Р Е Ф Е Р А Т

Пояснювальна записка до дипломного проекту: 89 с., 14 рис., 10 табл., 17 джерел, 20 електронних плакатів.

Об'єкт дослідження: Інформаційна система комунального підприємства.

Мета роботи: розроблення програмно-апаратних засобів інформаційної системи муніципального підприємства.

У проекті виконано: розроблення алгоритмів роботи підсистем, організація інформаційної взаємодії, формування звітних документів, розробка методичного забезпечення, розроблення рекомендацій з охорони праці.

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, БАЗА ДАНИХ, СИСТЕМА УПРАВЛІННЯ БАЗАМИ ДАНИХ, КАРТОТЕКА, ПОПЕРЕДНІЙ ПЕРЕГЛЯД, ДРУК, ЗВІТ, АЛГОРИТМ, ПРОГРАМА.

Умови одержання дипломного проекту

93400 м. Сєвєродонецьк, пр.Центральний 59«А», СНУ ім. В.Даля

ЗМІСТ

ВСТУП

Сьогодні можна говорити про інформаційну стадію розвитку цивілізації, оскільки якість і ефективність практичного використання інформації, яка є одним з важливих стратегічних державних ресурсів, відкриває нові горизонти для її прогресу і підвищення рівня життя людей, обґрунтовує зміни в соціально-культурній, науковій, загальноосвітній і інформаційній інфраструктурах суспільства. У контексті цього Україна, безперечно, не може залишатися збоку коли йдеться про розв'язку проблеми забезпечення загальнодоступності інформації.

Одним з чинників прискорення інформаційної стадії розвитку цивілізації є широке впровадження засобів обчислювальної техніки у всі сфери діяльності людини з метою значного підвищення продуктивності праці, ефективності і якості рішень, що приймаються, і виробів, що випускаються.

Сучасні персональні комп'ютери стають усе більш пристосованими для впорядковування і формалізації професійних знань в самих різних областях людської діяльності.

Учені, інженери, проектувальники, конструктора, бухгалтера, економісти, диспетчери і багато інших фахівців, що зайняті в науці, на виробництві і в управлінні, дістали можливість полегшити і в той же час в значній мірі упорядкувати і інтенсифікувати свою трудову діяльність. Не дивлячись на те, що засоби програмування, складові базове програмне забезпечення персональних професійних ЕОМ, як правило, досить розвинені для широкого їх використання непрофесіоналами в області програмування, потрібна адаптація програмних засобів до професійної сфери діяльності користувача. Іншими словами, користувачеві має бути забезпечений такий режим роботи на ЕОМ, що б він міг оперувати поняттями і правилами, які йому звичні і складають суть його професії.

На першому етапі зазвичай автоматизується рутинна робота, наприклад підготовка тексту звітного документа, складання звідного звіту, балансу і тому подібне Наступний, складніший етап – розробка спеціальних, переважно не процедурних, призначених для користувача мов. З їх допомогою фахівець може самостійно виконувати на ЕОМ певні дії, наприклад складати зведення, проводити аналіз зв'язаних груп економічних показників, займатися планеруванням і так далі Нарешті, на третьому етапі – і це найбільш процес, що складно формалізується, – може бути частково автоматизована і творча діяльність. ЕОМ в цьому випадку виступає вже в ролі “кваліфікованого охоронця” ряду фактів з даної наочної області. При роботі над деякою проблемою, наприклад аналізом господарського стану об'єкту управління, персональна ЕОМ підказує альтернативні рішення або пропонує вже відомі рішення в аналогічних ситуаціях і так далі

На всіх трьох етапах користувачі активно використовують персональні професійні ЕОМ в своїй безпосередній роботі. Для ефективної роботи користувача і ЕОМ розробляється так звані автоматизовані робочі місця, упорядковуються і класифікуються використовувані дані, тобто створюються відповідні бази даних. Ці бази проектуються простими по структурі і зручними для перебудови самими ж користувачами, що не мають спеціальних навиків в програмуванні. Головним чинником успішного функціонування автоматизованого робочого місця є наявність спеціальних діалогових мов високого рівня, що забезпечують зручний для людини контакт з ЕОМ (“дружній інтерфейс”). Необхідно звести до мінімуму можливість виникнення таких ситуацій, коли користувач вимушений звертатися для продовження роботи до спеціальної літератури. Також поважно, аби користувач спілкувався з ЕОМ, в основному формулюючи, що потрібно зробити, а не як треба зробити, що б отримати бажані результат. Ці мови мають бути, по суті справи, вхідними мовами деяких спеціально розроблених пакетів прикладних програм, що

забезпечують уведення-виведення необхідних даних, введення бази даних, діалог “людина - ЕОМ”.

У «Рубіжанськом теплокомуненерго» в рамках створення єдиної інформаційно-аналітичної системи розробляється пакет програм для обліку платежів за надане теплопостачання. Одна з підсистем повинна виконувати автоматизоване ведення обліку платежів житлового фонду міста за спожите тепло на протязі всього опалювального сезону.

1 АНАЛІЗ ТА ОБҐРУНТУВАННЯ РОЗРОБКИ

1.1 Сучасний стан житлово-комунального господарства

Житлово-комунальне господарство (ЖКГ) – це важлива соціальна галузь, яка забезпечує населення, підприємства та організації необхідними житлово-комунальними послугами, суттєво впливає на розвиток економіки країни.

У галузі функціонує кілька тисяч підприємств та організацій різних форм власності, які надають населенню понад 40 видів послуг на суму більше 6 млрд. грн. щороку. Тут використовується майже чверть основних фондів держави, зайнято 5% працездатного населення країни.

Прискорений розвиток житлового будівництва та неадекватне йому інженерне забезпечення житлово-комунального господарства, доступність та дешевизна енергоносіїв, що стали звичними у роки СРСР, створили ситуацію практично повного ігнорування питань енергозбереження та запровадження ефективних маловитратних технологій у житлово-комунальній сфері. Наприкінці 80-х років держава надавала житлово-комунальні послуги населенню фактично безкоштовно, громадяни оплачували лише 2% вартості фактично спожитих послуг [1].

Однак в умовах переходу до ринкової економіки держава більше не могла собі дозволити утримання всього житлово-господарського комплексу, особливо при різкому подорожчанні енергоносіїв на початку 90-х років. Тому одним з найактуальніших завдань того періоду було проведення реформи тарифної політики у галузі.

Вперше житлово-комунальних послуг подорожчали наприкінці грудня 1991 року - це було перше з п’яти підвищень тарифів впродовж 1992-1994 рр. Всі вони були зумовлені, насамперед, високими темпами інфляції, що мала місце на той час в економіці країни. Небачене досі

зростання цін на всі товари та послуги спричинило подорожчання і житлово-комунальних послуг. Однак, навіть за таких темпів підвищення частка вартості послуг, яку сплачувало населення, все ще залишалася мізерною, що змушувало державу продовжувати виплачувати дотації підприємствам житлово-комунального сектора економіки.

