

1. Цель работы

Смоделировать систему массового обслуживания для заданной предметной области средствами объектно-ориентированного языка программирования.

2. Задание

I. Реализовать приложение с графическим интерфейсом, моделирующее описанную в варианте систему массового обслуживания на объектно-ориентированном языке программирования (при написании программы самостоятельно можно не реализовывать только генератор равномерно распределённых псевдослучайных чисел).

II. Протестировать правильность работы разработанного приложения.

III. С помощью реализованной модели ответить на поставленные в варианте вопросы о функционировании системы массового обслуживания и сделать вывод о том, как работает смоделированная система.

3. Вариант задания

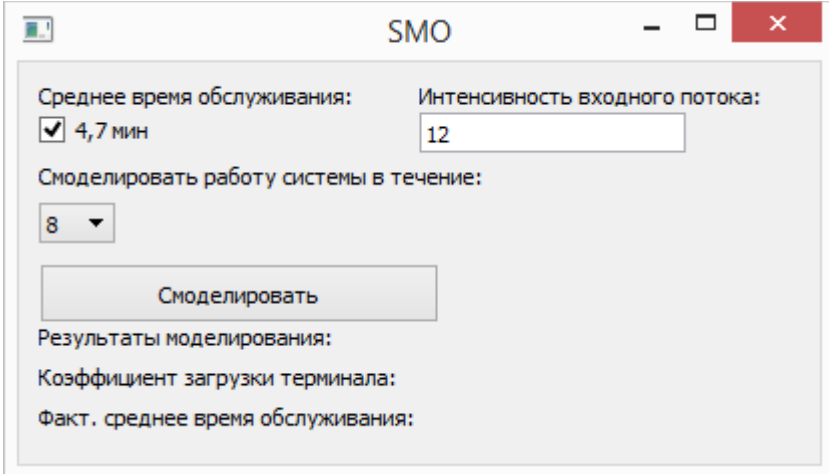
В справочной информационно-поисковой службе с одним терминалом имеет место пуассоновский входящий поток требований с интенсивностью 12 приходов в час. Обслуживание требований является экспоненциальным, но среднее время обслуживания зависит от числа требований, находящихся в очереди к терминалу. Эта зависимость представлена в таблице:

Длина очереди, чел	Среднее время обслуживания, мин
0	5.5
1 или 2	5
3, 4 или 5	4.5
6	4

Смоделировать работу системы в течение 1, 8, 40 часов. Найти фактическое среднее время обслуживания, коэффициент загрузки терминала. Повторить эти же действия при условии, что среднее время обслуживания всегда 4.7 мин. Сравнить результаты.

4. Описание разработанного программного средства

Программа разработана средствами Python + Qt5. Предназначена для моделирования систем массового обслуживания.



Интерфейс программного средства позволяет пользователю задавать режим работы (среднее время обслуживания – 4,7 мин либо в зависимости от очереди к терминалу), задавать интенсивность входного потока, а также выбрать время, в течение которого

моделируется работа системы. Результатом моделирования будет коэффициент загрузки терминала и фактическое среднее время обслуживания.

