МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА ПРОГРАМУВАННЯ ТА МАТЕМАТИКИ

Пояснювальна записка

до дипломної роботи

_____ бакалавр _____

(освітньо-кваліфікаційний рівень)

на тему «Комп'ютерне моделювання системи масового обслуговування»

Виконав: студент 4 курсу, групи ІТ-651
напряму підготовки 6.040302 „ Програмна інженерія ”

_____ Вертела Д.І.
(підпис)

Керівник,
доцент, к. т.н._____ Іванов В.Г.
(підпис)

Рецензент,
Ст. викладач _____ Батурін О.І.
(підпис)

СЄВЄРОДОНЕЦЬК
2019 року

СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет інформаційних технологій та електроніки

Кафедра програмування та математики

Освітньо-кваліфікаційний рівень бакалавр

Напрям підготовки 6. 050103 „ Програмна інженерія ”

Спеціальність 7.050103 „ Програмна інженерія ”

ЗАТВЕРДЖУЮ
Завідувач кафедри ПМ,
д.т.н., доцент
_____Лифар В.О.
«___» _____ 2019
р.

З А В Д А Н Н Я

НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

ВЕРТЕЛА ДМИТРО ІВАНОВИЧ

1. Тема роботи Комп'ютерне моделювання системи масового обслуговування

керівник роботи доцент Іванов Віталій Геннадьович

2. Строк подання студентом роботи 06 червня 2019 р.

3. Вихідні дані до роботи

Об'єктом даної роботи є комп’ютерне моделювання систем масового обслуговування.

3.1 Літературні джерела:

Саати Т. Л. Элементы теории обслуживания. М.: Советское радио , 1971.

Вентцель Е.С. Исследование операций . М.: Наука, 1988.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

4.1 Вступ

4.2 Аналіз предметної галузі (огляд літератури), з висвітленням наступних питань:

Системи масового обслуговування.

Імітаційне моделювання системи масового обслуговування.

4.3 Основна частина, в якої висвітлити:

Інформаційна модель об’єкту.

Програмну реалізацію моделі.

4.4 Висновки

4.5 Перелік використаних джерел

5. Перелік графічного матеріалу немає

6. Дата видачі завдання 02 лютого 2019 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1	Одержання завдання на виконання роботи	01.02.19	
2	Укладання і погодження з керівником плану і етапів виконання роботи	20.02.19	
3	Узагальнення даних літературних джерел, укладання розділу «Аналіз предметної галузі»	1.03.19	
4	Аналіз шляхів виконання завдання. Вибір і погодження з керівником оптимального шляху	10.03.19	
5	Проектування інфологічної моделі задачі що реалізується.	01.04.19	
6	Укладання та тестування програмного продукту	20.04.19	
7	Укладання, оформлення та погодження пояснювальної записки з керівником	15.05.19	
8	Здача готової пояснювальної записки на кафедру	06.06.19	
9	Укладання доповіді і презентації	10.06.19	

Студент _____ Вертела Д.І.

(підпис)

Керівник роботи _____ Іванов В.Г.

(підпис)

ЛИСТ ПОГОДЖЕННЯ І ОЦІНЮВАННЯ

дипломної роботи студента гр. ІТ-651 Вертела Д.І.

Науковий керівник

Доцент, к.т.н. _____ Іванов В.Г.

Оцінка наукового керівника: _____

Рецензент Ст. викладач каф. ПМ СНУ ім.В.Даля Батурін О.І.

місто роботи, посада, ПІБ

Оцінка рецензента: _____

Кінцева оцінка за результатами захисту:

_____

Голова ЕК

_____ Лифар В.О.

підпис

РЕФЕРАТ

Текст – 68, рис. –12, табл. – 4, літературних джерел – 10

У ході виконання даної дипломної роботи було проведено дослідження предметної області, вивчені основні джерела й особливості розробки моделі системи масового обслуговування.

Відповідно до мети і задачі, дана робота складається із вступу, теоретичної частини, практичної частини, висновків і списку літератури. В теоретичній частині розглядається загальна теорія дипломної роботи, в практичній частині – розроблена програма, що реалізує модель роботи систем масового обслуговування з використанням методів розв’язку, приведених в теоретичній частині

Одержана програма може бути використані для оцінки ефективності роботи систем масового обслуговування.

Ключові слова: СМО, КОМП'ЮТЕРНА МОДЕЛЬ

ЗМІСТ

ВСТУП

Актуальність досліджень. Одним з видів моделей за допомогою яких можна описати фінансово економічні процеси, є системи масового обслуговування. Ними можна описати функціонування багатьох підприємств банків, кредитних установ страхових організацій, податкових інспекцій, закладів сфери обслуговування (магазинів, лікарень тощо) діяльність яких пов'язана з багатократною реалізацією виконання якихось однотипних завдань і операцій

Основи знань про черги, іноді звані теорією черг або теорією масового обслуговування, складають важливу частину теорії управління виробництвом. Черги — звичайне явище. Вони можуть носити форму очікування ремонту автомобіля в центрі автосервісу або очікування студентами консультації у професора. Моделі черг (як і лінійне програмування, моделі управління запасами, методи мережевого аналізу проектів) використовуються і у сфері управління матеріальним виробництвом, і у сфері обслуговування. Аналіз черг в термінах довжини черги, середнього часу очікування, середнього часу обслуговування і інших чинників допомагає нам краще зрозуміти принципи організації системи обслуговування.

Об’єкт досліджень: Алгоритми розв’язання роботи систем масового обслуговування.

Предмет досліджень: Побудова демонстраційної програми за допомогою будь-якого середовища візуального програмування, який був би доступний у використанні користувачу-початківцю.

Мета дослідження: Аналіз систем масового обслуговування, їх собливостей, а також вивчення методів розв’язку систем масового обслуговування і практичне їх застосування.

Завдання дослідження:

а) скласти інформаційну модель;

б) провести детальний аналіз систем масового обслуговування , а також практично застосувати методи їх розв’язку;

в) скласти програму моделі системи масового обслуговування;

Методологічна та теоретична основа дослідження: методи і алгоритми розв’язання роботи систем масового обслуговування

Методи дослідження: програмування за допомогою будь-якого програмного середовища Delphi.

Практичне значення отриманих результатів. Одержана програма може бути використані у навчальному процесі при перевірки та демонстрації роботи систем масового обслуговування.

1 АНАЛІТИЧНИЙ ОГЛЯД

1.1 Основні поняття. Класифікація систем масового обслуговування (СМО) [1]

При дослідженні операцій часто доводиться стикатися з системами, призначеними для багаторазового використання при рішенні однотипних задач. Виникаючі при цьому процеси одержали назву процесів обслуговування, а системи - систем масового обслуговування (СМО). Прикладами таких систем є телефонні системи, ремонтні майстерні, обчислювальні комплекси, квиткові каси, магазини, перукарні і т.п.

Кожна СМО складається з певного числа обслуговуючих одиниць (приладів, пристроїв, пунктів, станцій), які називатимемо каналами обслуговування. Каналами можуть бути лінії зв'язку, робочі крапки, обчислювальні машини, продавці і ін. По числу каналів СМО підрозділяють на одноканальні і багатоканальні.

Заявки поступають в СМО звичайно не регулярно, а випадково, утворюючи так званий випадковий потік заявок (вимог). Обслуговування заявок, взагалі кажучи, також продовжується якийсь випадковий час. Випадковий характер потоку заявок і часу обслуговування призводить до того, що СМО виявляється завантаженою нерівномірно: у якісь періоди часу скуплюється дуже велика кількість заявок (вони або стають в чергу, або покидають СМО не обслуженими), в інші ж періоди СМО працює з недовантаженням або простоює.

Предметом теорії масового обслуговування є побудова математичних моделей, що пов'язують задані умови роботи СМО (число каналів, їх продуктивність, характер потоку заявок і т.п.) з показниками ефективності СМО, що описують її здатність справлятися з потоком заявок.

Як показники ефективності СМО використовуються: середнє число заявок, що обслуговуються в одиницю часу; середнє число заявок в черзі; середній час очікування обслуговування; вірогідність відмови в обслуговуванні без очікування; вірогідність того, що число заявок в черзі перевищить певне значення і т.п.

СМО ділять на два основні типи (класу): СМО з відмовами і СМО з очікуванням (чергою). У СМО з відмовами заявка, що поступила в мить, коли всі канали зайняті, дістає відмову, покидає СМО і надалі процесі обслуговування не бере участь (наприклад, заявка на телефонну розмову в мить, коли всі канали зайняті, дістає відмову і покидає СМО не обслуженої). У СМО з очікуванням заявка, що прийшла в мить, коли всі канали зайняті, не йде, а стає в чергу на обслуговування.

СМО з очікуванням підрозділяються на різні види залежно від того, яка організована черга: з обмеженою або необмеженою довжиною черги, з обмеженим часом очікування і т.п.

Для класифікації СМО важливе значення має дисципліна обслуговування, що визначає порядок вибору заявок з числа тих, що поступили і порядок розподілу їх між вільними каналами. За цією ознакою обслуговування заявки може бути організовано за принципом "перша прийшла - перша обслужена", "остання прийшла - перша обслужена" (такий порядок може застосовуватися, наприклад, при витяганні для обслуговування виробів з складу, бо останні з них виявляються часто доступнішими) або обслуговування з пріоритетом (коли в першу чергу обслуговуються найбільш важливі заявки). Пріоритет може бути як абсолютним, коли важливіша заявка "витісняє" з-під обслуговування звичайну заявку (наприклад, у разі аварійної ситуації планові роботи ремонтних бригад уриваються до ліквідації аварії), так і відносним, коли важливіша заявка одержує лише "краще" місце в черзі.

1.2 Поняття марківського випадкового процесу

Процес роботи СМО є випадковим процесом [1-2].

Під випадковим (імовірнісним або стохастичним) процесом розуміється процес зміни в часі стану якої-небудь системи відповідно до імовірнісних закономірностей.

Процес називається процесом з дискретними станами, якщо його можливі стани S1, S2, S3... можна наперед перерахувати, а перехід системи із стану в стан відбувається миттєво (стрибком). Процес називається процесом з безперервним часом, якщо моменти можливих переходів системи із стану в стан не фіксовані наперед, а випадкові.

Процес роботи СМО є випадковим процесом з дискретними станами і безперервним часом. Це означає, що стан СМО міняється стрибком у випадкові моменти появи якихось подій (наприклад, приходу нової заявки, закінчення обслуговування і т.п.).

Математичний аналіз роботи СМО істотно спрощується, якщо процес цієї роботи - марківський. Випадковий процес називається марківським або випадковим процесом без наслідку, якщо для будь-якого моменту часу t0 імовірнісні характеристики процесу в майбутньому залежать тільки від його стану в даний момент t_0 і не залежать від того, коли і як система прийшла в цей стан.

Приклад марківського процесу: система S - лічильник у таксі. Стан системи у момент t характеризується числом кілометрів (десятих доль кілометрів), пройдених автомобілем до даного моменту. Хай у момент t лічильник показує S_0. Вірогідність того, що у момент $t > t_0$ лічильник покаже те або інше число кілометрів (точніше, відповідне число рублів) S_1, залежить від S_0, але не залежить від того, в які моменти часу змінювалися свідчення лічильника до моменту t_0. Багато процесів можна приблизно вважати марківськими. Наприклад, процес гри в шахи; система S - група шахових

фігур. Стан системи характеризується числом фігур супротивника, що збереглися на дошці в момент t_0. Вірогідність того, що у момент t > t0 матеріальна перевага буде на стороні одного з супротивників, залежить в першу чергу від того, в якому стані знаходиться система в даний момент t0 а не того, коли і в якій послідовності зникли фігури з дошки до моменту t0.

У ряді випадків передісторією даних процесів можна просто нехтувати і застосовувати для їх вивчення марківські моделі.

При аналізі випадкових процесів з дискретними станами зручно користуватися геометричною схемою - так званим графом станів. Звичайно стани системи зображаються прямокутниками, а можливі переходи із стану в стан - стрілками (орієнтованими дугами), що сполучають стани.