Фінансова допомога житлово-комунальному господарству з боку держави складала у 1994 році близько 8% усіх бюджетних витрат, або 4,4% валового внутрішнього продукту України. В той же час, частка споживачів в оплаті послуг становила лише близько 4% від їх реальної вартості [1].

Звісно, що така система надання державою дотацій виявилась нежиттєздатною і стала гальмом на шляху соціально-економічних перетворень в Україні. Саме тому в лютому 1994 року було розпочато реалізацію програми реформування житлово-комунального господарства, в рамках якої, зокрема, передбачалося поетапне запровадження повної оплати житлово-комунальних послуг сім'ями з достатніми доходами з тим, щоб основний фінансовий тягар на покриття витрат житлово-комунального сектора поступово перенести з держави на споживачів цих послуг.

Першим етапом реалізації програми було досягнення 60-ти відсоткового рівня відшкодування витрат. З цією метою у 1995 р. Тарифи підвищувались декілька разів. Так, наприклад, у Києві фактично за рік, з січня 1995 по січень 1996 року, тарифи на житлово-комунальні послуги зросли в 10-30 разів. Зокрема, на послуги водопостачання - в 27,4 рази, на гарячу воду – в 21,5 рази, на центральне опалення – в 20,5 рази. Якщо на початку 1995 року в середньому по Україні населення відшкодовувало 6-8% реальної вартості послуг, то на початку 1996 р. обсяг відшкодування становила вже 60% [1].

Разом з тим, знецінені в реальному вимірі заробітна плата та інші доходи населення не давали змогу вчасно і в повному обсязі розраховуватися за спожиті послуги. Почала накопичуватися

заборгованість. Так, якщо загальна сума заборгованості по платежах на 1 квітня 1995 року становила 5,3 трлн. крб. (0,053 млрд. грн.), то на початку 1996 року вона сягнула 30 трлн. крб. (0,3 млрд. грн.), збільшившись за період в 5,7 разів. Більше половини сімей (54,7%) були боржниками, а їх кількість зросла на 12,6%.

Підвищення тарифів на житлово-комунальні послуги спричинило серйозні проблеми для малозабезпечених сімей, неспроможних оплачувати послуги за новими тарифами. Саме тому в 1995 році за ініціативою Президента України з метою створення надійної "сітки безпеки" для малозабезпечених родин було запроваджено систему соціального захисту населення - програму житлових субсидій. Цією програмою було передбачено надання безготівкової допомоги на оплату житлово-комунальних послуг сім'ям з низькими доходами.

З 1 травня 1995 року за умови призначення житлової субсидії сім'ї за житлово-комунальні послуги мали сплачувати не більше 15% свого середньомісячного сукупного доходу. Вже в травні 1995 року 756 відділів житлових субсидій по всій Україні почали прийом заяв на призначення допомоги. Протягом 1995 року було розширено надання субсидій на деякі додаткові види послуг. З 1 червня 1995 року було впроваджено субсидії на оплату скрапленого газу та твердого палива, а з 1 листопада 1995 року до послуг, на які надається субсидія, включено електроенергію.

Протягом 1996 року тарифи на житлово-комунальні послуги підвищувались декілька разів для забезпечення 80-ти відсоткового рівня відшкодування витрат. Так, зокрема, майже вдвічі подорожчала квартплата, у 1,5-2 рази було підвищено тарифи на комунальні послуги.

В цей період тарифи на житлово-комунальні послуги зростали випереджаючими темпами порівняно з цінами на інші споживчі товари та послуги. Цю тенденцію вперше було змінено у 1997 році, коли ціни на споживчі товари та послуги зросли на 10,1%, а тарифи на житлово-комунальні послуги підвищилися лише на 0,9%. Разом з тим, гострою

проблемою продовжувала залишатися заборгованість населення, загальна сума якої на кінець 1997 року досягла 3,4 млрд. грн. Частка боргу місцевих бюджетів за призначені субсидії в загальній сумі заборгованості склала в кінці 1997 року 15%. Решта – 85% - була боргом населення, який також містив в собі заборгованість місцевих бюджетів за надані пільги по сплаті житлово-комунальних послуг [1].

У 1997 році середній рівень сплати населення склав 75%, а місцевих бюджетів за субсидіями – 63% від нарахованих сум. Борг середньостатистичної української сім'ї досяг наприкінці 1997 року позначки 6,8 місяців (збільшившись за рік на 3,1 місяця). Дещо вищим порівняно з населенням був наприкінці року термін заборгованості місцевих бюджетів за субсидіями – 8,6 місяця, більшою була впродовжроку і амплітуда його зростання (3,7 місяця).

Впродовж 1998 року розмір відшкодування населенням вартості житлово-комунальних послуг не змінювався і становив 80%. Збільшення тарифів на електроенергію та газ в цей період практично не позначилося на рівні цін на житлово-комунальні послуги. Практично всі збитки від цього підвищення понесли підприємства ЖКГ.

З метою вдосконалення практики фінансового та капітального планування підприємств і забезпечення реформування галузі в період 1995- 1999 років відбулася майже повна передача державних об'єктів житлово-комунального господарства в комунальну власність територіальних громад, виконавчим органам місцевого самоврядування було делеговано встановлення тарифів на комунальні послуги підприємств комунальної власності, а також розроблено інструкцію планування витрат, які підприємства комунальної галузі всіх форм власності можуть включати в повному обсязі в собівартість реалізованих послуг при розрахунку тарифів тощо.

Новий етап розвитку житлово-комунального господарства розпочався з підписанням Президентом України Указу "Про прискорення

реформування житлово-комунального господарства" від 19 жовтня 1999 р., яким були затверджені Основні напрями реформування житлово-комунального господарства.

Основними напрямами було визнано поглиблення демонополізації та розвиток конкурентного середовища у галузі, удосконалення системи управління житлово-комунальним господарством через його реструктуризацію та запровадження договірних відносин між споживачами і виробниками послуг із забезпеченням їх правового захисту, а також реформування системи фінансування житлово-комунального господарства.

В ході реалізації Указу у 2000 р. були проведені розрахунки з погашення заборгованості бюджетів усіх рівнів підприємствам житлово-комунального господарства за послуги, спожиті бюджетними установами і організаціями, субсидії та пільги населенню всього на суму 1,8 млрд. грн.

На початку 2015 року було посилено адресність надання житлових субсидій через врахування майновогостановища сім'ї при визначенні права на допомогу. Крім того було призупинено та частково обмежено дію пільг на оплату житлово-комунальних послуг, які надавалися певним категоріям громадян [1].

