Отчет по 3 модулю по проектно-технологической практике.

Горелкина. РК6-32Б. Вариант 4.

Задание: Требуется разработать программу, реализующую дискретно-событийное моделирование системы, рассмотренной в задании 2 домашнего задания №4. Обратите внимание, что все интервалы времени подчиняются законам распределений, носящим непрерывный характер. Поэтому категорически неверными является выбор целочисленных типов данных для моментов и интервалов времени, и тем более инкремент модельного времени с единичным шагом. Нужно реализовать именно переход от события к событию, как это сделано в GPSS и других проблемно-ориентированных системах. Для упрощения можно ограничиться использованием единственного потока случайных чисел для генерации всех необходимых случайных величин. Результатом работы программы должен быть лог-файл, содержащий записи типа: «В момент времени 12.345 транзакт с идентификатором 1 вошёл в модель», «В момент времени 123.456 транзакт с идентификатором 123 встал в очередь 1», «В момент времени 234.567 транзакт с идентификатором 234 занял устройство 2», «В момент времени 345.678 транзакт с идентификатором 345 освободил устройство 1», «В момент времени 456.789 транзакт с идентификатором 456 вышел из модели».

Импортируем модуль для генерации случайных чисел.

In [1]:

```
import random
```

Задаем исходные данные и время моделирования системы. А также создаем список, в котором будут храниться константы для передачу в функцию rand().

In [2]:

```
r1 = 11
g1 = 6
b1 = 8
n1 = r1 + g1 + b1
end_time = 300.0
ar = [0, r1, g1]
```

Функция rand() получает аргумент, который определяет выбор элемента списка для передачи в фунцию unoform() модуля random. Создадим два класса TransAct и Operator. В классе Transact будет три поля: следующее состояние, время перехода в следующее состояние и индекс. В классе Operator будет два поля: время обслуживания и логическая переменная, идентифицирующая занятость канала. Сеттер класса Operator задает время обслуживания.

In [3]:

```
def rand(i):
    return random.uniform(ar[i], n1)

class TransAct():

    def __init__(self, state, time):
        global n
        self.state = state
        self.time = time
        self.iden = n
        n += 1

class Operator():

    def __init__(self, flag, time):
        self.flag = True
        self.time = time

    def set_time(self, n):
        self.time = rand(n)
```

Определим функции выбора кассы или очереди, куда направится очередной транзакт. Функция choose() служит для направления заявки на обслуживания одному из операторов. Функция state_choose() является одной из фунций словаря state, благодаря которому выбирается очередная функия в зависимости от состояния транзакта. В ней реализованы три условия. Первые два вызывают функцию choose() в зависимости от того, какой оператор свободен. Третье условие определяет выбор очереди и направление в нее транзакта в случае, если обе кассы заняты.

In [4]:

```
def choose(trans, i):

    file.write('В момент времени ' + str(trans.time) + ' с транзакт ' +
                    str(trans.iden) + ' занял устройство ' + str(i) + '\n')

    trans.time += oper[i-1].time
    trans.state += i

    oper[i-1].flag = False

    queues[i-1].append(trans)

def state_choose(trans):

    if oper[0].flag: choose(trans, 1)

    elif oper[1].flag: choose(trans, 2)

    else:
        i = 0 if len(queues[0]) <= len(queues[1]) else 1

        file.write('В момент времени ' + str(trans.time) + ' с транзакт ' +
                   str(trans.iden) + ' направился в очередь ' + str(1 + i) + '\n')

        trans.state = 1 + i
        trans.time = queues[i][-1].time

        queues[i].append(trans)
```

Три следующие функции определяют подобие блоков из системы GPSS (GENERATE, OPERATE, TERMINATE). В функции state_generate() происходит обработка текущей заявки и резервация следующей. В функции state_operate() просиходит обработка заявки. В функции state_terminate() происходит выход заявки из устройства. Все эти три функции также являются функциями из словаря state.

In [5]:

```
def state_generate(trans):

    file.write('В момент времени ' + str(trans.time) +
               ' с транзакт ' + str(trans.iden) + ' вошёл в модель ' + '\n')

    trans.state += 1

    new_trans = TransAct(-1, rand(0) + trans.time)
    transacts.append(new_trans)

def state_operate(trans):

    if ((not oper[trans.state - 1].flag) and
        (queues[trans.state - 1][0].iden == trans.iden)):

        file.write('В момент времени ' + str(trans.time) + ' с транзакт ' +
                   str(trans.iden) + ' освободил устройство ' + str(trans.state) + '\n')

        oper[trans.state - 1].flag = True
        oper[trans.state - 1].set_time(trans.state)

        del queues[trans.state - 1][0]

        trans.state = 3

    elif ((oper[trans.state - 1].flag) and
          (queues[trans.state - 1][0].iden == trans.iden)):

        file.write('В момент времени ' + str(trans.time) + ' с транзакт ' +
                   str(trans.iden) + ' занял устройство ' + str(trans.state) + '\n')

        trans.time += oper[trans.state - 1].time

        oper[trans.state- 1].flag = False

def state_terminate(trans):

    file.write('В момент времени ' + str(trans.time) +
               ' с транзакт ' + str(trans.iden) + ' вышел из модели ' + '\n')

    del transacts[transacts.index(trans)]
```

Сам словарь state задает набор функций, переходы к которым осуществляются по ключу состояния транзакта, все переходы реализованы в цикле.

