615 | 28 |
| 57 | 50.489 | 1 | 1 | 3.11391 | 1.741 | 29 |
| 58 | 52.23 | 1 | 2 | 1.37291 | 1.741 | 30 |
| 59 | 53.60291 | 2 | 1 | 0.35381 | 0.36809 | 29 |
| 60 | 53.95673 | 2 | 0 | -1 | 0.01427 | 30 |
| 61 | 53.971 | 1 | 1 | 1.759 | 1.741 | 31 |
| 62 | 55.712 | 1 | 2 | 0.018 | 1.741 | 32 |
| 63 | 55.73 | 2 | 1 | 0.3694 | 1.723 | 31 |
| 64 | 56.0994 | 2 | 0 | -1 | 1.3536 | 32 |
| 65 | 57.453 | 1 | 1 | 0.79767 | 1.741 | 33 |
| 66 | 58.25067 | 2 | 0 | -1 | 0.94333 | 33 |
| 67 | 59.194 | 1 | 1 | 4.0957 | 1.741 | 34 |
| 68 | 60.935 | 1 | 2 | 0.57565 | 1.741 | 35 |
| 69 | 61.51065 | 2 | 1 | 1.77905 | 1.16535 | 35 |
| 70 | 62.676 | 1 | 2 | 0.6137 | 1.741 | 36 |
| 71 | 63.2897 | 2 | 1 | 1.84583 | 1.1273 | 34 |
| 72 | 64.417 | 1 | 2 | 0.71853 | 1.741 | 37 |
| 73 | 65.13553 | 2 | 1 | 1.09326 | 1.02247 | 36 |
| 74 | 66.158 | 1 | 2 | 0.07079 | 1.741 | 38 |
| 75 | 66.22879 | 2 | 1 | 0.15069 | 1.67021 | 37 |
| 76 | 66.37948 | 2 | 0 | -1 | 1.51952 | 38 |
| 77 | 67.899 | 1 | 1 | 0.51915 | 1.741 | 39 |
| 78 | 68.41815 | 2 | 0 | -1 | 1.22185 | 39 |
| 79 | 69.64 | 1 | 1 | 0.27067 | 1.741 | 40 |
| 80 | 69.91067 | 2 | 0 | -1 | 1.47033 | 40 |
| 81 | 71.381 | 1 | 1 | 0.7228 | 1.741 | 41 |
| 82 | 72.1038 | 2 | 0 | -1 | 1.0182 | 41 |
| 83 | 73.122 | 1 | 1 | 1.49218 | 1.741 | 42 |
| 84 | 74.61418 | 2 | 0 | -1 | 0.24882 | 42 |
| 85 | 74.863 | 1 | 1 | 3.13286 | 1.741 | 43 |
| 86 | 76.604 | 1 | 2 | 0.0627 | 1.741 | 44 |
| 87 | 76.6667 | 2 | 1 | 1.32916 | 1.6783 | 44 |
| 88 | 77.99586 | 2 | 0 | -1 | 0.34914 | 43 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 89 | 78.345 | 1 | 1 | 6.26698 | 1.741 | 45 |
| 90 | 80.086 | 1 | 2 | 2.181 | 1.741 | 46 |
| 91 | 81.827 | 1 | 3 | 0.02296 | 1.741 | 47 |
| 92 | 81.84996 | 2 | 2 | 0.41704 | 1.71804 | 47 |
| 93 | 82.267 | 2 | 1 | 2.34498 | 1.301 | 46 |
| 94 | 83.568 | 1 | 2 | 0.02584 | 1.741 | 48 |
| 95 | 83.59384 | 2 | 1 | 1.01813 | 1.71516 | 48 |
| 96 | 84.61198 | 2 | 0 | -1 | 0.69702 | 45 |
| 97 | 85.309 | 1 | 1 | 0.79652 | 1.741 | 49 |
| 98 | 86.10552 | 2 | 0 | -1 | 0.94448 | 49 |
| 99 | 87.05 | 1 | 1 | 0.95301 | 1.741 | 50 |
| 100 | 88.00301 | 2 | 0 | -1 | 0.78799 | 50 |

Таблица 2

| $j$ | $t_з(j)$ | $q(j)$ | $t_{оч}(j)$ | $t_{ноб}(j)$ | $t_{обсл}(j)$ | $t_{коб}(j)$ |
|---|---|---|---|---|---|---|
| 1 | 1.741 | 0 | 0 | 1.741 | 0.65043 | 2.39143 |
| 2 | 3.482 | 0 | 0 | 3.482 | 1.17694 | 4.65894 |
| 3 | 5.223 | 0 | 0 | 5.223 | 3.03249 | 8.25549 |
| 4 | 6.964 | 0 | 0 | 6.964 | 2.46183 | 9.42583 |
| 5 | 8.705 | 0 | 0 | 8.705 | 0.23552 | 8.94052 |
| 6 | 10.446 | 0 | 0 | 10.446 | 2.75215 | 13.19815 |
| 7 | 12.187 | 0 | 0 | 12.187 | 0.13082 | 12.31782 |
| 8 | 13.928 | 0 | 0 | 13.928 | 0.32926 | 14.25726 |
| 9 | 15.669 | 0 | 0 | 15.669 | 3.53433 | 19.20333 |
| 10 | 17.41 | 0 | 0 | 17.41 | 2.12979 | 19.53979 |
| 11 | 19.151 | 0 | 0 | 19.151 | 3.44311 | 22.59411 |
| 12 | 20.892 | 0 | 0 | 20.892 | 2.94092 | 23.83292 |
| 13 | 22.633 | 0 | 0 | 22.633 | 1.50806 | 24.14106 |
| 14 | 24.374 | 0 | 0 | 24.374 | 1.37363 | 25.74763 |
| 15 | 26.115 | 0 | 0 | 26.115 | 0.62574 | 26.74074 |
| 16 | 27.856 | 0 | 0 | 27.856 | 1.60045 | 29.45645 |
| 17 | 29.597 | 0 | 0 | 29.597 | 0.33362 | 29.93062 |
| 18 | 31.338 | 0 | 0 | 31.338 | 2.44824 | 33.78624 |
| 19 | 33.079 | 0 | 0 | 33.079 | 6.86381 | 39.94281 |
| 20 | 34.82 | 0 | 0 | 34.82 | 2.01229 | 36.83229 |
| 21 | 36.561 | 0 | 0 | 36.561 | 0.06646 | 36.62746 |
| 22 | 38.302 | 0 | 0 | 38.302 | 0.38676 | 38.68876 |
| 23 | 40.043 | 0 | 0 | 40.043 | 3.17694 | 43.21994 |
| 24 | 41.784 | 0 | 0 | 41.784 | 1.41864 | 43.20264 |
| 25 | 43.525 | 0 | 0 | 43.525 | 4.53263 | 48.05763 |
| 26 | 45.266 | 0 | 0 | 45.266 | 4.92558 | 50.19158 |
| 27 | 47.007 | 0 | 0 | 47.007 | 1.08417 | 48.09117 |
| 28 | 48.748 | 0 | 0 | 48.748 | 1.47485 | 50.22285 |
| 29 | 50.489 | 0 | 0 | 50.489 | 3.11391 | 53.60291 |
| 30 | 52.23 | 0 | 0 | 52.23 | 1.72673 | 53.95673 |
| 31 | 53.971 | 0 | 0 | 53.971 | 1.759 | 55.73 |
| 32 | 55.712 | 0 | 0 | 55.712 | 0.3874 | 56.0994 |
| 33 | 57.453 | 0 | 0 | 57.453 | 0.79767 | 58.25067 |
| 34 | 59.194 | 0 | 0 | 59.194 | 4.0957 | 63.2897 |
| 35 | 60.935 | 0 | 0 | 60.935 | 0.57565 | 61.51065 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 36 | 62.676 | 0 | 0 | 62.676 | 2.45953 | 65.13553 |
| 37 | 64.417 | 0 | 0 | 64.417 | 1.81179 | 66.22879 |
| 38 | 66.158 | 0 | 0 | 66.158 | 0.22148 | 66.37948 |
| 39 | 67.899 | 0 | 0 | 67.899 | 0.51915 | 68.41815 |
| 40 | 69.64 | 0 | 0 | 69.64 | 0.27067 | 69.91067 |
| 41 | 71.381 | 0 | 0 | 71.381 | 0.7228 | 72.1038 |
| 42 | 73.122 | 0 | 0 | 73.122 | 1.49218 | 74.61418 |
| 43 | 74.863 | 0 | 0 | 74.863 | 3.13286 | 77.99586 |
| 44 | 76.604 | 0 | 0 | 76.604 | 0.0627 | 76.6667 |
| 45 | 78.345 | 0 | 0 | 78.345 | 6.26698 | 84.61198 |
| 46 | 80.086 | 0 | 0 | 80.086 | 2.181 | 82.267 |
| 47 | 81.827 | 0 | 0 | 81.827 | 0.02296 | 81.84996 |
| 48 | 83.568 | 0 | 0 | 83.568 | 0.02584 | 83.59384 |
| 49 | 85.309 | 0 | 0 | 85.309 | 0.79652 | 86.10552 |
| 50 | 87.05 | 0 | 0 | 87.05 | 0.95301 | 88.00301 |

## Задание 2

Система массового обслуживания (M|D|n).

