

МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**

МОДЕЛИРОВАНИЕ СИСТЕМ. ВАРИАНТ 2
ОТЧЕТ О ПРАКТИКЕ

студента 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные
технологии
факультета КНиИТ
Аношкина Андрея Алексеевича

Проверил
к. ф.-м. н., доцент

И. Е. Тананко

СОДЕРЖАНИЕ

1	ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ СИСТЕМ И СЕТЕЙ МАССОВОГО ОБСЛУЖИВАНИЯ.....	3
1.1	Системы массового обслуживания	3
1.1.1	Постановка задачи	3
1.1.2	Код.....	3
1.1.3	Результат	7

1 ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ СИСТЕМ И СЕТЕЙ МАССОВОГО ОБСЛУЖИВАНИЯ

1.1 Системы массового обслуживания

1.1.1 Постановка задачи

Дана СМО типа $M | M | 2$. Построить имитационную модель системы. На основании 1000 выборочных значений оценить \bar{u} и \bar{n} .

1.1.2 Код

Model.h

```
1  #pragma once
2  #include <queue>
3  #include <vector>
4  #include <time.h>
5
6  using namespace std;
7
8  class Model {
9  private:
10     queue<double> q;
11     double timeOfTaskDoneFirst;
12     double timeOfTaskDoneSecond;
13     double mu;
14     bool active_1;
15     bool active_2;
16     vector<double> timeInQueue;
17
18     double generateTime();
19
20 public:
21     Model(double mu);
22
23     void addToQueue(double time);
24     size_t getQueueLength();
25
26     double getTimeOfTaskDoneFirst();
27     double getTimeOfTaskDoneSecond();
28
29     void analyze(double time);
30
31     vector<double>& getTimeInQueue();
32 };
```

Model.cpp

```
1  #include "Model.h"
2
3  Model::Model(double mu) {
4      srand(time(0));
5      this->mu = mu;
6      this->active_1 = false;
7      this->active_2 = false;
8      this->timeOfTaskDoneFirst = 100000;
9      this->timeOfTaskDoneSecond = 100000;
10 }
11
12 void Model::addToQueue(double time) {
13     this->q.push(time);
14 }
15
16 size_t Model::getQueueLength() {
17     return this->q.size();
18 }
19
20 double Model::getTimeOfTaskDoneFirst() {
21     return this->timeOfTaskDoneFirst;
22 }
23
24 double Model::getTimeOfTaskDoneSecond() {
25     return this->timeOfTaskDoneSecond;
26 }
27
28 double Model::generateTime() {
29     int r = rand();
30     return -1.0 / this->mu * log(min(max(r, 1), RAND_MAX - 1) * 1.0 / RAND_MAX);
31 }
32
33 void Model::analyze(double time) {
34     if (this->active_1)
35         if (time >= this->timeOfTaskDoneFirst && time <=
36             ↪ this->timeOfTaskDoneFirst) {
37             this->active_1 = false;
38             this->timeOfTaskDoneFirst = 100000;
39         }
40     if (!this->active_1)
41         if (!this->q.empty()) {
```

```

42         this->active_1 = true;
43         this->timeInQueue.push_back(time - this->q.front());
44         this->q.pop();
45         this->timeOfTaskDoneFirst = time + this->generateTime();
46     }
47
48
49     if (this->active_2)
50         if (time >= this->timeOfTaskDoneSecond && time <=
51             ↪ this->timeOfTaskDoneSecond) {
52             this->active_2 = false;
53             this->timeOfTaskDoneSecond = 100000;
54         }
55
56     if (!this->active_2)
57         if (!this->q.empty()) {
58             this->active_2 = true;
59             this->timeInQueue.push_back(time - this->q.front());
60             this->q.pop();
61             this->timeOfTaskDoneSecond = time + this->generateTime();
62         }
63     }
64     vector<double>& Model::getTimeInQueue() {
65         return this->timeInQueue;
66     }

```

main.cpp

```

1  #include <iostream>
2  #include <vector>
3  #include <time.h>
4  #include <cmath>
5  #include "Model.h"
6
7  using namespace std;
8
9  double lambda = 20;
10 double mu = 15;
11
12 double generateTask() {
13     int r = rand();
14     return -1.0 / lambda * log(min(max(r, 1), RAND_MAX - 1) * 1.0 / RAND_MAX);
15 }