In [6]:

```
states = {
   -1 : state_generate,
    0 : state_choose,
    1 : state_operate,
    2 : state_operate,
    3 : state_terminate
}
```

Откроем файл на запись. Зададим начальные параметры моделирования системы. Переменная n служит счетчиком поступающих в систему транзактов. Создадим два объекта класса TransAct end и start, куда занесем исходные данные. В поле объекта end.time хранится врмея моделирования системы end_time. Добавим его в список транзактов, так как в дальнешем нам будет необходимо завершить цикл поступления заявок. В список операторов добавим объекты класса Operator. Список из двух очередей изначально пуст.

In [7]:

```
file = open('log_file.txt', 'w')

n = 0

end =  TransAct(-1, end_time)
start = TransAct(-1, rand(0))
transacts = [start, end]

oper = [Operator(0, rand(1)),
        Operator(0, rand(2))]

queues = [[], []]
```

Флаговая переменная позволяет выйти из цикла while при равенстве его True. В цикле находим транзакт с минимальным временем и вызываем функцию его состояния. Моделирование будем проводить в течение времени end_time.

In [8]:

```
end_flag = False

while(not end_flag):
    min_time = transacts[0].time
    for x in transacts:
        min_time = min_time if min_time < x.time else x.time
    if x.time > end_time:
        end_flag = True
    for x in transacts:
        if x.time == min_time:
            states[x.state](x)
```

Проверим содержимое файла log_file.txt.

In [9]:

```
file = open('log_file.txt', 'r')
for line in file.readlines():
    print (line)
```

```
В момент времени 10.400403000467948 с транзакт 1 вошёл в модель

В момент времени 10.400403000467948 с транзакт 1 занял устройство 1

В момент времени 13.292241049094304 с транзакт 2 вошёл в модель

В момент времени 13.292241049094304 с транзакт 2 занял устройство 2

В момент времени 19.083288760668548 с транзакт 3 вошёл в модель

В момент времени 19.083288760668548 с транзакт 3 направился в очередь 1

В момент времени 27.710087331941104 с транзакт 1 освободил устройство 1

В момент времени 27.710087331941104 с транзакт 3 занял устройство 1

В момент времени 27.710087331941104 с транзакт 1 вышел из модели

В момент времени 29.79485671030038 с транзакт 4 вошёл в модель

В момент времени 29.79485671030038 с транзакт 4 направился в очередь 1

В момент времени 31.85872394520123 с транзакт 5 вошёл в модель

В момент времени 31.85872394520123 с транзакт 5 направился в очередь 2

В момент времени 32.35207903252328 с транзакт 2 освободил устройство 2

В момент времени 32.35207903252328 с транзакт 5 занял устройство 2

В момент времени 32.35207903252328 с транзакт 2 вышел из модели

В момент времени 39.589705890265634 с транзакт 6 вошёл в модель

В момент времени 39.589705890265634 с транзакт 6 направился в очередь 2

В момент времени 42.2603405659814 с транзакт 5 освободил устройство 2

В момент времени 42.2603405659814 с транзакт 6 занял устройство 2

В момент времени 42.2603405659814 с транзакт 5 вышел из модели

В момент времени 43.979200206172294 с транзакт 3 освободил устройство 1

В момент времени 43.979200206172294 с транзакт 4 занял устройство 1

В момент времени 43.979200206172294 с транзакт 3 вышел из модели

В момент времени 63.231452877245694 с транзакт 7 вошёл в модель

В момент времени 63.231452877245694 с транзакт 7 направился в очередь 1

В момент времени 64.526972301845 с транзакт 4 освободил устройство 1

В момент времени 64.526972301845 с транзакт 7 занял устройство 1

В момент времени 64.526972301845 с транзакт 4 вышел из модели
```

