**Министерство образования и науки Российской Федерации**

**Московский государственный технический университет им. Н. Э. Баумана**

**Кафедра «РК6» Группа 36Б**

**Отчет по домашнему заданию № 3**

По курсу «Теория вероятностей и математическая статистика»

**Вариант 16**

Выполнил: Самойлов А. А.

Проверил: Берчун Ю.В.

Дата: 17.12.2020

Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Москва, 2020 г.**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **R1** | **G1** | **B1** | **R2** | **G2** | **B2** | **R3** | **G3** | **B3** |
| **6** | **5** | **8** | **7** | **6** | **7** | **11** | **10** | **10** |

Известно, что среднее время между звонками клиентов составляет TC = R1 + G1 + B1 = 19, секунд, а среднее время обслуживания TS = R2 = 7 секунд. Все потоки случайных событий считать пуассоновскими. Если все операторы заняты, звонок теряется.

Найдём интенсивность входного потока и интенсивность потока обслуживания: λ = = , μ = = .

**Задание 1**

Рассмотреть систему без очереди. Построить графики от числа операторов: вероятности отказа (вплоть до обеспечения отказов менее 1%); математического ожидания числа занятых операторов; коэффициента загрузки операторов.

В общем случае:

В этом задании схема будет иметь вид:

λ λ λ λ

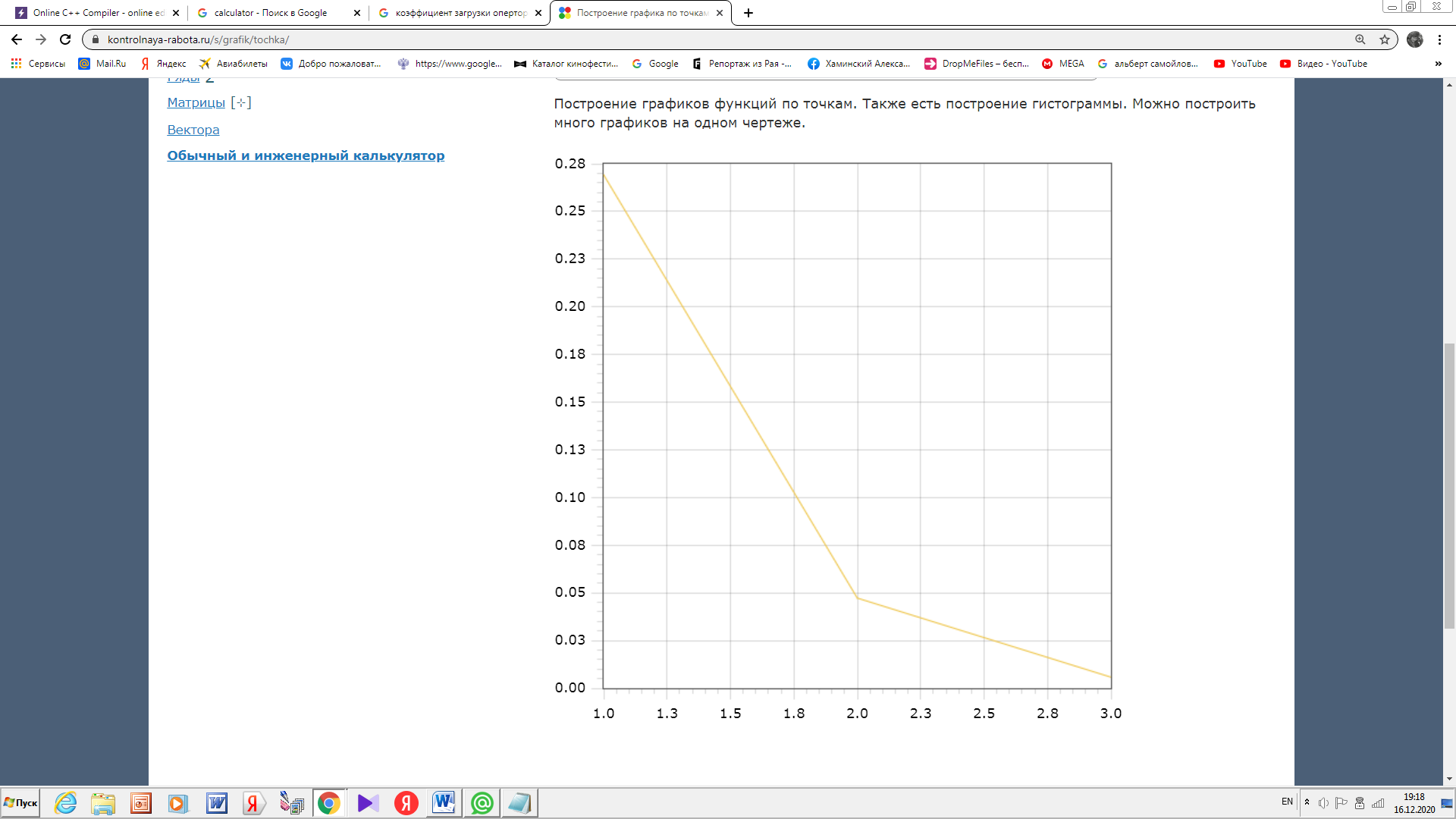
S0 S1 S2 … Sn

1μ 2μ 3μ nμ

Тогда:

Исходя из того, что , найдём , выразив все остальные через , как указано в предыдущей формуле.

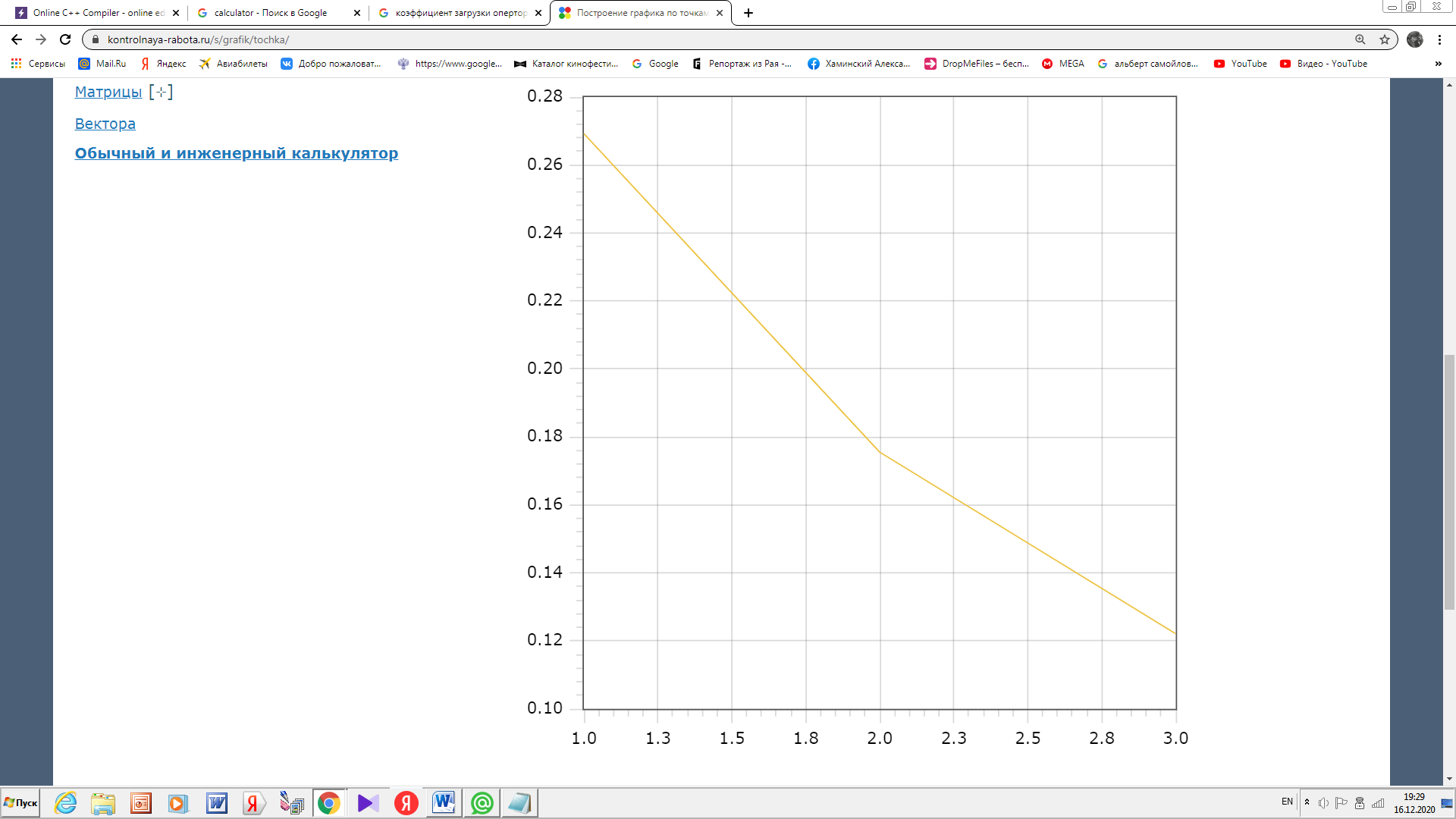
– найдём вероятность отказа. Построим график зависимости величины вероятности отказа от n. Будем искать значения до того, пока они не станут меньше 0.01.