Впровадження цих заходів дещо нівелювало негативні наслідки зростання тарифів у 1999-2015 р. Так, кількість сімей - учасників програми житлових субсидій впродовж 2000 р. не збільшилась і становила як і в попередні роки 20-23% загальної кількості сімей в Україні.

За рахунок відмови держави від дотування вартості житлово-комунальних послуг для всіх сімей та завдяки тому, що заможні сім'ї стали оплачувати повну вартість послуг, навантаження на бюджет у 1999 році зменшено орієнтовно на 2 млрд. грн. При цьому всі сім'ї, які є учасниками програми житлових субсидій, як правило, систематично і в повному обсязі оплачують житлово-комунальні послуги. Власне, завдяки цій програмі рівень оплати послуг населенням поступово почав зростати.

Впровадження адресної програми житлових субсидій дозволило Україні найшвидше серед країн СНД суттєво збільшити рівень відшкодування населенням тарифів на житлово-комунальні послуги, що стало першим кроком на шляху переведення житлово-комунального сектору економіки на ринкові рейки.

Рівень сплати населення за спожиті житлово-комунальні послуги впродовж 1997-2015 років залишався стабільним: несплаченою булла четверта-п’ята частина нарахувань. Зростання терміну заборгованості населення впродовж трьох попередніх років було однаковим (3 місяці), і лише протягом 2000 року цей показник зріс на 2,1 місяця. В цілому, за рахунок накопичення заборгованості за попередні роки борг середньостатистичної української сім’ї за спожиті житлово-комунальні послуги на початку 2016 року становив майже 12 місяців.

На 1 січня 2001 року заборгованість населення досягла 6,2 млрд. грн. В наступні два роки вона збільшилась ще на 19% і на 1 січня 2004 р. склала 7,45 млрд. грн. Водночас протягом 2001 – 2003 років вдалося скороти термін цієї заборгованості до 9 місяців, збільшити рівень оплати послуг ЖКГ населенням з 79% у 2000 р. до 96,4% у 2003 р.

З метоюствореннія умов для погашення цієї заборгованості у лютому 2003 р. був прийнятий Закон України “Про реструктуризацію заборгованості з квартирної плати, плати за житлово-комунальні послуги, спожиті газ та електроенергію” [1].

Згідно з Законом заборгованість з квартирної плати та плати за житлово-комунальні послуги громадян реструктурується на термін до 60 місяців залежно від суми боргу та рівня доходів громадян на дату реструктуризації.

На початок 2014 року в різних регіонах України рівень відшкодування населенням вартості комунальних послуг коливався в межах: на послуги водовідведення – від 30% у м. Севастополі до 100% у м. Києві та Одеській області, на послуги водопостачання – від 28% в АР Крим

до 100% у м. Києві, на послуги теплопостачання – від 67% у м. Севастополі до 100% у 12-ти областях.

Протягом 2016-2017 років майже трьохкратне перевищення темпів росту витрат житлово-комунальних підприємств (на 47%) над темпами росту тарифів (на 16%) на послуги ЖКГ призвело до ще більш збиткової роботи підприємств галузі. Від'ємна рентабельність склала 7,2%. У цей період вдалося зменшити дебіторську і кредиторську заборгованості по галузі (на 8 та 6% відповідно), збільшити майже на чверть обсяг інвестицій в основний капітал, за рахунок впровадження нових технологій досягти зростання продуктивності праці на 8%.

З цією метою з 1996 р. реалізується програма поетапного оснащення наявного житлового фонду засобами обліку та регулювання споживання води і теплової енергії.

Незадовільною є якість житлово-комунальних послуг. Найбільш проблемним є питання теплозабезпечення, якості та обсягів подачі води. Тепло у централізованій системі теплопостачання протягом опалювального сезону подається за мінімальними параметрами, порушується графік початку і закінчення опалювального сезону. Більше половини міст з населенням понад 100 тисяч чоловік забезпечується питною водою за графіком.

Місцеві органи влади мають віднайти ефективні заходи для зміцнення фінансово-економічного стану комунальних підприємств, підвищення рівня та якості надання житлово-комунальних послуг, поліпшення рівня управління житлово-комунальним господарством. Нестача коштів, зростання заборгованості споживачів – одна з найбільш гострих проблем коммунального господарства.

У 2001-2003 роках активізувався процес створення конкурентного середовища та розвитку договірних відносин у житлово-комунальному господарстві шляхом зміни форм власності діючих підприємств, залучення

приватного сектора і передачі об’єктів в експлуатацію за договорами на управління, в оренду або концесію.

З метою активізації проведення реформи житлово-комунального господарства Президентом України у березні 2002 р. створено окремий центральний орган виконавчої влади – Державний комітет України з питань житлово-комунального господарства. До основних завдань цього органу належить зокрема реалізація державної політики та здійснення міжгалузевої координації і функціонального регулювання з питань житлово-комунального господарства.

Функції ж управління об'єктами житлово-комунального господарства, транспорту і зв'язку та контролю за діяльністю відповідних підприємств Закон України "Про місцеве самоврядування" покладено на сільські, селищні та міські ради.

На виконання Указу Президента України № 1351/99 постановою Кабінету Міністрів України від 14.02.2017 було затверджено Програму реформування і розвитку житлово-комунального господарства на 2017 – 2015 роки та на період до 2017 року. Головними завданнями цієї програми були визначені:

– удосконалення системи управління підприємствами і організаціями житлово-комунального господарства всіх форм власності, розмежування функцій органів влади, підприємств - виробників послуг, споживачів житлово-комунальних послуг, розвиток ринкових відносин у галузі;

– поглиблення демонополізації житлово-комунального господарства, створення конкурентного середовища і ринку послуг, реструктуризація підприємств і організацій, формування єдиної соціальної та фінансової політики на території самоврядування, створення сільської комунальної служби в єдиній системі житлово-комунального господарства;

– зменшення витрат та втрат енергоносіїв у житлово-комунальному господарстві, проведення ефективної енергозберігаючої політики.

1.2 Передумови створення автоматизованої системи обліку платежів

Стан оплати споживачами житлово-комунальних послуг значною мірою впливає на розвиток всього житлово-комунального господарства.

Аналіз динаміки дебіторської та кредиторської заборгованості підприємств житлово-комунального господарства свідчить про уповільнення темпів її зростання у 2013 році. Разом з тим не досягнуто зменшення обсягів існуючих боргів [1].

У 2016 році обсяг дебіторської заборгованості підприємств житлово-комунального господарства загалом по Україні збільшився на 99 млн.грн., або на 1,2 відсотка (у 2014 році - на 4,5 відсотка). При досягненні позитивної динаміки в цілому по Україні у Кіровоградській області заборгованість зросла на 15,0%, Харківській – на 14,0%, Житомирській - на 9,6% , Львівській – на 9,7%. Разом з тим у Волинській області вона зменшилась на 25,0%, Запорізькій – на 17,2%, Луганській – на 18,5%, Сумській – на 37,7%.

