|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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**

*к лабораторной работе №4*

*По курсу: «Моделирование»*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ***ИУ7И-76Б*** |  |  | **Нгуен Ф. С.** |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | |  | | --- | | **Рудаков И.В.** | |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

*Москва, 2021 г.*

Оглавление

[**I.** **Теоретическая часть** 3](#_Toc91201925)

[Равномерное распределение: 3](#_Toc91201926)

[Нормальное распределение: 3](#_Toc91201927)

[Принцип ∆𝒕 . 4](#_Toc91201928)

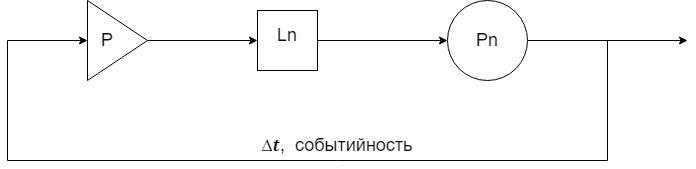
[Событийный принцип. 4](#_Toc91201929)

[**II.** **Экспериментальная часть** 6](#_Toc91201930)

[Входные данные: 6](#_Toc91201931)

[Выходные данные: 6](#_Toc91201932)

1. Теоретическая часть

Смоделировать систему, состоящую из генератора, очереди и ОА.

## Равномерное распределение:

𝑋 ~ 𝑅(𝑎, 𝑏), где 𝑎, 𝑏 ∈ R.

Функция распределения равномерной непрерывной случайной величины:

|  |  |
| --- | --- |
|  | (1) |

Плотность распределения равномерной непрерывной случайной величины:

|  |  |
| --- | --- |
|  | (2) |

## Нормальное распределение:

𝑋 ~ 𝑁(𝜇, 𝜎2), μ — математическое ожидание, σ2 - дисперсия

Плотность распределения равномерной непрерывной случайной величины:

|  |  |
| --- | --- |
|  | (3) |

Функция распределения:

|  |  |
| --- | --- |
|  | (4) |

## Принцип ∆𝒕 .

**Принцип** ∆𝒕 заключается в последовательном анализе состояний всех блоков в момент t + ∆𝑡 по заданному состоянию блоков в момент t. При этом новое состояние блоков определяется в соответствии с их алгоритмическим описанием с учетом действующих случайных факторов, задаваемых  
распределениями вероятности. В результате такого анализа принимается решение о том, какие общесистемные события должны имитироваться программной моделью на данный момент времени.

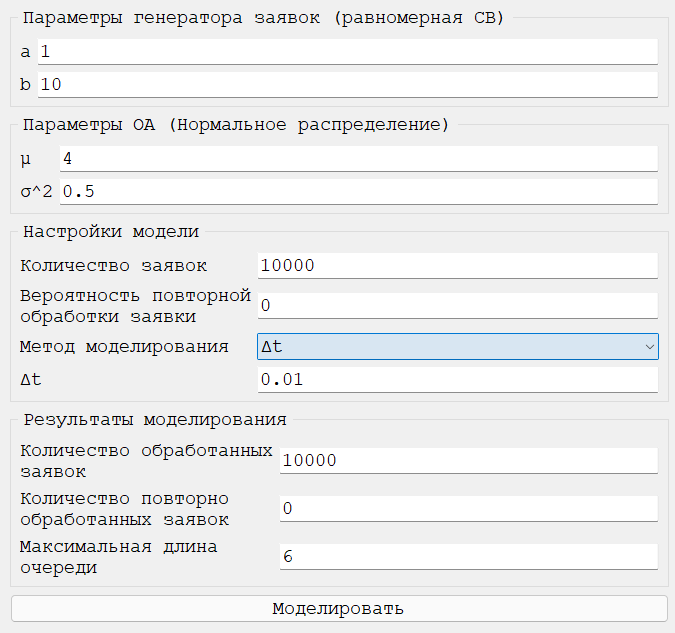
Основной недостаток этого принципа: значительные затраты машинного времени на реализацию моделирования системы. А при недостаточно малом ∆𝑡 появляется опасность пропуска отдельных событий в системе, что исключает возможность получения адекватных результатов при моделировании.

