

Федеральное государственное бюджетное образовательное учреждение высшего образования

**"МИРЭА – Российский технологический университет"**

**РТУ МИРЭА**

Институт кибербезопасности и цифровых технологий

ОТЧЕТ

**О ВЫПОЛНЕНИИ ПРАКТИЧЕСКОЙ РАБОТЫ №4**

**«Имитационное моделирование вычислительных систем» по дисциплине**

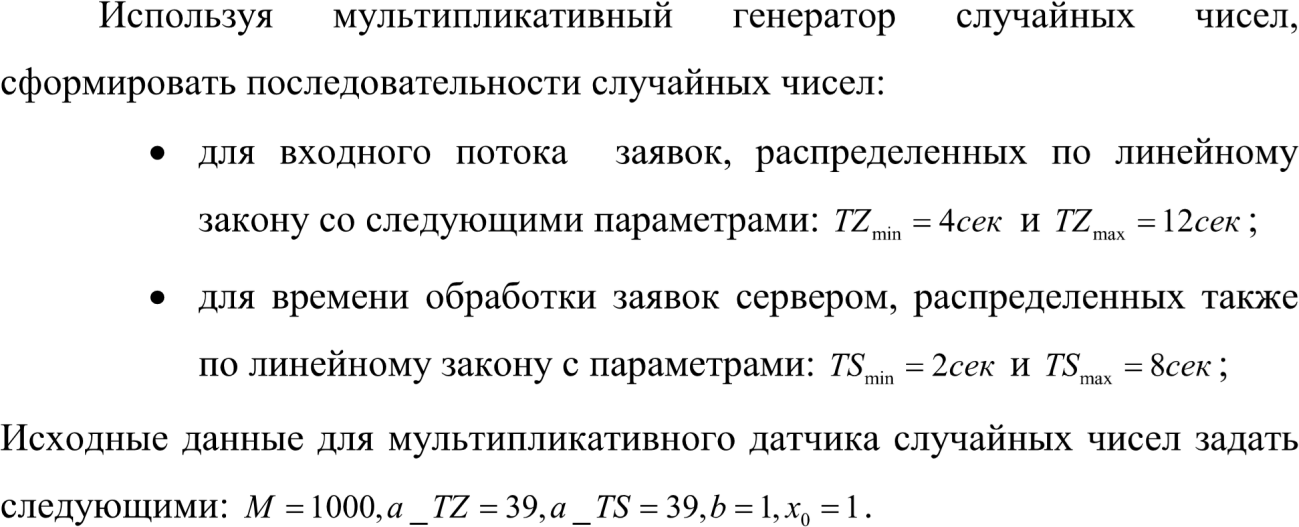
**«Моделирование систем»**

Выполнили: студенты 3 курса

группы БСБО-11-21 Смирнов И.А.

Проверил: Хорсик И.А.

**Задание №1.**



**Решение:**

# Task 1  
def multiplicative\_random\_generator(a, b, M, x, num\_values, min\_val, max\_val):  
 random\_values = []  
 for \_ in range(num\_values):  
 x = (a \* x + b) % M  
 random\_value = min\_val + (x / (M - 1)) \* (max\_val - min\_val)  
 random\_values.append(random\_value)  
 return random\_values  
  
  
a\_TZ = 39  
b = 1  
M = 1000  
x\_TZ = 1  
num\_values = 10  
min\_TZ = 4  
max\_TZ = 12  
  
a\_TS = 39  
x\_TS = 1  
min\_TS = 2  
max\_TS = 8  
  
random\_TZ = multiplicative\_random\_generator(  
 a\_TZ, b, M, x\_TZ, num\_values, min\_TZ, max\_TZ)  
random\_TS = multiplicative\_random\_generator(  
 a\_TS, b, M, x\_TS, num\_values, min\_TS, max\_TS)

Листинг 1. Код программы задания 1.