Загальна сума дебіторської заборгованості підприємств галузі на 1 січня 2013 року складала 8,7 млрд. грн., у тому числі заборгованість госпрозрахункових підприємств підприємствам житлово-комунального господарства за отримані послуги - 1,8 млрд. грн.

Протягом 2016 року в цілому по Україні ця заборгованість зменшилась на 212,8 млн.грн., або на 10,4% (в 2006 році зазначена заборгованість зросла на 7,7%). Найбільш суттєве зменшення мало місце у Сумській області – на 68%, Луганській – на 54,8%, Волинській – 37,4%,

Кіровоградській – 12,6%. Але в 9 областях продовжується зростання боргів госпрозрахункових підприємств, а саме: у Запорізькій, Львівській, Полтавській, Харківській [1].

Як позитивне явище необхідно відзначити те, що в минулому році на 45,4 млн. грн., або на 14,2%, зменшилась заборгованість бюджетних установ та організацій за спожиті комунальні послуги, яка на 1 січня 2013 року склала 319,3 млн.грн. (у той час, як у 2006 році вона зросла на 11,4 відсотка), у тому числі:

– установи та організації, які фінансуються з державного бюджету - 84,4 млн. грн.(зменшення у 2016 році склало 28,7%);

– установи та організації, які фінансуються з місцевих бюджетів - 234,9 млн. грн. (зменшення у 2016 році склало 7,5%).

Незважаючи на зменшення темпів приросту, протягом останніх років кредиторська заборгованість підприємств житлово-комунального господарства має тенденцію до зростання: у 2016 році вона зросла на 544,7 млн.грн. (7,1%), у 2016 році - на 294,6 млн.грн. (3,6%) і станом на 1 січня 2017 року склала 8,6 млрд.грн.

Значне збільшення відбулось в Закарпатській (29,5%), Донецькій (26,3%), Івано-Франківській (25,6%) та Кіровоградській (20,4%) областях. В деяких регіонах вдалось досягти зменшення кредиторської заборгованості, а саме: в Сумській та Волинській областях більше ніж на 50%, Запорізькій, Полтавській, Одеській та Дніпропетровській - на 10-18%. В загальній сумі кредиторської заборгованості заборгованість підприємств житлово-комунального господарства за енергоносії становить 4,5 млрд.грн., або 52,7%, у тому числі за газ – 2,3 млрд.грн., за електроенергію - 2,1 млрд.грн., покупне тепло – 0,1 млрд. гривень.

У цілому по Україні у 2016 році вдалось майже в три рази зменшити темпи приросту заборгованості за енергоносії. Разом з тим існують значні коливання між регіонами. Якщо в Сумській області заборгованість зменшилась на 70,6%, то в Житомирській вона зросла на 88,0%, у тому

числі за електроенергію - удвічі. Крім Сумської області необхідно відзначити досвід роботи Волинської, Запорізької, Дніпропетровської областей щодо зменшення боргів за енергоносії, де в 2012 році вони зменшились на 60,0% - 21,2%.

На сьогодні найбільшим боржником перед підприємствами житлово-комунального господарства є населення [1]**.** Хоча останнім часом рівень оплати населенням житлово-комунальних послуг зріс з 79% у 2012 р. до 96% у 2016 р., заборгованість оплати досягла 7,7 млрд. грн. і продовжує зростати: у 2016 р. – на 1,3%, у першому кварталі 2017 р. – на 3,2%. Водночас рівень розрахунків значно різниться по регіонах. У першому кварталі найкраще оплачували послуги ЖКГ у м. Києві, Волинській, Одеській, та Чернігівській областях (92-104%), найгірше – у Запорізькій, Луганській, Миколаївській та Херсонській (78-83%). Дніпропетровська та Донецька області мають найбільший термін заборгованості населення (відповідно 12 та 13 місяців), що наполовину більше середнього по Україні. При цьому в Дніпропетровській області порівняно з Чернівецькою, Чернігівською та Вінницькою доходи населення в 1,5-1,6 рази вищі, а заборгованість на одного жителя вища в 5 разів (і це при менших на 10-20% тарифах).

Ситуація з розрахунками не відповідає динаміці номінальних грошових доходів населення, які за останні 4 роки зросли в 3,4 рази (номінальна зарплата – в 2,3 рази), тоді як заборгованість населення з оплати послуг ЖКГ збільшилась в 1,5 рази.

Протягом 2011-2016 років ріст доходів населення в 1,8 рази випереджав зростання вартості послуг ЖКГ населенню. Водночас у структурі витрат домогосподарств оплата послуг ЖКГ за 2012-2016 рр. знизилась з 9 до 8,3%, у грошовому обчисленні вона зросла на 23,5%, тоді як витрати на освіту, охорону здоров’я, відпочинок і культуру – в 3,3 рази, на будівництво, ремонт житла та вклади до банків – в 1,7 рази, на послуги зв’язку – в 1,6 рази, на придбання меблів та побутової техніки – в 1,5 рази.

Оскільки поточні нарахування по оплаті послуг ЖКГ зменшились, населення могло б спрямовувати більше коштів на погашення заборгованості минулих років. Механізм для цього був запроваджений у 2011 р. шляхом прийняття Закону України "Про реструктуризацію заборгованості з квартирної плати, плати за житлово-комунальні послуги, спожиті газ та електроенергію". Втім, реструктуризація заборгованості відбувається досить повільно. З дати набуття зазначеним Законом та відповідним рішенням Уряду чинності (1 липня 2012 р.) і до сьогодні реструктуризовано лише 9,8% заборгованості, в рамках реструктуризації погашено тільки 1% боргів. Темпи проведення реструктуризації у I кварталі цього року та II півріччі минулого є однаковими.