$T_{об} = 0.178,\ \lambda = 6.178,$

| $l$ | $t_{соб}(l)$ | $Type(l)$ | $C(l)$ | $t_{ост}(l)$ | $t_{ожз}(l)$ | $j(l)$ |
|---|---|---|---|---|---|---|
| 1 | 0.12425 | 1 | 1 | 0.178 | 0.18398 | 1 |
| 2 | 0.30225 | 2 | 0 | -1 | 0.00598 | 1 |
| 3 | 0.30823 | 1 | 1 | 0.178 | 0.02848 | 2 |
| 4 | 0.33671 | 1 | 2 | 0.14952 | 0.24977 | 3 |
| 5 | 0.48623 | 2 | 1 | 0.02848 | 0.10025 | 2 |
| 6 | 0.51471 | 2 | 0 | -1 | 0.07177 | 3 |
| 7 | 0.58649 | 1 | 1 | 0.178 | 0.04247 | 4 |
| 8 | 0.62895 | 1 | 2 | 0.13553 | 0.0015 | 5 |
| 9 | 0.63045 | 1 | 3 | 0.13403 | 0.3289 | 6 |
| 10 | 0.76449 | 2 | 2 | 0.04247 | 0.19487 | 4 |
| 11 | 0.80695 | 2 | 1 | 0.0015 | 0.15241 | 5 |
| 12 | 0.80845 | 2 | 0 | -1 | 0.1509 | 6 |
| 13 | 0.95936 | 1 | 1 | 0.178 | 0.31052 | 7 |
| 14 | 1.13736 | 2 | 0 | -1 | 0.13252 | 7 |
| 15 | 1.26988 | 1 | 1 | 0.178 | 0.11959 | 8 |
| 16 | 1.38947 | 1 | 2 | 0.05841 | 0.08939 | 9 |
| 17 | 1.44788 | 2 | 1 | 0.11959 | 0.03098 | 8 |
| 18 | 1.47887 | 1 | 2 | 0.08861 | 0.52945 | 10 |
| 19 | 1.56747 | 2 | 1 | 0.08939 | 0.44084 | 9 |
| 20 | 1.65687 | 2 | 0 | -1 | 0.35145 | 10 |
| 21 | 2.00831 | 1 | 1 | 0.178 | 0.24658 | 11 |
| 22 | 2.18631 | 2 | 0 | -1 | 0.06858 | 11 |
| 23 | 2.2549 | 1 | 1 | 0.178 | 0.162 | 12 |
| 24 | 2.41689 | 1 | 2 | 0.016 | 0.05136 | 13 |
| 25 | 2.4329 | 2 | 1 | 0.162 | 0.03536 | 12 |
| 26 | 2.46826 | 1 | 2 | 0.12664 | 0.13827 | 14 |
| 27 | 2.59489 | 2 | 1 | 0.05136 | 0.01163 | 13 |
| 28 | 2.60652 | 1 | 2 | 0.03973 | 0.03206 | 15 |
| 29 | 2.63859 | 1 | 3 | 0.00767 | 0.28645 | 16 |
| 30 | 2.64626 | 2 | 2 | 0.13827 | 0.27878 | 14 |
| 31 | 2.78452 | 2 | 1 | 0.03206 | 0.14052 | 15 |
| 32 | 2.81659 | 2 | 0 | -1 | 0.10845 | 16 |
| 33 | 2.92504 | 1 | 1 | 0.178 | 0.12961 | 17 |
| 34 | 3.05466 | 1 | 2 | 0.04839 | 0.04612 | 18 |
| 35 | 3.10077 | 1 | 3 | 0.00227 | 0.22185 | 19 |
| 36 | 3.10304 | 2 | 2 | 0.12961 | 0.21958 | 17 |
| 37 | 3.23266 | 2 | 1 | 0.04612 | 0.08997 | 18 |
| 38 | 3.27877 | 2 | 0 | -1 | 0.04385 | 19 |
| 39 | 3.32262 | 1 | 1 | 0.178 | 0.01059 | 20 |
| 40 | 3.33321 | 1 | 2 | 0.16741 | 0.03097 | 21 |
| 41 | 3.36419 | 1 | 3 | 0.13644 | 0.04781 | 22 |
| 42 | 3.412 | 1 | 4 | 0.08862 | 0.13654 | 23 |

| 43 | 3.50062 | 2 | 3 | 0.01059 | 0.04791 | 20 |
|---|---|---|---|---|---|---|
| 44 | 3.51121 | 2 | 2 | 0.03097 | 0.03733 | 21 |
| 45 | 3.54219 | 2 | 1 | 0.04781 | 0.00635 | 22 |
| 46 | 3.54854 | 1 | 2 | 0.04146 | 0.00031 | 24 |
| 47 | 3.54885 | 1 | 3 | 0.04115 | 0.01378 | 25 |
| 48 | 3.56263 | 1 | 4 | 0.02737 | 0.01258 | 26 |
| 49 | 3.57521 | 1 | 5 | 0.01479 | 0.07422 | 27 |
| 50 | 3.59 | 2 | 4 | 0.13654 | 0.05944 | 23 |
| 51 | 3.64943 | 1 | 5 | 0.0771 | 0.07355 | 28 |
| 52 | 3.72298 | 1 | 6 | 0.00356 | 0.12137 | 29 |
| 53 | 3.72654 | 2 | 5 | 0.00031 | 0.11782 | 24 |
| 54 | 3.72685 | 2 | 4 | 0.01378 | 0.11751 | 25 |
| 55 | 3.74063 | 2 | 3 | 0.01258 | 0.10373 | 26 |
| 56 | 3.75321 | 2 | 2 | 0.07422 | 0.09114 | 27 |
| 57 | 3.82743 | 2 | 1 | 0.07355 | 0.01692 | 28 |
| 58 | 3.84436 | 1 | 2 | 0.05663 | 0.02951 | 30 |
| 59 | 3.87386 | 1 | 3 | 0.02712 | 0.03436 | 31 |
| 60 | 3.90098 | 2 | 2 | 0.12137 | 0.00724 | 29 |
| 61 | 3.90823 | 1 | 3 | 0.11413 | 0.03032 | 32 |
| 62 | 3.93854 | 1 | 4 | 0.08381 | 0.14985 | 33 |
| 63 | 4.02236 | 2 | 3 | 0.02951 | 0.06604 | 30 |
| 64 | 4.05186 | 2 | 2 | 0.03436 | 0.03654 | 31 |
| 65 | 4.08623 | 2 | 1 | 0.03032 | 0.00217 | 32 |
| 66 | 4.0884 | 1 | 2 | 0.02815 | 0.06314 | 34 |
| 67 | 4.11654 | 2 | 1 | 0.14985 | 0.03499 | 33 |
| 68 | 4.15154 | 1 | 2 | 0.11486 | 0.12779 | 35 |
| 69 | 4.2664 | 2 | 1 | 0.06314 | 0.01293 | 34 |
| 70 | 4.27933 | 1 | 2 | 0.05021 | 0.15196 | 36 |
| 71 | 4.32954 | 2 | 1 | 0.12779 | 0.10176 | 35 |
| 72 | 4.43129 | 1 | 2 | 0.02604 | 0.01502 | 37 |
| 73 | 4.44631 | 1 | 3 | 0.01102 | 0.17183 | 38 |
| 74 | 4.45733 | 2 | 2 | 0.15196 | 0.16081 | 36 |
| 75 | 4.60929 | 2 | 1 | 0.01502 | 0.00885 | 37 |
| 76 | 4.61814 | 1 | 2 | 0.00617 | 0.0544 | 39 |
| 77 | 4.62431 | 2 | 1 | 0.17183 | 0.04823 | 38 |
| 78 | 4.67254 | 1 | 2 | 0.1236 | 0.2873 | 40 |
| 79 | 4.79614 | 2 | 1 | 0.0544 | 0.1637 | 39 |
| 80 | 4.85054 | 2 | 0 | -1 | 0.1093 | 40 |
| 81 | 4.95984 | 1 | 1 | 0.178 | 0.0445 | 41 |
| 82 | 5.00434 | 1 | 2 | 0.1335 | 0.19852 | 42 |
| 83 | 5.13784 | 2 | 1 | 0.0445 | 0.06502 | 41 |
| 84 | 5.18234 | 2 | 0 | -1 | 0.02052 | 42 |
| 85 | 5.20286 | 1 | 1 | 0.178 | 0.6556 | 43 |
| 86 | 5.38086 | 2 | 0 | -1 | 0.4776 | 43 |
| 87 | 5.85846 | 1 | 1 | 0.178 | 0.18835 | 44 |
| 88 | 6.03646 | 2 | 0 | -1 | 0.01035 | 44 |
| 89 | 6.04681 | 1 | 1 | 0.178 | 0.04541 | 45 |
| 90 | 6.09222 | 1 | 2 | 0.13259 | 0.00497 | 46 |
| 91 | 6.09719 | 1 | 3 | 0.12762 | 0.34518 | 47 |
| 92 | 6.22481 | 2 | 2 | 0.04541 | 0.21756 | 45 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 93 | 6.27022 | 2 | 1 | 0.00497 | 0.17215 | 46 |
| 94 | 6.27519 | 2 | 0 | -1 | 0.16718 | 47 |
| 95 | 6.44237 | 1 | 1 | 0.178 | 0.00907 | 48 |
| 96 | 6.45144 | 1 | 2 | 0.16893 | 0.40338 | 49 |
| 97 | 6.62037 | 2 | 1 | 0.00907 | 0.23444 | 48 |
| 98 | 6.62944 | 2 | 0 | -1 | 0.22538 | 49 |
| 99 | 6.85481 | 1 | 1 | 0.178 | 0.16186 | 50 |
| 100 | 7.01667 | 1 | 2 | 0.01614 | 0.03443 | 51 |