Задание №1:

Заявка 1: Время заявки = 4.32 сек, Время обработки = 2.24 сек

Заявка 2: Время заявки = 8.49 сек, Время обработки = 5.37 сек

Заявка 3: Время заявки = 11.05 сек, Время обработки = 7.29 сек

Заявка 4: Время заявки = 6.57 сек, Время обработки = 3.93 сек

Заявка 5: Время заявки = 8.16 сек, Время обработки = 5.12 сек

Заявка 6: Время заявки = 6.25 сек, Время обработки = 3.69 сек

Заявка 7: Время заявки = 11.69 сек, Время обработки = 7.77 сек

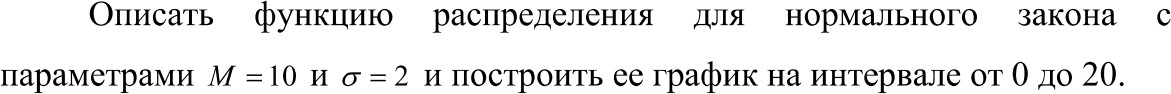
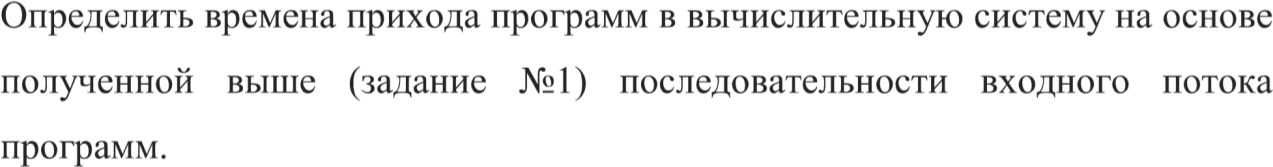
Заявка 8: Время заявки = 7.53 сек, Время обработки = 4.65 сек

Заявка 9: Время заявки = 5.60 сек, Время обработки = 3.20 сек

Заявка 10: Время заявки = 10.41 сек, Время обработки = 6.81 сек

Рис. 2. Результат задания 1.

**Задание №2.**



**Решение:**

# Task 2  
  
def arrival\_times(request\_times):  
 arrival\_times = [request\_times[0]]  
  
 for i in range(1, len(request\_times)):  
 arrival\_time = arrival\_times[i - 1] + request\_times[i]  
 arrival\_times.append(arrival\_time)  
  
 return arrival\_times  
  
  
request\_times = [7.21, 4.56, 9.12, 6.75, 4.98, 5.32, 8.11, 11.07, 6.89, 9.32]  
  
arrival\_times = arrival\_times(request\_times)

Листинг 3. Код программы задания 2.

Задание №2:

Программа 1: Время прихода = 7.21 сек

Программа 2: Время прихода = 11.77 сек

Программа 3: Время прихода = 20.89 сек

Программа 4: Время прихода = 27.64 сек

Программа 5: Время прихода = 32.62 сек

Программа 6: Время прихода = 37.94 сек

Программа 7: Время прихода = 46.05 сек

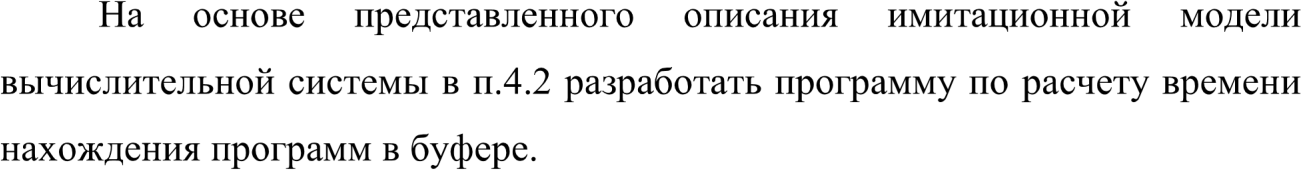
Программа 8: Время прихода = 57.12 сек

Программа 9: Время прихода = 64.01 сек

Программа 10: Время прихода = 73.33 сек

Рис. 4. Результат задания 2.