1.3 Рівняння для аналізу СМО [3-6]

1.3.1 Основні рівняння СМО

Закон Бернуллі. В основі аналізу СМО лежить біноміальний закон Бернуллі, який дозволяє розраховувати ймовірність появи події "А" точно К разів при п незалежних іспитах (тільки для дискретних випадкових взаємно несумісних незалежних подій):

$$P_n(K) = C_n^K p^K q^{n-K} = \left(\frac{n!}{K!(n-K)!} \right) p^K q^{n-K}, \qquad (1.1)$$

$$C_n^K = \frac{n!}{K!(n-K)!}; \qquad (1.2)$$

де п - кількість незалежних іспитів; P_n(К) - ймовірність появи подій "А" точно К разів при п незалежних іспитах; р - ймовірність появи однієї події "А"; q =1- р — ймовірність протилежної події (не появи події "А"); К —

кількість появи події "А" при и спостереженнях; C_n^K - сполучення по К елементів із п спостережень.

У принципі закон Бернуллі (і закони Лапласа та Пуассона, що з нього випливають) може використовуватись для визначення конструкції СМО - кількості каналів обслуговування та середнього строку обслуговування однієї вимоги.

Якщо прийняти К = 0, 1,2 ..., n, то отримаємо повну групу взаємно несумісних подій, для якої сума відповідних ймовірностей

$$\sum_{k=0}^{n} P_n(K) = 1. \qquad (1.3)$$

Формула Стірлінга. Формула Бернуллі дуже складна для обчислення, тому використовується формула Стерлінга

$$m! = \sqrt{2\pi m} \cdot m^m e^{-m} \qquad (1.4)$$

у якій точність розрахунків збільшується при $n \to \infty$.

Формула Лапласа. Подальші перетворення формули Бернуллі дозволяють отримати формулу Лапласа

$$P_n(K) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{\delta^2 K}{2\sigma^2}}, \qquad (1.5)$$

де $\sigma = \sqrt{npq}; \delta_K = K - (n+1)p \approx K - np$

Із формули Лапласа випливає, що при $\delta_K = 0$ або при К= $К_о$ (тут $К_о$ - найімовірніша кількість подій "А") отримуємо

$$\delta_K = К_0 - пр = 0;\ К_0 = пр \qquad (1.6)$$

і ймовірність найімовірнішого числа подій Ко дорівнює

$$P_n(K_0) = \frac{1}{\sqrt{2\pi\sigma^2}} \tag{1.7}$$

З точністю до 3σ (тобто з ймовірністю 0,997) можна вважати, що всі події відбудуться, якщо виконується умова К — $К_о \le 3\sigma$, або $K \le np + 3\sqrt{npq}$

Формула Пуассона. При великій кількості незалежних іспитів (n » 1), малій ймовірності одного з іспитів $p = \lambda/n$ (тут $\lambda = const$) формула Лапласа дає помилки, і тому використовують асимтотичну формулу Пуассона

$$P_n(K) = \frac{\lambda^K e^{-\lambda}}{K!}, \tag{1.8}$$

де $\lambda = pn = const$.

Ймовірність появи події "А" не більше за т разів

$$P\{K \le m\} = \sum_{K=0}^{m} \frac{\lambda^K e^{-\lambda}}{K!} \tag{1.9}$$

Ймовірність того, що подія "А" при п іспитах зовсім не з'являється $P_n(0) = e^{-\lambda}$.

Формула Пуассона використовується для розрахунку різних подій. При цьому вважається, що

$$\lambda = pn = \frac{vt}{n_T} = \frac{vt}{n}, \tag{1.10}$$

де *v* — середня кількість телефонних дзвінків за одиницю часу; *v* — середи відмов пристрою за одиницю часу; *v* — середня кількість розпаду

радіоактивних атомів за одиницю часу; ν — середня кількість електронів, яка попадає на анод за одиницю часу; ν - середня кількість зміни знаку при телеграфних повідомленнях за одиницю часу; $п_т = \nu T = п$ - число подій за досить великий період часу T .

Ймовірність появи одного телефонного дзвінка, однієї відмови пристрою і т.п.

$$p = \frac{t}{T} = \frac{t}{\frac{n_T}{\nu}} = \frac{\nu t}{n_T} = \frac{\nu t}{n} \tag{1.11}$$

Постійна величина формули Пуассона $\lambda = pn = \frac{\nu t n}{n} = \nu t$ і формула Пуассона набуває вигляду

$$P(K, \nu t) = \frac{(\nu t)^K e^{-\nu t}}{K!} \ . \tag{1.12}$$

1.3.2 Диференційні рівняння СМО

Стан S_i СМО визначається:

в одноканальній СМО з очікуванням - довжиною черги і;

у багатоканальній СМО з відмовами - кількістю зайнятих каналів і(у цій СМО черги немає: якщо всі канали зайняті то вимога не обслуговується і зникає);

у багатоканальній СМО з очікуванням - числом зайнятих каналів плюс довжиною черги.

Розглянемо досить велику кількість N таких ідентичних СМО. Позначимо через Рі ймовірність того, що СМО знаходиться на час t у стані S_i, а через ΔP_i - зміну ймовірності P_i за малий проміжок часу Δt.

У СМО вважається, що потік вимог підкоряється пуассонівському закону. Тому за малий термін Δt на жодну з N СМО не може надійти більше однієї вимоги, та жодна з N СМО за цей час не може обслужити більше за одну вимогу. Ймовірністю надходження або закінчення обслуговування двох або більшої кількості вимог за час Δt у N СМО можна знехтувати, бо ця ймовірність дуже мала (дорівнює нулю). Тобто СМО переходить із стану S_i, у стан S_{i+1} або стан S_{i-1}. СМО не може перейти, наприклад, із стану S_i, у стани S_{i+2}, або S_{i-2}

Тоді згідно з законом Пуассона (ймовірність появи точно k подій із п можливих за проміжок часу t) при умовах k= 1, t = Δt ймовірність появи точно однієї події дорівнює

$$Pn^{(k)} = \left[\frac{(\lambda t)^k e^{-\lambda t}}{k!} \right] = \frac{(\lambda \Delta t)^1 e^{-\lambda \Delta t}}{1!} = \lambda \Delta t \qquad (1.13)$$

Таким чином, ймовірність надходження точно однієї події дорівнює

$$P_{надх} = \lambda \Delta t, \qquad (1.14)$$

де λ - інтенсивність надходження вимог.

Одночасно вважається, що обслуговування теж підкоряється експоненціальному закону і тому ймовірність завершення обслуговування точно однієї події $P_{обслуг} = \mu \Delta t$, де μ - інтенсивність обслуговування

РОЗДІЛ 2 ІМІТАЦІЙНА МОДЕЛЬ

2.1. Основи імітаційного моделювання систем масового обслуговування

Із системами масового обслуговування (СМО) ми зустрічаємось повсякчас. Кожному з нас доводилось чекати обслуговування в черзі (у магазині, на автозаправці, в бібліотеці, кав'ярні тощо). Аналогічні ситуації виникають, коли треба скористатися телефонним зв'язком або виконати свою програму на комп'ютері. Будь-яке виробництво теж можна уявити як послідовність систем обслуговування. До типових систем обслуговування належать також ремонтні і медичні служби, транспортні системи, аеропорти, вокзали тощо. Особливого значення набули такі системи у процесах інформатики. Це передусім комп'ютерні системи, мережі передавання інформації, операційні системи, бази і банки даних. Системи обслуговування відіграють значну роль у повсякденному житті. Досвід моделювання різних типів дискретних систем свідчить про те, що приблизно 80% цих моделей ґрунтуються на СМО [7]. Систему масового обслуговування загалом можна уявити як сукупність послідовно пов'язаних між собою вхідних потоків вимог на обслуговування (потоків замовлень), черг, каналів обслуговування і потоків обслужених замовлень. Будь-який пристрій, який безпосередньо обслуговує замовлення, називають каналом обслуговування. Системи масового обслуговування за наявності тої чи іншої ознаки можна класифікувати так [8]:

1. За характером надходження замовлень у систему: на системи з регулярним і випадковим потоками замовлень. Якщо кількість замовлень, які надходять у ситему за одиницю часу (інтенсивність потоку), стала або є заданою функцією часу, то маємо систему з регулярним потоком замовлень, в іншому разі – з випадковим. Випадковий потік замовлень може бути

стаціонарним або нестаціонарним. Якщо параметри потоку замовлень не залежать від розташування інтервалу часу, який розглядають, на осі часу, то маємо стаціонарний потік замовлень, в протилежному випадку – нестаціонарний. Наприклад, якщо кількість покупців, які приходять до магазину, не залежить від часу доби, то потік замовлень (покупців) – стаціонарний.

2. За кількістю замовлень, які надходять за одиницю часу: на системи з ординарним і неординарним потоками замовлень. Якщо ймовірність надходження двох або більше замовлень в один момент часу дорівнює нулеві або настільки мала, що нею можна знехтувати, то маємо систему з ординарним потоком замовлень. Наприклад, потік літаків, які прибувають на злітну смугу аеродрому (ЗС), можна вважати ординарним, оскільки ймовірність надходження двох і більше літаків до каналу обслуговування (ЗС) в один і той самий момент часу дуже мала.

3. За зв’язком між замовленнями: на системи без післядії від замовлень, які надійшли, і з післядією. Якщо ймовірність надходження замовлень у систему в деякий момент часу не залежить від того, скільки вимог уже надійшло до системи, тобто не залежить від передісторії процесу, який вивчають, то ми маємо задачу без післядії, у протилежному випадку – з післядією. Прикладом задачі з післядією може слугувати потік студентів на складання заліку викладачеві.

4. За характером поведінки замовлень у системі: з відмовами, з обмеженим очікуванням і з очікуванням без обмеження: – якщо нове замовлення, яке прибуло на обслуговування, застає усі канали обслуговування уже зайнятими і покидає систему, то маємо систему з відмовами. Замовлення може покинути систему і тоді, коли черга досягла певних розмірів. Якщо ракета супротивника з’являється в час, коли всі протиракетні пристрої обслуговують інші ракети, то вона без проблем залишає зону обслуговування; – якщо нове замовлення, яке прибуло на

обслуговування, застає усі канали обслуговування зайнятими і стає у чергу, але перебуває у ній обмежений час і, не дочекавшись обслуговування, покидає систему, то маємо систему з обмеженим очікуванням. Прикладом такого „нетерплячого” замовлення може бути самоскид із цементним розчином. Якщо час очікування великий, то щоб запобігти затвердненню розчину, він може бути розвантажений в іншому місці; – якщо нове замовлення, яке прибуло на обслуговування, заставши усі канали обслуговування зайнятими, змушене очікувати своєї черги до того часу, поки не буде обслужене, то маємо систему з очікуванням без обмеження. Приклад: літак, який перебуває на аеродромі до того часу, поки не звільниться злітна смуга.

5. За способом вибору замовлень на обслуговування: з пріоритетом, за часом надходження, випадково, останнього обслуговують першим. Іноді в такому випадку кажуть про дисципліну обслуговування: – якщо система масового обслуговування охоплює кілька категорій замовлень і з певних міркувань необхідно дотримуватись різного підходу до їхнього відбору, то маємо систему з пріоритетом. Зокрема, під час надходження виробів на будмайданчик, перш за все монтують ті, які необхідні у цей момент; – якщо канал, який звільнився, обслуговує замовлення, яке раніше за інших надійшло до системи, то маємо систему з обслуговуванням замовлень за часом надходження. Це найпоширеніший клас систем. Наприклад, покупця, який підійшов до продавця першим, обслуговують раніше за інших. Цей спосіб вибору замовлень на обслуговування застосовують там, де внаслідок технічних, технологічних або організаційних умов замовлення не можуть випереджати одне одного;

– якщо замовлення з черги надходять до каналу обслуговування у випадковому порядку, то маємо систему з випадковим вибором замовлень на обслуговування. Приклад: вибір слюсарем-сантехніком одного з декількох замовлень на усунення несправностей, які надійшли від мешканців. Вибір

тут, зазвичай, визначають місцезнаходженням самого слюсаря: він надасть перевагу замовленню мешканця, який перебуває від нього найближче, якщо інші чинники не визначають вибору; – останнього обслуговують першим. Цей спосіб вибору вимог на обслуговування використовують у тих випадках, коли зручніше й економніше брати на обслуговування замовлення, яке найпізніше надійшло до системи. Зокрема, якщо будівельні вироби складені один на одному, то зручніше спочатку брати виріб, який надійшов останнім.

6. За характером обслуговування замовлень: на системи з детермінованим і випадковим часом обслуговування. Якщо інтервал часу між моментами надходження замовлення до каналу обслуговування і моментом виходу замовлення з цього каналу є сталим, то йдеться про систему з детермінованим часом обслуговування, в іншому разі – з випадковим.