Таблица 2

| $j$ | $t_з(j)$ | $q(j)$ | $t_{оч}(j)$ | $t_{ноб}(j)$ | $t_{обсл}(j)$ | $t_{коб}(j)$ |
|---|---|---|---|---|---|---|
| 1 | 0.12425 | 0 | 0 | 0.12425 | 0.178 | 0.30225 |
| 2 | 0.30823 | 0 | 0 | 0.30823 | 0.178 | 0.48623 |
| 3 | 0.33671 | 0 | 0 | 0.33671 | 0.178 | 0.51471 |
| 4 | 0.58649 | 0 | 0 | 0.58649 | 0.178 | 0.76449 |
| 5 | 0.62895 | 0 | 0 | 0.62895 | 0.178 | 0.80695 |
| 6 | 0.63045 | 0 | 0 | 0.63045 | 0.178 | 0.80845 |
| 7 | 0.95936 | 0 | 0 | 0.95936 | 0.178 | 1.13736 |
| 8 | 1.26988 | 0 | 0 | 1.26988 | 0.178 | 1.44788 |
| 9 | 1.38947 | 0 | 0 | 1.38947 | 0.178 | 1.56747 |
| 10 | 1.47887 | 0 | 0 | 1.47887 | 0.178 | 1.65687 |
| 11 | 2.00831 | 0 | 0 | 2.00831 | 0.178 | 2.18631 |
| 12 | 2.2549 | 0 | 0 | 2.2549 | 0.178 | 2.4329 |
| 13 | 2.41689 | 0 | 0 | 2.41689 | 0.178 | 2.59489 |
| 14 | 2.46826 | 0 | 0 | 2.46826 | 0.178 | 2.64626 |
| 15 | 2.60652 | 0 | 0 | 2.60652 | 0.178 | 2.78452 |
| 16 | 2.63859 | 0 | 0 | 2.63859 | 0.178 | 2.81659 |
| 17 | 2.92504 | 0 | 0 | 2.92504 | 0.178 | 3.10304 |
| 18 | 3.05466 | 0 | 0 | 3.05466 | 0.178 | 3.23266 |
| 19 | 3.10077 | 0 | 0 | 3.10077 | 0.178 | 3.27877 |
| 20 | 3.32262 | 0 | 0 | 3.32262 | 0.178 | 3.50062 |
| 21 | 3.33321 | 0 | 0 | 3.33321 | 0.178 | 3.51121 |
| 22 | 3.36419 | 0 | 0 | 3.36419 | 0.178 | 3.54219 |
| 23 | 3.412 | 0 | 0 | 3.412 | 0.178 | 3.59 |
| 24 | 3.54854 | 0 | 0 | 3.54854 | 0.178 | 3.72654 |
| 25 | 3.54885 | 0 | 0 | 3.54885 | 0.178 | 3.72685 |
| 26 | 3.56263 | 0 | 0 | 3.56263 | 0.178 | 3.74063 |
| 27 | 3.57521 | 0 | 0 | 3.57521 | 0.178 | 3.75321 |
| 28 | 3.64943 | 0 | 0 | 3.64943 | 0.178 | 3.82743 |
| 29 | 3.72298 | 0 | 0 | 3.72298 | 0.178 | 3.90098 |
| 30 | 3.84436 | 0 | 0 | 3.84436 | 0.178 | 4.02236 |
| 31 | 3.87386 | 0 | 0 | 3.87386 | 0.178 | 4.05186 |
| 32 | 3.90823 | 0 | 0 | 3.90823 | 0.178 | 4.08623 |
| 33 | 3.93854 | 0 | 0 | 3.93854 | 0.178 | 4.11654 |
| 34 | 4.0884 | 0 | 0 | 4.0884 | 0.178 | 4.2664 |
| 35 | 4.15154 | 0 | 0 | 4.15154 | 0.178 | 4.32954 |
| 36 | 4.27933 | 0 | 0 | 4.27933 | 0.178 | 4.45733 |
| 37 | 4.43129 | 0 | 0 | 4.43129 | 0.178 | 4.60929 |
| 38 | 4.44631 | 0 | 0 | 4.44631 | 0.178 | 4.62431 |
| 39 | 4.61814 | 0 | 0 | 4.61814 | 0.178 | 4.79614 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 40 | 4.67254 | 0 | 0 | 4.67254 | 0.178 | 4.85054 |
| 41 | 4.95984 | 0 | 0 | 4.95984 | 0.178 | 5.13784 |
| 42 | 5.00434 | 0 | 0 | 5.00434 | 0.178 | 5.18234 |
| 43 | 5.20286 | 0 | 0 | 5.20286 | 0.178 | 5.38086 |
| 44 | 5.85846 | 0 | 0 | 5.85846 | 0.178 | 6.03646 |
| 45 | 6.04681 | 0 | 0 | 6.04681 | 0.178 | 6.22481 |
| 46 | 6.09222 | 0 | 0 | 6.09222 | 0.178 | 6.27022 |
| 47 | 6.09719 | 0 | 0 | 6.09719 | 0.178 | 6.27519 |
| 48 | 6.44237 | 0 | 0 | 6.44237 | 0.178 | 6.62037 |
| 49 | 6.45144 | 0 | 0 | 6.45144 | 0.178 | 6.62944 |
| 50 | 6.85481 | 0 | 0 | 6.85481 | 0.178 | 7.03281 |
| 51 | 7.01667 | 0 | 0 | 7.01667 | 0.178 | 7.19467 |

## Задание 3

Система массового обслуживания (M|M|n).

$\lambda = 6.178,\ \mu = 0.571$

| $l$ | $t_{\text{соб}}(l)$ | $Type(l)$ | $C(l)$ | $t_{\text{ост}}(l)$ | $t_{\text{ожз}}(l)$ | $j(l)$ |
|---|---|---|---|---|---|---|
| 1 | 0.04947 | 1 | 1 | 1.15165 | 0.04996 | 1 |
| 2 | 0.09943 | 1 | 2 | 0.55276 | 0.05136 | 2 |
| 3 | 0.15079 | 1 | 3 | 0.5014 | 0.20726 | 3 |
| 4 | 0.35805 | 1 | 4 | 0.29414 | 0.14826 | 4 |
| 5 | 0.50631 | 1 | 5 | 0.14588 | 0.00234 | 5 |
| 6 | 0.50865 | 1 | 6 | 0.14354 | 0.17005 | 6 |
| 7 | 0.65219 | 2 | 5 | 0.54894 | 0.02651 | 2 |
| 8 | 0.6787 | 1 | 6 | 0.52242 | 0.15305 | 7 |
| 9 | 0.83175 | 1 | 7 | 0.36937 | 0.31272 | 8 |
| 10 | 1.14447 | 1 | 8 | 0.05665 | 0.07969 | 9 |
| 11 | 1.20112 | 2 | 7 | 0.20185 | 0.02304 | 1 |
| 12 | 1.22416 | 1 | 8 | 0.17881 | 0.12538 | 10 |
| 13 | 1.34954 | 1 | 9 | 0.05343 | 0.02906 | 11 |
| 14 | 1.3786 | 1 | 10 | 0.02437 | 0.23742 | 12 |
| 15 | 1.40297 | 2 | 9 | 0.36911 | 0.21305 | 7 |
| 16 | 1.61602 | 1 | 10 | 0.14508 | 0.08133 | 13 |
| 17 | 1.69736 | 1 | 11 | 0.06375 | 0.08781 | 14 |
| 18 | 1.7611 | 2 | 10 | 0.01098 | 0.02407 | 13 |
| 19 | 1.77208 | 2 | 9 | 0.201 | 0.01309 | 5 |
| 20 | 1.78517 | 1 | 10 | 0.18791 | 0.02368 | 15 |
| 21 | 1.80886 | 1 | 11 | 0.16422 | 0.15416 | 16 |
| 22 | 1.96302 | 1 | 12 | 0.01006 | 0.1166 | 17 |
| 23 | 1.97308 | 2 | 11 | 0.09816 | 0.10654 | 10 |
| 24 | 2.07124 | 2 | 10 | 0.08665 | 0.00838 | 14 |
| 25 | 2.07962 | 1 | 11 | 0.07827 | 0.1194 | 18 |
| 26 | 2.15789 | 2 | 10 | 0.02304 | 0.04113 | 8 |
| 27 | 2.18093 | 2 | 9 | 0.05621 | 0.01809 | 9 |
| 28 | 2.19902 | 1 | 10 | 0.03813 | 0.19487 | 19 |
| 29 | 2.23714 | 2 | 9 | 0.05092 | 0.15675 | 6 |
| 30 | 2.28806 | 2 | 8 | 0.37863 | 0.10583 | 15 |