**Задание №3.**



**Решение:**

class Buffer:  
 def \_\_init\_\_(self, capacity):  
 self.capacity = capacity  
 self.queue = []  
 self.time\_in\_buffer = 0  
  
 def is\_full(self):  
 return len(self.queue) >= self.capacity  
  
 def enqueue(self, program):  
 if not self.is\_full():  
 self.queue.append(program)  
  
 def dequeue(self):  
 if self.queue:  
 return self.queue.pop(0)  
 else:  
 return None  
  
 def process(self):  
 if self.queue:  
 program = self.dequeue()  
 return program  
 else:  
 return None  
  
  
class Program:  
 def \_\_init\_\_(self, arrival\_time):  
 self.arrival\_time = arrival\_time  
 self.processing\_time = random.uniform(min\_TS, max\_TS)  
 self.enter\_buffer\_time = None  
 self.leave\_buffer\_time = None  
  
  
min\_TZ = 4  
max\_TZ = 12  
min\_TS = 2  
max\_TS = 8  
buffer\_capacity = 3  
  
arrival\_times = [7.21, 4.56, 9.12, 6.75, 4.98, 5.32, 8.11, 11.07, 6.89, 9.32]  
  
buffer = Buffer(buffer\_capacity)  
programs = []  
  
for arrival\_time in arrival\_times:  
 program = Program(arrival\_time)  
 programs.append(program)  
  
current\_time = 0  
  
for program in programs:  
 if current\_time < program.arrival\_time:  
 current\_time = program.arrival\_time  
 program.enter\_buffer\_time = current\_time  
 while buffer.is\_full():  
 current\_program = buffer.process()  
 current\_program.leave\_buffer\_time = current\_time  
 buffer.enqueue(program)  
 current\_time += program.processing\_time  
 program.leave\_buffer\_time = current\_time  
  
total\_time\_in\_buffer = sum(  
 (program.leave\_buffer\_time - program.enter\_buffer\_time) for program in programs)  
average\_time\_in\_buffer = total\_time\_in\_buffer / len(programs)  
print("\nЗадание №3:\n ")  
print(f"Общее время в буфере: {total\_time\_in\_buffer:.2f} сек")  
print(f"Среднее время в буфере: {average\_time\_in\_buffer:.2f} сек")

Листинг 5. Программа по расчету времени нахождения программ в буфере.

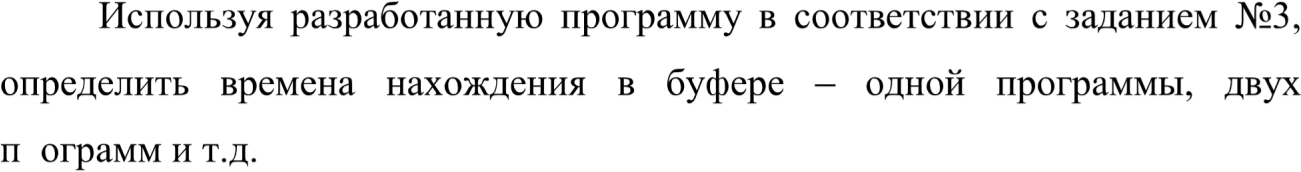
Задание №3:

Общее время в буфере: 122.02 сек

Среднее время в буфере: 12.20 сек

Рис. 6. Результат задания 3.

**Задание №4.**



**Решение:**

# Task 4  
print("\nЗадание №4:\n ")  
buffer\_times = defaultdict(list)  
  
for program in programs:  
 time\_in\_buffer = program.leave\_buffer\_time - program.enter\_buffer\_time  
 if time\_in\_buffer > 0:  
 buffer\_times[int(time\_in\_buffer)].append(time\_in\_buffer)  
  
for count, times in buffer\_times.items():  
 if count == 1:  
 print(f"Времена нахождения одной программы в буфере: {times}")  
 else:  
 print(f"Времена нахождения {count} программ в буфере: {times}")

Листинг 7. Код решения задания 4.