Математическое ожидание числа занятых операторов в данной задаче можно найти по формуле: . Построим график зависимости величины мат ожидания числа занятых операторов от n.



Коэффициент загрузки операторов равен – построим график зависимости этой величины от n.



Полученные данные в табличном виде:

|  |  |  |  |
| --- | --- | --- | --- |
| **n** | **Pотказ** |  | **kзагр\_опер** |
| **1** | **0.269231** | **0.269231** | **0.269231** |
| **2** | **0.0472517** | **0.351013** | **0.175506** |
| **3** | **0.00576936** | **0.366295** | **0.122098** |

**Задание 2**

Рассмотреть систему с ограниченной очередью. Варьируя число операторов (вплоть до числа каналов, соответствующего 1% отказов в системе без очереди), построить семейства графиков от числа мест в очереди: вероятности отказа; математического ожидания числа занятых операторов; коэффициента загрузки операторов; вероятности существования очереди; математического ожидания длины очереди; коэффициента занятости мест в очереди. Варьируя число мест в очереди, построить семейства графиков от числа операторов: вероятности отказа; математического ожидания числа занятых операторов; коэффициента загрузки операторов; вероятности существования очереди; математического ожидания длины очереди; коэффициента занятости мест в очереди.

В этом задании схема будет иметь вид:

λ λ λ λ λ λ

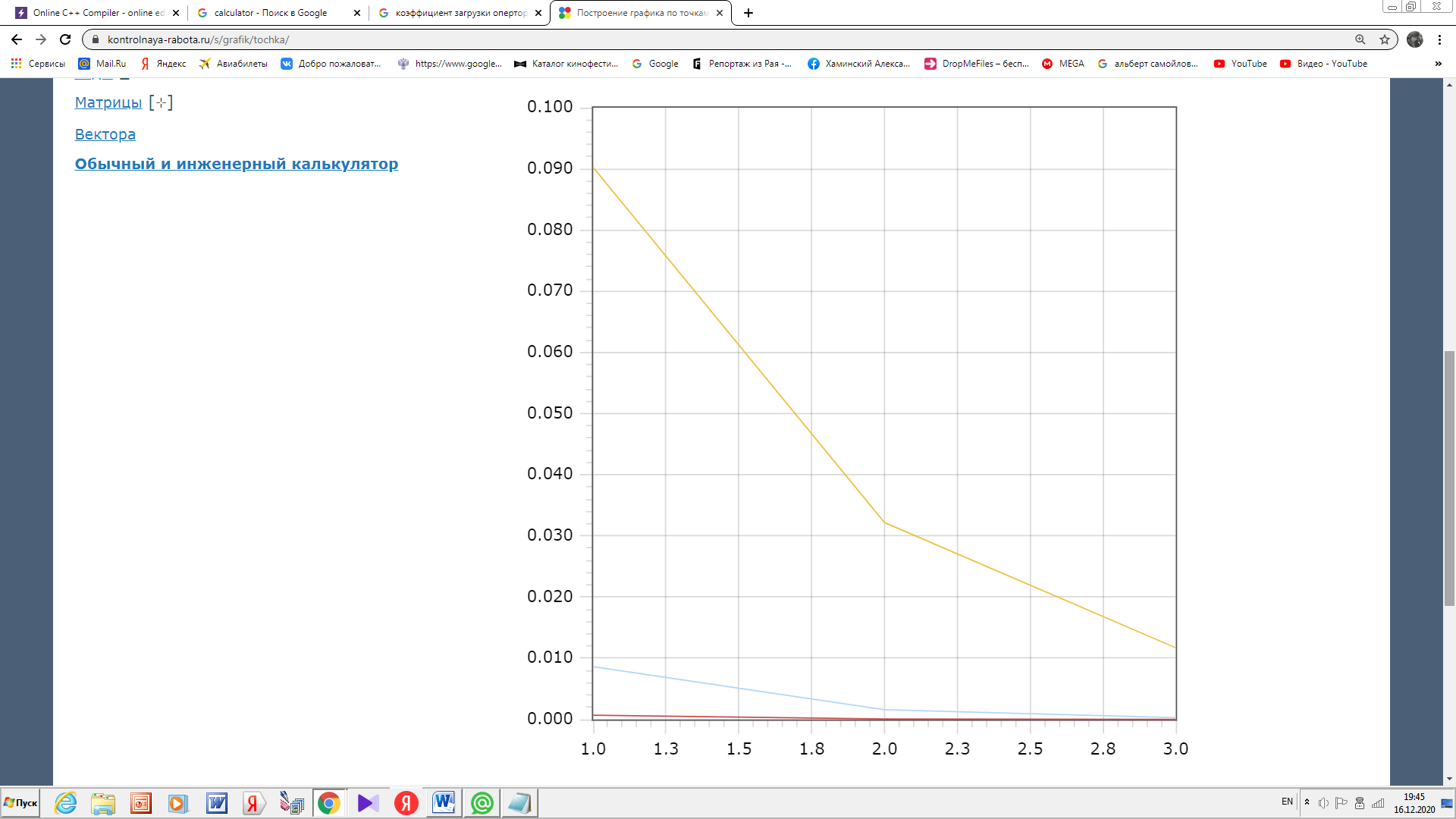
S0 S1 … Sn Sn+1 … Sn+m

1μ 2μ nμ nμ nμ nμ

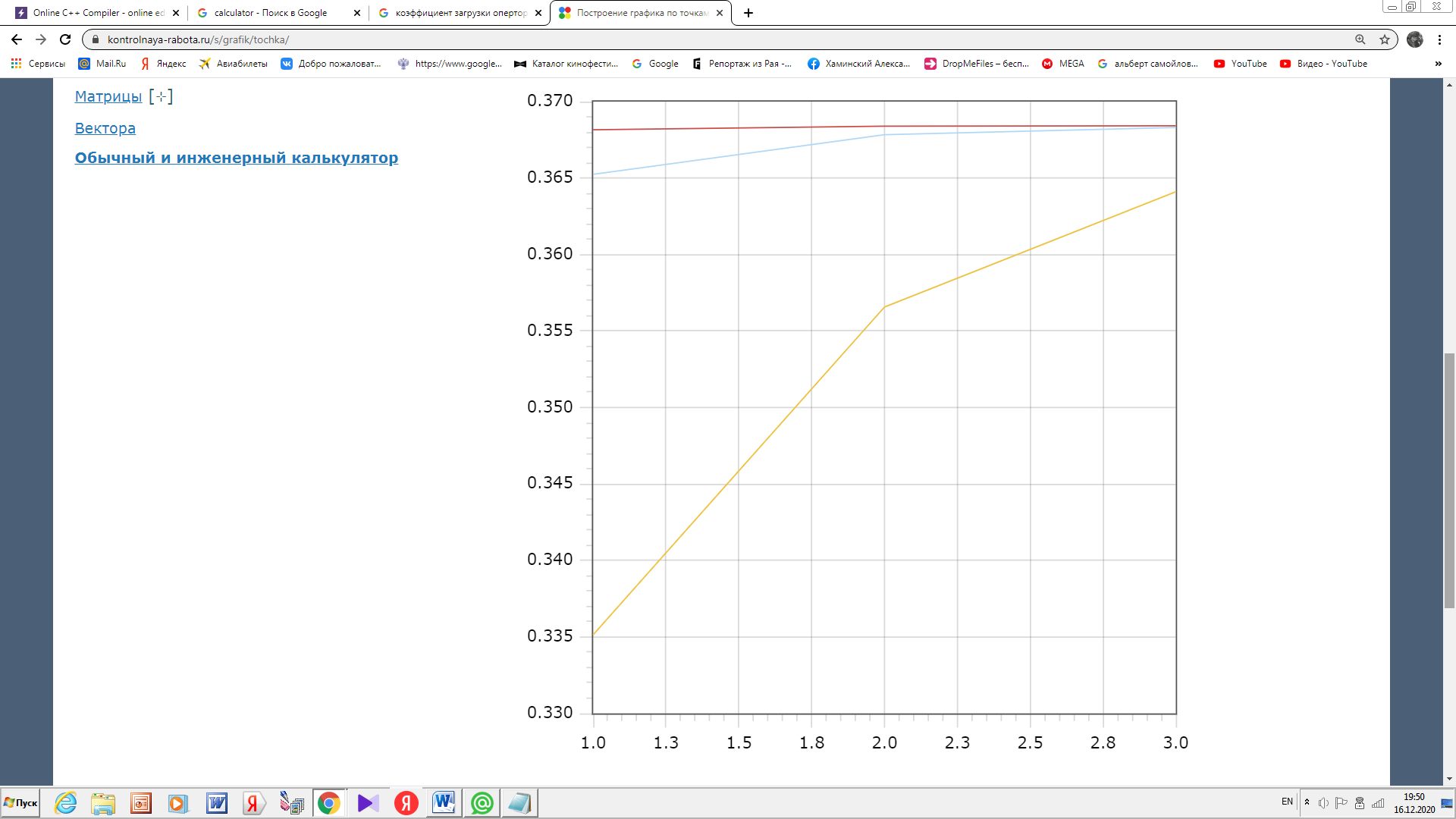
Тогда:

Исходя из того, что , найдём , выразив все остальные через , как указано в предыдущей формуле.