| 31 | 2.39389 | 1 | 9 | 0.2728 | 0.11795 | 20 |
|---|---|---|---|---|---|---|
| 32 | 2.51184 | 1 | 10 | 0.15485 | 0.56908 | 21 |
| 33 | 2.66669 | 2 | 9 | 0.12696 | 0.41423 | 18 |
| 34 | 2.79365 | 2 | 8 | 0.11383 | 0.28727 | 4 |
| 35 | 2.90748 | 2 | 7 | 0.51311 | 0.17344 | 19 |
| 36 | 3.08092 | 1 | 8 | 0.33966 | 0.33709 | 22 |
| 37 | 3.41801 | 1 | 9 | 0.00257 | 0.01018 | 23 |
| 38 | 3.42058 | 2 | 8 | 0.54358 | 0.00761 | 20 |
| 39 | 3.42819 | 1 | 9 | 0.53598 | 0.1988 | 24 |
| 40 | 3.62699 | 1 | 10 | 0.33717 | 0.13649 | 25 |
| 41 | 3.76348 | 1 | 11 | 0.20068 | 0.04914 | 26 |
| 42 | 3.81263 | 1 | 12 | 0.15154 | 0.67448 | 27 |
| 43 | 3.96416 | 2 | 11 | 0.21961 | 0.52294 | 11 |
| 44 | 4.18377 | 2 | 10 | 0.09791 | 0.30334 | 16 |
| 45 | 4.28168 | 2 | 9 | 0.20412 | 0.20543 | 21 |
| 46 | 4.4858 | 2 | 8 | 0.28446 | 0.00131 | 27 |
| 47 | 4.48711 | 1 | 9 | 0.28316 | 0.06864 | 28 |
| 48 | 4.55575 | 1 | 10 | 0.21451 | 0.06671 | 29 |
| 49 | 4.62246 | 1 | 11 | 0.1478 | 0.13027 | 30 |
| 50 | 4.75273 | 1 | 12 | 0.01753 | 0.03972 | 31 |
| 51 | 4.77027 | 2 | 11 | 0.03473 | 0.02219 | 22 |
| 52 | 4.79245 | 1 | 12 | 0.01254 | 0.32636 | 32 |
| 53 | 4.805 | 2 | 11 | 0.24098 | 0.31381 | 30 |
| 54 | 5.03348 | 2 | 10 | 0.0125 | 0.08533 | 32 |
| 55 | 5.04598 | 2 | 9 | 0.15197 | 0.07283 | 26 |
| 56 | 5.11881 | 1 | 10 | 0.07914 | 0.055 | 33 |
| 57 | 5.17382 | 1 | 11 | 0.02414 | 0.39446 | 34 |
| 58 | 5.19796 | 2 | 10 | 0.20829 | 0.37032 | 12 |
| 59 | 5.40624 | 2 | 9 | 0.39178 | 0.16204 | 3 |
| 60 | 5.56828 | 1 | 10 | 0.22974 | 0.05351 | 35 |
| 61 | 5.62179 | 1 | 11 | 0.17624 | 0.14081 | 36 |
| 62 | 5.76259 | 1 | 12 | 0.03543 | 0.19068 | 37 |
| 63 | 5.79803 | 2 | 11 | 0.13895 | 0.15524 | 31 |
| 64 | 5.93697 | 2 | 10 | 0.08189 | 0.0163 | 33 |
| 65 | 5.95327 | 1 | 11 | 0.0656 | 0.03439 | 38 |
| 66 | 5.98766 | 1 | 12 | 0.0312 | 0.29306 | 39 |
| 67 | 6.01887 | 2 | 11 | 0.0866 | 0.26185 | 17 |
| 68 | 6.10547 | 2 | 10 | 0.17904 | 0.17525 | 29 |
| 69 | 6.28072 | 1 | 11 | 0.00379 | 0.37172 | 40 |
| 70 | 6.28451 | 2 | 10 | 0.16276 | 0.36793 | 38 |
| 71 | 6.44727 | 2 | 9 | 0.04866 | 0.20517 | 25 |
| 72 | 6.49594 | 2 | 8 | 0.28567 | 0.15651 | 40 |
| 73 | 6.65244 | 1 | 9 | 0.12917 | 0.01817 | 41 |
| 74 | 6.67061 | 1 | 10 | 0.111 | 0.06888 | 42 |
| 75 | 6.73949 | 1 | 11 | 0.04212 | 0.04086 | 43 |
| 76 | 6.78035 | 1 | 12 | 0.00125 | 0.12856 | 44 |
| 77 | 6.78161 | 2 | 11 | 0.19239 | 0.1273 | 35 |
| 78 | 6.90891 | 1 | 12 | 0.06509 | 0.05126 | 45 |
| 79 | 6.96017 | 1 | 13 | 0.01383 | 0.06241 | 46 |
| 80 | 6.974 | 2 | 12 | 0.12602 | 0.04859 | 24 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 81 | 7.02259 | 1 | 13 | 0.02949 | 0.00233 | 47 |
| 82 | 7.02491 | 1 | 14 | 0.02716 | 0.05708 | 48 |
| 83 | 7.05208 | 2 | 13 | 0.04794 | 0.02991 | 45 |
| 84 | 7.08199 | 1 | 14 | 0.01803 | 0.2341 | 49 |
| 85 | 7.10002 | 2 | 13 | 0.82051 | 0.21607 | 44 |
| 86 | 7.16336 | 2 | 12 | 0.75717 | 0.15273 | 47 |
| 87 | 7.31609 | 1 | 13 | 0.21235 | 0.35687 | 50 |
| 88 | 7.52844 | 2 | 12 | 0.39209 | 0.14452 | 48 |
| 89 | 7.67296 | 1 | 13 | 0.24757 | 0.13374 | 51 |
| 90 | 7.8067 | 1 | 14 | 0.11383 | 0.06877 | 52 |
| 91 | 7.87547 | 1 | 15 | 0.04505 | 0.14134 | 53 |
| 92 | 7.92053 | 2 | 14 | 0.06827 | 0.09629 | 41 |
| 93 | 7.9888 | 2 | 13 | 0.07723 | 0.02802 | 36 |
| 94 | 8.01682 | 1 | 14 | 0.04922 | 0.026 | 54 |
| 95 | 8.04281 | 1 | 15 | 0.02322 | 0.14585 | 55 |
| 96 | 8.06603 | 2 | 14 | 0.25198 | 0.12263 | 34 |
| 97 | 8.12673 | 2 | 13 | 0.19129 | 0.06193 | 52 |
| 98 | 8.18866 | 1 | 14 | 0.12935 | 0.16103 | 56 |
| 99 | 8.31801 | 2 | 13 | 0.11103 | 0.03168 | 39 |
| 100 | 8.34969 | 1 | 14 | 0.07935 | 0.4794 | 57 |