5. Текст программы

```
import sys
from PyQt5.QtWidgets import QWidget, QLabel, QApplication, QCheckBox, QComboBox,
QPushButton, QLineEdit
from PyQt5.QtCore import Qt
import queue as qq
import numpy as np
import scipy as sp

#время моделирования входного потока
def model_time_input(lamdb, Tmax):
    t = [0]
    alpha = [np.random.uniform(0,1) for i in range(1,lamdb * Tmax)]
    t.extend(- 1. / lamdb * np.log(alpha))
    time_input = list(filter(lambda x: x < Tmax, sp.cumsum(t)))
    mean_time_input = (np.sum(t) / len(t)) * 60.
    return time_input, mean_time_input

#время моделирования обслуживания
def model_time_service(q_size, flag):
    #
    if flag == True:
        t_ser = 4.7 / 60. #среднее время обслуживания
    else:
        if q_size == 0:
            t_ser = 5.5/ 60.
        elif q_size == 1 or q_size == 2:
            t_ser = 5./ 60.
        elif q_size == 3 or q_size == 4 or q_size == 5:
            t_ser = 4.5/ 60.
        elif q_size == 6:
            t_ser = 4./ 60.
    nu = 1. / t_ser
    alpha = np.random.uniform(0,1)
    time_service = - 1. / nu * np.log(alpha)
    return time_service

def SMO(lamdb, flag, Tmax):
    q_ser = False #обслуживается ли кто-то терминалом
    arr_time_ser = []
    qqe = qq.Queue(maxsize = 6) #очередь
    time_input, mean_time_input = model_time_input(lamdb, Tmax)
    #обслуживание клиентов
    tmp = 0.0
    count_ser = 0
    count_fail = 0
    for el in time_input:
        time_ser = model_time_service(qqe.qsize(), flag)
        arr_time_ser.append(time_ser)
        if q_ser == False and qqe.empty():
            q_ser =True
            tmp = el + time_ser
        else:
            if q_ser == False and qqe.qsize() !=0:
                el1 = qqe.get()
                q_ser =True
                tmp = tmp + time_ser
                qqe.put(el1)
            else:
                if el >= tmp:
                    q_ser = False
                    count_ser += 1
```

```

        if qqe.qsize() != 0:
            el1 = qqe.get()
            tmp = tmp + time_ser
            q_ser = True
            qqe.put(el)
        else:
            q_ser = True
            tmp = el + time_ser
    else:
        if q_ser == True and qqe.qsize() <= 5:
            qqe.put(el)
        else:
            count_fail += 1

mean_time_ser = (np.sum(arr_time_ser)/len(arr_time_ser))*60.
koef_load = mean_time_input / mean_time_ser
return koef_load, mean_time_ser

class Interface(QWidget):
    def __init__(self):
        super().__init__()
        self.Tmax = 1 #время, в течение которого моделируется работа системы
        self.flag = True
        self.lambd = 12 #интенсивность входного потока в час
        self.initUI()

    def initUI(self):
        self.lbl1 = QLabel("Среднее время обслуживания:", self)
        cb = QCheckBox('4,7 мин', self)
        cb.move(10, 25)
        self.lbl1.move(10, 10)
        cb.toggle()
        cb.stateChanged.connect(self.changeTitle)

        self.lbl11 = QLabel("Интенсивность входного потока:", self)
        self.lbl11.move(200, 10)
        self.le = QLineEdit("12",self)
        self.le.move(200, 25)
        self.le.textEdited[str].connect(self.onChanged)

        self.lbl2 = QLabel("Смоделировать работу системы в течение:", self)
        self.lbl2.move(10, 50)
        combo = QComboBox(self)
        combo.addItem("1")
        combo.addItem("8")
        combo.addItem("40")
        combo.move(10, 70)
        combo.activated[str].connect(self.onActivated)
        btn = QPushButton('Смоделировать', self)
        btn.move(10, 100)
        btn.resize(200,30)
        self.lbl6 = QLabel("Результаты моделирования:", self)
        self.lbl6.move(10, 130)
        self.lbl3 = QLabel("Коэффициент загрузки терминала:", self)
        self.ql1 = QLabel(" ",self)
        self.ql1.move(200, 150)
        self.ql1.resize(200,15)
        self.ql2 = QLabel(" ",self)
        self.ql2.move(200, 170)
        self.ql2.resize(200,15)
        self.lbl5 = QLabel("Факт. среднее время обслуживания:", self)
        self.lbl3.move(10, 150)
        self.lbl5.move(10, 170)
        # связывает событие нажатия на кнопку с методом
        btn.clicked.connect(self.buttonClicked)
        self.setGeometry(300, 300, 400, 200)
        self.setWindowTitle('SMO')
        self.show()

