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 (ИУ)»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии (ИУ7)»

ОТЧЕТ

по лабораторной работе № 6

по курсу «Моделирование»

на тему: «Моделирование работы приемной комиссии»

Студент ИУ7-73Б _____ Р. Р. Хамзина
(Группа) (Подпись, дата) (И. О. Фамилия)

Преподаватель _____ И. В. Рудаков
(Подпись, дата) (И. О. Фамилия)

2022 г.

СОДЕРЖАНИЕ

1 Задание

Реализовать программу с графическим интерфейсом для моделирования процесса обработки 400 заявок абитуриентов приемной комиссией факультета ИУ МГТУ им. Н. Э. Баумана и определения числа абитуриентов, подавших документы на кафедры ИУ1-ИУ4, ИУ5-ИУ8 и ИУ9-ИУ12. Приемная комиссия работает следующим образом:

1. Абитуриенты приходят через интервал времени, равный 10 ± 5 мин.

2. Абитуриент заполняет анкету у одного из двух сотрудников. Сотрудники имеют разную производительность и могут обеспечивать обработку анкеты абитуриента за 25 ± 10 мин. и 15 ± 15 мин. соответственно. Абитуриенты стремятся занять свободного сотрудника с максимальной производительностью. Если оба сотрудника заняты, абитуриент подает документы в другой день.

3. Абитуриент попадает в одну из трех очередей в зависимости от кафедры, студентом которой хочет стать абитуриент. С вероятностью 0.2 абитуриент выбирает одну из кафедр ИУ1-ИУ4, с вероятностью 0.5 — одну из кафедр ИУ5-ИУ8, с вероятностью 0.3 — одну из кафедр ИУ9-ИУ12.

4. Три сотрудника принимают документы у абитуриентов из очередей следующим образом: первый сотрудник обслуживает абитуриентов из первой очереди за 15 ± 2 мин., второй сотрудник — из второй очереди за 25 ± 10 мин., третий сотрудник — из третьей очереди за 20 ± 5 мин.

1.1 Схемы модели

На рисунке 1.1 показана структурная схема модели, а на рисунке 1.2 представлена схема модели в терминах СМО.

