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.

Используя мультипликативный генератор случайных чисел, сформировать последовательности случайных чисел:

- для входного потока заявок, распределенных по линейному закону со следующими параметрами: $TZ_{\min} = 4сек$ и $TZ_{\max} = 12сек$;
- для времени обработки заявок сервером, распределенных также по линейному закону с параметрами: $TS_{\min} = 2сек$ и $TS_{\max} = 8сек$;

Исходные данные для мультипликативного датчика случайных чисел задать следующими: $M = 1000, a\_TZ = 39, a\_TS = 39, b = 1, x_0 = 1$.

Решение:

```python
# 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.

Определить времена прихода программ в вычислительную систему на основе полученной выше (задание №1) последовательности входного потока программ.

Описать функцию распределения для нормального закона с параметрами $M = 10$ и $\sigma = 2$ и построить ее график на интервале от 0 до 20.

Решение:

```python
# 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.

На основе представленного описания имитационной модели вычислительной системы в п.4.2 разработать программу по расчету времени нахождения программ в буфере.

Решение:

```python
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.

Используя разработанную программу в соответствии с заданием №3, определить времена нахождения в буфере – одной программы, двух п ограмм и т.д.

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.

Рассчитать вероятности нахождения в буфере – одной программы, двух программ и т.д.

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.

Используя встроенный генератор случайных чисел MathCAD (функция *rnd*) сформировать последовательности случайных чисел:

- для входного потока заявок, распределенных по экспоненциальному закону с параметром $lambda = \frac{1}{3}$;
- для времени обработки заявок сервером, распределенных по экспоненциальному закону с параметром $mu = \frac{1}{4}$

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.

Используя разработанную программу в соответствии с заданием №3, определить времена и вероятность нахождения в буфере – одной программы, двух программ и т.д. для экспоненциальных законов распределения входного потока заявок и времени обработки заявок сервером.

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.