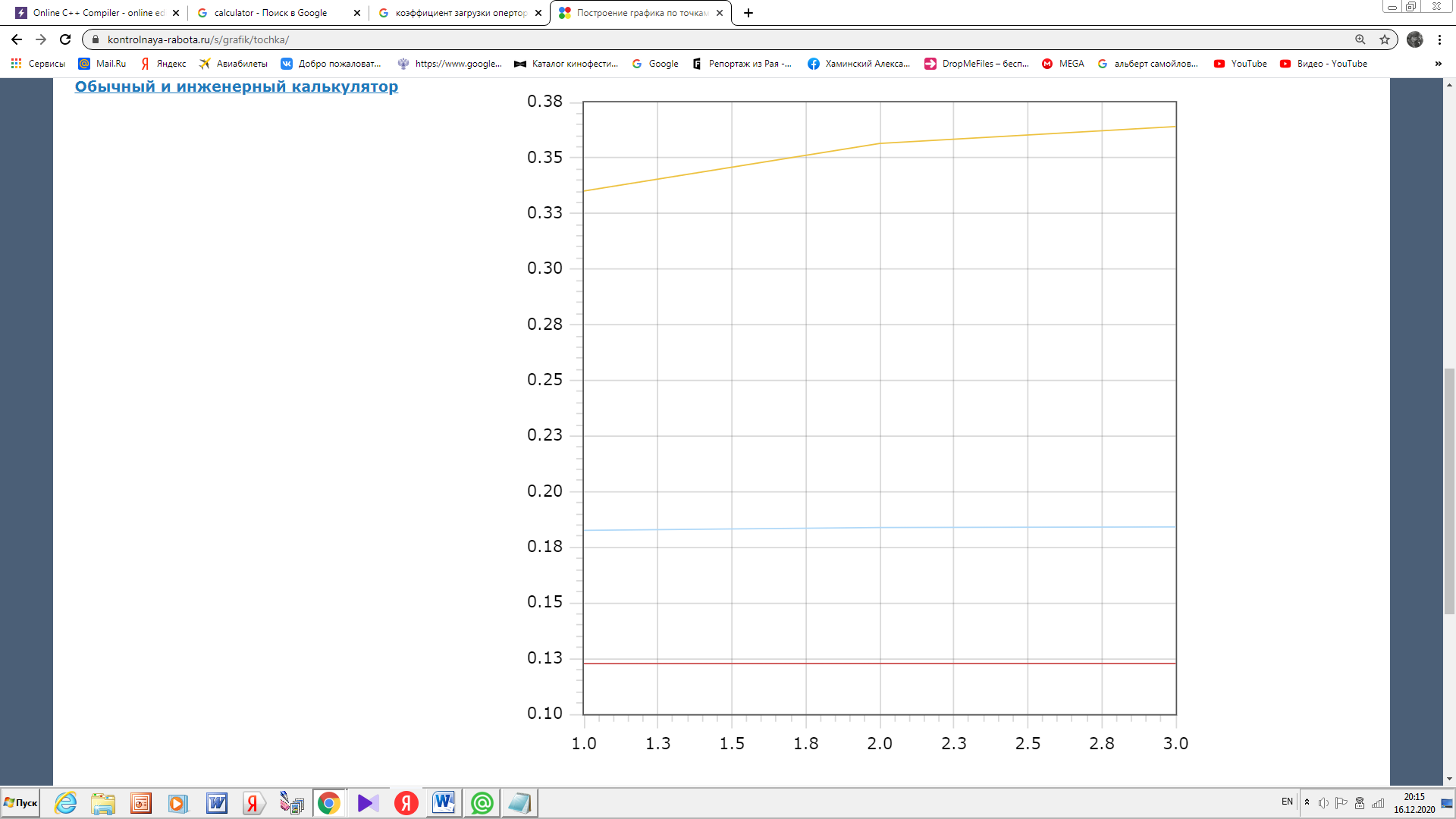
– найдём вероятность отказа. Построим график зависимости величины вероятности отказа от m для разных n.



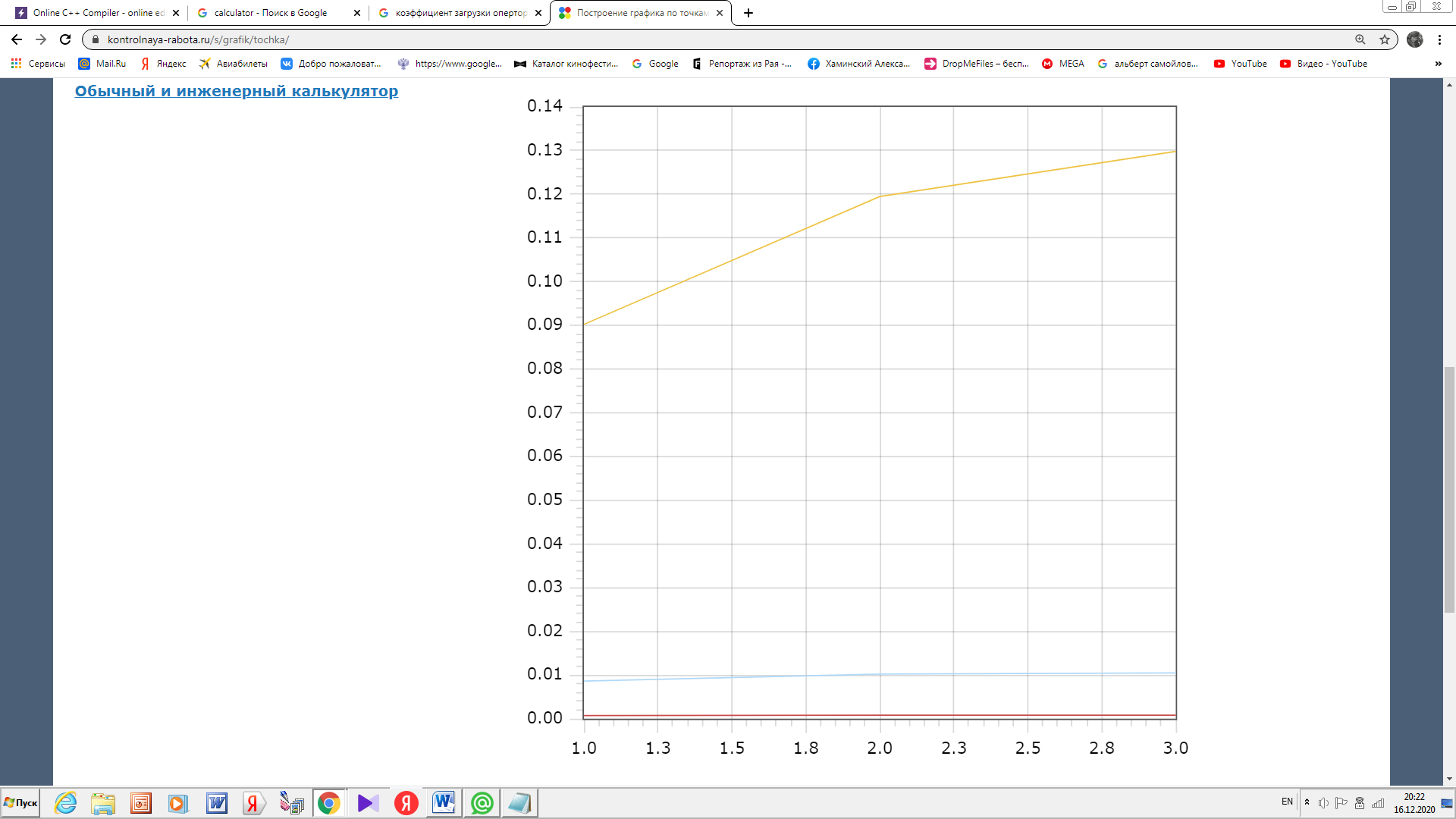
Математическое ожидание числа занятых операторов в данной задаче можно найти по формуле: . Построим график зависимости величины мат ожидания числа занятых операторов от m для разных n.