Достоинство: равномерная протяжка времени.

## Событийный принцип.

Характерное свойство систем обработки информации то, что состояние отдельных устройств изменяются в дискретные моменты времени, совпадающие с моментами времени поступления сообщений в систему, временем поступления окончания задачи, времени поступления аварийных  
сигналов и т.д. Поэтому моделирование и продвижение времени в системе удобно проводить, используя **событийный принцип**, при котором состояние всех блоков имитационной модели анализируется лишь в момент появления какого-либо события. Момент поступления следующего события определяется минимальным значением из списка будущих событий, представляющего собой совокупность моментов ближайшего изменения состояния каждого из блоков системы

1. Экспериментальная часть



Входные данные:

|  |  |  |  |
| --- | --- | --- | --- |
| Количество заявок = 10000 | | | |
| 𝑅(𝑎, 𝑏) | | 𝑁(𝜇, 𝜎2) | |
| a | b | 𝜇 | 𝜎2 |
| 1 | 10 | 4 | 0.5 |

## Выходные данные:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | ∆𝒕 = 0.01 | | **Событийный** | |
| Вероятность повтор. oбраб. заявки | Количество повторно обраб. заявок | Максимальная длина очереди | Количество повторно обраб. заявок | Максимальная длина очереди |
| 0% | 0 | 6 | 0 | 5 |
| 10% | 991 | 11 | 1020 | 9 |
| 20% | 2004 | 14 | 1994 | 14 |
| 50% | 4977 | 2181 | 4951 | 2218 |
| 100% | 10000 | 7278 | 10000 | 7261 |

1. **Код программы:**

**class** **Modeller:**

**def** \_\_init\_\_**(**self**,** uniform\_a**,** uniform\_b**,** normal\_mu**,** normal\_sigma**,** reenter\_prop**):**

self**.**\_generator **=** RequestGenerator**(**UniformGenerator**(**uniform\_a**,** uniform\_b**))**

self**.**\_processor **=** RequestProcessor**(**ErlangGenerator**(**normal\_mu**,** normal\_sigma**),** reenter\_prop**)**

self**.**\_generator**.**add\_receiver**(**self**.**\_processor**)**

**def** event\_based\_modelling**(**self**,** request\_count**):**

generator **=** self**.**\_generator

processor **=** self**.**\_processor

gen\_period **=** generator**.**next\_time\_period**()**

proc\_period **=** gen\_period **+** processor**.**next\_time\_period**()**

**while** processor**.**processed\_requests **<** request\_count**:**

**if** gen\_period **<=** proc\_period**:**

generator**.**emit\_request**()**

gen\_period **+=** generator**.**next\_time\_period**()**

**else:**

processor**.**process**()**

**if** processor**.**current\_queue\_size **>** 0**:**

proc\_period **+=** processor**.**next\_time\_period**()**

**else:**

proc\_period **=** gen\_period **+** processor**.**next\_time\_period**()**

**return** **(**processor**.**processed\_requests**,** processor**.**reentered\_requests**,**

processor**.**max\_queue\_size**,** proc\_period**)**

**def** time\_based\_modelling**(**self**,** request\_count**,** dt**):**

generator **=** self**.**\_generator

processor **=** self**.**\_processor

gen\_period **=** generator**.**next\_time\_period**()**

proc\_period **=** gen\_period **+** processor**.**next\_time\_period**()**

current\_time **=** 0

**while** processor**.**processed\_requests **<** request\_count**:**

**if** gen\_period **<=** current\_time**:**

generator**.**emit\_request**()**

gen\_period **+=** generator**.**next\_time\_period**()**

**if** current\_time **>=** proc\_period**:**

processor**.**process**()**

**if** processor**.**current\_queue\_size **>** 0**:**

proc\_period **+=** processor**.**next\_time\_period**()**

**else:**

proc\_period **=** gen\_period **+** processor**.**next\_time\_period**()**

current\_time **+=** dt

**return** processor**.**processed\_requests**,** processor**.**reentered\_requests**,** processor**.**max\_queue\_size**,** current\_time