7. За кількістю каналів обслуговування: на одноканальні і багатоканальні системи. Наприклад, для зведення будинку можна використати один будівельний кран (один канал обслуговування) або декілька (багато каналів) для обслуговування виробів, які прибувають на будову.

8. За кількістю етапів обслуговування: на однофазні і багатофазні системи. Якщо канали обслуговування розташовані послідовно, і вони неоднорідні, оскільки виконують різні операції обслуговування, то йдеться про багатофазну систему масового обслуговування. Прикладом такої системи може бути обслуговування автомобілів на станції технічного обслуговування (миття, діагностування тощо).

9. За однорідністю замовлень, які надходять на обслуговування: на системи з однорідними і неоднорідними потоками замовлень. Наприклад, якщо для розвантаження прибувають фургони однакової вантажомісткості, то такі замовлення називають однорідними, якщо різної – то неоднорідними.

10. За обмеженістю потоку замовлень: на замкнені і розімкнені системи. Якщо потік замовлень обмежений і замовлення, які покинули

систему, через деякий час до неї повертаються, то маємо замкнену систему, в протилежному випадку – розімкнену. Прикладом замкненої ситеми може слугувати бригада робітників, які налагоджують станки в ткацькому цеху.

2.2 СМО з відмовами [3-6]

Як показники ефективності СМО з відмовами розглядатимемо:

A — абсолютну пропускну спроможність СМО, тобто середнє число заявок, що обслуговуються в одиницю часу;

Q — відносну пропускну спроможність, тобто середню часткузаявок, що прийшли, обслуговуються системою;

$P_{отк}$—ймовірність відмови, тобто того, що заявка покине СМО не обслуженою;

k— середнє число зайнятих каналів (для багатоканальної системи).

2.2.1 Одноканальна система з відмовами

Розглянемо задачу.

Є один канал, на який поступаєпотік заявок з інтенсивністю λ. Потік обслуговуванні має інтенсивність μ . Знайти граничну ймовірність станів системи і показники її ефективності.

Система S (СМО) має два стани: S0— канал вільний S1 — канал зайнятий. Розмічений граф станів представлений на рис. 2.1.

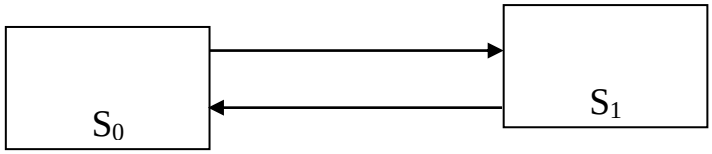

Рисунок 2.1

У граничному, стаціонарному режимі система рівнянь, алгебри, для ймовірності|станів має

$$\begin{cases} \lambda p_0 = \mu p_1 \\ \mu p_1 = \lambda p_0 \end{cases} \quad (2.1)$$

тобто система вироджується в одне рівняння. Враховуючи умову, нормування $p_0+p_1=1$, знайдемо з (1.1) гранична ймовірністьстанів

$$p_0 = \frac{\mu}{\lambda + \mu}, p_1 = \frac{\lambda}{\lambda + \mu} \quad (2.2)$$

які виражають середній відносний час перебування системи в станіS$_0$ (коли канал вільний) і S$_1$ (коли канал зайнятий| тобто визначають відповідно відносну пропускну спроможність Q системи і ймовірність відмови $P_{отк}$:

$$Q = \frac{\mu}{\lambda + \mu} \quad (2.3)$$

$$P_{отк} = \frac{\lambda}{\mu + \lambda} \quad (2.4)$$

Абсолютну пропускну спроможність знайдемо, помноживши відносну пропускну спроможність Q. на інтенсивність потоку відмов

$$\mathrm{A} = \frac{\lambda\mu}{\lambda + \mu} \quad (2.5)$$

2.2.2 Багатоканальна система з відмовами

Розглянемо класичне завдання Эрланга.

Є n каналів, на які поступає потік заявок з інтенсивністю λ. Потік обслуговувань має інтенсивність μ. Знайти граничну вірогідність станів системи і показники її ефективності.

Система S (СМО) має наступні стани (нумеруємо їх по числу заявок, що знаходяться в системі): S_o, S_1, S_2, ..., S_k ..., S_n, де S_k — стан системи, коли в ній знаходиться k заявок, тобто зайнято k каналів.

Граф станів СМО відповідає процесу загибелі і розмноження показаний на рис. 2.2

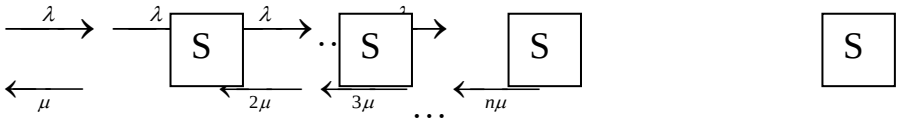

Рисунок 2.2 – Граф станів СМО

Потік заявок послідовно переводить систему з будь-якого лівого стану в сусідній правий з однією і тією ж інтенсивністю λ. Інтенсивність же потоку обслуговуванні, що переводять систему з будь-якого правого стану в сусідній лівий стан, постійно міняється залежно від стану. Дійсно, якщо СМО знаходиться в стані S_2 (два канали зайняті), то вона може перейти в стан S_1 (один канал зайнятий), коли закінчить обслуговування або перший, або другий канал, тобто сумарна інтенсивність їх потоків обслуговуванні буде 2μ. Аналогічно сумарний потік обслуговуванні, що переводить СМО із стану S_3 (три канали зайняті) у S_2 матиме інтенсивність 3μ, тобто може звільнитися будь-який з трьох каналів і т.д.

Для схеми загибелі і розмноження одержимо для граничної вірогідності стану

$$p_0 = \left(1 + \frac{\lambda}{\mu} + \frac{\lambda^2}{2!\mu^2} + ... + \frac{\lambda^k}{k!\mu^k} + ... + \frac{\lambda^n}{n!\mu^n}\right)^{-1} \qquad (2.6)$$

де члени розкладання $\frac{\lambda}{\mu}$, $\frac{\lambda^n}{n!\mu^n}$, $\frac{\lambda^2}{2!\mu^2}$ —,будуть представляти собою коефіцієнти при p_o у виразах для граничної вірогідності p_1, p_2, p_k ., p_n.

Величина називається *приведеною інтенсивністю потоку заявок* або *інтенсивністю навантаження каналу.* Вона виражає середнє число заявок, що приходить за середній час обслуговування однієї заявки. Тепер

$$p_0 = \left(1 + p + \frac{p^2}{2!} + ... + \frac{p^k}{k!} + ... + \frac{p^n}{n!}\right)^{-1} \qquad (2.7)$$

Формула (2.7) для граничної вірогідності одержала назву формула Эрланга на честь засновника теорії масового обслуговування.

Вірогідність відмови СМО є гранична вірогідність того що всі n каналів системи будуть зайняті, тобто

$$P_{отк.} = \frac{p^n}{n!} p_0 \qquad (2.8)$$

Відносна пропускна спроможність — вірогідність того, що заявка буде обслужена:

$$Q = 1 - P_{отк} = 1 - \frac{p^n}{n!} p_0 \qquad (2.9)$$

Абсолютна пропускна спроможність:

$$A = \lambda Q = \lambda\left(1 - \frac{p^n}{n!}\, p_0\right) \tag{2.10}$$

Середнє число зайнятих каналів є математичне очікування числа зайнятих каналів:

$$\bar{k} = \sum_{k=0}^{n} k p_k \tag{2.11}$$

де p_k — гранична вірогідність станів, визначуваних по формулі (1.8).

Проте середнє число зайнятих каналів можна знайти простіше, якщо врахувати, що абсолютна пропускна спроможність системи А є не що інше, як інтенсивність потоку обслужених системою заявок (у одиницю часу). Оскільки кожен зайнятий канал обслуговує в середньому заявок (у одиницю часу), то середнє число зайнятих каналів

$$\bar{k} = \frac{A}{\mu} \tag{2.12}$$

або, враховуючи (2.10) та (2.6):

$$\bar{k} = p\left(1 - \frac{p^n}{n!}\, p_0\right) \tag{2.13}$$

2.3 СМО з очікуванням (чергою) [5]

Як показники ефективності СМО з очікуванням, окрім вже відомих показників — абсолютної А і відносної *Q* пропускної спроможності, вірогідності відмови $P_{отк}$,, середнього числа зайнятих каналів $\bar{k}$ (для багатоканальної системи) розглядатимемо також наступні: $L_{сист}$— середнє число заявок в системі; $T_{сист}$ — середній час перебування заявки в системі; $L_{оч}$ — середнє число заявок в черзі { довжина черги); $T_{оч}$ — середній час перебування заявки в черзі; $P_{зан}$ — вірогідність того, що канал зайнятий (ступінь завантаження каналу).

2.3.1 Одноканальна система з необмеженою чергою

На практиці часто зустрічаються одноканальні СМО з необмеженою чергою (наприклад, телефон-автомат з однією будкою).

Розглянемо завдання.

Є одноканальна СМО з чергою, на яку не накладені ніякі обмеження (ні по довжині черги, ні за часом очікування). Потік заявок, що поступають в СМО, має інтенсивність, а потік обслуговуванні — інтенсивність μ . Необхідно знайти граничну вірогідність станів і показники ефективності СМО. Система може знаходитися в одному із станів S_o S_1, S_2, ., S_k, по числу заявок, що знаходяться в СМО: S_o — канал вільний; .S_1 — канал зайнятий (обслуговує заявку), черги немає; S_2— канал зайнятий, одна заявка стоїть в черзі; S_k — канал зайнятий, *(до - 1)* заявок стоять в черзі і т.д.

Граф станів СМО представлений на рис 2.3.

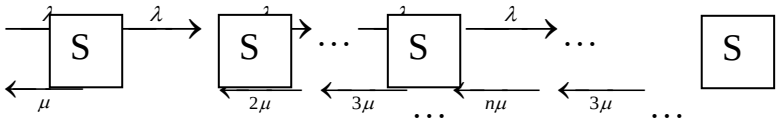

Рисунок 2.3 Граф станів СМО

Це процес загибелі і розмноження, але з нескінченним числом станів, в якому інтенсивність потоку заявок рівна, а інтенсивність потоку обслуговуванні μ.

Перш ніж записати формули граничної вірогідності, необхідно бути упевненим в їх існуванні, адже у разі, коли час $t \to \infty$, черга може необмежено зростати. Доведено, що якщо $p < 1$, тобто середнє число заявок, що приходять, менше середнього числа обслужених заявок (у одиницю часу), то гранична вірогідність існує. Якщо $p > 1$, черга росте до безкінечності.

Для визначення граничної вірогідності станів скористаємося формулами для процесу загибелі і розмноження (тут ми допускаємо відому не строгість, оскільки раніше ці формули були одержані для випадку кінцевого числа станів системи). Одержимо:

$$p_0 = \left[1 + \frac{\lambda}{\mu} + \left(\frac{\lambda}{\mu}\right)^2 + ... + \left(\frac{\lambda}{\mu}\right)^k + ...\right]^{-1} = \left(1 + p + p^2 + ... + p^k + ...\right) \qquad (2.14)$$

Оскільки гранична вірогідність існує лише при $p < 1$, то геометричний ряд із знаменником p < 1, записаний в дужках у формулі (1.29), сходиться до суми, рівної. Тому з урахуванням співвідношень

$$p_1 = \rho p_0,\ p_2 = \rho^2 p_0,...,\ p_k = \rho^k p_0,.... \qquad (2.15)$$

Знайдемо граничні ймовірності інших станів

$$p_1 = \rho(1-\rho),\ p_2 = \rho^2(1-\rho),\ldots,\ p_k = \rho^k(1-\rho),\ldots \qquad (2.16)$$

Гранична вірогідність p_0, p_1, p_2, p_k, утворюють убуваючу геометричну прогресію із знаменником $p < 1$, отже, вірогідність p_0 найбільша. Це означає, що якщо СМО справляється з потоком заявок (при $p < 1$), то найбільш вірогідною буде відсутність заявок в системі.

Середнє число заявок в системі $L_{сист}$ визначимо по формулі математичного очікування, яка з урахуванням (2.15) прийме вигляд

$$L_{сист} = \sum_{k=1}^{\infty} k p_k = (1-\rho)\sum_{k=1}^{\infty} k p_k \qquad (2.17)$$

(підсумовування від 1 до ∞, оскільки нульовий член $p_0 = 0$).