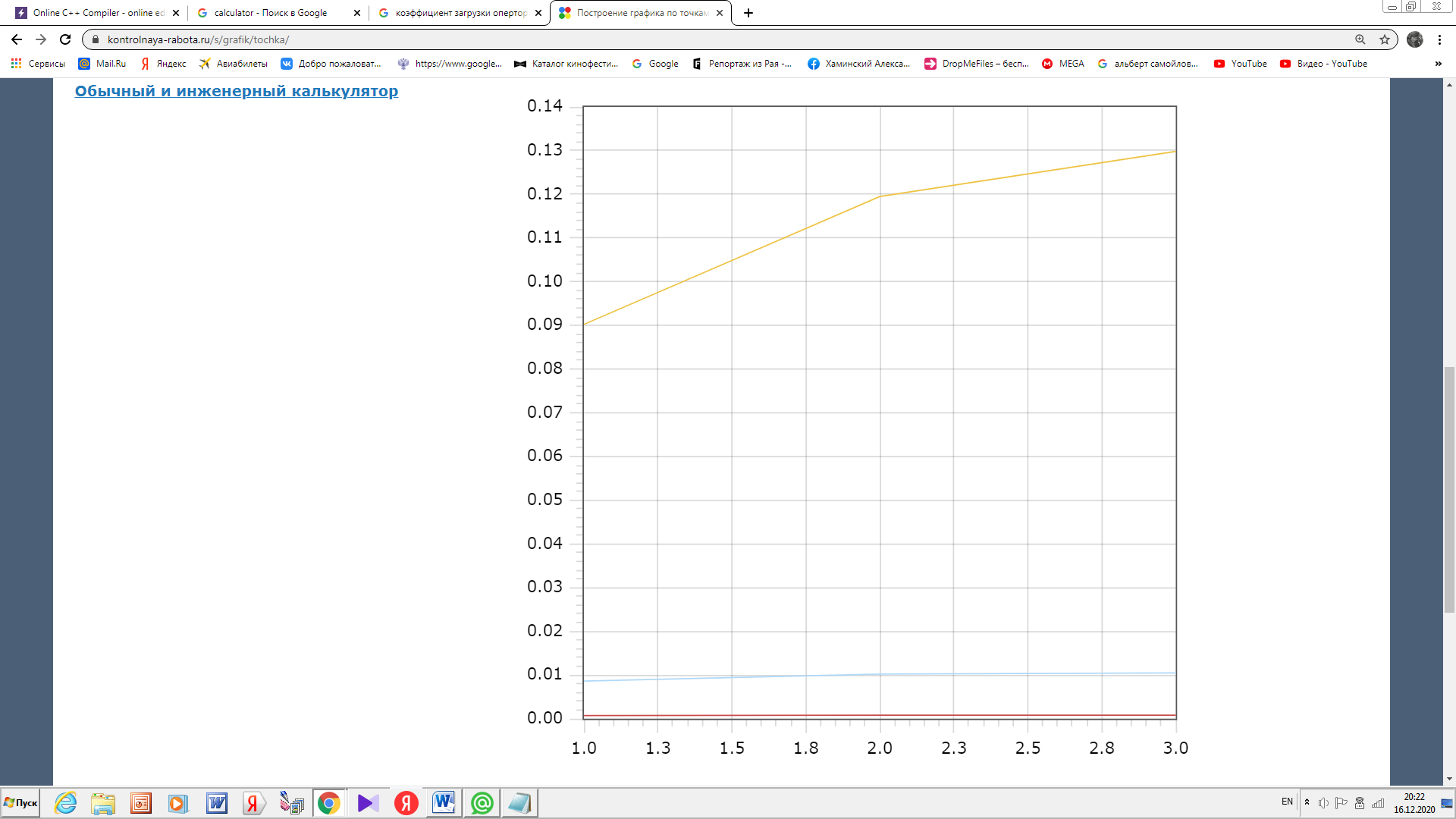
Коэффициент загрузки операторов: – построим график зависимости этой величины от m.



Вероятность существования очереди: . Построим график зависимости вероятности существования очереди от m для разных n.



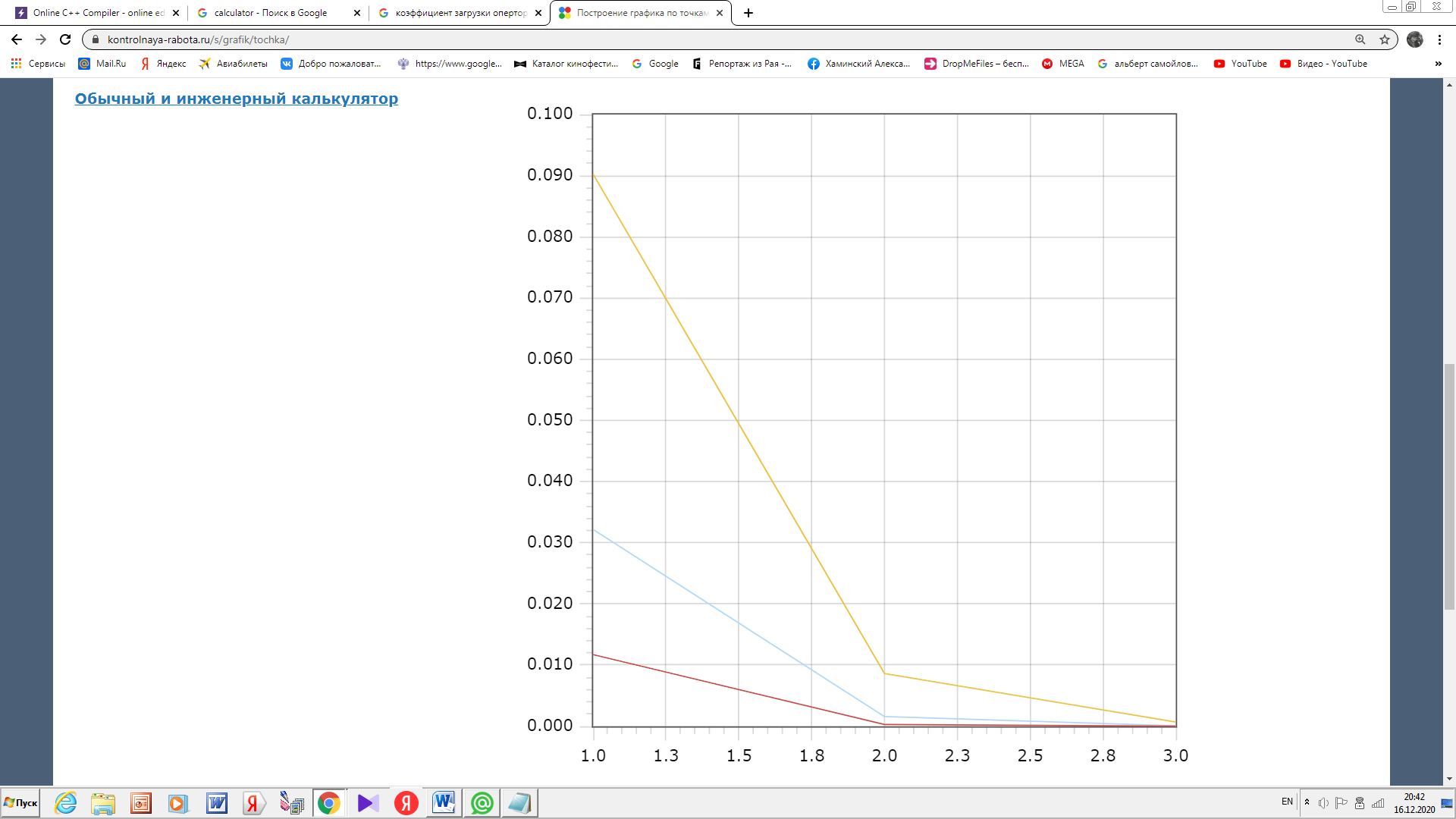
Мат ожидание длины очереди: . Построим график зависимости математического ожидания длины очереди от m для разных n.