Задание №4:

Времена нахождения 15 программ в буфере: [15.435554734592344, 15.327380466896756]

Времена нахождения 13 программ в буфере: [13.395180669505823, 13.781138750646583]

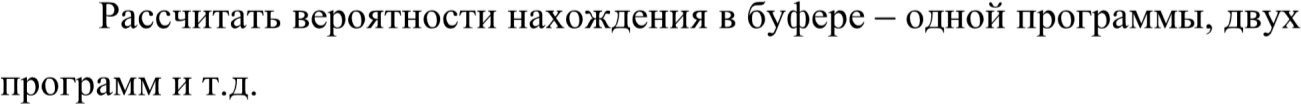
Времена нахождения 16 программ в буфере: [16.442116855281604, 16.072124179151245, 16.053445367573143]

Времена нахождения 4 программ в буфере: [4.196301552277447]

Времена нахождения 5 программ в буфере: [5.462114416149632, 5.855025594438189]

Рис. 8. Гистограммы относительных частот.

**Задание №5.**



**Решение:**

Task 5  
print("\nЗадание №5:\n ")  
buffer\_counts = defaultdict(int)  
  
for program in programs:  
 time\_in\_buffer = program.leave\_buffer\_time - program.enter\_buffer\_time  
 if time\_in\_buffer > 0:  
 buffer\_counts[time\_in\_buffer] += 1  
  
total\_programs = len(programs)  
  
probabilities = {}  
for count, events in buffer\_counts.items():  
 probability = events / total\_programs  
 probabilities[count] = probability  
  
for count, probability in probabilities.items():  
 print(  
 f"Вероятность нахождения {count} программ в буфере: {probability:.2%}")

Листинг 9. Код программы задания 5.

Задание №5:

Вероятность нахождения 15.435554734592344 программ в буфере: 10.00%

Вероятность нахождения 15.327380466896756 программ в буфере: 10.00%

Вероятность нахождения 13.395180669505823 программ в буфере: 10.00%

Вероятность нахождения 16.442116855281604 программ в буфере: 10.00%

Вероятность нахождения 16.072124179151245 программ в буфере: 10.00%

Вероятность нахождения 16.053445367573143 программ в буфере: 10.00%

Вероятность нахождения 13.781138750646583 программ в буфере: 10.00%

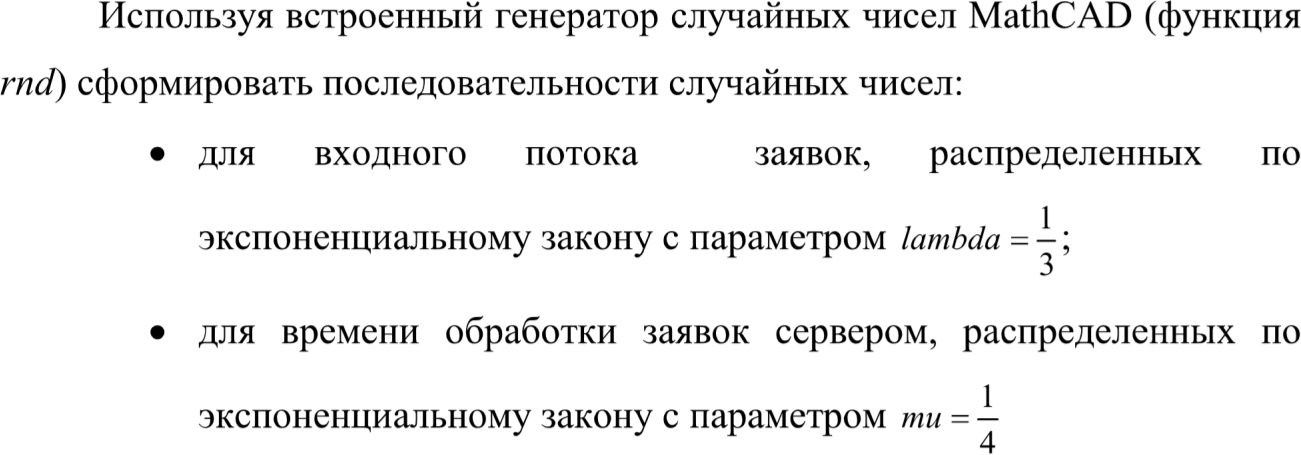
Вероятность нахождения 4.196301552277447 программ в буфере: 10.00%