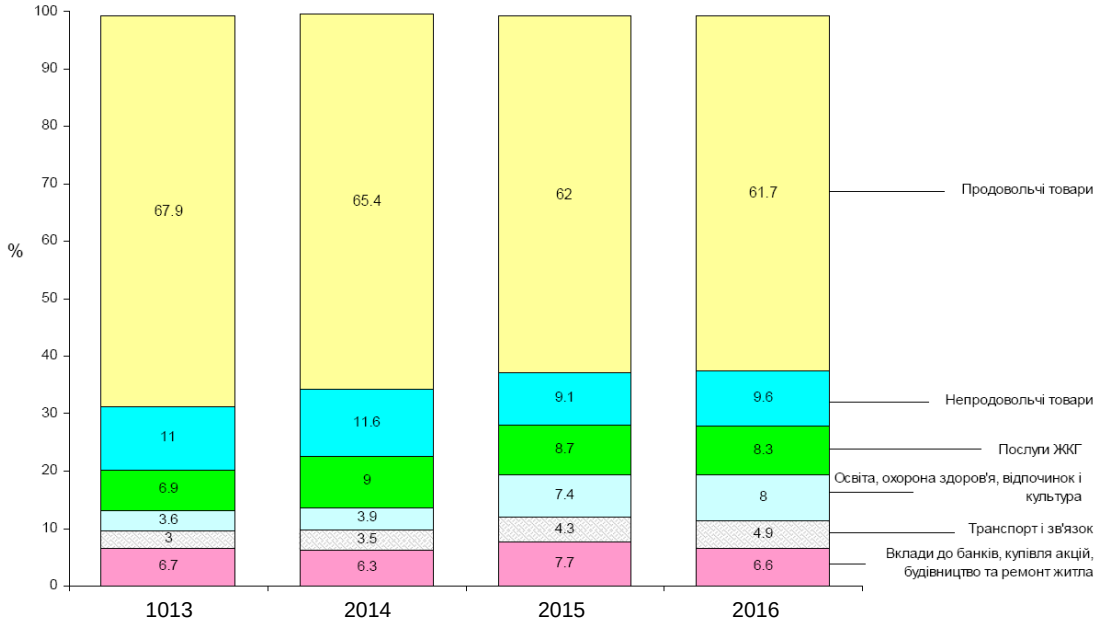

Рисунок 1.1 - Зміна структури сукупних витрат домогосподарств, 2013-2016 рр.

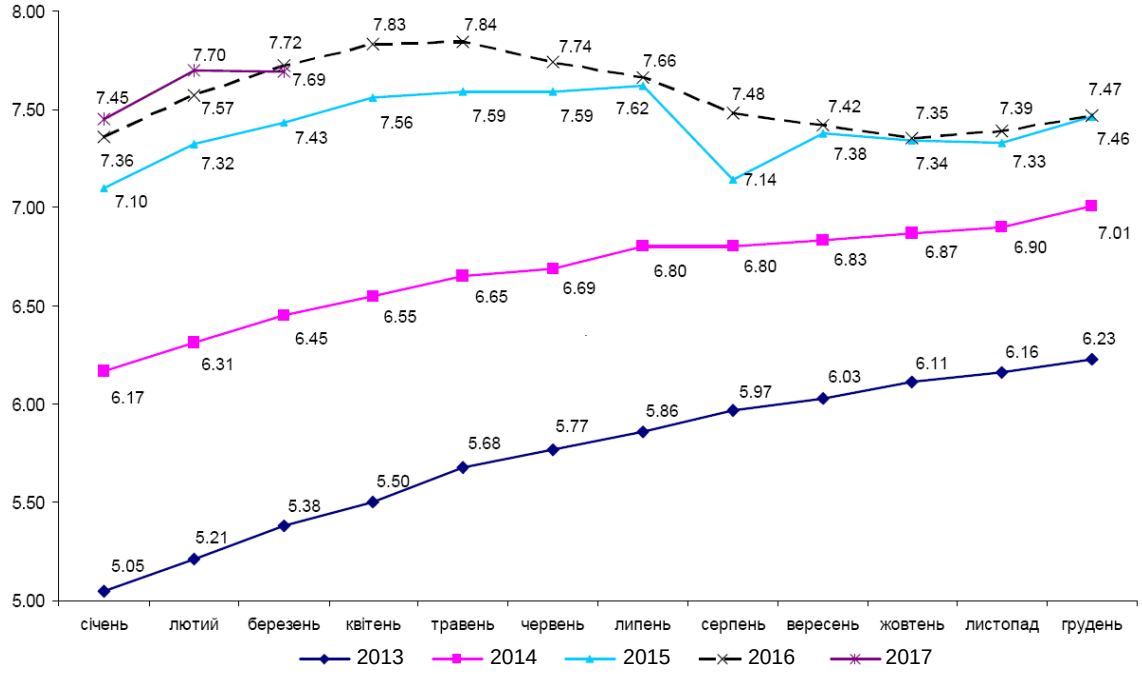

Рисунок 1.2 - Динаміка заборгованості населення по оплаті житлово-комунальних послуг за 2013-2017 роки, млрд.грн.

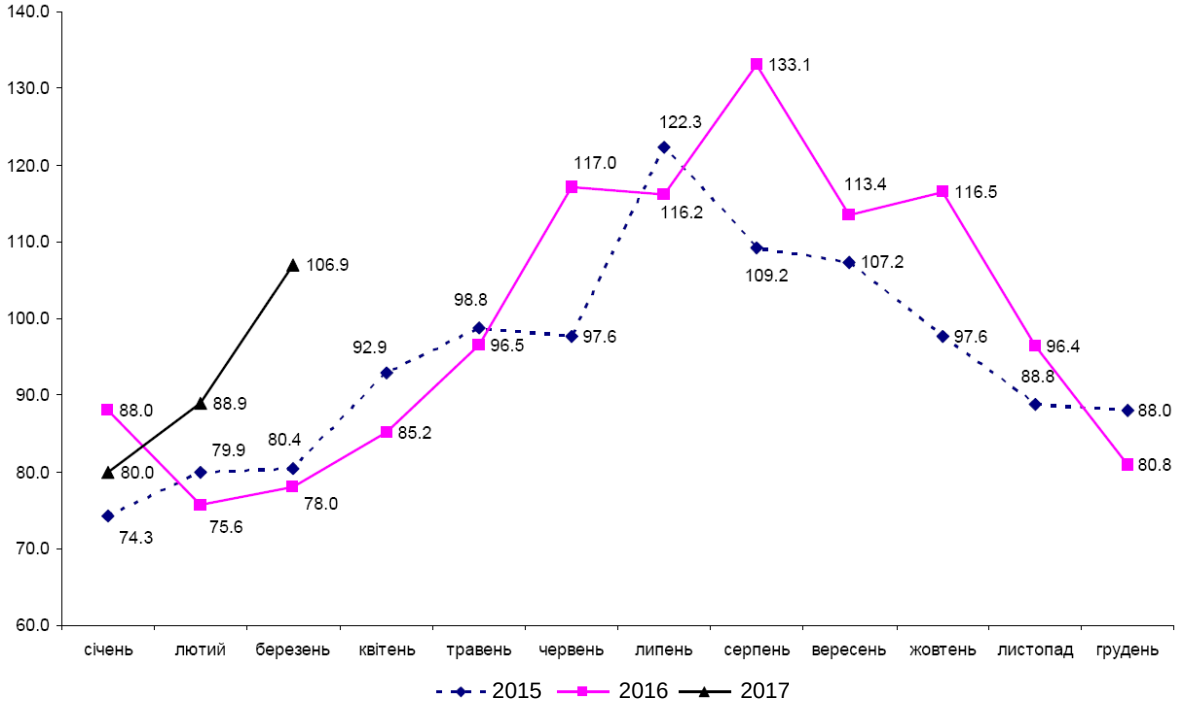

Рисунок 1.3 - Рівень оплати населенням житлово-комунальних послуг 2015-2017 роки,%