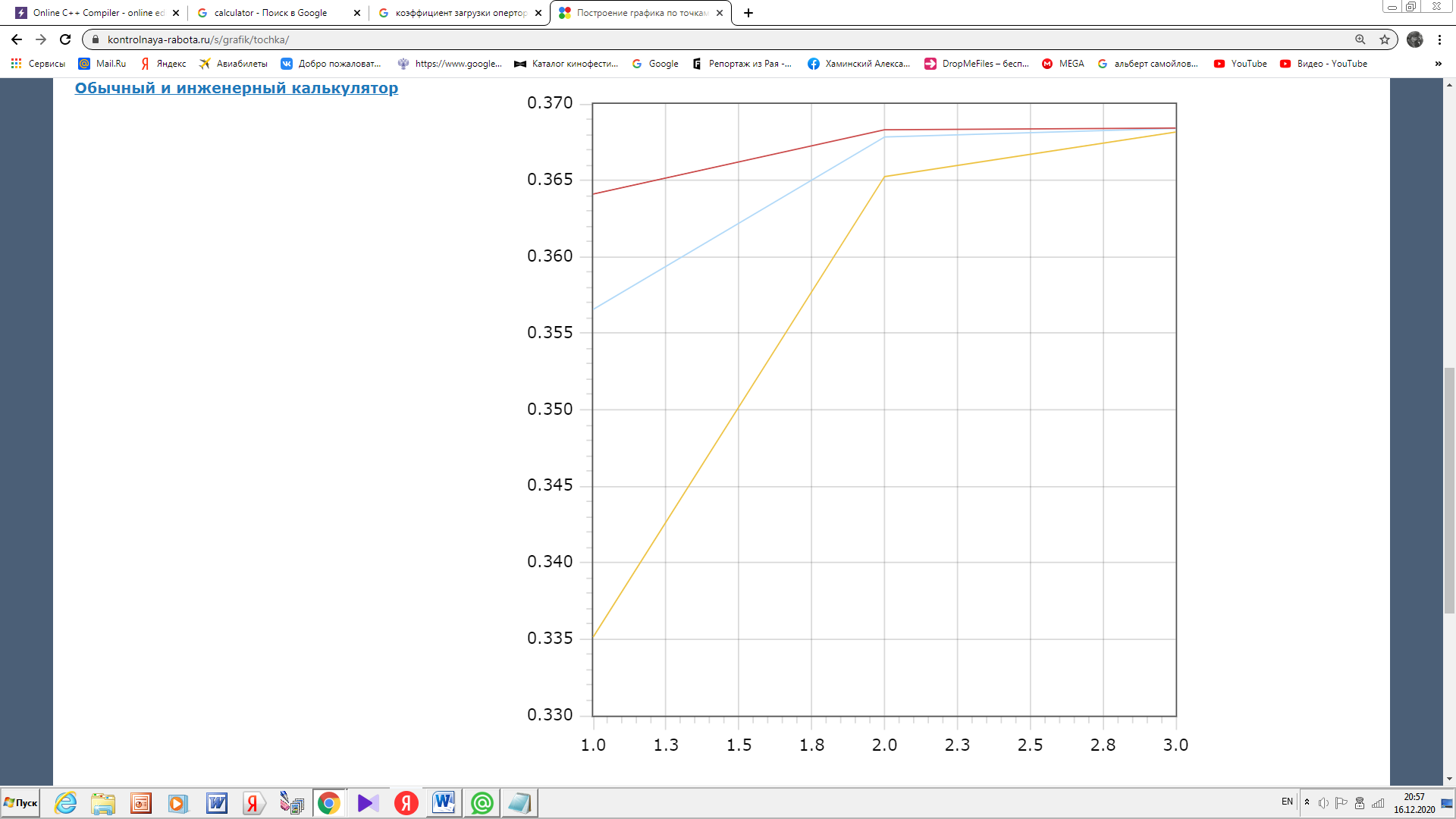
Коэффициент занятости мест в очереди: – построим график зависимости этой величины от m.



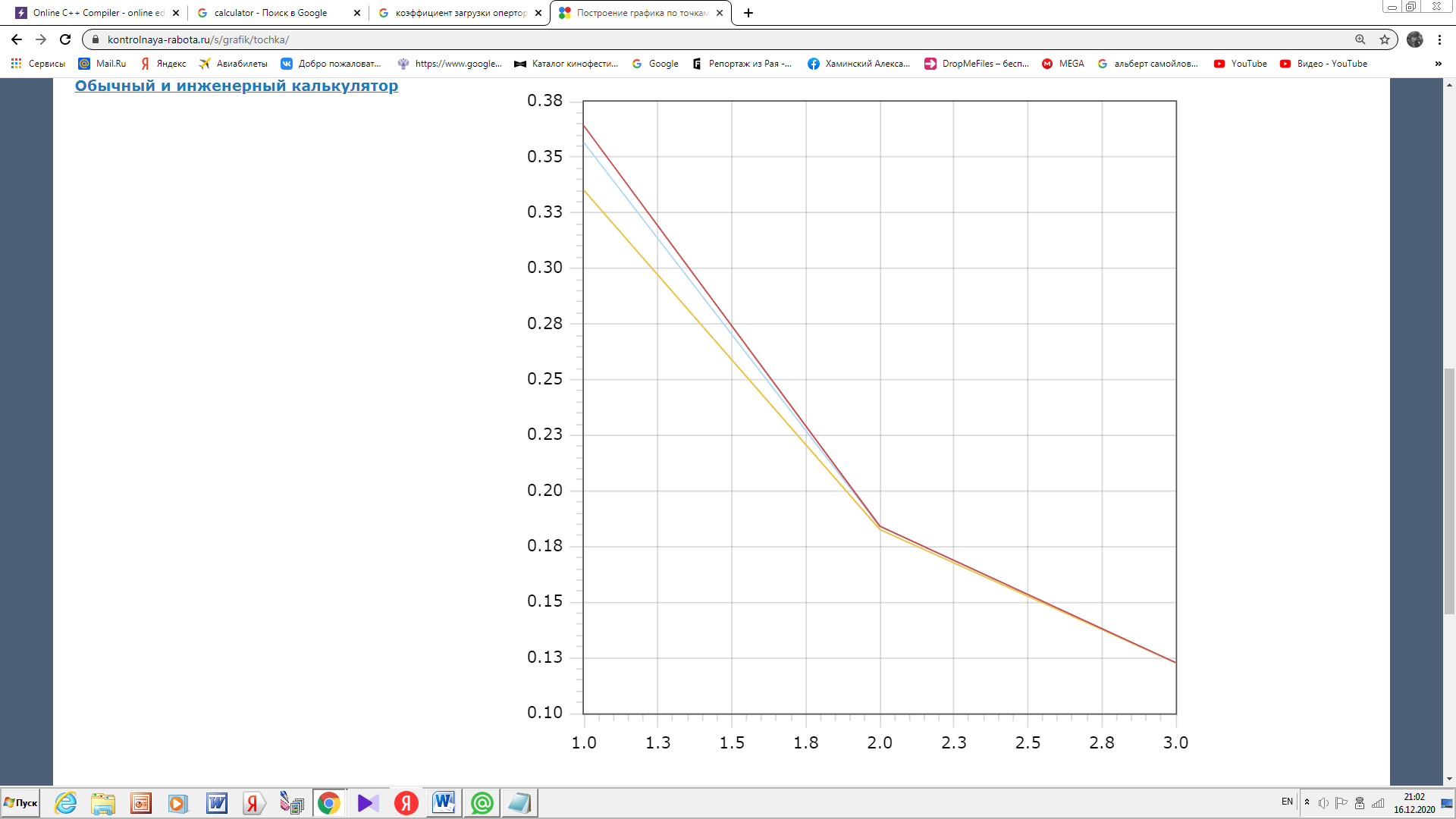
– найдём вероятность отказа. Построим график зависимости величины вероятности отказа от n для разных m.



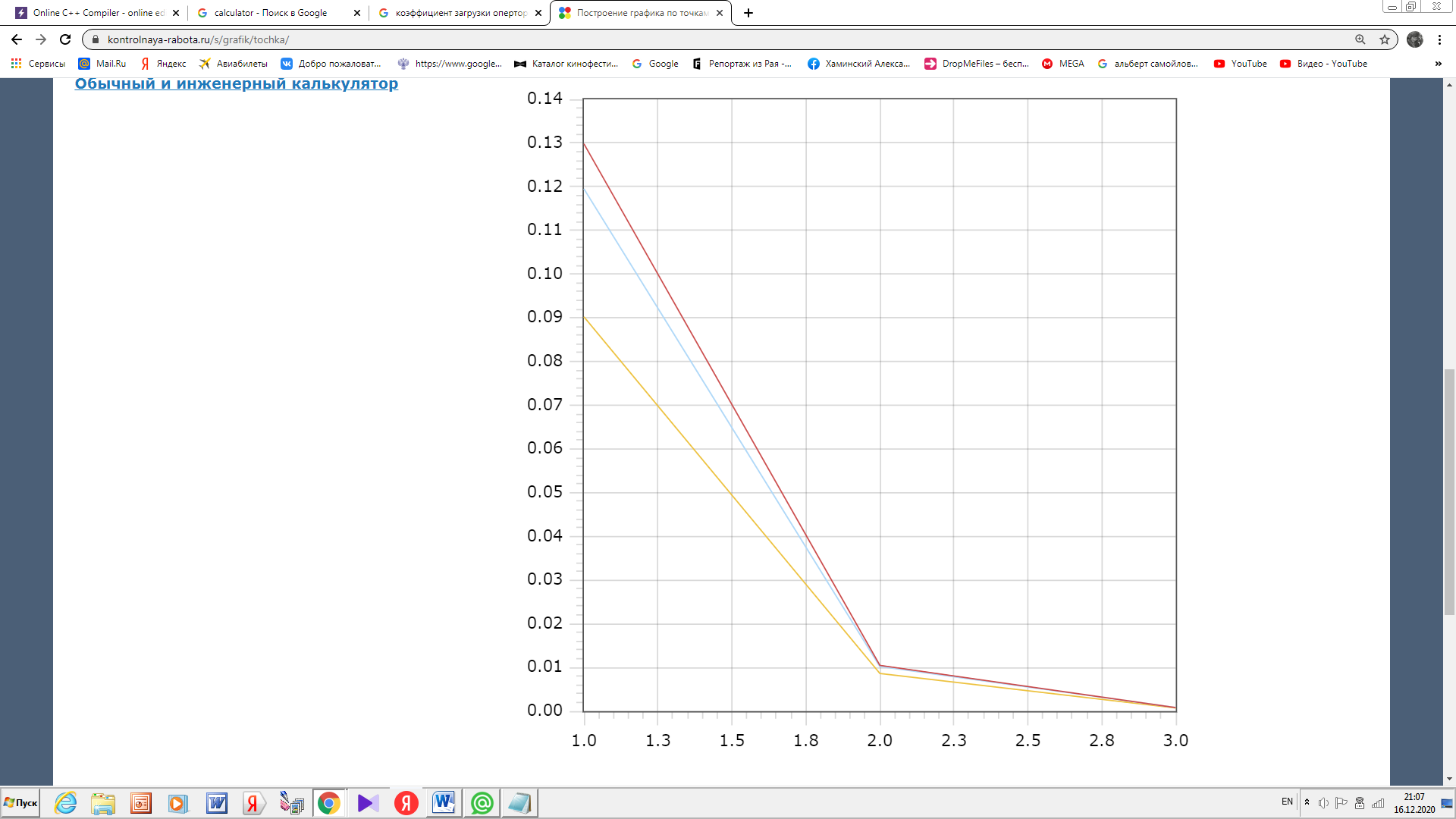
Математическое ожидание числа занятых операторов в данной задаче можно найти по формуле: . Построим график зависимости величины мат ожидания числа занятых операторов от n для разных m.



