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 ИУ «Информатика и системы управления»

КАФЕДРА ИУ-7 «Программное обеспечение эвм и информационные технологии»

ОТЧЕТ

По лабораторной работе №6

«Моделирование СМО»

По курсу «Моделирование»

Студент Т. А. Казаева

Группа ИУ7-76Б

Преподаватель И. В. Рудаков

2022 г.

1. Задание

На некоторой неизвестной лучшей кафедре в в некотором неизвестном лучшем техническом вузе преподаватель принимает лабораторные работы. Студентам осталось сдать ему три лабораторные работы:

- лабораторная работа «*ассиметричное шифрование*» (под номером 4) сдается 6 ± 2 мин.;
- лабораторная работа «*электронная подпись*» (под номером 5) сдается 4 ± 1 мин.;
- лабораторная работа «*сжатие*» (под номером 6) сдается 10 ± 5 мин.

Лабораторные работы сдаются по порядку. Если студент плохо подготовился, то он не может продолжить сдавать следующую лабораторную работу, даже если она у студента готова.

Возможности студентов в день сдачи представлены в таблице 1.1.

Таблица 1.1: Возможности студентов в день сдачи

№	время, мин.	описание
1	6±2	сдать 4-ю ЛР
2	4±1	сдать 5-ю ЛР
3	10±5	сдать 6-ю ЛР
4	10±3	сдать 4-ю и 5-ю ЛР
5	14±6	сдать 5-ю и 6-ю ЛР
6	20±8	закрыть курс

Преподаватель принимает принимает лабораторные 4,25 часа. Поток студентов у преподавателя постоянный.

Осталось 3 сдачи до Нового Года. Сколько студентов закроют лабораторные работы по курсу до Нового Года, а сколько студентов будут плакать горькими слезами?

2. Концептуальная модель

Система состоит из двух блоков:

- **блок имитатора воздействия внешней среды**:
  Система имеет шесть генераторов (по одному на каждую возможность студента). Чтобы поток студентов был постоянным, заявки генерируются с частотой равной минимальному времени сдачи лабораторной работы.

- **блок функций системы:**
  Система имеет шесть функционирующих блоков – преподаватель принимает студентов согласно возможностям в таблице 1.1. Каждая сдача в среднем занимает занимает преподавателя на время, приведенное выше в таблице.

Преподаватель принимает в день сдачи 4 часа 15 минут, то есть $4,5 \cdot 60 = 255$ минут. Необходимо смоделировать 3 дня сдачи – 3 периода от 0 до 255 единиц времени.

3. Результат

Результат выполнения программы:
Студентов принято: 69.

- студенты, закрывшие предмет: 13;
- студенты, сдавшие одну лабораторную: 38;
- студенты, сдавшие две лабораторные: 13.

4. Программный код

Для реализации программы был выбран язык Python.

На листинге 4.1 представлена генерация числа из заданного интервала.

Листинг 4.1: Генерация числа из заданного интервала

```
class UniformDistribution:
    def __init__(self, base, err):
        self.lo = base - err
        self.hi = base + err

    def getTime(self):
        return random.randint(self.lo, self.hi)
```

На листинге 4.2 представлена реализация класса «Преподаватель».

Листинг 4.2: Реализация класса «Преподаватель»

```
class Professor:
    def __init__(self, bases, errs):
        self.ops = [UniformDistribution(bases[0], errs[0]),
                    UniformDistribution(bases[1], errs[1]),
                    UniformDistribution(bases[2], errs[2]),
                    UniformDistribution(bases[3], errs[3]),
                    UniformDistribution(bases[4], errs[4]),
                    UniformDistribution(bases[5], errs[5])]
```

На листинге 4.3 представлена реализация класса «Студенты».

Листинг 4.3: Реализация класса «Студенты»

```
class Students:
    def __init__(self, bases, errs):
        self.students = [UniformDistribution(bases[0], errs[0]),
                         UniformDistribution(bases[1], errs[1]),
                         UniformDistribution(bases[2], errs[2]),
                         UniformDistribution(bases[3], errs[3]),
                         UniformDistribution(bases[4], errs[4]),
                         UniformDistribution(bases[5], errs[5])]
```

На листинге 4.4 представлена реализация СМО.

Листинг 4.4: Реализация СМО

```
class QueueNetwork:
    def __init__(self):
        self.startTime = 0
        self.endTime = 255
        self.opps = [0, 0, 0, 0, 0, 0]
        self.jobs = list()
        self.professor = Professor([6, 4, 10, 10, 14, 20], [2, 1,
           5, 3, 6, 8])
        self.students = Students([4, 3, 5, 7, 8, 12], [0, 0, 0, 0,
           0, 0])

    def NextRound(self):
        self.startTime = 0
        self.endTime = 255
        self.jobs = list()

    def GetDayInfo(self):
        print(self.startTime)
        print(self.endTime)
        print(self.days)
        print(self.opps)
        print(self.jobs)

    def generateStudents(self):
        for i in range(6):
            t = 0
            while t < self.endTime:
                t += self.students.students[i].getTime()
                self.addJob([t, i])

    def modeling(self):
        self.generateStudents()
        i = 0
        while self.startTime < self.endTime and i < len(self.jobs):
            while self.startTime > self.jobs[i][0]:
```

```
34                  i += 1
35              self.startTime = self.jobs[i][0]
36              self.startTime +=
                   self.professor.ops[self.jobs[i][1]].getTime()
37              if (self.startTime < self.endTime):
38                  self.opps[self.jobs[i][1]] += 1
39              i += 1
40
41      def addJob(self, event: list):
42          i = 0
43          while i < len(self.jobs) and self.jobs[i][0] < event[0]:
44              i += 1
45          self.jobs.insert(i, event)
```