Можна показати, що формула (2.17) перетвориться (при $p < 1$) до вигляду

$$L_{сист} = \frac{\rho}{1-\rho} \qquad (2.18)$$

Знайдемо середнє число заявок в черзі $L_{оч}$. Очевидно, що

$$L_{оч} = L_{сист} - L_{об} \qquad (2.19)$$

де $L_{об}$ — середнє число заявок, що знаходяться під обслуговуванням.

Середнє число заявок під обслуговуванням визначимо по формулі математичного очікування числа заявок під обслуговуванням, що приймає значення 0 (якщо канал вільний) або 1 (якщо канал зайнятий):

$$L_{об} = 0 \cdot p_0 + 1(1 - p_0)$$

тобто середнє число заявок під обслуговуванням рівне вірогідності того, що канал зайнятий:

$$L_{об} = P_{зан} = 1 - p_0 \tag{2.20}$$

По формулі (2.14)

$$L_{об} = P_{зан} = p \tag{2.21}$$

Тоді отримаємо:

$$L_{оч} = \frac{\rho^2}{1 - \rho} \tag{2.22}$$

Доведено, що при будь-якому характері потоку заявок, при будь-якому розподілі часу обслуговування, при будь-якій дисципліні обслуговування середній час перебування заявки в системі (черги) рівна середньому числу заявок в системі (у черзі), що ділиться на інтенсивність потоку заявок, тобто

$$T_{сист} = \frac{1}{\lambda} L_{сист} \tag{2.23}$$

$$T_{оч} = \frac{1}{\lambda} L_{оч} \tag{2.24}$$

Формули (2.23) і (2.24) називаються формулами Літтла. Вони витікають з того, що в граничному, стаціонарному режимі середнє число

заявок, що прибувають в систему, рівно середньому числу заявок, що покидають її: обидва потоки заявок мають одну і ту ж інтенсивність λ.

На підставі формул (2.23) і (2.24) з обліком (2.18) і (2.22) середній час перебування заявки в системі визначиться по формулі:

$$T_{сист} = \frac{\rho}{\lambda(1-\rho)} \tag{2.25}$$

а середній час перебування заявки в черзі

$$T_{оч} = \frac{\rho^2}{\lambda(1-\rho)} \tag{2.26}$$

2.3.2 Багатоканальна СМО з необмеженою чергою

Розглянемо завдання. Є *n*-канальна СМО з необмеженою чергою. Потік заявок, що поступають в СМО, має інтенсивність λ, а потік обслуговуванні інтенсивність μ. Необхідно знайти граничну вірогідність станів СМО і показники її ефективності.

Система може знаходитися в одному із станів S_o S_1, S_2, ..., S_k ..., S_n нумерованих по числу заявок, що знаходяться в СМО: S_o — в системі немає заявок (всі канали вільні); S_1— зайнятий один канал, інші вільні; S_2 — зайняті два канали, інші вільні; S_k — зайнято каналів, інші вільні; S_n — зайняті всі *n* каналів (черги немає); S_{n+1} — зайняті всі *n* каналів, в черзі одна заявка; S_{n+r} — зайняті всі *n* каналів, *r* заявок стоїть в черзі.

Звернемо увагу на те, що на відміну від попередньої СМО, інтенсивність потоку обслуговуванні (що переводить систему з одного стану в інший справа наліво) не залишається постійною, а у міру збільшення числа заявок в СМО від *0* до *n* збільшується від величини μ до $n\mu$, оскільки

відповідно збільшується число каналів обслуговування. При числі заявок в СМО більшому, ніж *n*, інтенсивність потоку обслуговуванні зберігається рівною $n\mu$.

Можна показати, що при *p/n < 1* гранична вірогідність існує. Якщо *p/n ≥ 1*, черга росте до безкінечності. Використовуючи формули для процесу загибелі і розмноження, можна одержати наступні формули для граничної вірогідності станів *n-канальної* СМО з необмеженою чергою

$$p_0 = \left(1 + \frac{\rho}{1!} + \frac{\rho^2}{2!} + ... + \frac{\rho^n}{n!} .... + \frac{\rho^{n+1}}{n!(n-\rho)}\right)^{-1} \quad (2..27)$$

$$p_1 = \frac{\rho}{1!} p_0 ,..., p_k = \frac{\rho^k}{k!} p_0 ,..., p_n = \frac{\rho^n}{n!} p_0 \quad (2.28)$$

$$p_{n+1} = \frac{\rho^{n+1}}{n \cdot n!} p_0 ,..., p_{n+r} = \frac{\rho^{n+r}}{n^r \cdot n!} p_0 ,.... \quad (2.29)$$

Ймовірність того, що заявка буде в очереді,

$$P_{оч} = \frac{\rho^{n+1}}{n!(n-p)} p_0 \quad (2.30)$$

Для *n-канальної* СМО з необмеженою чергою, використовуючи колишні прийоми, можна знайти: середнє число зайнятих каналів

$$\bar{k} = \frac{\lambda}{\mu} = p \quad (2.31)$$

середнє число заявок в черзі

$$L_{оч} = \frac{\rho^{n+1} \cdot p_0}{n \cdot n! \left(1 - \frac{\rho}{n}\right)^2} \quad (2.32)$$

середнє число заявок в системі

$$L_{сист} = L_{оч} + \rho \quad (2.33)$$

Середній час перебування заявки в черги і середній час перебування заявки в системі, як і раніше, знаходяться по формулах Літтла.

2.3.3 СМО з обмеженою чергою

СМО з обмеженою чергою відрізняються від розглянутих вище завдань лише тим, що число заявок в черзі обмежене (не може перевершувати деякого заданого *m).* Якщо нова заявка поступає в мить, коли всі місця в черзі зайняті, вона покидає СМО не обслуженої, тобто дістає відмову.

Очевидно: для обчислення граничної вірогідності станів і показників ефективності таких СМО може бути використаний той же підхід, що і вище, з тією різницею, що підсумовувати треба не нескінченну професію (як, наприклад, ми робили при виведенні формули 1.30, а кінцеву.

Середній час перебування заявки в черзі і в системі, як і раніше, визначаємо по формулах Літтла.

Припустимо, що в деякий момент часу до обслуговуючої системи надходить заявка. Якщо в цей момент часу в системі є вільні лінії, то заявка приймається до обслуговування. У протилежному випадку, тобто коли усі лінії зайняті, заявка лишається на вході у систему на протязі деякого часу τп ($\tau_п$ – час перебування заявки у системі) як претендент на обслуговування. За

інтервал часу $\tau_п$ заявка повинна бути прийнята до обслуговування, в протилежному випадку вона вважається загубленою (отримає відмову).

2.4 Модель обєкту

Дослідження поведінки СМО, тобто дослідження залежностей фазових змінних від часу при подачі на входи будь-яких, необхідних відповідно до завдання, потоків заявок, називають імітаційним моделюванням СМО. Імітаційне моделювання проводиться шляхом відтворення подій, які відбуваються одночасно або послідовно в модельному часі. При цьому подією є факт зміни значення будь-якої фазової змінної [9-10].

Для імітаційного моделювання СМО необхідні імітаційна модель, мова для її представлення і програмна система, що реалізує цю мову.

Імітаційні моделі СМО називаються мережними імітаційними моделями (МІМ). Використовують декілька підходів до побудови МІМ, що відрізняються вибором структури моделі. У структурі МІМ виділяються її елементарні частини (моделі елементів), такими елементами можуть бути або події, або засоби обслуговування, або пункти маршрутів заявок. Відповідно розрізняють моделі і мови їх опису, орієнтовані на події, пристрої або процеси. Тому вибір мови моделювання визначає структуру моделі і методику її побудови.

У структурі МІМ, яка орієнтована на процеси, використовуються моделі джерел вхідних потоків заявок, пристроїв, накопичувачів і вузлів.

Джерело вхідного потоку заявок у моделі являє собою алгоритм, по якому обчислюються моменти появи заявок на вході. Джерела можуть бути залежними і незалежними. У залежних джерелах моменти появи заявок залежать від настання визначених подій, наприклад, від приходу іншої заявки на вхід деякого пристрою. Типовою моделлю незалежного джерела є

алгоритм генерації значень випадкової величини з заданим законом розподілу.

Пристрої в імітаційній моделі представляються алгоритмами генерації значень інтервалів (тривалостей) обслуговування. Частіше всього це алгоритми генерації значень випадкових величин із заданим законом розподілу. Крім того, модель пристрою відображає задану дисципліну обслуговування, оскільки в модель входить алгоритм, який управляє чергами на входах пристрою.

Накопичувачі моделюються алгоритмами визначення обсягів пам'яті, які займаються заявками, що приходять на вхід накопичувача. Звичайно, обсяг пам'яті, що займається заявкою, обчислюється як значення випадкової величини, закон і (або) числові характеристики розподілу залежать від типу заявки.

Вузли виконують сполучні, керуючі і допоміжні функції в імітаційній моделі, наприклад, для вибору напрямків руху заявок у МІМ, зміни їхніх параметрів і пріоритету, поділу заявок на частини, об'єднання і т.п.

Звичайно кожному типу елементарної моделі, за винятком лише деяких вузлів, у програмній системі відповідає визначена процедура (підпрограма). Тоді МІМ можна представити як алгоритм, котрий складається з впорядкованих звертань до цих процедур, відображаючих поведінку системи.

В процесі моделювання відбуваються зміни модельного часу, котрий частіше всього приймається дискретним, і вимірюється в тактах. Час змінюється після того, як закінчена імітація чергової групи подій, що відносяться до поточного моменту часу t_k. Імітація супроводжується накопиченням в окремому файлі статистики даних: кількості заявок, які вийшли із системи опрацьованими і неопрацьованими, сумарного часу зайнятості для кожного пристрою, середньої довжини черг і т.п. Імітація закінчується коли поточний час перевищить заданий відрізок часу або коли вхідні джерела опрацюють задане число заявок. Після цього проводиться

опрацювання накопичених даних, у результаті одержуємо значення необхідних вихідних параметрів.

У програмах імітаційного моделювання СМО переважно реалізується подійний метод організації обчислень. Суть подійного методу полягає в відслідковуванні на моделі послідовності подій у тому ж порядку, у якому вони відбувалися б у реальній системі. Обчислення проводяться тільки для тих моментів часу і тих частин (процедур) моделі, до яких відносяться події. Даний метод може істотно прискорити моделювання в порівнянні з інкрементним методом, у якому на кожному такті аналізуються стани всіх елементів моделі.

Розглянемо можливу схему реалізації подійного методу імітаційного моделювання.

Моделювання починається з перегляду операторів генерації заявок, тобто з звертання до моделей джерел вхідних потоків. Для кожного незалежного джерела таке звертання дозволяє розрахувати момент генерації першої заявки. Цей момент разом із іменем – вказівкою на заявку – заноситься в список майбутніх подій (СМП), а відомості про заявку, що генерується – у список заявок (СЗ). Запис у СЗ містить у собі ро’я заявки, значення її параметрів (атрибутів), місце, що займається в даний момент у МІМ. У СМП події впорядковуються по збільшенню моментів настання.

Далі зі СМП вибирається сукупність відомостей про події, що відносяться до найбільш раннього моменту часу. Ця сукупність переноситься в список поточних подій (СПП), із якого беруться вказівки на події. Звертання по вказівці до СЗ дозволяє встановити місце в МІМ заявки А, з якою пов’язана подія, що моделюється. Нехай цим місцем є пристрій В. Далі програма моделювання виконує такі дії:

1) змінює параметри стану пристрою В; наприклад, якщо заявка А звільняє В, а черга до В не була порожня, то відповідно до заданої

дисципліни обслуговування з черги до В вибирається заявка С и надходить на обслуговування в В;

2) прогнозується час настання наступної події, пов’язаної з заявкою С, шляхом звертання до моделі пристрою В, у якій розраховується тривалість обслуговування заявки С; відомості про цю майбутню подію заносяться в СМП і СЗ;

3) відбувається імітація руху заявки А в МІМ по маршруту, визначеному заданою програмою моделювання, доти, поки заявка не прийде на вхід деякого ОА; тут або заявка затримується в черзі, або шляхом звертання до моделі цього ОА прогнозується настання деякої майбутньої події, пов’язаної з наступним шляхом заявки А; відомості про цю майбутню подію також заносяться в СМП і СЗ;

4) у файл статистики добавляються згадані вище дані.