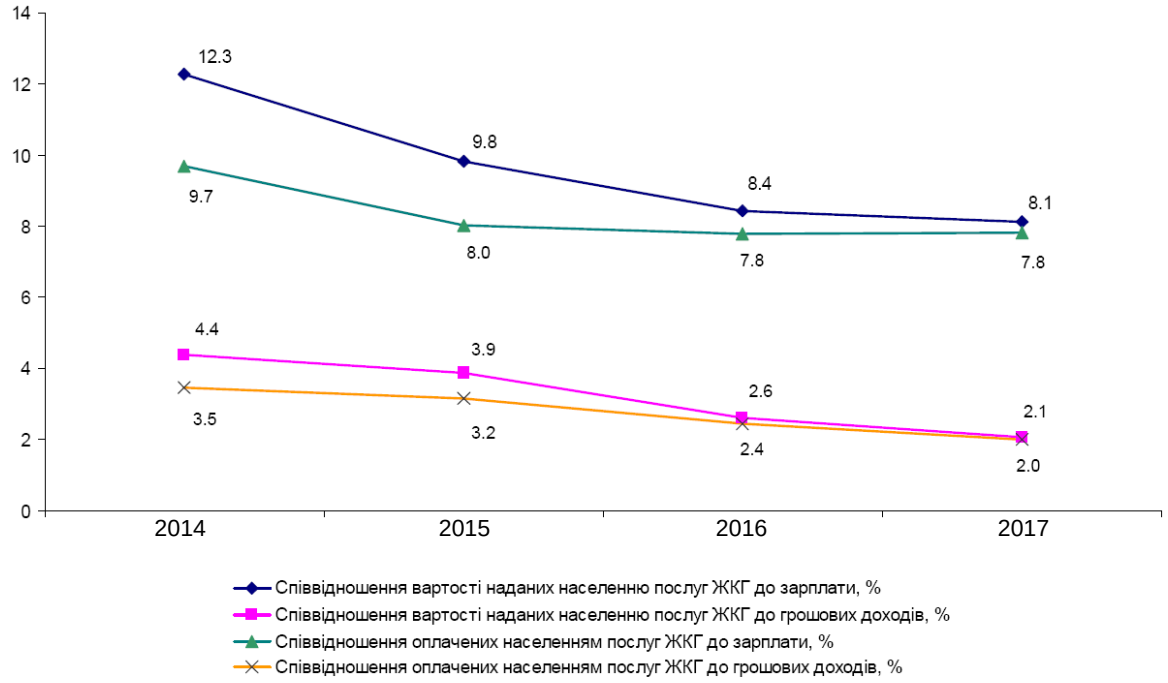

Рисунок 1.4 - Співвідношення доходів громадян та оплати послуг ЖКГ у 2014-2017 рр.

1.3 Огляд аналогів

У деяких містах України (наприклад м. Донецьк) створюються системи централізованого прийому платежів від населення. Наприклад, платіжна система "Город" є об'єднаною базою даних всіх заборгованостей населення: за комунальні послуги, тепло і електроенергію, телефонний зв'язок, охорону, навчання, штрафи - за все, чим живе місто. Вважається що для платників це зручна система здійснення платежів, а для постачальників послуг - швидкість вступу платежів і контроль над проходженням грошей. Такі системи функціонують на фінансовій і технологічній базі серйозних банків за підтримки адміністрації. Питань нарахування квартплати платіжна система "Город" не вирішує.

Програма “Управдом” («ТурбоБухгалтерия», м. Київ) призначена для автоматизації обліку комунальних платежів і розрахунку квартплати, а

також для ведення бухгалтерського обліку в товариствах власників житла, житлово-будівельних кооперативах, житлово-експлуатаційних конторах і тому подібне

Згідно вимогам Нового Житлового Кодексу вводяться нові правові форми для організацій власників житла Товариство Власників Житла (ТВЖ), що має на увазі ведення обліку в ТВЖ. Завдання програмного продукту «Управдом» допомогти керівникам ТВЖ в автоматизації обліку.

Функції «Блоку розрахунку комунальних платежів»:

– ведення картотеки будинків, квартир і нежитлових приміщень;

– формування кошторису витрат;

– автоматичний розрахунок тарифів поточних платежів на покриття витрат по комунальних послугах, послугах радіозв'язку, телебачення та інші;

– створення обмеженої кількості видів комунальних платежів;

– автоматичний розрахунок щомісячних платежів мешканців за споживані комунальні і експлуатаційні послуги;

– розрахунок на підставі свідчень індивідуальних лічильників, встановлених в квартирах;

– облік пільг, субсидій;

– автоматичне формування квитанцій на комунальні платежі;

– формування звітів по розрахунках з мешканцями (платниками), по пільговиках;

– аналіз і контроль заборгованості мешканців (платників).

Програма засобами «Блоку ведення бухгалтерського обліку» дозволяє вести в повному об'ємі бухгалтерський і податковий облік, банківські і касові операції, розрахунки з контрагентами. У програму включені системи для розрахунку зарплати співробітників, обліку основних засобів. Програма містить необхідні уніфіковані форми первинних документів, формує звіти для податкових органів, забезпечує експорт звітності у формати ФНС для здачі на магнітних носіях (дискетах).

"Управдом" є зручною програмою, в якій є всі необхідні функції для ведення обліку в домоволодінні всіх форм власності. Проте, адаптація програми "Управдома" під індивідуальні потреби підприємств ЖКХ коштує від 5 до 20 тис. грн. залежно від структури і складності підприємства ЖКХ.

Програма «Кварта-С» призначена для розрахунку і обліку квартплати, паспортного обліку в ТЗЖ, ЖКХ, управляючих компаніях (УК), а також розрахункових центрах, гаражних і дачних кооперативах, і інших організаціях, вирішальних завдання по проведенню періодичних розрахунків по нарахуванню і прийому платежів від населення. Програма підійде як для невеликого ТЗЖ, так і для крупної управляючої компанії, і може експлуатуватися як в локальному, так і в мережевому варіанті.

У одній базі даних - необмежене число ТЗЖ, ЖКХ, що зберігаються в довіднику організацій.

Сальдо і звороти, нарахування і оплати по кожному особовому рахунку і організації в цілому передбачені в двох розрізах:

– по місяцях нарахувань;

– по послугах (статтям нарахувань).