Таблица 2

| $j$ | $t_з(j)$ | $q(j)$ | $t_{оч}(j)$ | $t_{ноб}(j)$ | $t_{обсл}(j)$ | $t_{коб}(j)$ |
|---|---|---|---|---|---|---|
| 1 | 0.04947 | 0 | 0 | 0.04947 | 1.15165 | 1.20112 |
| 2 | 0.09943 | 0 | 0 | 0.09943 | 0.55276 | 0.65219 |
| 3 | 0.15079 | 0 | 0 | 0.15079 | 5.25546 | 5.40624 |
| 4 | 0.35805 | 0 | 0 | 0.35805 | 2.4356 | 2.79365 |
| 5 | 0.50631 | 0 | 0 | 0.50631 | 1.26578 | 1.77208 |
| 6 | 0.50865 | 0 | 0 | 0.50865 | 1.7285 | 2.23714 |
| 7 | 0.6787 | 0 | 0 | 0.6787 | 0.72427 | 1.40297 |
| 8 | 0.83175 | 0 | 0 | 0.83175 | 1.32614 | 2.15789 |
| 9 | 1.14447 | 0 | 0 | 1.14447 | 1.03646 | 2.18093 |
| 10 | 1.22416 | 0 | 0 | 1.22416 | 0.74892 | 1.97308 |
| 11 | 1.34954 | 0 | 0 | 1.34954 | 2.61462 | 3.96416 |
| 12 | 1.3786 | 0 | 0 | 1.3786 | 3.81935 | 5.19796 |
| 13 | 1.61602 | 0 | 0 | 1.61602 | 0.14508 | 1.7611 |
| 14 | 1.69736 | 0 | 0 | 1.69736 | 0.37389 | 2.07124 |
| 15 | 1.78517 | 0 | 0 | 1.78517 | 0.50289 | 2.28806 |
| 16 | 1.80886 | 0 | 0 | 1.80886 | 2.37491 | 4.18377 |
| 17 | 1.96302 | 1 | 0.01006 | 1.97308 | 4.04579 | 6.01887 |
| 18 | 2.07962 | 0 | 0 | 2.07962 | 0.58707 | 2.66669 |
| 19 | 2.19902 | 0 | 0 | 2.19902 | 0.70846 | 2.90748 |
| 20 | 2.39389 | 0 | 0 | 2.39389 | 1.02669 | 3.42058 |
| 21 | 2.51184 | 0 | 0 | 2.51184 | 1.76984 | 4.28168 |
| 22 | 3.08092 | 0 | 0 | 3.08092 | 1.68934 | 4.77027 |
| 23 | 3.41801 | 0 | 0 | 3.41801 | 6.59968 | 10.01769 |
| 24 | 3.42819 | 0 | 0 | 3.42819 | 3.54581 | 6.974 |
| 25 | 3.62699 | 0 | 0 | 3.62699 | 2.82028 | 6.44727 |
| 26 | 3.76348 | 0 | 0 | 3.76348 | 1.2825 | 5.04598 |
| 27 | 3.81263 | 1 | 0.15154 | 3.96416 | 0.52164 | 4.4858 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 28 | 4.48711 | 0 | 0 | 4.48711 | 4.04694 | 8.53405 |
| 29 | 4.55575 | 0 | 0 | 4.55575 | 1.54972 | 6.10547 |
| 30 | 4.62246 | 0 | 0 | 4.62246 | 0.18254 | 4.805 |
| 31 | 4.75273 | 1 | 0.01753 | 4.77027 | 1.02776 | 5.79803 |
| 32 | 4.79245 | 1 | 0.01254 | 4.805 | 0.22848 | 5.03348 |
| 33 | 5.11881 | 0 | 0 | 5.11881 | 0.81816 | 5.93697 |
| 34 | 5.17382 | 0 | 0 | 5.17382 | 2.89222 | 8.06603 |
| 35 | 5.56828 | 0 | 0 | 5.56828 | 1.21333 | 6.78161 |
| 36 | 5.62179 | 0 | 0 | 5.62179 | 2.36701 | 7.9888 |
| 37 | 5.76259 | 1 | 0.03543 | 5.79803 | 2.74995 | 8.54798 |
| 38 | 5.95327 | 0 | 0 | 5.95327 | 0.33124 | 6.28451 |
| 39 | 5.98766 | 1 | 0.0312 | 6.01887 | 2.29915 | 8.31801 |
| 40 | 6.28072 | 0 | 0 | 6.28072 | 0.21522 | 6.49594 |
| 41 | 6.65244 | 0 | 0 | 6.65244 | 1.26809 | 7.92053 |
| 42 | 6.67061 | 0 | 0 | 6.67061 | 6.78964 | 13.46025 |
| 43 | 6.73949 | 0 | 0 | 6.73949 | 2.64381 | 9.3833 |
| 44 | 6.78035 | 1 | 0.00125 | 6.78161 | 0.31841 | 7.10002 |
| 45 | 6.90891 | 1 | 0.06509 | 6.974 | 0.07808 | 7.05208 |
| 46 | 6.96017 | 2 | 0.0919 | 7.05208 | 3.54013 | 10.59221 |
| 47 | 7.02259 | 2 | 0.07743 | 7.10002 | 0.06334 | 7.16336 |
| 48 | 7.02491 | 3 | 0.13845 | 7.16336 | 0.36508 | 7.52844 |
| 49 | 7.08199 | 3 | 0.44645 | 7.52844 | 0.9006 | 8.42904 |
| 50 | 7.31609 | 2 | 0.60444 | 7.92053 | 0.58823 | 8.50876 |
| 51 | 7.67296 | 2 | 0.31584 | 7.9888 | 1.13704 | 9.12584 |
| 52 | 7.8067 | 3 | 0.25933 | 8.06603 | 0.06069 | 8.12673 |
| 53 | 7.87547 | 4 | 0.25125 | 8.12673 | 2.18987 | 10.3166 |
| 54 | 8.01682 | 3 | 0.3012 | 8.31801 | 0.94009 | 9.2581 |
| 55 | 8.04281 | 4 | -1 | -1 | -1 | -1 |
| 56 | 8.18866 | 3 | -1 | -1 | -1 | -1 |
| 57 | 8.34969 | 3 | -1 | -1 | -1 | -1 |

# Анализ результатов

### Система массового обслуживания (D|M|n)

Таблица 3

| $k$ | $N(k)$ | $t_{зан}(k)$ | Коэффициент простоя |
|---|---|---|---|
| 0 | 31 | 36.22704 | 0.58834 |
| 1 | 15 | 19.02818 | 0.78378 |
| 2 | 4 | 2.79163 | 0.96828 |
| 3 | 0 | 0 | 1.0 |
| 4 | 0 | 0 | 1.0 |
| 5 | 0 | 0 | 1.0 |
| 6 | 0 | 0 | 1.0 |
| 7 | 0 | 0 | 1.0 |
| 8 | 0 | 0 | 1.0 |
| 9 | 0 | 0 | 1.0 |
| 10 | 0 | 0 | 1.0 |

### Система массового обслуживания (M|D|n)

Таблица 3

| $k$ | $N(k)$ | $t_{зан}(k)$ | Коэффициент простоя |
|---|---|---|---|
| 0 | 22 | 1.69705 | 0.75814 |
| 1 | 16 | 1.71998 | 0.75487 |
| 2 | 8 | 1.25406 | 0.82127 |
| 3 | 3 | 0.41684 | 0.94059 |
| 4 | 1 | 0.16542 | 0.97643 |
| 5 | 1 | 0.15088 | 0.9785 |
| 6 | 0 | 0 | 1.0 |
| 7 | 0 | 0 | 1.0 |
| 8 | 0 | 0 | 1.0 |
| 9 | 0 | 0 | 1.0 |
| 10 | 0 | 0 | 1.0 |

Относительные частоты пребывания СМО в состояниях

| | (D\|M\|n) | (M\|D\|n) |
|---|---|---|
| $i$ | $v_i(100)$ | $v_i(100)$ |
| 0 | 0.25 | 0.14 |
| 1 | 0.46 | 0.35 |
| 2 | 0.25 | 0.3 |
| 3 | 0.04 | 0.12 |
| 4 | | 0.05 |
| 5 | | 0.03 |
| 6 | | 0.01 |

## Система массового обслуживания (M|M|n)

Таблица 3

| $k$ | $N(k)$ | $t_{зан}(k)$ | Коэффициент простоя |
|---|---|---|---|
| 0 | 5 | 8.04991 | 0.0359 |
| 1 | 8 | 7.54706 | 0.09613 |
| 2 | 6 | 7.5336 | 0.09774 |
| 3 | 4 | 7.52445 | 0.09884 |
| 4 | 3 | 7.27269 | 0.12899 |
| 5 | 6 | 6.64416 | 0.20426 |
| 6 | 3 | 6.77418 | 0.18869 |
| 7 | 3 | 5.35002 | 0.35926 |
| 8 | 7 | 5.56746 | 0.33321 |
| 9 | 3 | 6.45484 | 0.22694 |
| 10 | 6 | 5.0699 | 0.3928 |

r = (0, 3e-05, 0.00015, 0.00054, 0.00147, 0.00319, 0.00575, 0.00889, 0.01202, 0.01445, 0.01564, 0.01538, 0.01513, 0.01488, 0.01464, 0.0144)

| $i$ | $r_i$ | $v_i(100)$ | $\lvert v_i(100) - r_i \rvert$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 1 | 3e-05 | 0.01 | 0.00997 |
| 2 | 0.00015 | 0.01 | 0.00985 |
| 3 | 0.00054 | 0.01 | 0.00946 |
| 4 | 0.00147 | 0.01 | 0.00853 |
| 5 | 0.00319 | 0.02 | 0.01681 |
| 6 | 0.00575 | 0.02 | 0.01425 |
| 7 | 0.00889 | 0.03 | 0.02111 |
| 8 | 0.01202 | 0.08 | 0.06798 |
| 9 | 0.01445 | 0.15 | 0.13555 |
| 10 | 0.01564 | 0.19 | 0.17436 |
| 11 | 0.01538 | 0.17 | 0.15462 |
| 12 | 0.01513 | 0.11 | 0.09487 |
| 13 | 0.01488 | 0.09 | 0.07512 |
| 14 | 0.01464 | 0.08 | 0.06536 |
| 15 | 0.0144 | 0.02 | 0.0056 |
|  | 0.13655 | 1.0 | 0.17436 |

Таблица из задания 5

| | (D\|M\|n) | (M\|D\|n) | (M\|M\|n) |
|---|---|---|---|
| Число заявок $J(100)$, поступивших в СМО на интервале $[0, t_{\text{соб}}(100)]$ | 50 | 51 | 57 |
| Число $JF(100)$, полностью обслуженных заявок на интервале $[0, t_{\text{соб}}(100)]$ | 50 | 49 | 43 |
| Среднее число заявок, находившихся в СМО, на интервале$[0, t_{\text{соб}}(100)]$ | 1.08 | 1.72 | 10.2 |
| Среднее время пребывания заявок в очереди на интервале $[0, t_{\text{соб}}(100)]$ | 0 | 0 | 0.06537 |
| Среднее время пребывания заявок в СМО на интервале $[0, t_{\text{соб}}(100)]$ | 1.8009 | 0.18527 | 2.12693 |

# Список литературы


1. Гнеденко Б.В., Коваленко И.Н. Введение в теорию массового обслуживания. – М.: ЛКИ, 2021. – 400 с.
2. Кирпичников А.П. Методы прикладной теории массового обслуживания. – М.: URSS, 2018. – 224 с.
3. Ивченко Г.И., Каштанов В.А., Коваленко И.Н. Теория массового обслуживания. – М.: URSS, 2012. – 304 с.
4. Смирнов С.Н. Введение в прикладную теорию массового обслуживания. – М.: Гелиос АРВ, 2016. – 176 с.
5. Лобузов А.А., Гумляева С.Д., Норин Н.В. Задачи по теории случайных процессов. – М.: МИРЭА, 1993. – 68 с.
6. Алпатов Ю. Н. Моделирование процессов и систем управления. – СПб: Лань, 2021. - 140 с.
7. Самусевич Г. А. Моделирование процессов функционирования СМО. – М.: Издательство Юрайт, 2021. — 117 с.