Вероятность нахождения 5.462114416149632 программ в буфере: 10.00%

Вероятность нахождения 5.855025594438189 программ в буфере: 10.00%

Рис. 10. Результат задания 5.

**Задание №6.**



**Решение:**

# Task 6  
print("\nЗадание №6:\n ")  
lambda\_value = 1/3  
num\_values = 10  
print("\nlambda = 1/3:\n ")  
for \_ in range(num\_values):  
 random\_value = -1 / lambda\_value \* math.log(1 - random.random())  
 print(f"Случайное число с экспоненциальным распределением: {random\_value:.2f}")  
  
lambda\_value = 1/4  
num\_values = 10  
print("\nmu = 1/4:\n ")  
for \_ in range(num\_values):  
 random\_value = -1 / lambda\_value \* math.log(1 - random.random())  
 print(f"Случайное число с экспоненциальным распределением: {random\_value:.2f}")

Листинг 11. Код программы задания 6.

Задание №6:

lambda = 1/3:

Случайное число с экспоненциальным распределением: 3.97

Случайное число с экспоненциальным распределением: 2.93

Случайное число с экспоненциальным распределением: 5.99

Случайное число с экспоненциальным распределением: 5.86

Случайное число с экспоненциальным распределением: 1.32

Случайное число с экспоненциальным распределением: 0.78

Случайное число с экспоненциальным распределением: 0.89

Случайное число с экспоненциальным распределением: 0.47

Случайное число с экспоненциальным распределением: 2.16

Случайное число с экспоненциальным распределением: 5.53

mu = 1/4:

Случайное число с экспоненциальным распределением: 0.63

Случайное число с экспоненциальным распределением: 0.31

Случайное число с экспоненциальным распределением: 0.09

Случайное число с экспоненциальным распределением: 1.23

Случайное число с экспоненциальным распределением: 4.73

Случайное число с экспоненциальным распределением: 2.32

Случайное число с экспоненциальным распределением: 0.48

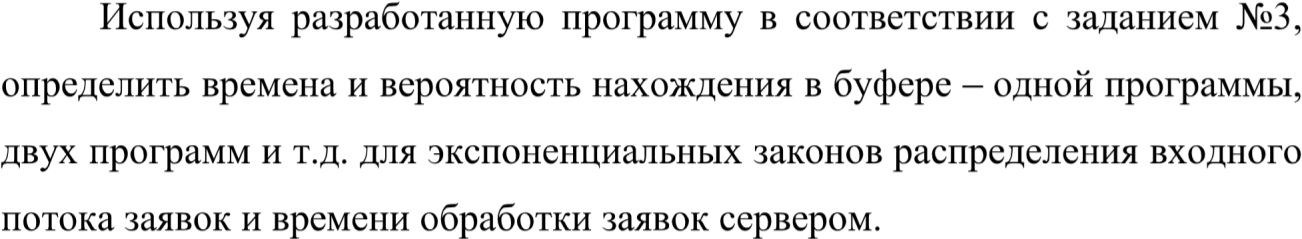
Случайное число с экспоненциальным распределением: 1.94

Случайное число с экспоненциальным распределением: 0.80

Случайное число с экспоненциальным распределением: 0.10

Рис. 12. Результат задания 6.