Після відпрацювання всіх подій, що відносяться до моменту часу t_k, відбувається збільшення модельного часу до значення, що відповідає найближчій майбутній події, і розглянутий процес імітації повторюється.

Імітаційне моделювання базується на відтворені на ЕОМ розгорнутого в часі процесу функціонування системи з обов’язковим врахуванням її взаємодії із зовнішнім середовищем. Для створення імітаційної моделі необхідно провести:

Побудову моделі досліджуваної системи на основі моделей підсистем, які об’єднані одним з видів взаємодій.

Вибір інформативних характеристик об’єкта, способів їх одержання та методів їх аналізу.

Побудову моделі впливу зовнішнього середовища на систему у вигляді імітаційних моделей зовнішніх факторів.

Вибір способу дослідження імітаційної моделі відповідно до методів планування імітаційного експерименту.

Умовно імітаційну модель представляють у вигляді діючих програмно чи апаратно реалізованих блоків:

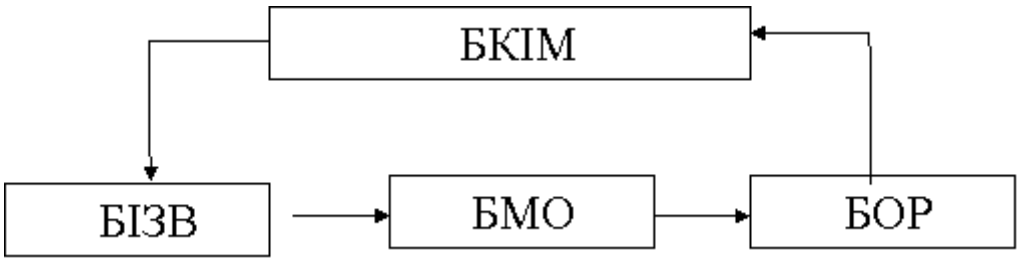

Рисунок 2.4 –Структура імітаційної моделі

Структура імітаційної моделі (рис.2.4)

Блок імітації зовнішніх впливів (БІЗВ) – формує реалізації випадкових чи детермінованих процесів, які імітують впливи зовнішнього середовища на об’єкти.

Блок опрацювання результатів (БОР) – дає змогу одержати інформативні характеристики об’єкта. Для цього він використовує інформацію, що надходить від блоку математичної моделі об’єкта.

Блок математизації об’єкта (БМО).

Блок керування імітаційною моделлю (БКІМ) – реалізовує способи дослідження моделі та автоматизовує процес проведення імітаційного експерименту.

Метою імітаційного моделювання є конструювання або побудова імітаційної моделі та проведення імітаційного експерименту з цією моделлю для вивчення функціювання та поведінки досліджуваної системи обо об’єкта.

До переваг імітаційного моделювання відносяться:

1) можливість проведення експериментів над моделями системи, для яких натурні експерименти неможливі з етичних та небезпечних для життя причин.

2) вирішення задач, до яких не застосовні аналітичні методи за причини їх складності або відсутності.

3) можливість аналізу загальносистемних функцій для складних систем.

4) скорочення термінів для прийняття проектних рішень та проведення оцінки їх ефективності.

5) можливість проведення аналізу варіантів структури великих систем та їх поведінки під дією тих чи інших впливів.

Структурна модель показує, з яких компонентів складається система і які зв’язки існують між ними:

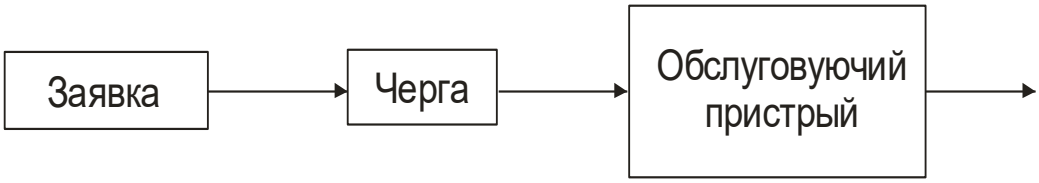

Рисунок 2.5 –Побудова структурної моделі

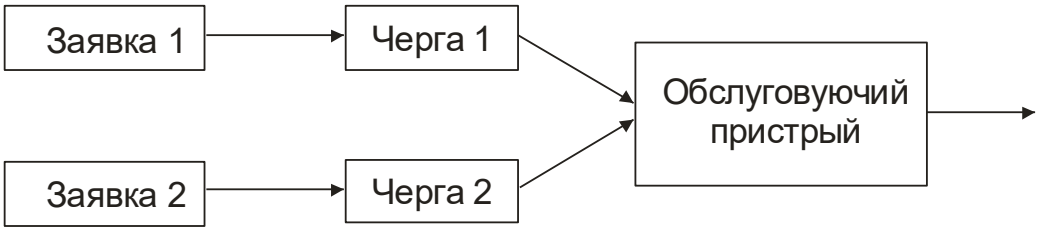

Рисунок 2.6 –Побудова функціональної моделі

Концептуальною моделлю об’єкта називають сукупність характеристик, властивостей та особливостей функціонування об’єкта і припущення щодо ступеня впливу на вихідні потоки . Це абстрактна модель, що визначає склад і структуру системи, властивості елементів та причинно-

наслідкові зв’язки, притаманні досліджуваній системі і необхідні для досягнення цілі моделювання.

Основними поняттями концептуальної моделі є: дані, дані користувача, керуючі дані, пакет, вузол комутації, канал зв’язку і мережа передачі даних.

Дані – це факти описані у формалізованому вигляді. Виділяють дані користувача(вводяться ни сисему) і керуючі дані(служать для керування роботи системи).

Мережа передачі даних – сукупність пристроїв, що служить для передачі даних.

Вузол комутації – виконує функцію комутації даних. Їх перерозподілу.

Канал зв’язку – сукупність засобів, що забезпечують доставку даних в необхідне місце призначення.

Пакет – дані, що мають заголовок і обмежену довжину.

Концептуальна модель є основою для розробки математичної моделі. Створенням математичної моделі досягається дві мети:

– для однозначного розуміння процесу функціонування системи дається її формалізований опис;

– дає можливість представити опис системи у такому вигляді, який дозволяє проведеня аналітичного дослідженя.

РОЗДІЛ 3 ОПИС ТА РЕЗУЛЬТАТИ РОБОТИ ПРОГРАМИ

Лістинг програми показаний в додатку А.

Дана програма написана в середовищі розробки Delphi з використанням візуальних компонентів управління.

Результуюча програма CMO.exe є повністю закінченим програмним продуктом.

При запуску пограмми з'являється вікно, в якому користувач може вибрати тип системи масового обслужевания рис. 3.1:

одноканальна СМО з відмовами;

багатоканальна СМО з відмовами;

одноканальна СМО з необмеженою чергою;

багатоканальна СМО з необмеженою чергою.

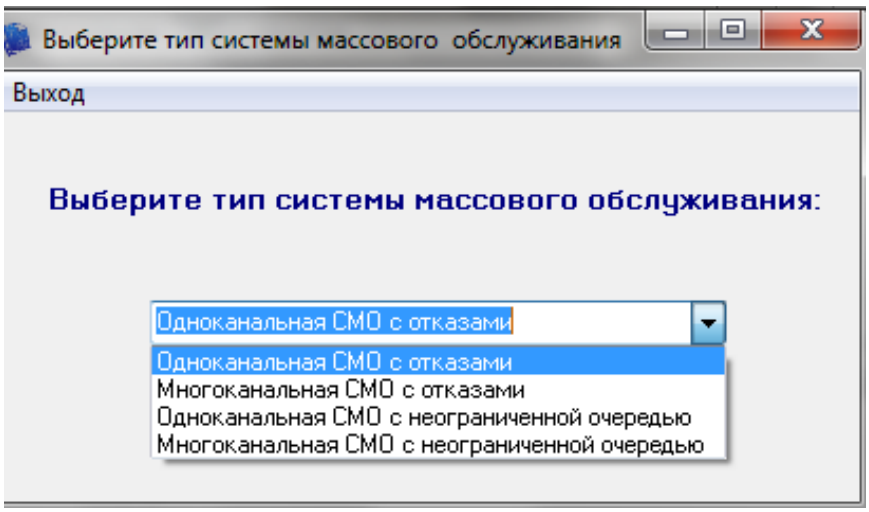

Рисунок 3.1 –Вікно програми

Розглянемо роботу програми на прикладі одноканальна СМО з відмовами рис. 3.2.

Розглянемо роботу програми на прикладі одноканальна СМО з відмовами

Після вибору під меню одноканальна СМО з'являється вікно містять наступне вхідні дані:

кількість заявок;

час обслуговування (однієї заявки):

або інтенсивність обслуговування.

Також користувачеві надається можливість вибрати відповідні одиниці вимірювання часу

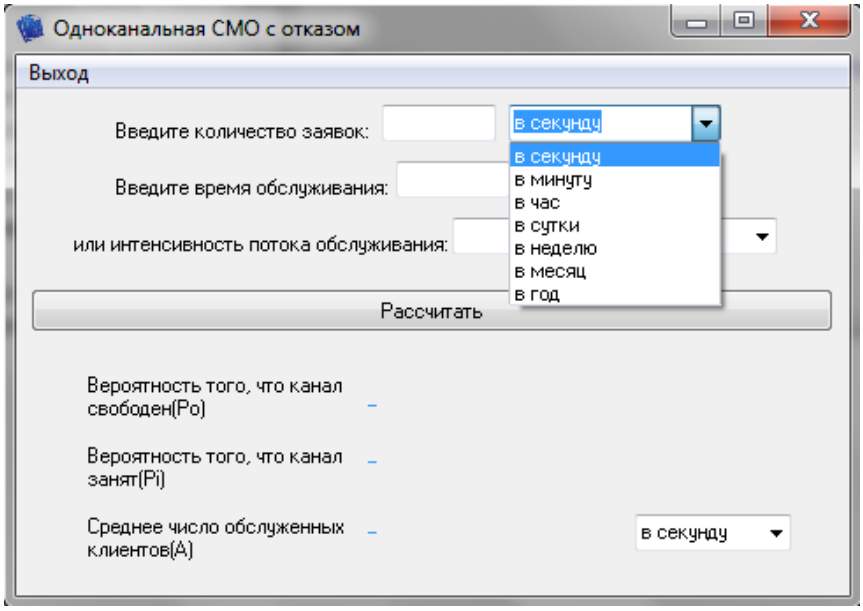

Рисунок 3.2 – Робота програми на прикладі одноканальна СМО

Після заповнення вхідних даних натискається кнопка «Розрахувати» і програма виводить результат у тому ж вікні рис. 3.3.

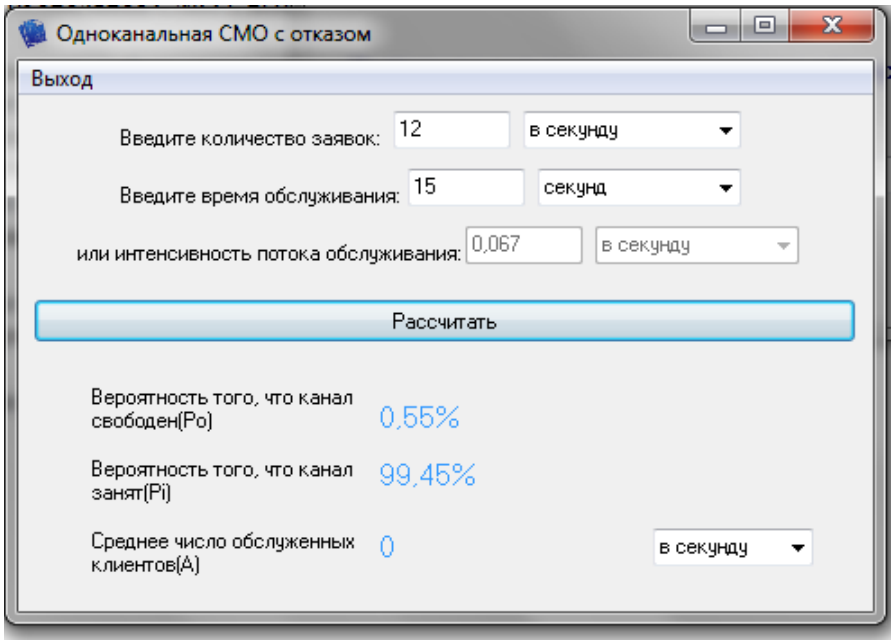

Рисунок 3.3 – Робота програми на прикладі одноканальна СМО

Результатом роботи програми є показники:

ймовірність того, що канал вільний;

ймовірність того, що канал зайнятий;

середнє число обслужених каналів.