Розрахунок квартплати в програмі здійснюється відповідно до законодавства, з врахуванням пільг. Можливі нарахування за експлуатаційні і комунальні послуги, а також за будь-які інші послуги або цільові збори.

Користувач може самостійно додати нову послугу і задати тарифи для різних будівель. Автоматичний розрахунок нарахувань по всіх особових рахунках вибраної будови з врахуванням пільг, субсидій, перерахунку і недопостачі послуг виробляється в документі «нарахування». Нарахування можуть бути періодичними або разовими. Реалізовані нарахування за свідченнями квартирних і загальнобудинкових лічильників. Будь-яка з будівель, що враховуються, може мати свій набір

послуг і тарифів відповідно до категорії житла. Передбачений механізм щомісячного нарахування пені (до тих, пір доки не поступить оплата).

Можливі аванси, часткова оплата квитанції, а також залік суми, що поступила, як за всі нараховані послуги, так і лише за деякі послуги (вказані користувачем). Особові рахунки в програмі відкриваються для житлових і нежитлових приміщень, а також для парковочных місць і боксів в паркінгу. Довідник особових рахунків містить всю необхідну інформацію для нарахування квартплати. Користувач самостійно вводить нові і редагує наявні особові рахунки.

У програмі представлена безліч звітів, включаючи:

– оборотно-сальдові відомості з деталізацією по особових рахунках, по послугах, по місяцях нарахування, по документам (операціям), по будовам, по організаціям;

– довідки про пільговиків і сумарні пільги по категоріях;

– довідки про субсидії;

– звіти по боржникам;

– відомості оплати та інші.

Конфігурація «Кварта-С» призначена для нарахування квартплати і введення вступів від мешканців, тоді як бухгалтерія ведеться в типовій конфігурації 1С:Бухгалтерія. Вартість програми представлена в таблиці 1.1.

Таблиця 1.1 - Вартість програми «Кварта-С»

Найменування	Ціна, грн
Типова версія: (ліцензія на 5 будівель, ТЗЖ, ЖКХ)	3 100
Для управляючих компаній: (ліцензія на 50 будівель звідні звіти по організаціях)	9 100
Вартість ліцензії на необмежене число будов для крупних компаній, що управляють	за договором
Ліцензія на платформу 1С:Предприятие 8.1 отримується окремо:	
1С:Бухгалтерія 8 для України: (ліцензія на 1 робоче місце)	1 920

Додаткова ліцензія на одне робоче місце	1 080
Додаткова ліцензія на 5 робочих місць	3 744

Програма «Кварта-С» не має механізму автоматичного перерахунку із-за недопостачі послуг зі всієї будови або лише вказаних під'їздів (в разі перерви в постачанні послуги лише у вказаному під'їзді будівлі). Це не дозволяє розглядати програму «Кварта-С» як придатну для використання в РТКЕ.

Крім того, усі розглянуті аналоги орієнтовані на розрахунок квартплати і не враховують специфіки роботи підприємства ЖКГ з продажу і обліку теплопостачання споживачам. Тому ставиться завдання розробки підсистеми розрахунку витрат теплоенергії споживачам РТКЕ.

1.4 Технічне завдання

Актуальність розробки спеціалізованого програмного забезпечення для розрахунку витрат теплоенергії приватними споживачам полягає у тому, що в теперішній час рівень платежів постійно змінюється, змінюються тарифи на послуги, змінюються кількість пільгових категорій користувачів. Все це ускладнює облік та аналіз рівня розрахунків. Зараз немає сертифікованого програмного забезпечення, котре виконує подібній облік та аналіз і придатне для використання усіма підприємствами з продажу послуг населенню.

1.4.1 Функціональні вимоги

Розробці підлягає програмне забезпечення розрахунку витрат теплоенергії споживачам РТКЕ, яке охоплює:

– фіксування показників тепло лічильників, які встановлені на будинках, кварталах, та інших суб’єктів теплопостачання;

– встановлення періоду опалювального сезону с урахуваннях терміну реально наданих послуг по кожному об’єкту теплопостачання;

– фіксування щомісячних тарифів сплати;

– нарахування суми сплати послуг теплопостачання для кожного суб’єкта з урахуванням площі опалювання;

– формування звітів сплати наданих послуг теплопостачання;

– формування довідок про пільговиків і сумарні пільги по категоріях;

– надання довідок про субсидії;

– надання звітів по боржникам.

По своєму складу, вимоги, якими повинна володіти будь-яка програма складаються з вимог до функціональних характеристик, вимоги до показників надійності, складу і параметрам технічних і програмних засобів.

1.4.2 Робота з даними

Програма автоматизації розрахунку повинна забезпечувати виконання наступних функцій:

– використовувати БД у наступних режимах:

– створення основних таблиць БД (тарифи, картотека суб’єктів теплопостачання, та таке інше);

– підключення до основних таблиць БД;

– введення вихідної інформації;

– звернення до довідників таблиць;

– безпосередні розрахунки;

– робота з довідниками:
– створення довідкових таблиць БД (пільги, субсидії, та таке інше);
– підключення до довідників;
– можливість перегляду і редагування довідкової інформації;
– засоби виводу на друк довідкових даних.
– забезпечення зберігання інформації;
– забезпечення цілісності даних;
– видача коректних повідомлень;
– збереження особистих налаштувань користувача.

1.4.3 Вимоги до складу та параметрам технічних засобів

Технічне забезпечення повинне включати наступні компоненти:

– персональний комп’ютер повинен використовуватися типа IBM PC, який задовольняє наступним технічним вимогам:
– процесор Intel Pentim з тактової частотою 2ГГц і вище;
– об'єм оперативної пам'яті не менше 1 Гбайт;
– жорсткий диск ємкістю не менше 40 Гбайт;
– відеоадаптер і дисплей;
– маніпулятор типа ‘миша’;
– стандартна клавіатура;
– друкуючий пристрій.

Програмне забезпечення повинне включати:

– операційну систему сімейства Windows (бажано Windows XP/7);
– систему управління базою даних в якій реалізований проект.

2 ВИБІР ІНСТРУМЕНТАЛЬНИХ ЗАСОБІВ РОЗРОБЛЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

У даному розділі проводиться огляд існуючих систем управління базами даних. На підставі вимог, викладених в технічному завданні складається алгоритм функціонування програми. Проводиться аналіз і обґрунтування вибору середовища програмування.

2.1 Проектування бази даних

База даних (БД) - це сукупність даних, які описують деяку систему. У добре спроектованій БД міститися декілька таблиць, кожна з яких включає інформацію про однотипні об'єкти системи, властивості яких перетворяться в стовпці таблиці, звані на мові БД полями. Кожен параметр в системі представляється окремим рядком таблиці, званим записом.