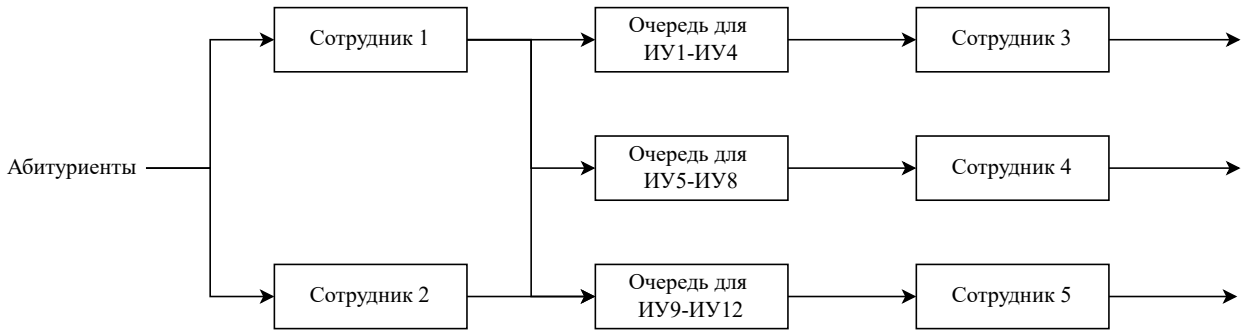

Рисунок 1.1 – Структурная схема модели

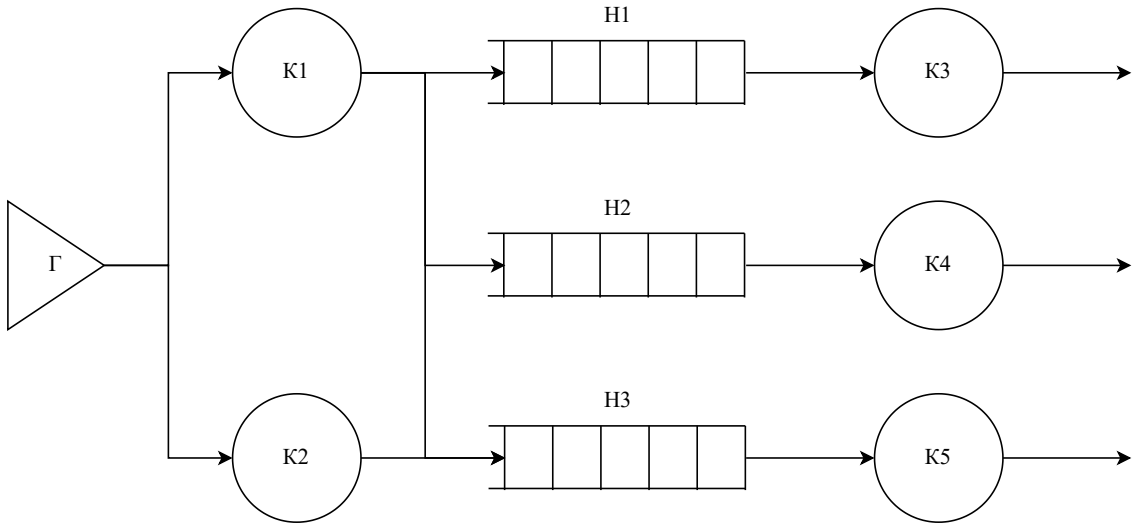

Рисунок 1.2 – Схема модели в терминах СМО

2 Реализация

2.1 Детали реализации

На листинге 2.1 показана реализация класса генератора абитуриентов.

Листинг 2.1 – Моделирование работы генератора абитуриентов

```
class EnrolleeGenerator:
    def __init__(self, time_value, time_limit, handlers, number):
        self.time_generator =
             TimeGenerator(time_value - time_limit,
                           time_value + time_limit)
        self.handlers = self.__sort_handlers(handlers)
        self.time_next = 0
        self.number = number

    def generate_enrollee(self, time_prev):
        self.time_next = time_prev + \
            self.time_generator.get_time_interval()

    def choose_handler(self):
        for handler in self.handlers:
            if handler.is_free():
                return handler

        return None

    def __sort_handlers(self, handlers):
        return sorted(handlers, key= lambda handler:
           handler.max_time)

def choose_department_type(probabilities):
    num = random()
    value = 0

    for type, probability in enumerate(probabilities):
        value += probability

        if value > num:
            return type
```

На листинге 2.2 представлена реализация класса сотрудника, обрабатывающего анкеты.

Листинг 2.2 – Моделирование работы сотрудника, обрабатывающего анкеты

```
class Handler:
    def __init__(self, time_value, time_limit):
        self.time_generator =
             TimeGenerator(time_value - time_limit,
                           time_value + time_limit)
        self.max_time = time_value + time_limit
        self.time_next = 0
        self.free = True

    def generate_time(self, prev_time):
        self.time_next = prev_time + \
        self.time_generator.get_time_interval()

    def is_free(self):
        return self.free

    def set_free(self):
        self.free = True

    def set_busy(self):
        self.free = False
```

На листинге 2.3 показана реализация класса сотрудника, принимающего документы.

Листинг 2.3 – Моделирование работы сотрудника, принимающего документы

```
class Receiver:
    def __init__(self, time_value, time_limit):
        self.time_generator =
             TimeGenerator(time_value - time_limit,
                           time_value + time_limit)
        self.queue = []
        self.time_next = 0
        self.free = True

    def generate_time(self, prev_time):
        self.time_next = prev_time + \
        self.time_generator.get_time_interval()

```

```
14      def is_free(self):
15          return self.free
16
17      def set_free(self):
18          self.free = True
19
20      def set_busy(self):
21          self.free = False
22
23      def queue_empty(self):
24          if self.queue:
25              return False
26          return True
27
28      def append_enrollee(self, department_type):
29          self.queue.append(department_type)
30
31      def pop_enrollee(self):
32          return self.queue.pop(0)
```

На листинге 2.4 представлена реализации класса приемной комиссии и функции выбора кафедры абитуриентом.

