Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ

по Лабораторной работе №7
по курсу «Моделирование»
на тему: «Обслуживающий аппарат»

Студент	ИУ7-73Б (Группа)	(Подпись, дата)	Миронов Г. А. (И. О. Фамилия)
Преподаватель		(Подпись, дата)	Рудаков И. В. (И. О. Фамилия)

2022 г.

1 Задание

Провести моделирование системы, состоящей из генератора, блока памяти, и обслуживающего аппарата.

Генератор подает сообщения, распределенные по равномерному закону, они приходят в память и выбираются на обработку по закону из ЛР1. Количество заявок конечно и задано. Предусмотреть случай, возвращат обработанной заявки обратно в очередь.

Необходимо определить оптимальную длину очереди, при которой в системе не будет потерянных сообщений. Реализовать используя GPSS.

2 Теоретическая часть

2.1 Равномерное распределение

Говорят, что случайная величина X имеет равномерное распределение на отрезке $[a; b]$, если её функция плотности имеет вид

$$f(x) = \begin{cases} \frac{1}{b-a}, a \leq x \leq b \\ 0, \text{иначе} \end{cases} \tag{2.1}$$

Обозначается $X \sim R[a, b]$.

Соответствующая функция распределения:

$$F(x) = \begin{cases} 0, a < x \\ \frac{x-a}{b-a}, a \leq x \leq b \\ 1, x > b \end{cases} \tag{2.2}$$

2.2 Распределение Пуассона

Говорят, что случайная величина X распределена по закону Пуассона с параметром $\lambda > 0$, если она принимает значения $0, 1, 2, \ldots$ с вероятностями

$$P\{X = k\} = \frac{\lambda^k}{k!} * e^{-\lambda}, k \in 0, 1, 2, \ldots \tag{2.3}$$

Обозначается $X \sim \Pi(\lambda)$.

Функция плотности распределения имеет вид:

$$P\{x = k\} = \frac{\lambda^k}{k!} * e^{-\lambda}, k \in 0, 1, 2, \ldots \tag{2.4}$$

Тогда соответствующая функция распределения имеет вид:

$$F(x) = P\{X < x\}, X \sim \Pi(\lambda) \tag{2.5}$$

2.3 Формализация задачи

2.3.1 Δt модель

Данная модель заключается в последовательном анализе состояний всех блоков системы в момент времени $t + \Delta t$. Новое состояние определяется в соответствии с их алгоритмическим описанием с учетом действия случайных факторов. В результате этого анализа принимается решение о том, какие системные события должны имитироваться на данный момент времени.

Основной недостаток модели: значительные затраты и вероятность пропуска события при больших Δt.

3 Результат работы

3.1 Без повторов, 1000 заявок

Входные данные:

```
a, b = 1, 10

lambda = 4

total_tasks = 1000
repeat_percentage = 0.0
step = 0.01
```

Результаты моделирования:

```
REQUIRED_QUEUE_LEN        4.000
```

3.2 10% повторов, 1000 заявок

Входные данные:

```
a, b = 1, 10

lambda = 4

total_tasks = 1000
repeat_percentage = 0.1
step = 0.01
```

Результаты моделирования:

```
REQUIRED_QUEUE_LEN        6.000
```

3.3 25% повторов, 1000 заявок

Входные данные:

```
a, b = 1, 10

lambda = 4

total_tasks = 1000
repeat_percentage = 0.25
step = 0.01
```

Результаты моделирования:

```
REQUIRED_QUEUE_LEN        15.000
```

3.4 25% повторов, 10000 заявок

Входные данные:

```
a, b = 1, 10

lambda = 4

total_tasks = 10000
repeat_percentage = 0.25
step = 0.01
```

Результаты моделирования:

```
REQUIRED_QUEUE_LEN        52.000
```

3.5 50% повторов, 10000 заявок

Входные данные:

```
a, b = 1, 10

lambda = 4

total_tasks = 10000
repeat_percentage = 0.5
step = 0.01
```

Результаты моделирования:

```
REQUIRED_QUEUE_LEN        4669.000
```

3.6 Итоговая таблица сравения

Полученные в ходе эксперимента данные, представлены в таблице 3.1.

Таблица 3.1 – Сводные данные по проведенным моделированиям системы и их параметрам

Кол-во сообщений	**Процент повторов**	Шаг Δt	**Равномерное**		**Пуассона**	**Размер очереди**
			a	**b**	λ	Δt
1000	0	0.1	1	10	4	4
1000	10	0.1	1	10	4	6
1000	25	0.1	1	10	4	15
10000	25	0.1	1	10	4	52
10000	50	0.1	1	10	4	4669

4 Исходный код программы

Листинг 4.1 – Исходный код программы

```
        SIMULATE
        GENERATE    (UNIFORM(1,1,10))

M_MEM   QUEUE       qMemory
        SEIZE       Processor
        DEPART      qMemory

        ADVANCE     (POISSON(2,4))
        RELEASE     Processor
        TRANSFER    .10,M_END,M_MEM

M_END   SAVEVALUE   REQUIRED_QUEUE_LEN,QM$qMemory

        TERMINATE   1
        START       1000
```