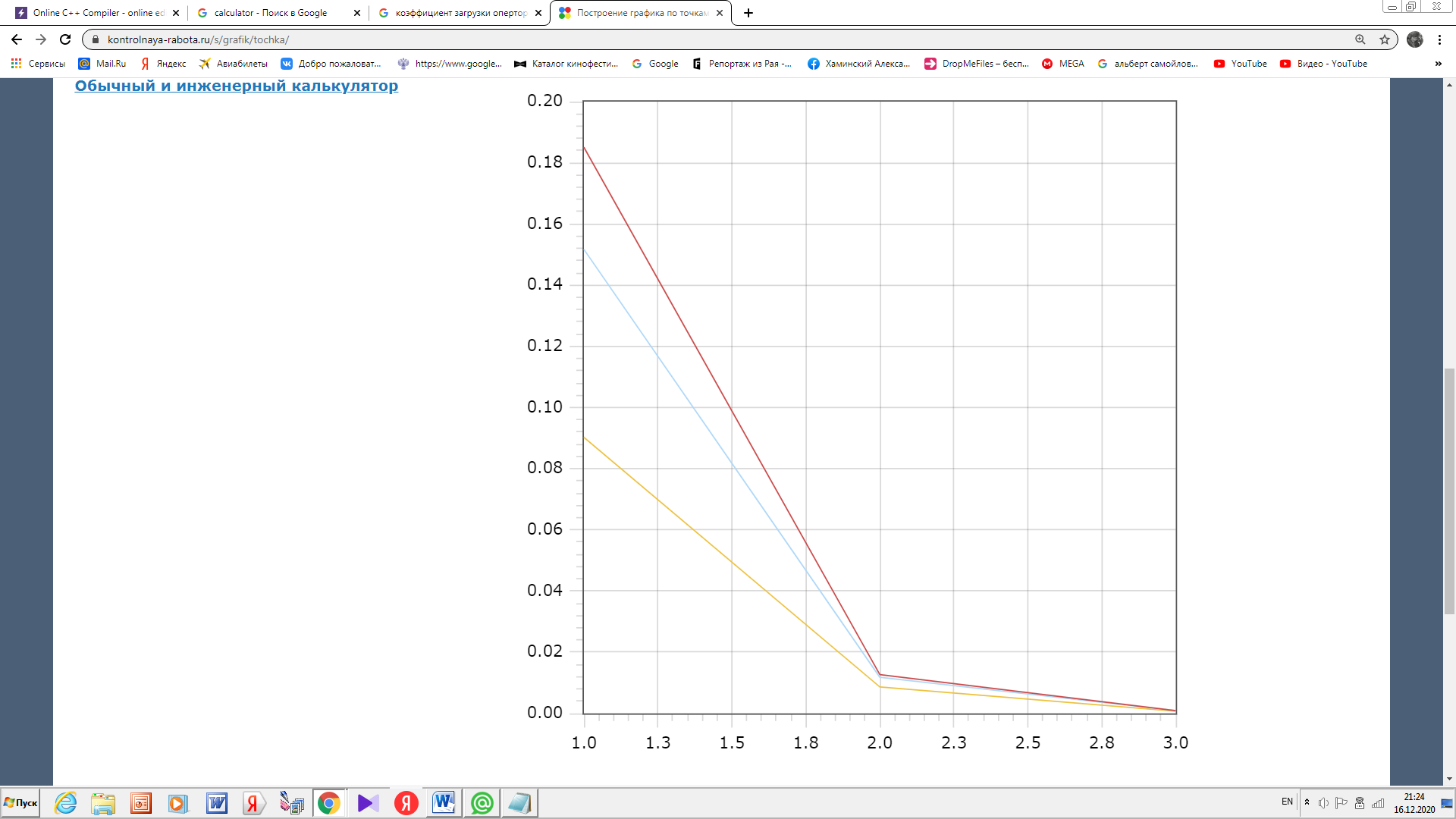
Коэффициент загрузки операторов: – построим график зависимости этой величины от n.



Вероятность существования очереди: . Построим график зависимости вероятности существования очереди от n для разных m.



Мат ожидание длины очереди: . Построим график зависимости математического ожидания длины очереди от n для разных m.



Коэффициент занятости мест в очереди: – построим график зависимости этой величины от n.



Полученные данные в табличном виде:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **n** | **m** | **Pотказ** |  | **kоператор** | **Pочередь** |  | **kочередь** |
| **1** | **1** | **0.0902394** | **0.335175** | **0.335175** | **0.0902394** | **0.0902394** | **0.0902394** |
| **1** | **2** | **0.0321764** | **0.356567** | **0.356567** | **0.119512** | **0.151689** | **0.0758443** |
| **1** | **3** | **0.0117156** | **0.364105** | **0.364105** | **0.129828** | **0.185058** | **0.061686** |
| **2** | **1** | **0.0086292** | **0.365242** | **0.182621** | **0.0086292** | **0.0086292** | **0.0086292** |
| **2** | **2** | **0.0015871** | **0.367836** | **0.183918** | **0.0102025** | **0.0117896** | **0.0058948** |
| **2** | **3** | **0.0002923** | **0.368313** | **0.184157** | **0.0104918** | **0.0126629** | **0.0042210** |
| **3** | **1** | **0.0007080** | **0.36816** | **0.12272** | **0.0007080** | **0.0007080** | **0.0007080** |
| **3** | **2** | **8.6942e-5** | **0.368389** | **0.122796** | **0.0007949** | **8.8184e-4** | **0.0004409** |
| **3** | **3** | **1.0677e-5** | **0.368417** | **0.122806** | **8.05565e-4** | **9.1386e-4** | **0.0003046** |

**Задание 3**

Рассмотреть систему без ограничений на длину очереди. Построить графики от числа операторов (вплоть до числа каналов, соответствующего 1% отказов в системе без очереди): математического ожидания числа занятых операторов; коэффициента загрузки операторов; вероятности существования очереди; математического ожидания длины очереди.

В этом задании схема будет иметь вид:

λ λ λ λ λ λ

S0 S1 … Sn Sn+1 Sn+2 ...