# Приложение

main.py

```
from create_a_report import create_report


def main():
    create_report(72)


if __name__ == '__main__':
    main()
```

create_a_report.py

```
from docx import Document
from docx.shared import Inches

from Application import Application
from Event import Event
from ListWrapper import ListWrapper

from constants import set_constants as set_c
from solution import event_handler, get_data_for_an_calc
from get_data import get_conditions
from work_with_document import fill_table_for_report,
fill_table_analysis_of_calculations


def write_report_on_task(n_task: int, document, tables: list[[list[Application]
| list[Event]]], conditions: str):
    document.add_heading(f' Задание {n_task + 1}', 2)
    document.add_paragraph(conditions)
    dic = {
        1: 'Система массового обслуживания (D|M|n).',
        2: 'Система массового обслуживания (M|D|n).',
        3: 'Система массового обслуживания (M|M|n).'
    }
    document.add paragraph(dic[n task+1])
    widths = (Inches(0.4), Inches(1), Inches(0.3), Inches(0.3), Inches(1),
Inches(1), Inches(0.3))
    fill_table_for_report(document, tables[0], widths)
    document.add_paragraph(f'Таблица 2')
    widths = (Inches(0.4), Inches(1), Inches(0.3), Inches(0.3), Inches(1),
Inches(1), Inches(0.3))
    fill_table_for_report(document, tables[1], widths)
    print(f'Выполнил задачу № {n_task + 1}')


def get_data_for_report() -> tuple:
    """  Получает данные для заполнения таблиц 1, 2 для задач 1, 2, 3, 4  """

    smo1, tables_task_1 = event_handler(1)
    smo2, tables_task_2 = event_handler(2)
    smo3, tables_task_3 = event_handler(3)
    analytic_calc = get_data_for_an_calc([smo1, smo2, smo3])
    return tables_task_1, tables_task_2, tables_task_3, analytic_calc


def create_report(variant, path_to_cond='lab_3.txt', doc_name='Report.doc'):
    """Заполняет черновую версию в файл doc name"""
    data = get_conditions(variant, path_to_cond)
```

```
    print(data)
    variant = data.variant
    name = data.name
    set_c(*data.data)
    data_for_report = get_data_for_report()

    from constants import NUM_SMO, SERVICE_TIME, DELTA_T, LAMBD, MU

    conditions = f'Вариант №{variant}\n кол-во СМО = {NUM_SMO},T
об={SERVICE_TIME}, Тэ={DELTA_T} lambda={LAMBD}, mu = {MU}'
    print(conditions)

    document = Document()
    document.add_paragraph(name)
    document.add_paragraph(conditions)
    for i in range(3):
        write_report_on_task(i, document, data_for_report[i], conditions)
    analytic_calc = data_for_report[3]
    fill_table_analysis_of_calculations(document, analytic_calc)
    document.save(doc_name)
```

solution.py

```
"""

Содержит решение 1-3 задач
Вычисление таблиц аналитического раздела

"""
import math
from Controller_SMO import Controller_SMO


def event handler(n task: int):
    """Обработчик событий. Заполняет таблицы 1 и 2"""
    from constants import NUM_EVENTS, NUM_SMO

    f_name = f'table1_task{n_task}.txt'
    smo = Controller_SMO(NUM_SMO, n_task, f_name)
    result = smo.start_system(NUM_EVENTS)

    return smo, result


def get_table_with_device_data(smo):
    """ Собирает данными о приборах для таблицы 3"""
    t_device_data = smo.get_data_for_report()
    return t_device_data


def get_data_for_table_5(smo):
    return smo.get_column_for_table_5()


def get_frequency_table(smo):
    return smo.get_frequency_table()


def get_vector_r(length):
    from constants import MU, LAMBD, NUM_SMO
    vector = []
    p = LAMBD / MU
    v = p / NUM_SMO
    r0 = 0
```

```
    for k in range(NUM_SMO):
        r0 += p ** k / math.factorial(k)
    r0 = (r0 + (p ** NUM_SMO / math.factorial(NUM_SMO)) * (1 / (1 - v))) ** (-1)

    vector.append(r0)

    for k in range(1, NUM_SMO + 1):
        vector.append((r0 * p ** k) / math.factorial(k))

    for l in range(1, length - NUM_SMO):
        vector.append((v ** l) * vector[NUM_SMO])

    return vector


def get_frequency_table_task_3(vector_r, vector_v):
    table = []
    sum_r = 0
    sum_v = 0
    max_value = 0
    for i in range(len(vector_v)):
        sum_r += vector_r[i]
        sum_v += vector_v[i]
        value = abs(vector_v[i] - vector_r[i])
        if value > max_value:
            max_value = value
        table.append([i, vector_r[i], vector_v[i], value])
    table.append(['', sum_r, sum_v, max_value])
    return table


def get_data_for_an_calc(smo_list):
    """
    Формирует данные для отчета в аналитическом разделе

    В отчете 6 таблиц и 1 вектор r(список)

    """
    # с данными о приборах
    table_3 = []
    table_for_task_5 = []

    for i in range(3):
        table_3.append(get_table_with_device_data(smo_list[i]))
        table_for_task_5.append(get_data_for_table_5(smo_list[i]))

    frequency_tables = [get_frequency_table(smo) for smo in smo_list]

    vector_r = get_vector_r(len(frequency_tables[2]))

    frequency_table_task_3 = get_frequency_table_task_3(vector_r,
frequency_tables[2])

    return [table_3, vector_r, table_for_task_5, frequency_tables[:2],
frequency_table_task_3]
```

Application.py

```
class Application:
    """

    Класс для представления Заявки, поступившей в СМО

    Атрибуты
```

```
    --------
    ...
    Методы
    ------
    __len__():
        Возвращает количество атрибутов в классе
    get_data_for_report(self):
        Возвращает список из всех атрибутов класса

    """

    def __init__(self, number=0, application_time=0, place_in_queue=0,
staying_in_queue=-1, start_service=-1,
                 service_time=-1, end_time=-1):
        """

        :param number: номер заявки (default 0)
        :param application_time: момент появления заявки (default 0)
        :param place in queue: номер места в очереди (default 0)
        :param staying_in_queue: время пребывания заявки в очереди (default -1)
        :param start_service: момент начала обслуживания заявки (default -1)
        :param service_time: время обслуживания (default -1)
        :param end_time: момент окончания обслуживания заявки (default -1)

        """
        self.number = number
        self.app_time = application_time
        self.place_in_queue = place_in_queue
        self.stay_in_queue = staying_in_queue
        self.start_service = start_service
        self.service_time = service_time
        self.end_time = end_time
        self._data_len = 7

    def __len__(self):
        """Возвращает количество атрибутов в классе."""
        return self._data_len

    def get_data_for_report(self):
        """Возвращает список из всех атрибутов класса."""
        return [self.number, self.app_time, self.place_in_queue,
self.stay_in_queue, self.start_service,
                self.service_time, self.end_time]
```