Таблиця 3.1– Вхідних даних

Назва вхідних даних	Значення вхідних даних
кількість заявок	12
час обслуговування	15
або інтенсивність	0,067

Таблиця 3.2– Результат одноканальної СМО

Назва показника	Значення показника
ймовірність того, що канал вільний	0,55 %
ймовірність того, що канал зайнятий	99,45 %
середнє число обслужених каналів	0

Аналогічно працює програма і по іншим типам систем масового обслуговування рис. 3.4.

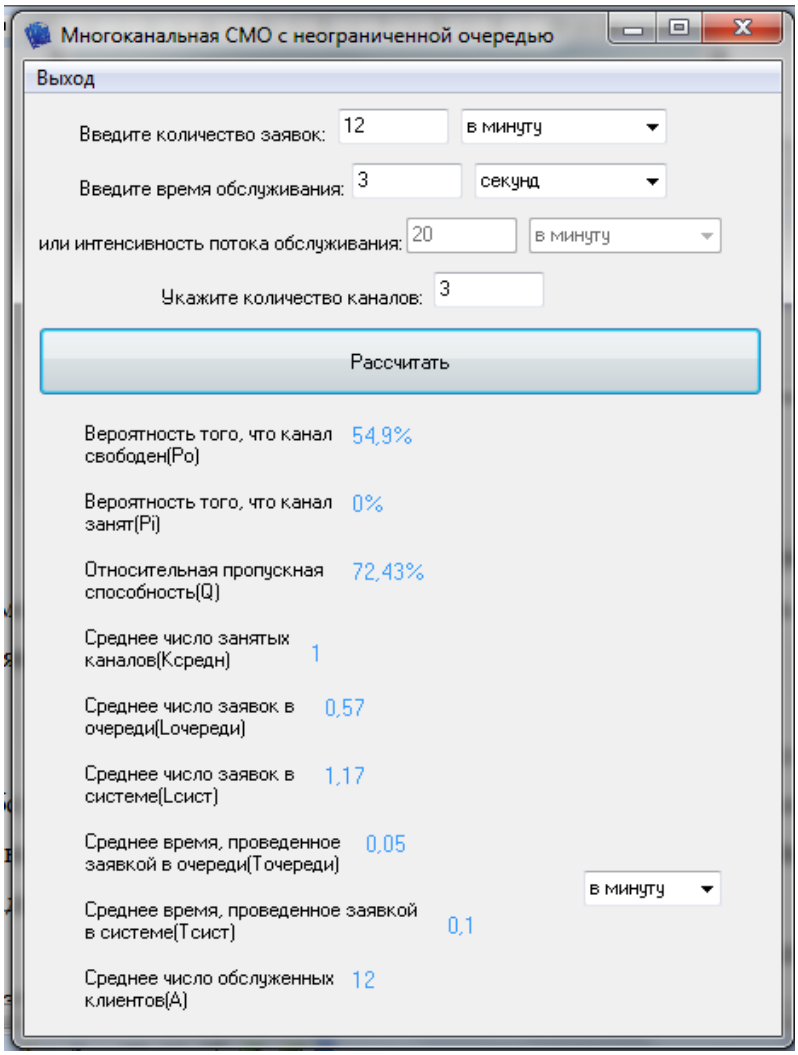

Рисунок 3.4 – Робота програми на прикладі багатоканальна СМО з необмеженою чергою.

Таблиця 3.3 Вхідних даних

Назва вхідних даних	Значення вхідних даних
кількість заявок	12
час обслуговування	3
або інтенсивність	20
кількість каналів	3

Таблиця 3.4– Результат багатоканальної СМО з необмеженою чергою

Назва показника	Значення показника
ймовірність того, що канал вільний	54,9 %
ймовірність того, що канал зайнятий	0 %
відносна пропускна здатність	72,43 %
середнє число зайнятих каналів	1
середнє число заявок в черги	0,57
середнє число заявок в системі	1,17
середній час, проведений заявкою в черги	0,05
середній час, проведений заявкою в системі	0,1
середнє число обслужених клієнтів	12

Так само варто відзначити, що в разі введення даних не відповідають роботі СМО з'являється вікно з оповіщенням «Система масового обслуговування не має показників, чергу зростає до нескінченності» рис. 3.5

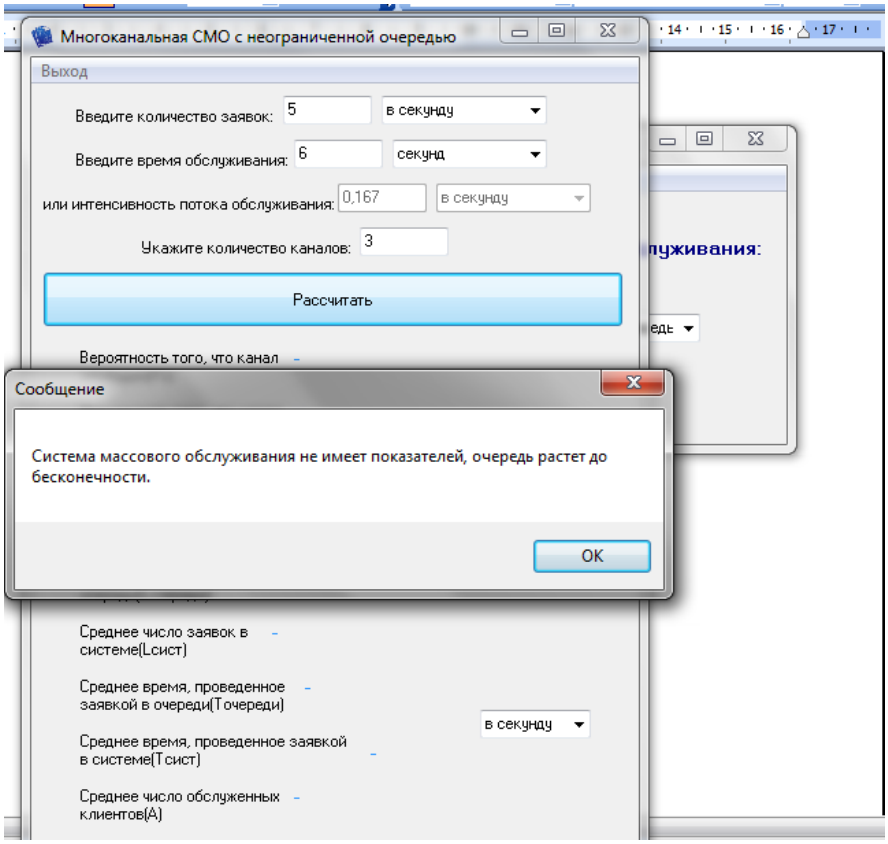

Рисунок 3.5 – «Система масового обслуговування не має показників, чергу зростає до нескінченності»

А в разі не введення даних і натискання клавіші «Розрахувати» з'являється вікно з оповіщенням «Будь ласка, введіть дані» рис. 3.6

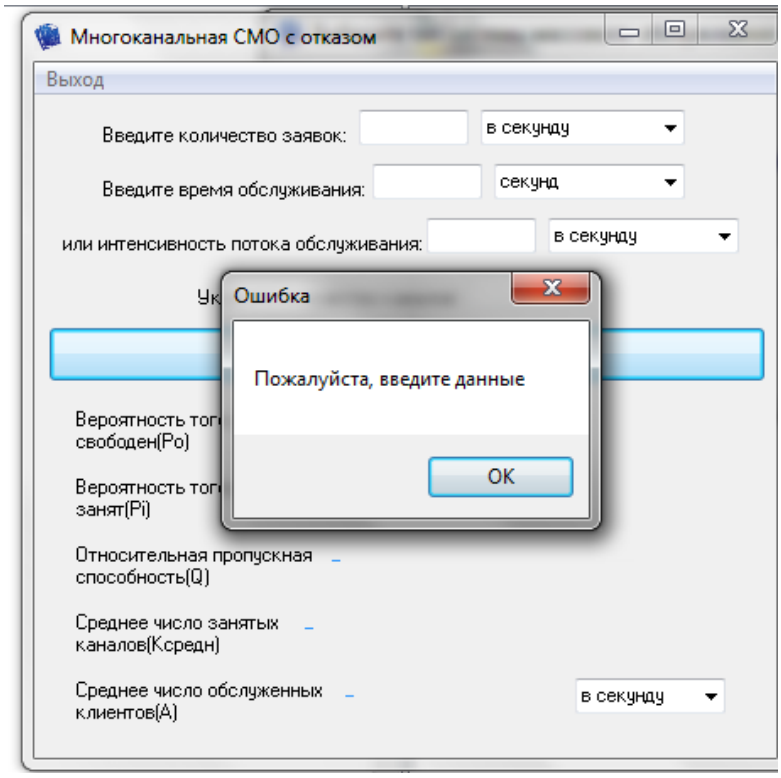

Рисунок 3.6 – В разі не введення даних

Таким чином, поставлена задача про створення програми вирішена.

Програма дозволяє змоделювати:

одноканальний СМО з відмовами;

оагатоканальну СМО з відмовами;

одноканальнаую СМО з необмеженою чергою;

багатоканальну СМО з необмеженою чергою.

ВИСНОВКИ

1. Зроблено аналіз предметної області, вивчені основні джерела й особливості розробки моделі системи масового обслуговування.

2. У роботі була розроблена програма, що реалізує модель роботи систем масового обслуговування.

3. Розроблена програма може бути використана при моделюванні роботи:

одноканальний СМО з відмовами;

оагатоканальну СМО з відмовами;

одноканальнаую СМО з необмеженою чергою;

багатоканальну СМО з необмеженою чергою.

СПИСОК ЛІТЕРАТУРИ

1. Томашевський В.М. Моделювання систем [Текст]. – К.: Видавнича група ВН, 2005. – 352 с.

2. Жожикашвили В.А., Вишневский В.М. Сети массового обслуживания. Теория и применение к сетям ЭВМ [Текст]. – М.: Радио и связь, 1988. -192 с.

3. Кузин Л.Т. Основы кибернетики: В2-х тт. Т. 2. Основы кибернетических моделей [Текст]. Учеб. Пособие для вузов. – М.: Энергия,1979. – 584 с.

4. Беляков В.Г., Митрофанов Ю.И. к исследованию замкнутых сетей массового обслуживания большой размерности [Текст]. Автоматика и телемеханика. – 1981. - №7. – С.61-69.

5. Вишневский В.М., Герасимов А.И. Исследование потоков в замкнутых экспоненциальных сетях массового обслуживания [Текст]. Проблемы управления и теории информации. – 1983. Т. 12, №6.

6. Яшков С.Ф. Анализ очередей в ЭВМ [Текст]. – М.:Радио и связь, 1989, 216с.

7. Шварц М. Сети ЭВМ. Анализ и проектирование [Текст]. – М.: Радио и свіязь, 1981. – 336 с.

8. Саати Т. Л. Элементы теории обслуживания [Текст]. М.: Советское радио , 1971.

9. Гнеденко Б. В., Коваленко И. Н. Введение в теорию массового обслуживания [Текст]. 2-е изд. М.:Наука , 1987.

10. Вентцель Е.С. Исследование операций [Текст]. Изд-во «Советское радио»,1972.