1μ 2μ nμ nμ nμ nμ

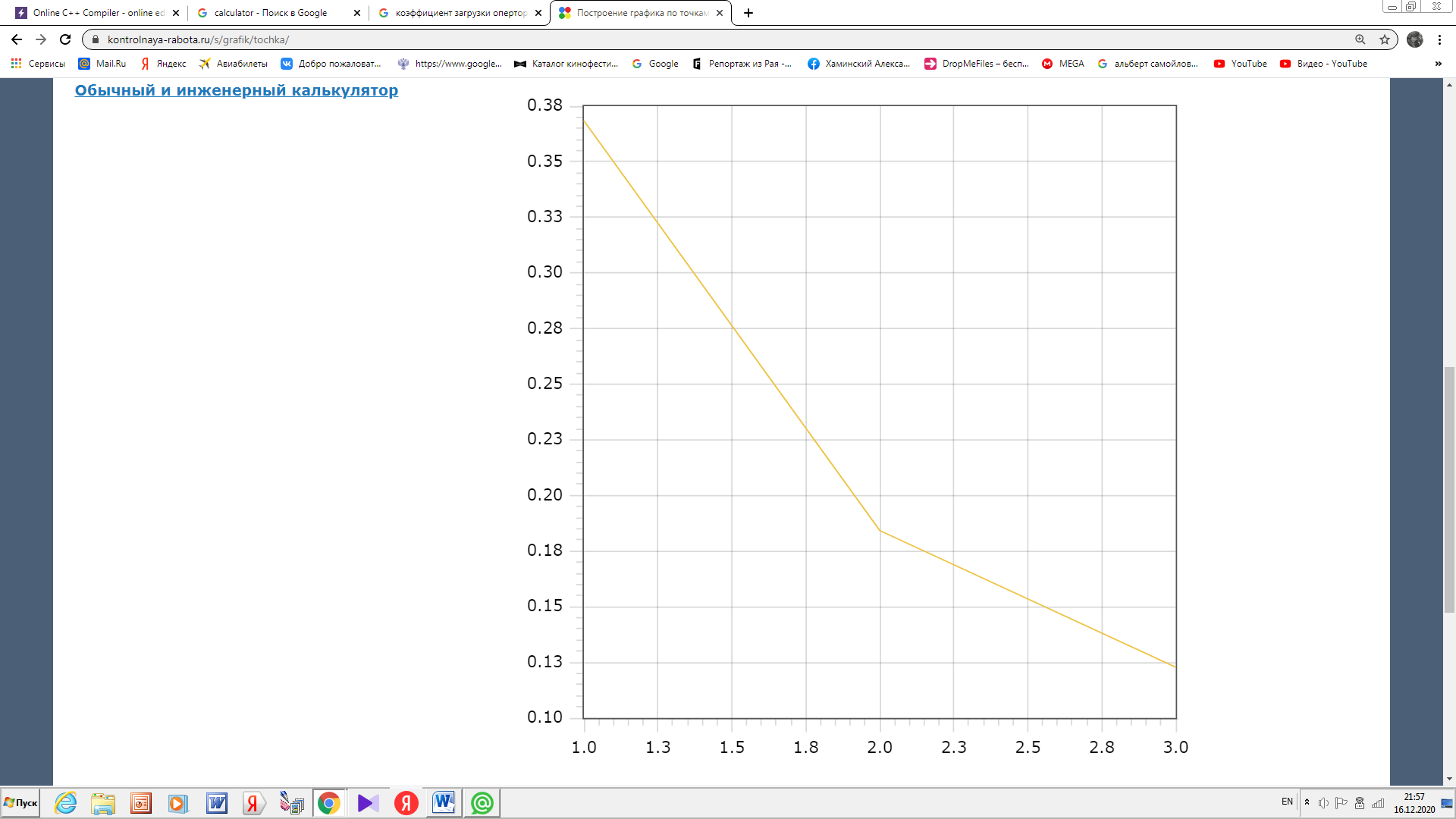
Тогда:

Исходя из того, что , найдём по формуле , где

Математическое ожидание числа занятых операторов в данной задаче можно найти по формуле: . Построим график зависимости величины мат ожидания числа занятых операторов от n.



Коэффициент загрузки операторов: – построим график зависимости этой величины от n.



Вероятность существования очереди: . Построим график зависимости вероятности существования очереди от n.



Мат ожидание длины очереди: . Построим график зависимости математического ожидания длины очереди от n.



Полученные данные в табличном виде:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **n** |  | **kоператор** | **Pочередь** |  |
| **1** | **0.368421** | **0.368421** | **0.135734** | **0.214912** |
| **2** | **0.368421** | **0.184211** | **0.0105571** | **0.012941** |
| **3** | **0.368421** | **0.122807** | **0.000807059** | **0.00092047** |

**Задание 4**

Рассмотреть систему без ограничений на длину очереди, учитывающей фактор ухода клиентов из очереди (среднее приемлемое время ожидания - Тw = R3 + G3 + B3 = 31 секунд). Построить графики от числа операторов (вплоть до числа каналов, соответствующего 1% отказов в системе без очереди): математического ожидания числа занятых операторов; коэффициента загрузки операторов; вероятности существования очереди; математического ожидания длины очереди.

В этом задании схема будет иметь вид:

λ λ λ λ λ λ

S0 S1 … Sn Sn+1 Sn+2 ...

1μ 2μ nμ nμ + ν nμ + 2ν nμ + 3 ν

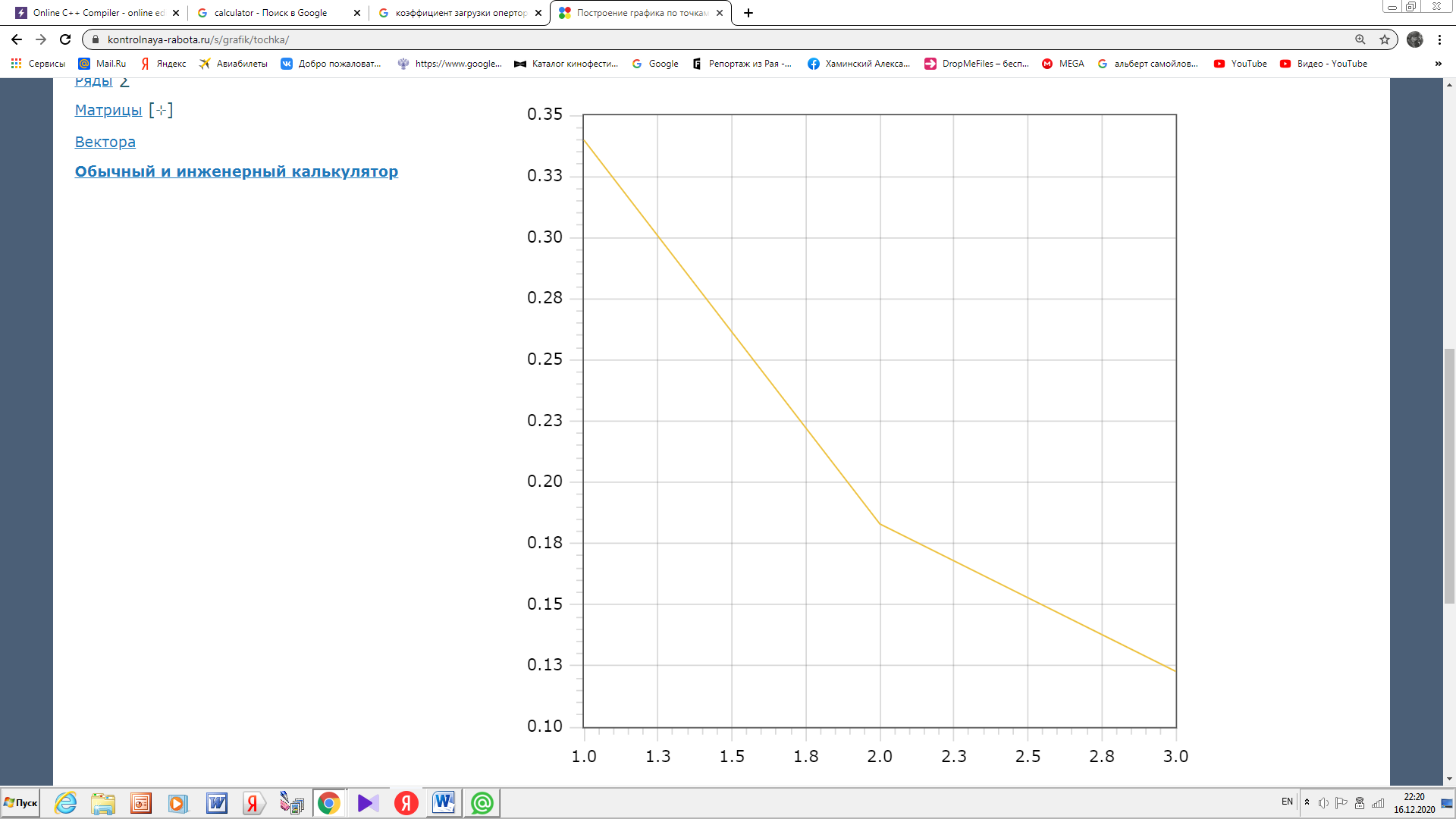
Тогда:

Согласно данному выражению, , и найти P0 возможно только вычислительным методом с определённой точностью. Будем вычислять значение P0 до тех пор, пока изменение значения на новой итерации не станет меньше 10-4.

Математическое ожидание числа занятых операторов в данной задаче можно найти по формуле: . Данные значения также можно найти только вычислительным методом. Будем вычислять полученные значения до тех пор, пока изменение значения на новой итерации не станет меньше 10-4. Построим график зависимости величины мат ожидания числа занятых операторов от n.



Коэффициент загрузки операторов: – построим график зависимости этой величины от n.



Вероятность существования очереди: . Данные значения также можно найти только вычислительным методом. Будем вычислять полученные значения до тех пор, пока изменение значения на новой итерации не станет меньше 10-4. Построим график зависимости вероятности существования очереди от n.



Мат ожидание длины очереди: . И эти значения можно найти только вычислительным методом. Будем вычислять полученные значения до тех пор, пока изменение значения на новой итерации не станет меньше 10-4. Построим график зависимости математического ожидания длины очереди от n.