Controller_SMO.py

```
"""
    Обрабатывает полученные заявки
        - следит за очередью заявок
        - хранит наименьшее время, оставшееся до завершения заявки прибором
    - поручает приборам выполнение заявок
    - записывает результаты в таблицу 1, 2
    - собирает данные для аналитической части
        - собирает частоту пребывания СМО в состояниях

"""
import numpy as np
import json
import os
from collections import deque, Counter

from Event import Event
from Application import Application
```

```
from Device import Device
from DeviceData import DeviceData


class Controller_SMO:
    """

       Класс для управления приборами из системы массового обслуживания (СМО)

        ...

       Атрибуты
       --------
        num_devices : int (default 1)
                Количество приборов, находящихся в подчинении
        type_system : int (default 1)
                Тип системы
                1 : Система массового обслуживания (D|M|n)
                2 : Система массового обслуживания (M|D|n)
                3 : Система массового обслуживания (M|M|n)


        device_id_completing_app : ind (default -1)
            Номер прибора, который быстрее всех заканчивает обслуживание заявки
        number_app_now : int (default 0)
                Количество заявок, находящихся в СМО в данный момент
        current_event_number : int (default 1)
                Номер события, которое обслуживается сейчас
        current_app_number : int (default 1)
                Номер заявки, которая обслуживается сейчас

        min_app_service_time : float (default 0.)
                Минимальное время обслуживания заявки
        time_arrival_next_app : float (default 0.)
                Время прихода следующей заявки
        event_start_time : float (default 0.)
                Время наступления события

        devices_list : list[Device]  (default [])
                Список подчиненных приборов
        event_table : list[Event] (default [])
                Таблица событий
        application_table : list[Application] (default [])
                Таблица заявок
        _app_list_need_to_complete : list[int] (default [])
                Номера заявок, которые надо завершить

        q :   deque (default deque())
                Очередь заявок

        selection : dict (default {})
                Выборка данный, для работы приборов. Нужна чтоб повторить
прошлый результат выполнения программы

        f_selection_to_file : bool (default True)
                Флаг, нужно ли записывать выборку в файл или нет.
                    True : Файл нужно записать
                    False : Файл с выборкой записан, записывать не надо

        f_name_with_selection : str (default' selection.txt')
                Имя файла в котором, либо содержится выборка, либо ее надо в
него записать\
       Методы
       ------
        ***
```

```
                Обрабатывает пришедшую заявку

        ***
                Собирает данные для отчета
        ****

       """
    selection = {}
    f_selection_to_file = True

    def __init__(self, num: int, type_: int, f_name: str):

        self.num_devices = num
        self.devices_list: list[Device] = [Device(number=i) for i in range(num)]
        self.type_system = type_
        self.selection = {}
        self._get_selection(f_name)
        self.f_name_with_selection = f_name
        self.event_table: list[Event] = []
        self.application_table: list[Application] = []
        self.q = deque()

        self.min_app_service_time = 0.
        self.number_app_now = 0
        self.time_arrival_next_app = 0.
        self.event_start_time = 0.
        self.current_event_number = 1
        self.current_app_number = 1
        self.device_id_completing_app = 0
        self._app_list_need_to_complete = []

    def __repr__(self):
        return f'{self.devices_list}'

    def service_first_app(self):
        """Пришла заявка, обрабатываем ее"""

        self.event_start_time = self.selection['1'][0]
        self.min_app_service_time = self.selection['1'][1]
        self.time_arrival_next_app = self.selection['1'][2]

        event = Event(self.current_event_number, self.event_start_time, 1, 1,
self.min_app_service_time, self.time_arrival_next_app, 1)

        device = self._search_free_device()
        device.give_task(self.current_app_number, self.min_app_service_time)

        end_time = event.event_time + event.time_until_end_service
        self.number_app_now = 1
        app = Application(self.number_app_now, event.event_time, 0, 0,
event.event_time, event.time_until_end_service, end_time)
        self._app_list_need_to_complete.append(self.current_app_number)
        self.event_table.append(event)
        self.application_table.append(app)

    def _get_selection(self, f_name: str):
        """
        Получаем начальные данные из файла или генерируем их

        Меняет значение self.f_selection_to_file

        """

        if os.path.exists(f_name):
            self.f_selection_to_file = False
```

```
            self.selection = _load_selection(f_name)
        else:
            self.f_selection_to_file = True
            self.selection = {'1':
[_get_time_between_applications(self.type_system),

_get_service_time_by_requests(self.type_system),

_get_time_between_applications(self.type_system)]}

    def start_system(self, num_event: int):
        """Запустить моделирование событий"""

        self.service_first_app()  # обработка 1-й заявки
        while self.current_event_number < num_event:
            self._define_event_type()

        _selection_to_file(self.selection, self.f_name_with_selection)
        return [self.event_table, self.application_table]

    def _define_event_type(self):
        """
        Определяет тип события
                1) СМО обрабатывает заявку
                    1.1) В СМО нет заявок, которые были обслужены
                        1.1.1) Поступившую заявку
                        1.1.2) Заявку из очереди
                    1.2) В СМО есть заявки, которые были обслужены и их нужно
завершить
                        1.2.1 Завершаем работу над заявкой
                2) СМО завершает работу над заявкой
                3) СМО добавляет поступившую заявку в очередь
        В функции происходит:

        - генерация событий;
        - заполнение таблицы event table
        - заполнение таблицы application table

        """
        device = self._search_free_device()
        self._update_min_app_service_time(0)

        if device:  # есть хоть 1 свободный прибор
            # если есть свободный прибор, то берем заявку из очереди
            # очередь пуста, принимаем заявку без очереди
            # В очереди есть заявки и скоро прейдет новая:
            #       - достаем заявку из очереди и даем ее на обслуживание
            #       - новую заявку отправляем в очередь

            if self.q:  # достаем заявку из очереди
                while device:  # раскидываем все заявки по свободным приборам
                    self._process_app_from_queue(device)
                    self._update_min_app_service_time(0)
                    device = self._search_free_device()
                if self.min_app_service_time > self.time_arrival_next_app:
                    self._add_app_to_queue()
                else:

self._completes_app_processing(self.devices_list[self.device_id_completing_app])
            else:  # очередь пуста
                if self.min_app_service_time == -1 or \
                        self.min_app_service_time > self.time_arrival_next_app:
# Все приборы свободны
                    self._process_current_app(device)
                else:
```

```
self._completes_app_processing(self.devices_list[self.device_id_completing_app])

        elif self.min_app_service_time < self.time_arrival_next_app:  # Заявка
завершится быстрее, чем придет новая

self._completes_app_processing(self.devices_list[self.device_id_completing_app])
        elif self.min_app_service_time > self.time_arrival_next_app:  # пришла
заявка, но все приборы заняты
            self._add_app_to_queue()
        else:
            raise Exception('Необработанный случай')

    def _search_free_device(self) -> Device | None:
        """ Опрос приборов о выполнении заявок """
        for device in self.devices_list:
            if device.is_free():
                return device

        return None

    def _ger_app_service_time_devises(self) -> dict[Device, float]:
        """Спрашивает у приборов время окончания обслуживания заявки"""
        result = {}
        for devise in self.devices_list:
            if not devise.is_free():  # прибор занят
                result[devise] = devise.get_time_until_end_service_app()

        return result

    def _update_min_app_service_time(self, time):
        """
        Опрашивает все занятые приборы и получает номер  прибора и минимальное
время до завершения обслуживания заявки

        Изменяет
            self.min_app_service_time
            self.device_id_completing_app
        """
        self.min_app_service_time = np.inf
        for devise in self.devices_list:
            if not devise.is_free():  # прибор работает над заявкой
                value = devise.update_time_until_end_service_app(time)
                if self.min_app_service_time > value > 0:  # 0 - прибор закончил
работу
                    self.min_app_service_time = value
                    self.device_id_completing_app = devise.get_number()
        if np.inf == self.min_app_service_time:  # все приборы свободны
            self.min_app_service_time = -1
            self.device_id_completing_app = -1

    def _processing_application(self, device):
        """ Обрабатывает полученную заявку или достает ее из очереди"""
        pass

    def _process_current_app(self, device: Device):
        """ Отдаем заявку на обслуживание прибору """
        self.current_event_number += 1
        self.current_app_number += 1
        self.number_app_now += 1

        self.event_start_time = self.event_start_time +
self.time_arrival_next_app
        if self._app_list_need_to_complete:
            self._update_min_app_service_time(self.time_arrival_next_app)
```

```
        else:
            self._update_min_app_service_time(0)

        if self.f_selection_to_file:
            time_until_end_service =
 get service time by requests(self.type system)
            self.time_arrival_next_app =
_get_time_between_applications(self.type_system)
            self.selection[str(self.current_event_number)] =
[time_until_end_service, self.time_arrival_next_app]
        else:
            time_until_end_service =
self.selection[str(self.current_event_number)][0]
            self.time_arrival_next_app =
self.selection[str(self.current_event_number)][1]
        self._app_list_need_to_complete.append(self.current_app_number)

        device.give_task(self.current_app_number, time_until_end_service)
        self._update_min_app_service_time(0)
        event = Event(number=self.current_event_number,
                      event_time=self.event_start_time,
                      event_type=1,
                      status_system=self.number_app_now,
                      time_until_end_service=self.min_app_service_time,
                      wait_time=self.time_arrival_next_app,
                      number_application=self.current_app_number,
                      )
        app = Application(number=self.current_app_number,
                          application_time=self.event_start_time,
                          place_in_queue=0,
                          staying_in_queue=0,
                          start_service=self.event_start_time,
                          service_time=time_until_end_service,
                          end_time=self.event_start_time +
time_until_end_service,
                          )
        self.event_table.append(event)
        self.application_table.append(app)

    def _completes_app_processing(self, device: Device):
        """Завершает обработку заявки"""
        self.current_event_number += 1
        self.number_app_now -= 1

        self.event_start_time = self.event_start_time +
self.min_app_service_time
        self.time_arrival_next_app = self.time_arrival_next_app -
self.min_app_service_time
        self._update_min_app_service_time(self.min_app_service_time)
        app_num = device.end_task()
        event = Event(number=self.current_event_number,
                      event_time=self.event_start_time,
                      event_type=2,
                      status_system=self.number_app_now,
                      time_until_end_service=self.min_app_service_time,
                      wait_time=self.time_arrival_next_app,
                      number_application=app_num,
                      )
        self.event_table.append(event)
        self._app_list_need_to_complete.remove(app_num)

    def _process_app_from_queue(self, device: Device):
        """Обрабатывает заявку из очереди"""
        num_app, num_event = self.q.popleft()
        if self.f_selection_to_file:
```

```
            service_time = self.selection[str(num_event)][0]
            self.selection[str(num_event)] = [service_time,
self.selection[str(num_event)][1]]
        else:
            service_time = self.selection[str(num_event)][0]

        device.give_task(num_app, service_time)
        self._app_list_need_to_complete.append(num_app)
        app = self.application_table[num_app - 1]
        app.start_service = self.event_start_time
        app.stay_in_queue = app.start_service - app.app_time
        app.service_time = service_time
        app.end_time = app.start_service + app.service_time

    def _add_app_to_queue(self):
        """Добавляем заявку в очередь"""
        self.current_event_number += 1
        self.current_app_number += 1
        self.number_app_now += 1

        self.event_start_time = self.event_start_time +
self.time_arrival_next_app
        self._update_min_app_service_time(self.time_arrival_next_app)
        if self.f_selection_to_file:
            service_time = _get_service_time_by_requests(self.type_system)
            self.time_arrival_next_app =
_get_time_between_applications(self.type_system)
            self.selection[str(self.current_event_number)] = [service_time,
self.time_arrival_next_app]
        else:
            self.time_arrival_next_app =
self.selection[str(self.current_event_number)][1]

        self.q.append((self.current_app_number, self.current_event_number))

        event = Event(number=self.current_event_number,
                      event_time=self.event_start_time,
                      event_type=1,
                      status_system=self.number_app_now,
                      time_until_end_service=self.min_app_service_time,
                      wait_time=self.time_arrival_next_app,
                      number_application=self.current_app_number,
                      )
        app = Application(number=self.current_app_number,
                          application_time=self.event_start_time,
                          place_in_queue=len(self.q),
                          )
        self.event_table.append(event)
        self.application_table.append(app)

    def get_frequency_table(self):

        table_1 = []
        reversed_table_1 = []
        for elem1 in self.event_table:
            table_1.append(elem1.get_data_for_report())
        for i, row in enumerate(zip(*table_1)):
            reversed_table_1.append(list(row))
        counter_states = Counter(sorted(reversed_table_1[3]))  # количество
входа в определенное состояние

        frequency_states = _get_frequency_states(counter_states)[:]

        return frequency_states
```

```
    def get_data_for_report(self):
        """Собирает с прибора данные, необходимые для отчета"""
        table: list[DeviceData] = []
        for device in self.devices_list:
            work_time = self.event_table[-1].event_time
            device.device data.calculate device downtime ratio(work time)
            table.append(device.device_data.get_data_for_report())
        return table

    def get_column_for_table_5(self):
        num_apps_received = 0  # Число поступивших на обслуживание заявок
        num_apps_served = 0  # Число обслуженных заявок
        for device in self.devices_list:
            num_apps_received += device.device_data.num_applications_received
            num_apps_served += device.device_data.num_applications_served
        num_apps_received += len(self.q)

        sum_column_status_system = 0.
        queue_time = 0.
        application_time_in_smp = 0.
        for event in self.event_table:
            sum_column_status_system += event.status_system
        for app in self.application_table:
            if app.stay_in_queue != -1:
                queue_time += app.stay_in_queue
                application_time_in_smp += app.service_time

        return [num_apps_received, num_apps_served,
                sum_column_status_system / 100,
                queue_time / num_apps_served,
                application_time_in_smp / num_apps_served,
                ]


def _get_frequency_states(counter_states: dict) -> list:
    """ Находит частоты состояний СМО"""

    frequency_states_1 = {}
    frequency_states = []  # .clean
    for state in counter_states:
        frequency_states_1[state] = counter_states[state] / 100
    try:
        frequency_states_1[0]
    except:
        frequency_states_1[0] = 0.0

    for i in range(len(frequency_states_1)):
        frequency_states.append(frequency_states_1[i])

    return frequency_states


def _get_time_between_applications(data):
    """получить время между заявками."""
    from constants import DELTA_T, LAMBD
    if data in (2, 3):
        return abs(np.random.exponential(1 / LAMBD))
    elif data == 1:
        return DELTA_T


def _get_service_time_by_requests(data):
    """получить время обслуживания заявками."""
    from constants import SERVICE_TIME, MU
    if data in (1, 3):
```

```
        return abs(np.random.exponential(1 / MU))
    elif data == 2:
        return SERVICE_TIME


def  load selection(f name, dir ='') -> dict[str: float]:
    """Загружает выборку из файла"""
    if os.path.exists(dir_ + f_name):
        with open(dir_+f_name, 'r') as file:
            return json.load(file)
    else:
        raise Exception(f'Не могу найти {dir_ + f_name}')


def _selection_to_file(data, f_name, dir_=''):
    """ Записывает выборку в файл"""
    if not os.path.exists(dir_ + f_name):
        with open(dir_ + f_name, 'w') as file:
            json.dump(data, file)
```