ДОДАТОК А

Листинг програми

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Menus, XPMan;
type
  TForm1 = class(TForm)
    Label1: TLabel;
    ComboBox1: TComboBox;
    MainMenu1: TMainMenu;
    D1: TMenuItem;
    XPManifest1: TXPManifest;
    procedure D1Click(Sender: TObject);
    procedure ComboBox1Change(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
uses Unit2, Unit3, Unit4, Unit5;
{$R *.dfm}
procedure TForm1.D1Click(Sender: TObject);
begin
application.Terminate;
end;
procedure TForm1.ComboBox1Change(Sender: TObject);
begin
if combobox1.ItemIndex=0 then form2.Show;
```

```
if combobox1.ItemIndex=1 then form3.Show;
if combobox1.ItemIndex=2 then form4.Show;
if combobox1.ItemIndex=3 then form5.Show;
end;
end.

unit Unit2;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Menus, StdCtrls, Math, XPMan, ExtCtrls;
type
  TForm2 = class(TForm)
    MainMenu1: TMainMenu;
    D1: TMenuItem;
    Label1: TLabel;
    Label2: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Button1: TButton;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    ComboBox1: TComboBox;
    ComboBox2: TComboBox;
    ComboBox3: TComboBox;
    XPManifest1: TXPManifest;
    Edit3: TEdit;
    ComboBox4: TComboBox;
    Label9: TLabel;
    Timer1: TTimer;
```

```
    Edit4: TEdit;
    Edit5: TEdit;
    procedure D1Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
      private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form2: TForm2;
  l,m,p0,p1,a,f:double;
  implementation
{$R *.dfm}
procedure TForm2.D1Click(Sender: TObject);
begin
form2.Close;
end;
procedure TForm2.Button1Click(Sender: TObject);
begin
if (edit1.Text='') or ((edit2.Text='') and (edit3.Text='')) then
Application.MessageBox('Пожалуйста, введите данные','Ошибка', MB_OK)
else
begin
case combobox1.ItemIndex of
0:begin l:=strtofloat(edit1.Text); f:=1; end;
1:begin l:=strtofloat(edit1.Text)/60; f:=60; end;
2:begin l:=strtofloat(edit1.Text)/3600; f:=3600; end;
3:begin l:=strtofloat(edit1.Text)/86400; f:=86400; end;
4:begin l:=strtofloat(edit1.Text)/604800; f:=604800; end;
5:begin l:=strtofloat(edit1.Text)/2592000; f:=2592000; end;
6:begin l:=strtofloat(edit1.Text)/31536000; f:=31536000; end;
end;
```

```
if (edit2.Enabled=true) then
case combobox2.ItemIndex of
0:begin m:=1/strtofloat(edit2.Text); edit5.Text:=floattostr(roundto(m*f, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
1:begin m:=1/strtofloat(edit2.Text)/60;
edit5.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/60, -3));
combobox4.ItemIndex:=combobox1.ItemIndex;; end;
2:begin m:=1/strtofloat(edit2.Text)/3600;
edit5.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/3600, -3));
combobox4.ItemIndex:=combobox1.ItemIndex;; end;
3:begin m:=1/strtofloat(edit2.Text)/86400;
edit5.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/86400, -3));
combobox4.ItemIndex:=combobox1.ItemIndex;; end;
4:begin m:=1/strtofloat(edit2.Text)/604800;
edit5.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/604800, -3));
combobox4.ItemIndex:=combobox1.ItemIndex;; end;
5:begin m:=1/strtofloat(edit2.Text)/2592000;
edit5.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/2592000, -3));
combobox4.ItemIndex:=combobox1.ItemIndex;; end;
6:begin m:=1/strtofloat(edit2.Text)/31536000;
edit5.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/31536000, -3));
combobox4.ItemIndex:=combobox1.ItemIndex;; end;
end
else
case combobox4.ItemIndex of
0:begin m:=strtofloat(edit3.Text); edit4.Text:=floattostr(roundto(1/strtofloat(edit3.Text), -3));
combobox2.ItemIndex:=0; end;
1:begin m:=strtofloat(edit3.Text)/60; edit4.Text:=floattostr(roundto(1/strtofloat(edit3.Text), -3));
combobox2.ItemIndex:=1; end;
2:begin m:=strtofloat(edit3.Text)/3600; edit4.Text:=floattostr(roundto(1/strtofloat(edit3.Text), -
3)); combobox2.ItemIndex:=2; end;
3:begin m:=strtofloat(edit3.Text)/86400; edit4.Text:=floattostr(roundto(1/strtofloat(edit3.Text), -
3)); combobox2.ItemIndex:=3; end;
```

```
4:begin m:=strtofloat(edit3.Text)/604800; edit4.Text:=floattostr(roundto(1/strtofloat(edit3.Text), -3)); combobox2.ItemIndex:=4; end;
5:begin m:=strtofloat(edit3.Text)/2592000; edit4.Text:=floattostr(roundto(1/strtofloat(edit3.Text), -3)); combobox2.ItemIndex:=5; end;
6:begin m:=strtofloat(edit3.Text)/31536000; edit4.Text:=floattostr(roundto(1/strtofloat(edit3.Text), -3)); combobox2.ItemIndex:=6; end;
end;
p0:=m/(m+l);
p1:=l/(m+l);
case combobox3.ItemIndex of
0:a:=l*p0;
1:a:=l*60*p0;
2:a:=l*3600*p0;
3:a:=l*86400*p0;
4:a:=l*604800*p0;
5:a:=l*2592000*p0;
6:a:=l*31536000*p0;
end;
label6.Caption:=floattostr(Roundto(p1*100, -2)) + '%';
label7.Caption:=floattostr(Roundto(p0*100, -2)) + '%';
label8.Caption:=floattostr(Round(a));
end;
end;
procedure TForm2.Timer1Timer(Sender: TObject);
begin
if edit2.Text<>'' then begin edit3.Enabled:=false; edit3.visible:=false; edit5.Visible:=true; combobox4.Enabled:=false; end else begin edit3.Enabled:=true; edit3.visible:=true; edit5.Visible:=false ;combobox4.Enabled:=true; end;
if edit3.Text<>'' then begin edit2.Enabled:=false; edit2.visible:=false; edit4.Visible:=true; combobox2.Enabled:=false; end else begin edit2.Enabled:=true; edit2.visible:=true; edit4.Visible:=false;combobox2.Enabled:=true; end;
end;
end.
```

```
unit Unit3;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Menus, StdCtrls, Math, XPMan, ExtCtrls;
type
  TForm3 = class(TForm)
    MainMenu1: TMainMenu;
    D1: TMenuItem;
    Button1: TButton;
    Label3: TLabel;
    Edit3: TEdit;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label8: TLabel;
    Label13: TLabel;
    Label9: TLabel;
    Label14: TLabel;
    Label1: TLabel;
    Label2: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    ComboBox1: TComboBox;
    ComboBox2: TComboBox;
    ComboBox3: TComboBox;
    XPManifest1: TXPManifest;
    Timer1: TTimer;
    Edit4: TEdit;
    ComboBox4: TComboBox;
    Label7: TLabel;
```

```
    Edit5: TEdit;
    Edit6: TEdit;
    function fact(chisl:integer): extended;
    procedure D1Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form3: TForm3;
  k,n:integer;
  ks,a,l,m,p0,p1,q,f,z:double;
  r:extended;
implementation
{$R *.dfm}
procedure TForm3.D1Click(Sender: TObject);
begin
form3.Close;
end;
function Tform3.fact(chisl:integer) : extended;
var i:integer;
begin
result:=1;
for i:=1 to chisl do
begin
result:= result * i;
end;
end;
procedure TForm3.Button1Click(Sender: TObject);
begin
```

```
if (edit1.Text='') or ((edit2.Text='') and (edit4.Text='')) or (edit3.Text='') then Application.MessageBox('Пожалуйста, введите данные','Ошибка', MB_OK)
else
begin
case combobox1.ItemIndex of
0:begin l:=strtofloat(edit1.Text); f:=1; end;
1:begin l:=strtofloat(edit1.Text)/60; f:=60; end;
2:begin l:=strtofloat(edit1.Text)/3600; f:=3600; end;
3:begin l:=strtofloat(edit1.Text)/86400; f:=86400; end;
4:begin l:=strtofloat(edit1.Text)/604800; f:=604800; end;
5:begin l:=strtofloat(edit1.Text)/2592000; f:=2592000; end;
6:begin l:=strtofloat(edit1.Text)/31536000; f:=31536000; end;
end;
if (edit2.Enabled=true) then
case combobox2.ItemIndex of
0:begin m:=1/strtofloat(edit2.Text); edit6.Text:=floattostr(roundto(m*f, -3)); combobox4.ItemIndex:=combobox1.ItemIndex; end;
1:begin m:=1/strtofloat(edit2.Text)/60; edit6.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/60, -3)); combobox4.ItemIndex:=combobox1.ItemIndex; end;
2:begin m:=1/strtofloat(edit2.Text)/3600; edit6.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/3600, -3)); combobox4.ItemIndex:=combobox1.ItemIndex; end;
3:begin m:=1/strtofloat(edit2.Text)/86400; edit6.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/86400, -3)); combobox4.ItemIndex:=combobox1.ItemIndex; end;
4:begin m:=1/strtofloat(edit2.Text)/604800; edit6.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/604800, -3)); combobox4.ItemIndex:=combobox1.ItemIndex; end;
5:begin m:=1/strtofloat(edit2.Text)/2592000; edit6.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/2592000, -3)); combobox4.ItemIndex:=combobox1.ItemIndex; end;
```

```
6:begin m:=1/strtofloat(edit2.Text)/31536000;
edit6.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/31536000, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
end
else
case combobox4.ItemIndex of
0:begin m:=strtofloat(edit3.Text); edit5.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -3));
combobox2.ItemIndex:=0; end;
1:begin m:=strtofloat(edit3.Text)/60; edit5.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -3));
combobox2.ItemIndex:=1; end;
2:begin m:=strtofloat(edit3.Text)/3600; edit5.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -
3)); combobox2.ItemIndex:=2; end;
3:begin m:=strtofloat(edit3.Text)/86400; edit5.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -
3)); combobox2.ItemIndex:=3; end;
4:begin m:=strtofloat(edit3.Text)/604800; edit5.Text:=floattostr(roundto(1/strtofloat(edit4.Text),
-3)); combobox2.ItemIndex:=4; end;
5:begin m:=strtofloat(edit3.Text)/2592000;
edit5.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -3)); combobox2.ItemIndex:=5; end;
6:begin m:=strtofloat(edit3.Text)/31536000;
edit5.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -3)); combobox2.ItemIndex:=6; end;
end;
k:=strtoint(edit3.Text);
r:=l/m;
p0:=1+r;
for n:=2 to k do
begin
p0:=p0 + exp(n*ln(r))/fact(n);
end;
p0:=1/p0;
p1:=(exp(k*ln(r))/fact(k))*p0;
q:=1-p1;
case combobox3.ItemIndex of
0:begin a:=l*q; m:=m; end;
1:begin a:=l*60*q; m:=m*60; end;
```

```
2:begin a:=l*3600*q; m:=m*3600; end;
3:begin a:=l*86400*q; m:=m*86400; end;
4:begin a:=l*604800*q; m:=m*604800; end;
5:begin a:=l*2592000*q; m:=m*2592000; end;
6:begin a:=l*31536000*q; m:=m*31536000; end;
end;
ks:=a/m;
label10.Caption:=floattostr(roundto(p1*100, -2)) + '%';
label11.Caption:=floattostr(roundto(p0*100, -2)) + '%';
label12.Caption:=floattostr(Round(a));
label13.Caption:=floattostr(roundto(q*100,-2)) + '%';
label14.Caption:=floattostr(round(ks))
end;
end;
procedure TForm3.Timer1Timer(Sender: TObject);
begin
if edit2.Text<>'' then begin edit4.Enabled:=false; edit4.visible:=false; edit6.Visible:=true; combobox4.Enabled:=false; end else begin edit4.Enabled:=true; edit4.visible:=true; edit6.Visible:=false; combobox4.Enabled:=true; end;
if edit4.Text<>'' then begin edit2.Enabled:=false; edit2.visible:=false; edit5.Visible:=true;combobox2.Enabled:=false; end else begin edit2.Enabled:=true; edit2.visible:=true; edit5.Visible:=false; combobox2.Enabled:=true; end;
end;
end.

unit Unit4;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Menus, StdCtrls, Math, XPMan, ExtCtrls;
type
  TForm4 = class(TForm)
    MainMenu1: TMainMenu;
    D1: TMenuItem;
```

```
  Label3: TLabel;
  Label4: TLabel;
  Label5: TLabel;
  Label6: TLabel;
  Label7: TLabel;
  Label8: TLabel;
  Label9: TLabel;
  Label10: TLabel;
  Label11: TLabel;
  Label12: TLabel;
  Label13: TLabel;
  Label14: TLabel;
  Button1: TButton;
  Label1: TLabel;
  Label2: TLabel;
  Edit1: TEdit;
  Edit2: TEdit;
  ComboBox1: TComboBox;
  ComboBox2: TComboBox;
  MainMenu2: TMainMenu;
  MenuItem1: TMenuItem;
  ComboBox3: TComboBox;
  XPManifest1: TXPManifest;
  Edit4: TEdit;
  ComboBox4: TComboBox;
  Label15: TLabel;
  Timer1: TTimer;
  Edit3: TEdit;
  Edit5: TEdit;
  procedure D1Click(Sender: TObject);
  function fact(chisl: integer) : extended;
  procedure Button1Click(Sender: TObject);
  procedure Timer1Timer(Sender: TObject);
private
```