**Задание №7.**



**Решение:**

# Task 7  
print("\nЗадание №7:\n ")  
def generate\_exponential(lambda\_value, num\_values):  
 random\_values = []  
 for \_ in range(num\_values):  
 random\_value = -1 / lambda\_value \* math.log(1 - random.random())  
 random\_values.append(random\_value)  
 return random\_values  
  
lambda\_TZ = 1/3  
lambda\_TS = 1/4  
num\_programs = 1000  
  
  
random\_TZ = generate\_exponential(lambda\_TZ, num\_programs)  
random\_TS = generate\_exponential(lambda\_TS, num\_programs)  
  
buffer\_capacity = 3  
buffer = Buffer(buffer\_capacity)  
programs = []  
  
for i in range(num\_programs):  
 arrival\_time = random\_TZ[i]  
 program = Program(arrival\_time)  
 programs.append(program)  
  
current\_time = 0  
  
for program in programs:  
 if current\_time < program.arrival\_time:  
 current\_time = program.arrival\_time  
 program.enter\_buffer\_time = current\_time  
 while buffer.is\_full():  
 current\_program = buffer.process()  
 current\_program.leave\_buffer\_time = current\_time  
 buffer.enqueue(program)  
 current\_time += program.processing\_time  
 program.leave\_buffer\_time = current\_time  
  
total\_time\_in\_buffer = sum(  
 (program.leave\_buffer\_time - program.enter\_buffer\_time) for program in programs)  
average\_time\_in\_buffer = total\_time\_in\_buffer / num\_programs  
print(f"Общее время в буфере: {total\_time\_in\_buffer:.2f} сек")  
print(f"Среднее время в буфере: {average\_time\_in\_buffer:.2f} сек\n")  
  
buffer\_counts = defaultdict(int)  
  
for program in programs:  
 time\_in\_buffer = program.leave\_buffer\_time - program.enter\_buffer\_time  
 if time\_in\_buffer > 0:  
 buffer\_counts[int(time\_in\_buffer)] += 1  
  
total\_programs = len(programs)  
  
probabilities = {}  
for count, events in buffer\_counts.items():  
 probability = events / total\_programs  
 probabilities[count] = probability  
  
for count, probability in probabilities.items():  
 print(f"Вероятность нахождения {count} программ в буфере: {probability:.2%}")

Листинг 13. Код программы задания 7.

Задание №7:

Общее время в буфере: 15015.56 сек

Среднее время в буфере: 15.02 сек

Вероятность нахождения 17 программ в буфере: 10.40%

Вероятность нахождения 19 программ в буфере: 3.20%

Вероятность нахождения 18 программ в буфере: 7.40%

Вероятность нахождения 16 программ в буфере: 10.50%

Вероятность нахождения 15 программ в буфере: 13.30%

Вероятность нахождения 11 программ в буфере: 6.90%

Вероятность нахождения 8 программ в буфере: 1.80%

Вероятность нахождения 7 программ в буфере: 0.50%

Вероятность нахождения 14 программ в буфере: 11.30%

Вероятность нахождения 12 программ в буфере: 11.30%

Вероятность нахождения 13 программ в буфере: 11.30%

Вероятность нахождения 10 программ в буфере: 4.20%

Вероятность нахождения 9 программ в буфере: 2.00%

Вероятность нахождения 22 программ в буфере: 0.90%

Вероятность нахождения 20 программ в буфере: 3.50%

Вероятность нахождения 23 программ в буфере: 0.10%

Вероятность нахождения 21 программ в буфере: 1.20%

Вероятность нахождения 6 программ в буфере: 0.10%

Вероятность нахождения 4 программ в буфере: 0.10%

Рис. 14. Результат задания 7.