Device.py

```
from DeviceData import DeviceData


class Device:
    """

       Класс для представлений прибор из системы массового обслуживания (СМО)

        ...

       Атрибуты
       --------
        _number : int (default 0)
                Номер прибора
         num app : int (default 0)
                Номер обслуживаемой заявки
        _free : bool (default True)
                True : Прибор свободен
                False : Прибор занят
         time until end service app : float (default 0.)
                Время до окончания обслуживания заявки
        _num_serviced_applications : list (default [])
                Список заявок, которые были обслужены прибором

         busy time : float (default 0.)
                Время работы (занятости) прибора
        _idle_time : float (default 0.)
                Время бездействия (простоя) прибора



       Методы
       ------


       """

    def __init__(self, number=0):
        self._number = number
        self._num_app = 0
        self._free = True
        self._time_until_end_service_app = 0.0
```

```
        self._num_serviced_applications = []
        self.device_data = DeviceData(self._number)

    def __repr__(self):
        return f'device №{self._number} - {"free" if self._free else "busy"};' \
               f' app serviced through {self. time until end service app}
обслуживаю заявку №{self._num_app}'

    def is_free(self):
        return self._free

    def give_task(self, number_app, time):
        self._num_app = number_app
        self._time_until_end_service_app = time
        self._free = False

        self.device_data.num_applications_received += 1

    def end_task(self):
        self._num_serviced_applications.append(self._num_app)
        self.device_data.num_applications_served += 1
        tmp = self._num_app
        self._num_app = -1
        self._free = True
        self._time_until_end_service_app = 0
        return tmp

    def get_number(self):
        return self._number

    def get_number_app(self):
        return self._num_app

    def update_time_until_end_service_app(self, time: float) -> float:
        """Обновляет значение _time_until_end_service_app"""
        if self._time_until_end_service_app < time:
            raise Exception(f'У прибора №{self._number} получается отрицательное
время до конца заявки'
                            f' \n{self._time_until_end_service_app} - { time}')
        if self._time_until_end_service_app != 0 and
self._time_until_end_service_app != time:
            self._time_until_end_service_app = self._time_until_end_service_app
- time
            self.device_data.operating_time += time
        else:
            return 0

        return self._time_until_end_service_app

    def get_time_until_end_service_app(self) -> float:
        """Обновляет значение _time_until_end_service_app и возвращает его"""
        return self._time_until_end_service_app
```

Event.py

```
class Event:
    """

    Класс для представления события

    Атрибуты
    --------
    ...
    Методы
```

```
    ------
    __len__():
        Возвращает количество атрибутов в классе
    get_data_for_report(self):
        Возвращает список из всех атрибутов класса

    """

    def __init__(self, number=0, event_time=0., event_type=0, status_system=0,
time_until_end_service=0., wait_time=0.,
                 number_application=0):
        """

        :param number: номер события (default 0)
        :param event_time: момент наступления события (default 0)
        :param event_type: тип события (default 0)
        :param status_system: состояния СМО (default 0)
        :param time_until_end_service: оставшееся время обслуживания (default 0)
        :param wait_time: время ожидания новой заявки (default 0)
        :param number_application номер заявки, участвующей в событии (default
0)

        """
        self.num = number
        self.event_time = event_time
        self.event_type = event_type
        self.status_system = status_system
        self.time_until_end_service = time_until_end_service
        self.wait_time = wait_time
        self.num_application = number_application
        self._data_len = 7

    def __len__(self):
        """Возвращает количество атрибутов в классе."""
        return self._data_len

    def __repr__(self):
        return f'{self.num}, {self.event_time}, {self.event_type},
{self.status_system}, {self.time_until_end_service},' \
               f'{self.wait_time}, {self.num_application}'

    def get_data_for_report(self):
        """Возвращает список из всех атрибутов класса."""
        return [self.num, self.event_time, self.event_type, self.status_system,
self.time_until_end_service,
                self.wait_time, self.num_application]
```

ListWrapper.py

```
class ListWrapper:
    """Обертка вокруг списка"""

    def __init__(self, lst: list[float | int]):
        self.lst = lst

    def get_data_for_report(self) -> list[float | int]:
        """Возвращает список"""
        return self.lst

    def __len__(self):
        """Возвращает количество атрибутов в классе."""
        return len(self.lst)
```

```
    def __iter__(self):
        return self.lst.__iter__()
```

constants.py

```
NUM_FOR_ROUND = 5
NUM_EVENTS = 100

NUM_SMO = None  # количество доступных приборов
SERVICE_TIME = None  # время между приходом заявок
DELTA_T = None  # время между приходом заявок
LAMBD = None
MU = None  # параметр показательного распределения


def set_constants(num_smo, service_time, delta_t, lambd, mu):
    """Устанавливает значения констант."""
    global NUM_SMO, SERVICE_TIME, DELTA_T, LAMBD, MU
    NUM_SMO = num_smo
    SERVICE_TIME = service_time
    DELTA_T = delta_t
    LAMBD = lambd
    MU = mu
```