```
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form4: TForm4;
  l,m,r,p0,p1,lo,ls,vo,vs,f:double;
implementation
{$R *.dfm}
procedure TForm4.D1Click(Sender: TObject);
begin
form4.Close;
end;
function Tform4.fact(chisl: integer) : extended;
var i:integer;
begin
result:=1;
for i:=1 to chisl do
result:= i* result;
end;
procedure TForm4.Button1Click(Sender: TObject);
begin
if (edit1.Text='') or ((edit2.Text='') and (edit4.Text='')) then
Application.MessageBox('Пожалуйста, введите данные','Ошибка', MB_OK)
else
begin
case combobox1.ItemIndex of
0:begin l:=strtofloat(edit1.Text); f:=1; end;
1:begin l:=strtofloat(edit1.Text)/60; f:=60; end;
2:begin l:=strtofloat(edit1.Text)/3600; f:=3600; end;
3:begin l:=strtofloat(edit1.Text)/86400; f:=86400; end;
4:begin l:=strtofloat(edit1.Text)/604800; f:=604800; end;
5:begin l:=strtofloat(edit1.Text)/2592000; f:=2592000; end;
6:begin l:=strtofloat(edit1.Text)/31536000; f:=31536000; end;
```

```
end;
if (edit2.Enabled=true) then
case combobox2.ItemIndex of
0:begin m:=1/strtofloat(edit2.Text); edit5.Text:=floattostr(roundto(m*f, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
1:begin m:=1/strtofloat(edit2.Text)/60;
edit5.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/60, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
2:begin m:=1/strtofloat(edit2.Text)/3600;
edit5.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/3600, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
3:begin m:=1/strtofloat(edit2.Text)/86400;
edit5.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/86400, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
4:begin m:=1/strtofloat(edit2.Text)/604800;
edit5.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/604800, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
5:begin m:=1/strtofloat(edit2.Text)/2592000;
edit5.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/2592000, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
6:begin m:=1/strtofloat(edit2.Text)/31536000;
edit5.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/31536000, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
end
else
case combobox4.ItemIndex of
0:begin m:=strtofloat(edit4.Text); edit3.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -3));
combobox2.ItemIndex:=0; end;
1:begin m:=strtofloat(edit4.Text)/60; edit3.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -3));
combobox2.ItemIndex:=1; end;
2:begin m:=strtofloat(edit4.Text)/3600; edit3.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -
3)); combobox2.ItemIndex:=2; end;
3:begin m:=strtofloat(edit4.Text)/86400; edit3.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -
3)); combobox2.ItemIndex:=3; end;
```

```
4:begin m:=strtofloat(edit4.Text)/604800; edit3.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -3)); combobox2.ItemIndex:=4; end;
5:begin m:=strtofloat(edit4.Text)/2592000; edit3.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -3)); combobox2.ItemIndex:=5; end;
6:begin m:=strtofloat(edit4.Text)/31536000; edit3.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -3)); combobox2.ItemIndex:=6; end;
end;
r:=l/m;
if r<1 then
begin
p0:=1-r;
p1:=1-p0;
lo:=sqr(r)/(1-r);
ls:=r/(1-r);
case combobox3.ItemIndex of
0:begin vo:=lo/l; vs:=ls/l; end;
1:begin vo:=lo/(l*60); vs:=ls/(l*60); end;
2:begin vo:=lo/(l*3600); vs:=ls/(l*3600); end;
3:begin vo:=lo/(l*86400); vs:=ls/(l*86400); end;
4:begin vo:=lo/(l*604800); vs:=ls/(l*604800); end;
5:begin vo:=lo/(l*2592000); vs:=ls/(l*2592000); end;
6:begin vo:=lo/(l*31536000); vs:=ls/(l*31536000); end;
end;
label9.Caption:=floattostr(roundto(p0*100, -2)) + '%';
label10.Caption:=floattostr(roundto(p1*100, -2)) + '%';
label11.Caption:=floattostr(roundto(lo, -2));
label12.Caption:=floattostr(roundto(ls, -2));
label13.Caption:=floattostr(roundto(vo , -2));
label14.Caption:=floattostr(roundto(vs, -2));
end
else
Application.MessageBox('Система массового обслуживания не имеет показателей, очередь растет до бесконечности.','Сообщение', MB_OK)
```

```
end;
end;
procedure TForm4.Timer1Timer(Sender: TObject);
begin
if edit2.Text<>'' then begin edit4.Enabled:=false; edit4.visible:=false; edit5.Visible:=true; combobox4.Enabled:=false; end else begin edit4.Enabled:=true; edit4.visible:=true; edit5.Visible:=false; combobox4.Enabled:=true; end;
if edit4.Text<>'' then begin edit2.Enabled:=false; edit2.visible:=false; edit3.Visible:=true; combobox2.Enabled:=false; end else begin edit2.Enabled:=true; edit2.visible:=true; edit3.Visible:=false; combobox2.Enabled:=true; end;
end;
end.

unit Unit5;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Menus, StdCtrls, Math, XPMan, ExtCtrls;
type
  TForm5 = class(TForm)
    MainMenu1: TMainMenu;
    D1: TMenuItem;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    Label1: TLabel;
    Label2: TLabel;
```

```
    Button1: TButton;
    Edit1: TEdit;
    Edit2: TEdit;
    ComboBox1: TComboBox;
    ComboBox2: TComboBox;
    ComboBox3: TComboBox;
    Label15: TLabel;
    Edit3: TEdit;
    Label16: TLabel;
    Label17: TLabel;
    Label18: TLabel;
    Label19: TLabel;
    Label20: TLabel;
    Label21: TLabel;
    Label8: TLabel;
    Label14: TLabel;
    XPManifest1: TXPManifest;
    Timer1: TTimer;
    Edit4: TEdit;
    ComboBox4: TComboBox;
    Label22: TLabel;
    Edit5: TEdit;
    Edit6: TEdit;
    function fact(chisl:integer): extended;
    procedure D1Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form5: TForm5;
```

```
  k,n:integer;
  l,m,p0,p1,q,r,a,ks,vs,vo,ls,lo,f:double;
implementation
uses Unit1, Unit2, Unit3, Unit4;
{$R *.dfm}
procedure TForm5.D1Click(Sender: TObject);
begin
form5.Close;
end;
function Tform5.fact(chisl:integer) : extended;
var i:integer;
begin
result:=1;
for i:=1 to chisl do
begin
result:= result * i;
end;
end;
procedure TForm5.Button1Click(Sender: TObject);
begin
if (edit1.Text='') or ((edit2.Text='') and (edit4.Text='')) or (edit3.Text='') then
Application.MessageBox('Пожалуйста, введите данные','Ошибка', MB_OK)
else
begin
case combobox1.ItemIndex of
0:begin l:=strtofloat(edit1.Text); f:=1; end;
1:begin l:=strtofloat(edit1.Text)/60; f:=60; end;
2:begin l:=strtofloat(edit1.Text)/3600; f:=3600; end;
3:begin l:=strtofloat(edit1.Text)/86400; f:=86400; end;
4:begin l:=strtofloat(edit1.Text)/604800; f:=604800; end;
5:begin l:=strtofloat(edit1.Text)/2592000; f:=2592000; end;
6:begin l:=strtofloat(edit1.Text)/31536000; f:=31536000; end;
end;
if (edit2.Enabled=true) then
```

```
case combobox2.ItemIndex of
0:begin m:=1/strtofloat(edit2.Text); edit6.Text:=floattostr(roundto(m*f, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
1:begin m:=1/strtofloat(edit2.Text)/60;
edit6.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/60, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
2:begin m:=1/strtofloat(edit2.Text)/3600;
edit6.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/3600, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
3:begin m:=1/strtofloat(edit2.Text)/86400;
edit6.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/86400, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
4:begin m:=1/strtofloat(edit2.Text)/604800;
edit6.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/604800, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
5:begin m:=1/strtofloat(edit2.Text)/2592000;
edit6.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/2592000, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
6:begin m:=1/strtofloat(edit2.Text)/31536000;
edit6.Text:=floattostr(roundto(1/strtofloat(edit2.Text)*f/31536000, -3));
combobox4.ItemIndex:=combobox1.ItemIndex; end;
end
else
case combobox4.ItemIndex of
0:begin m:=strtofloat(edit4.Text); edit5.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -3));
combobox2.ItemIndex:=0; end;
1:begin m:=strtofloat(edit4.Text)/60; edit5.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -3));
combobox2.ItemIndex:=1; end;
2:begin m:=strtofloat(edit4.Text)/3600; edit5.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -
3)); combobox2.ItemIndex:=2; end;
3:begin m:=strtofloat(edit4.Text)/86400; edit5.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -
3)); combobox2.ItemIndex:=3; end;
4:begin m:=strtofloat(edit4.Text)/604800; edit5.Text:=floattostr(roundto(1/strtofloat(edit4.Text),
-3)); combobox2.ItemIndex:=4; end;
```

```
5:begin                                                    m:=strtofloat(edit4.Text)/2592000;
edit5.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -3)); combobox2.ItemIndex:=5; end;
6:begin                                                    m:=strtofloat(edit4.Text)/31536000;
edit5.Text:=floattostr(roundto(1/strtofloat(edit4.Text), -3)); combobox2.ItemIndex:=6; end;
end;
k:=strtoint(edit3.Text);
r:=l/m;
if r/k<1 then
begin
p0:=1+r;
for n:=2 to k+1 do
begin
p0:=p0 + exp(n*ln(r))/fact(n);
end;
p0:=1/p0;
p1:=(exp(k+1*ln(r))/fact(k+1))*p0;
q:=1-p1;
lo:=((exp(k+1*ln(r)))*p0)/(k*fact(k)*sqr(1-(r/k)));
ls:=lo+r;
case combobox3.ItemIndex of
0:begin a:=l/q; vo:=lo/l; vs:=ls/l; if r<1 then a:=l end;
1:begin a:=l*60*q; vo:=lo/(l*60); vs:=ls/(l*60); if r<1 then a:=l*60 end;
2:begin a:=l*3600*q; vo:=lo/(l*3600); vs:=ls/(l*3600); if r<1 then a:=l*3600 end;
3:begin a:=l*86400*q; vo:=lo/(l*86400); vs:=ls/(l*86400); if r<1 then a:=l*86400 end;
4:begin a:=l*604800*q; vo:=lo/(l*604800); vs:=ls/(l*604800); if r<1 then a:=l*604800 end;
5:begin a:=l*2592000*q; vo:=lo/(l*2592000); vs:=ls/(l*2592000); if r<1 then a:=l*2592000 end;
6:begin a:=l*31536000*q; vo:=lo/(l*31536000); vs:=ls/(l*31536000); if r<1 then a:=l*31536000
end;
end;
if r<1 then
begin
p1:=0;
end;
label9.Caption:=floattostr(roundto(p0*100, -2)) + '%';
```

```
label10.Caption:=floattostr(roundto(p1*100, -2)) + '%';
label19.Caption:=floattostr(roundto(q*100, -2)) + '%';
label17.Caption:=floattostr(round(a));
label21.Caption:=floattostr(round(r));
label11.Caption:=floattostr(roundto(lo, -2));
label12.Caption:=floattostr(roundto(ls, -2));
label13.Caption:=floattostr(roundto(vo, -2));
label14.Caption:=floattostr(roundto(vs, -2));
end
else
Application.MessageBox('Система массового обслуживания не имеет показателей, очередь растет до бесконечности.','Сообщение', MB_OK)
end;
end;
procedure TForm5.Timer1Timer(Sender: TObject);
begin
if edit2.Text<>'' then begin edit4.Enabled:=false; edit4.visible:=false; edit6.Visible:=true; combobox4.Enabled:=false; end else begin edit4.Enabled:=true; edit4.visible:=true; edit6.Visible:=false; combobox4.Enabled:=true; end;
if edit4.Text<>'' then begin edit2.Enabled:=false; edit2.visible:=false; edit5.Visible:=true; combobox2.Enabled:=false; end else begin edit2.Enabled:=true; edit2.visible:=true; edit5.Visible:=false; combobox2.Enabled:=true; end;
end;
end.
```