```

```

16
17 double nextMoment(Model& model, double time_of_task) {
18     double min_time =
19         (model.getTimeOfTaskDoneFirst() <= model.getTimeOfTaskDoneSecond()) ?
20         model.getTimeOfTaskDoneFirst() : model.getTimeOfTaskDoneSecond();
21
22     double next_moment = (min_time <= time_of_task) ? min_time : time_of_task;
23
24     return next_moment;
25 }
26
27 double m(Model& model) {
28     float M = 0;
29     vector<double>::iterator begin = model.getTimeInQueue().begin();
30     vector<double>::iterator end = model.getTimeInQueue().end();
31     for (vector<double>::iterator it = begin; it != end; M += *it, ++it);
32
33     return M / model.getTimeInQueue().size();
34 }
35
36 double n(vector<double>& v, double T) {
37     float n = 0;
38     for (int i = 0; i < v.size(); n += i * v[i] / T, ++i);
39
40     return n;
41 }
42
43 int main() {
44     srand(time(0));
45
46     double current_time = 0;
47     int current_count = 0;
48     int total_count = 1000;
49
50     cout << "Model M|M|2\n\n";
51
52     cout << "lambda = "; cin >> lambda;
53     cout << "mu = "; cin >> mu;
54     cout << "\n";
55
56     Model model(mu);
57
58     double time_of_task = 0;

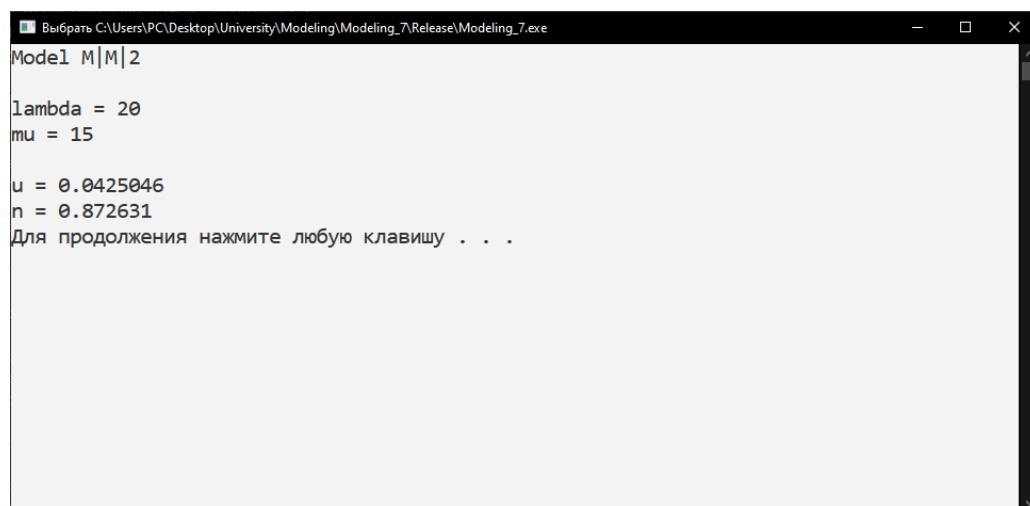
```

```

59     vector<double> countInQueue;
60
61     while ((current_count < total_count) || (model.getTimeInQueue().size() <
        ⇨ total_count)) {
62         if (current_time == time_of_task) {
63             current_count++;
64
65             model.addToQueue(time_of_task);
66
67             time_of_task += generateTask();
68         }
69
70         model.analyze(current_time);
71         for (int i = countInQueue.size(); i < model.getQueueLength() + 1;
        ⇨ countInQueue.push_back(0), ++i);
72
73         countInQueue[model.getQueueLength()] -= current_time;
74         current_time = nextMoment(model, time_of_task);
75         countInQueue[model.getQueueLength()] += current_time;
76     }
77
78     cout << "m = " << m(model) << "\n";
79     cout << "n = " << n(countInQueue, current_time) << "\n";
80
81     system("pause");
82
83     return 0;
84 }

```

1.1.3 Результат



```
Выбрать C:\Users\PC\Desktop\University\Modeling_7\Release\Modeling_7.exe
Model M|M|2
lambda = 20
mu = 15
u = 0.0425046
n = 0.872631
Для продолжения нажмите любую клавишу . . .
```

Рисунок 1 – $\lambda = 20, \mu = 15$