Листинг 2.4 – Моделирование работы приемной комиссии и выбора кафедры абитуриентом

```
1  class Commission:
2      def __init__(self, enrollee_generator, receivers):
3          self.enrollee_generator = enrollee_generator
4          self.receivers = receivers
5
6      def service_enrollees(self, probabilities):
7          received_documents = [0] * 3
8
9          self.enrollee_generator.generate_enrollee(0)
10         generated_enrollees = 1
11
12         events = [Event(self.enrollee_generator,
13                         self.enrollee_generator.time_next)]
14
15         while generated_enrollees <
              self.enrollee_generator.number:
16             events = sort_events(events)
```

```
            event = events.pop(0)

            if isinstance(event.creator, EnrolleeGenerator):
                handler =
                   self.enrollee_generator.choose_handler()

                if handler is not None:
                    handler.set_busy()
                    handler.generate_time(event.time)
                    events.append(Event(handler,
                       handler.time_next))

                self.enrollee_generator.generate_enrollee
                                (event.time)
                generated_enrollees += 1
                events.append(Event(self.enrollee_generator,
                              self.enrollee_generator.time_next))
            elif isinstance(event.creator, Handler):
                handler = event.creator
                handler.set_free()

                type = choose_department_type(probabilities)
                receiver = self.receivers[type]
                receiver.append_enrollee(type + 1)

                if receiver.is_free() and not
                   receiver.queue_empty():
                    department_type = receiver.pop_enrollee()
                    receiver.set_busy()
                    receiver.generate_time(event.time)
                    events.append(Event(receiver,
                       receiver.time_next))
                    received_documents[department_type - 1] += 1
            elif isinstance(event.creator, Receiver):
                receiver = event.creator
                receiver.set_free()

                if not receiver.queue_empty():
                    department_type = receiver.pop_enrollee()
                    receiver.set_busy()
                    receiver.generate_time(event.time)
```

```
54                        events.append(Event(receiver,
                              receiver.time_next))
55                        received_documents[department_type - 1] += 1
56
57            return received_documents
```

2.2 Полученный результат

На рисунке 2.1 показана страница программы для определения числа абитуриентов, подавших документы на кафедры ИУ1-ИУ4, ИУ5-ИУ8 и ИУ9-ИУ12, при моделировании работы приемной комиссии с заданными в условии параметрами.

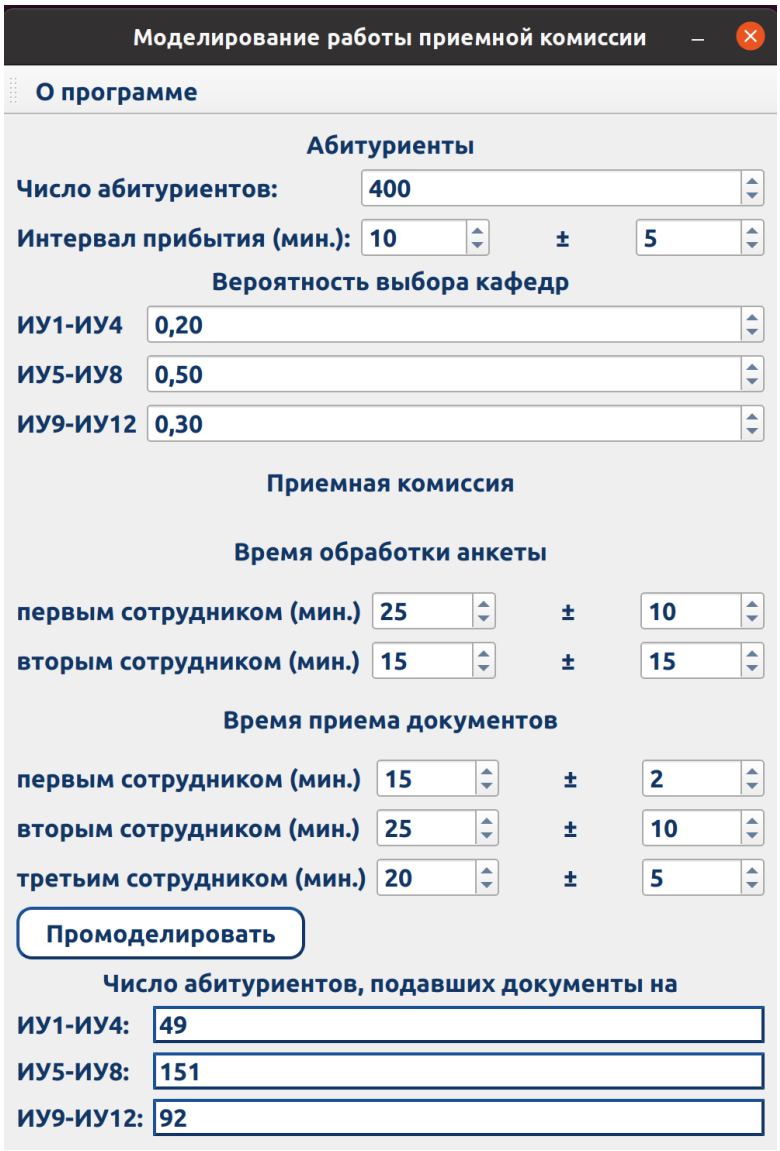

Рисунок 2.1 – Моделирование работы приемной комиссии