```

```

def onActivated(self, text):
    self.Tmax = int(text)

def changeTitle(self, state):

    if state == Qt.Checked:
        self.flag = True
    else:
        self.flag = False

def onChanged(self, text):
    if text == "":
        self.lambd = 12
        self.le.setText(str(self.lambd))
    else:
        self.lambd = int(text)
    self.setFocus()

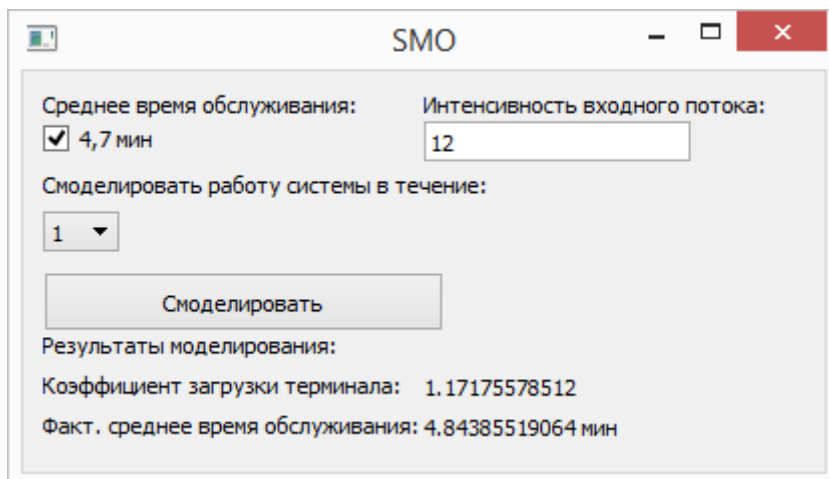
def buttonClicked(self):
    koef_load, mean_time_ser = SMO(self.lambd, self.flag, self.Tmax)
    self.qle1.setText(str(koef_load))
    self.qle2.setText(str(mean_time_ser) + ' мин')

if __name__ == '__main__':

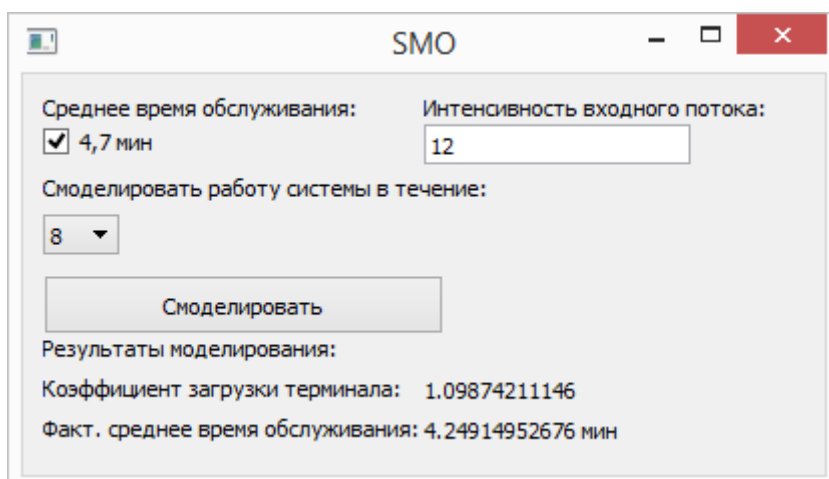
    app = QApplication(sys.argv)
    ex = Interface()
    sys.exit(app.exec_())

```

6. Набор тестов



The screenshot shows the SMO application window. The title bar says "SMO". Inside, there are two input fields at the top: "Среднее время обслуживания:" with a checked checkbox and "4,7 мин", and "Интенсивность входного потока:" with a text box containing "12". Below these is a label "Смоделировать работу системы в течение:" followed by a dropdown menu showing "1". A button labeled "Смоделировать" is below the dropdown. At the bottom, under "Результаты моделирования:", it displays "Коэффициент загрузки терминала: 1.17175578512" and "Факт. среднее время обслуживания: 4.84385519064 мин".



The screenshot shows the SMO application window with the same layout as the previous one. The "Смоделировать работу системы в течение:" dropdown menu now shows "8". The "Смоделировать" button is still present. The results at the bottom are: "Коэффициент загрузки терминала: 1.09874211146" and "Факт. среднее время обслуживания: 4.24914952676 мин".

SMO

Среднее время обслуживания: ☒ 4,7 мин Интенсивность входного потока:

Смоделировать работу системы в течение:

Результаты моделирования:
Коэффициент загрузки терминала: 1.03761219544
Факт. среднее время обслуживания: 5.09769445524 мин

SMO

Среднее время обслуживания: ☐ 4,7 мин Интенсивность входного потока:

Смоделировать работу системы в течение:

Результаты моделирования:
Коэффициент загрузки терминала: 0.73552104657
Факт. среднее время обслуживания: 5.4910611604 мин

SMO

Среднее время обслуживания: ☐ 4,7 мин Интенсивность входного потока:

Смоделировать работу системы в течение:

Результаты моделирования:
Коэффициент загрузки терминала: 0.922887982784
Факт. среднее время обслуживания: 4.53944432046 мин

SMO

Среднее время обслуживания: ☐ 4,7 мин Интенсивность входного потока:

Смоделировать работу системы в течение:

Результаты моделирования:
Коэффициент загрузки терминала: 1.36514301754
Факт. среднее время обслуживания: 3.96731052198 мин