В момент времени 65.25399567836519 с транзакт 6 освободил устройство 2

В момент времени 65.25399567836519 с транзакт 6 вышел из модели

В момент времени 77.3057004068395 с транзакт 7 освободил устройство 1

В момент времени 77.3057004068395 с транзакт 7 вышел из модели

В момент времени 83.96124249093822 с транзакт 8 вошёл в модель

В момент времени 83.96124249093822 с транзакт 8 занял устройство 1

В момент времени 100.76349204641974 с транзакт 9 вошёл в модель

В момент времени 100.76349204641974 с транзакт 9 занял устройство 2

В момент времени 100.90948543713841 с транзакт 8 освободил устройство 1

В момент времени 100.90948543713841 с транзакт 8 вышел из модели

В момент времени 116.1838199310754 с транзакт 10 вошёл в модель

В момент времени 116.1838199310754 с транзакт 10 занял устройство 1

В момент времени 116.48176936658254 с транзакт 9 освободил устройство 2

В момент времени 116.48176936658254 с транзакт 9 вышел из модели

В момент времени 127.3616812584377 с транзакт 10 освободил устройство 1

В момент времени 127.3616812584377 с транзакт 10 вышел из модели

В момент времени 140.59774442496146 с транзакт 11 вошёл в модель

В момент времени 140.59774442496146 с транзакт 11 занял устройство 1

В момент времени 150.7264313227639 с транзакт 12 вошёл в модель

В момент времени 150.7264313227639 с транзакт 12 занял устройство 2

В момент времени 160.95213952608535 с транзакт 13 вошёл в модель

В момент времени 160.95213952608535 с транзакт 13 направился в очередь 1

В момент времени 164.16504237163764 с транзакт 11 освободил устройство 1

В момент времени 164.16504237163764 с транзакт 13 занял устройство 1

В момент времени 164.16504237163764 с транзакт 11 вышел из модели

В момент времени 166.90790417382084 с транзакт 12 освободил устройство 2

В момент времени 166.90790417382084 с транзакт 12 вышел из модели

В момент времени 172.61532177864467 с транзакт 14 вошёл в модель

В момент времени 172.61532177864467 с транзакт 14 занял устройство 2

В момент времени 185.2398167406572 с транзакт 14 освободил устройство 2

В момент времени 185.2398167406572 с транзакт 14 вышел из модели

В момент времени 186.05917344558503 с транзакт 15 вошёл в модель

В момент времени 186.05917344558503 с транзакт 15 занял устройство 2

В момент времени 187.2813380514712 с транзакт 16 вошёл в модель

В момент времени 187.2813380514712 с транзакт 16 направился в очередь 1

В момент времени 187.9869888501134 с транзакт 13 освободил устройство 1

В момент времени 187.9869888501134 с транзакт 16 занял устройство 1

В момент времени 187.9869888501134 с транзакт 13 вышел из модели

В момент времени 198.27126550313292 с транзакт 15 освободил устройство 2

В момент времени 198.27126550313292 с транзакт 15 вышел из модели

В момент времени 201.98482495267058 с транзакт 16 освободил устройство 1

В момент времени 201.98482495267058 с транзакт 16 вышел из модели

В момент времени 211.6903796268228 с транзакт 17 вошёл в модель

В момент времени 211.6903796268228 с транзакт 17 занял устройство 1

В момент времени 211.7394754666629 с транзакт 18 вошёл в модель

В момент времени 211.7394754666629 с транзакт 18 занял устройство 2

В момент времени 220.53252648726956 с транзакт 18 освободил устройство 2

В момент времени 220.53252648726956 с транзакт 18 вышел из модели

В момент времени 231.3722270371263 с транзакт 19 вошёл в модель

В момент времени 231.3722270371263 с транзакт 19 занял устройство 2

```
В момент времени 232.87931698758308 с транзакт 17 освободил устройство 1
В момент времени 232.87931698758308 с транзакт 17 вышел из модели
В момент времени 239.77303166920933 с транзакт 19 освободил устройство 2
В момент времени 239.77303166920933 с транзакт 19 вышел из модели
В момент времени 244.9363344704559 с транзакт 20 вошёл в модель
В момент времени 244.9363344704559 с транзакт 20 занял устройство 1
В момент времени 261.7665156579732 с транзакт 20 освободил устройство 1
В момент времени 261.7665156579732 с транзакт 20 вышел из модели
В момент времени 269.81951786055737 с транзакт 21 вошёл в модель
В момент времени 269.81951786055737 с транзакт 21 занял устройство 1
В момент времени 276.2586683345197 с транзакт 22 вошёл в модель
В момент времени 276.2586683345197 с транзакт 22 занял устройство 2
В момент времени 285.3624626530195 с транзакт 21 освободил устройство 1
В момент времени 285.3624626530195 с транзакт 21 вышел из модели
В момент времени 289.91124848879934 с транзакт 23 вошёл в модель
В момент времени 289.91124848879934 с транзакт 23 занял устройство 1
```

Как видим, информация о транзактах записалась в файл, программа работает корректно. Закроем файл.

In [10]:

```
file.close()
```