Полученные данные в табличном виде:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **n** |  | **k** | **P** |  |
| **1** | **0.33981** | **0.339881** | **0.0966713** | **0.126413** |
| **2** | **0.366011** | **0.183005** | **0.00909534** | **0.0106415** |
| **3** | **0.368236** | **0.122745** | **0.000728723** | **0.000820127** |

**Код программы**

#include <iostream>

#include <cmath>

using namespace std;

long double Factorial(int a) {

long double r = 1.0L;

for(int i = 2; i <= a; ++i) {

r \*= i;

}

return r;

}

long double SumMult(int amount, long double lambda, long double mu, long double nu, int p) {

long double sum\_p = 0.0L;

long double mult\_p = 1.0L;

for(int i = 1; i <= p; ++i) {

mult\_p \*= lambda;

mult\_p /= (amount\*mu + i\*nu);

sum\_p += mult\_p;

}

return sum\_p;

}

long double SumMultExp(int amount, long double lambda, long double mu, long double nu, int p) {

long double sum\_p = 0.0L;

long double mult\_p = 1.0L;

for(int i = 1; i <= p; ++i) {

mult\_p \*= lambda;

mult\_p /= (amount\*mu + i\*nu);

sum\_p += i \* mult\_p;

}

return sum\_p;

}

long double CalculateDenominator(int amount,long double lambda, long double mu) {

long double denominator = 0.0L;

for (int i = 0; i <= amount; ++i) {

denominator += pow(lambda, i) / (Factorial(i) \* pow(mu, i));

}

return denominator;

}

long double CalculateDenominatorOchered(int amount, int ochered, long double lambda, long double mu)

{

long double denominator = CalculateDenominator(amount, lambda, mu);

long double amountFactorial = Factorial(amount);

for (int i = amount + 1; i <= ochered + amount; ++i) {

denominator += pow(lambda, i) / (amountFactorial \* pow(amount, i - amount) \* pow(mu, i));

}

return denominator;

}

long double CalculateDenominatorInfiniteOchered(int amount, long double lambda, long double mu) {

long double denominator = CalculateDenominator(amount, lambda, mu);

long double a = lambda / (amount \* mu);

denominator += (pow(lambda, amount) / (Factorial(amount) \* pow(mu, amount))) \* (a / (1-a));

return denominator;

}

long double CalculateDenominatorInfiniteOcheredWithLeaving(int amount, long double lambda, long

double mu, long double nu, int p) {

long double denominator = CalculateDenominator(amount, lambda, mu);

denominator += (pow(lambda, amount) / (Factorial(amount) \* pow(mu, amount))) \* SumMult(amount,

lambda, mu, nu, p);

return denominator;

}

void RejectWithoutWaiting(int amount, long double lambda, long double mu) {

long double denominator = CalculateDenominator(amount, lambda, mu);

long double numerator = 1.0L;

long double ro = lambda / mu;

long double p = numerator / denominator;

long double matexpecting = 0.0L;

cout << "P0 = " << p << endl;

for (int i = 1; i <= amount; ++i) {

p \*= ro;

p /= i;

cout << "P" << i << " = " << p << endl;

matexpecting += i \* p;

}

cout << "Math expecting is " << matexpecting << endl;

long double n\_occuped = ro \* (1-p);

long double k\_occuped = n\_occuped / amount;

cout << "Coefficient occuped is " << k\_occuped;

}

void RejectWithWaiting(int amount, int ochered, long double lambda, long double mu) {

long double denominator = CalculateDenominatorOchered(amount, ochered, lambda, mu);

long double numerator = 1.0L;

long double ro = lambda / mu;

long double p = numerator / denominator;

long double matexpecting = 0.0L;

cout << "P0 = " << p << endl;

for (int i = 1; i <= amount; ++i) {

p \*= ro;

p /= i;

cout << "P" << i << " = " << p << endl;

matexpecting += i \* p;

}

long double p\_och = 0.0L;

long double matexpectingOchered = 0.0L;

for (int i = 1; i <= ochered; ++i) {

p \*= ro;

p /= amount;

p\_och += p;

cout << "P" << amount << "+" << i << " = " << p << endl;

matexpecting += amount \* p;

matexpectingOchered += i \* p;

}

cout << "Probability for waiting is " << p\_och << endl;

cout << "Math expecting is " << matexpecting << endl;

cout << "Math expecting for waiting is " << matexpectingOchered << endl;

long double n\_occuped = ro \* (1-p);

long double k\_occuped = n\_occuped / amount;

cout << "Coefficient occuped is " << k\_occuped << endl;

long double n\_occuped\_och = ro \* (1-p\_och);

long double k\_occuped\_och = n\_occuped\_och / (amount + ochered);

cout << "Coefficient occuped for waiting is " << k\_occuped\_och<< endl;

}

void RejectWithInfiniteWaiting(int amount, long double lambda, long double mu) {

long double denominator = CalculateDenominatorInfiniteOchered(amount, lambda, mu);

long double numerator = 1.0L;

long double ro = lambda / mu;

long double a = lambda / (amount \* mu);

long double p0 = numerator / denominator;

long double p = p0;

long double matexpecting = 0.0L;

cout << "P0 = " << p0 << endl;

for (int i = 1; i <= amount; ++i) {

p \*= ro;

p /= i;

cout << "P" << i << " = " << p << endl;

matexpecting += i \* p;

}

matexpecting += amount \* p \* (a/(1.0L-a));

long double p\_och = p \* (a/(1.0L-a));

long double matexpectingOchered = p \* a \* (1.0L/pow(1.0L - a, 2));

cout << "Probability for waiting is " << p\_och << endl;

cout << "Math expecting is " << matexpecting << endl;

cout << "Math expecting for waiting is " << matexpectingOchered << endl;

/\*long double n\_occuped = ro \* (1-p);

long double k\_occuped = n\_occuped / amount;

cout << "Coefficient occuped is " << k\_occuped << endl;

long double n\_occuped\_och = ro \* (1-p\_och);

long double k\_occuped\_och = n\_occuped\_och / (amount + ochered);

cout << "Coefficient occuped for waiting is " << k\_occuped\_och<< endl;\*/

}

void RejectWithInfiniteWaitingWithLeaving(int amount, long double lambda, long double mu, long double

nu) {

long double denominatorPrev = CalculateDenominatorInfiniteOcheredWithLeaving(amount, lambda, mu,

nu, 0);

long double denominator = CalculateDenominatorInfiniteOcheredWithLeaving(amount, lambda, mu, nu,

1);

long double numerator = 1.0L;

long double ro = lambda / mu;

int i = 2;

long double p0\_prev = numerator / denominatorPrev;

long double p0 = numerator / denominator;

while (abs(p0\_prev - p0) >= 0.0001L) {

denominator = CalculateDenominatorInfiniteOcheredWithLeaving(amount, lambda, mu, nu, i);

++i;

p0\_prev = p0;

p0 = numerator / denominator;

}

long double p = p0;

long double matexpecting\_const = 0.0L;

cout << "P0 = " << p0 << endl;

for (int i = 1; i <= amount; ++i) {

p \*= ro;

p /= i;

cout << "P" << i << " = " << p << endl;

matexpecting\_const += i \* p;

}

long double matexpecting\_prev = matexpecting\_const;

long double matexpecting = matexpecting\_const;

matexpecting\_prev += amount \* p \* SumMult(amount, lambda, mu, nu, 1);

matexpecting += amount \* p \* SumMult(amount, lambda, mu, nu, 2);

i = 3;

while (abs(matexpecting - matexpecting\_prev) >= 0.0001L) {

matexpecting\_prev = matexpecting;

matexpecting = matexpecting\_const;

matexpecting += amount \* p \* SumMult(amount, lambda, mu, nu, i);

++i;

}

i = 3;

long double p\_och\_prev = p \* SumMult(amount, lambda, mu, nu, 1);

long double p\_och = p \* SumMult(amount, lambda, mu, nu, 2);

while (abs(p\_och - p\_och\_prev) >= 0.0001L) {

p\_och\_prev = p\_och;

p\_och = p \* SumMult(amount, lambda, mu, nu, i);

++i;

}

i = 3;

long double matexpectingOchered\_prev = p \* SumMultExp(amount, lambda, mu, nu, 1);

long double matexpectingOchered = p \* SumMultExp(amount, lambda, mu, nu, 2);

while (abs(matexpectingOchered - matexpectingOchered\_prev) >= 0.0001L) {

matexpectingOchered\_prev = matexpectingOchered;

matexpectingOchered = p \* SumMultExp(amount, lambda, mu, nu, i);

++i;

}

cout << "Probability for waiting is " << p\_och << endl;

cout << "Math expecting is " << matexpecting << endl;

cout << "Math expecting for waiting is " << matexpectingOchered << endl;

/\*long double n\_occuped = ro \* (1-p);

long double k\_occuped = n\_occuped / amount;

cout << "Coefficient occuped is " << k\_occuped << endl;

long double n\_occuped\_och = ro \* (1-p\_och);

long double k\_occuped\_och = n\_occuped\_och / (amount + ochered);

cout << "Coefficient occuped for waiting is " << k\_occuped\_och<< endl;\*/

}

int main(int argc, char \*\*argv)

{

if (argc < 3) { return -1; }

int amount = atoi(argv[1]);

//int amountOchered = atoi(argv[4]);

long double lambda = 1.0L / atoi(argv[2]);

long double mu = 1.0L / atoi(argv[3]);

long double nu = 1.0L / atoi(argv[4]);

RejectWithInfiniteWaitingWithLeaving(amount, lambda, mu, nu);

//RejectWithInfiniteWaiting(amount, lambda, mu);

//RejectWithWaiting(amount, amountOchered, lambda, mu);

//RejectWithoutWaiting(amount, lambda, mu);

return 0;

}