Кожна таблиця локальної БД подібна до документа або електронної таблиці. Локальні таблиці зберігаються на жорсткому диску комп'ютера або централізований записуються на мережевий диск. Відмінність таблиці від документів або електронних таблиць полягає в тому, що вони можуть використовуватися декількома користувачами одночасно (тобто підтримують спільний доступ) [4].

Для управління спільним використанням даних, необхідний набір процедур для контролю і захисту доступу до записів і таблиць, наприклад, якщо двоє користувачів намагаються дістати доступ до одного і того ж запису для його зміни. Загальних методів для такого контролю не існує - кожен тип бази даних (локальної або видаленої) забезпечує виконання протоколу блокування своїм власним способом.

Одним з найбільш відомих і широко поширених підходів проектування баз даних є тип об'єкт / відношення. Цей підхід був заснований на запропонованій в 1976 році Ченом [4] "моделі об'єкт / відношення " (ER-модели) і з тих пір неодноразово удосконалився Ченом і багатьма іншими дослідниками.

Згідно визначенню, даному Ченом [4], об'єктом називається "предмет, який може бути чітко і однозначно ідентифікований". При цьому об'єкти підрозділяються на правильні об'єкти і слабкі об'єкти. Слабким об'єктом називається об'єкт, який знаходиться залежно від деякого іншого об'єкту, тобто він не може існувати, якщо не існує цей інший об'єкт.

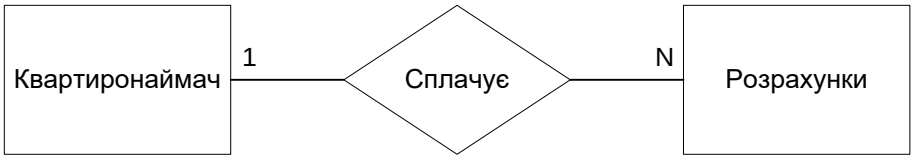

Рисунок 2.1 – Діаграма моделі об'єкт / відношення

Наприклад, на рисунку 2.1 об'єкти “Розрахунки” є слабкими, оскільки вони не можуть існувати (у контексті даної бази даних), якщо не існує відповідний об'єкт “Квартиронаймач”. Зокрема, якщо даний “Квартиронаймач” буде видалений, то всі залежні від нього об'єкти “Розрахунки” також будуть видалені. Правильним об'єктом називається об'єкт, який не є слабким: наприклад, правильними об'єктами можуть бути квартиронаймачі.

Об'єкти (і відношення) мають деякі властивості (їх також називають атрибутами). Всі об'єкти одного типу мають деякі загальні властивості: наприклад, всі квартиронаймачі мають ПІБ, дата народження, категорія населення й т.д. Значення властивостей кожного типу знаходяться у відповідній множини значень, що у реляційних термінах називається доменом. Нижче перераховані деякі види властивостей та їх особливості:

– проста або складена властивість (наприклад, складена властивість "ПІБ" може складатися із простих властивостей “прізвище”, “ініціали”);

– ключова властивість, унікальне в деякому контексті (наприклад, номер квартири, є унікальним тільки в контексті окремого будинку);

– базова або довільна властивість (наприклад, загальне число квартри мікрорайону може бути виведене на основі підсумовування окремого числа квартир, які розташовані у кожному будинку цього мікрорайону).

Відношення визначається як “асоціація об'єктів”. Наприклад, між квартиронаймачами й рахунками задане відношення (“Сплачує”), що означає, що заданий квартиронаймач здійснює оплату по декільком рахункам.

Розглянемо підхід на основі моделі об'єкт/відношення з іншого погляду. Ідеї, засновані на такому підході або близькі до них, становили неформальну основу при створенні формальної реляційної моделі. Загальний підхід до семантичного моделювання включає чотири етапи; ціль кожного з них можна коротко сформувати так:

– ідентифікувати корисні семантичні концепції;
– вивести формальні об'єкти;
– вивести формальні правила цілісності;
– вивести формальні оператори.

При проектуванні варто звернути увагу на те, що ці чотири етапи також застосовні для проектування базової реляційної моделі (і для будь-якої формальної моделі даних), а не тільки для "розширених" моделей типу об'єкт/відношення. Інакше кажучи, для того щоб створити (формальну) базову реляційну модель, проектувальник, насамперед, повинен розробити деякі (неформальні) "корисні семантичні концепції", засновані або на ідеях моделі об'єкт/відношення. Або на подібні їм концепціях.

Пропонується, щоб відносини використовувалися для моделювання й "об'єктів", і "залежностей". Але саме головне полягає в тому, що відносини є формальними об'єктами, а реляційна система - формальною системою. Модель об'єкт/відносини не є (або, по крайній мірі, не в першу

чергу) формальною моделлю [5]. Дійсно, вона переважно складається з набору неформальних концепцій, що відповідають (тільки) першому із чотирьох наведених вище етапів. Для проектування бази даних корисно мати "непробивні", тобто повністю ідентифіковані концепції, представлені на етапі 1. Однак немаловажним залишається той факт, що проектування бази даних не може бути виконане без формальних об'єктів і правил. Представлених на етапах 2 і 3, а багата кількість задач зовсім не можуть бути виконані без використання формальних операторів етапу 4. Термін "моделювання типу об'єкт/відношення" звичайно використовується для позначення процесу вибору структури бази даних.

Ставиться задача спроектувати бази даних для системи “Облік розрахунків”.

Число відносин у БД і конкретні атрибути, приписувані кожному відношенню, визначаються в процесі проектування. Декомпозиційний підхід до проектування, є цілком придатним за умови невеликого числа задіяних атрибутів. Якщо число атрибутів перевищує 20, то метод, заснований на декомпозиції, стає зайво громіздким і проектувальник повинен звернути увагу на інші методи проектування. Один з таких методів називається “сутність - зв'язок”. Він відрізняється від методу декомпозиції тим, що функціональні залежності залучаються не на початковому, а на кінцевому етапі проектування.

Сутність визначається як деякий об'єкт, що представляє інтерес для організації. Цей об'єкт повинен мати екземпляри, що відрізняють друг від друга й допускають однозначну ідентифікацію. Єдина визначальна ознака, що може допомогти в знаходженні сутностей, полягає в тому, що сутність - це, як правило, іменник.

У нашім випадку сутностями є:

– квартиронаймач;
– рахунки;
– РТКЕ;

– послуги;

– квартира.

Для проектування недостатньо поняття “сутність”. Таке поняття як “зв'язок” являє собою з'єднання між двома або більше сутностями. При пошуку зв'язків в основному варто покладатися на та обставину, що зв'язок звичайно виражається дієсловом. Типовими прикладами зв'язків між двома сутностями є:

– квартиронаймач сплачує рахунки;

– РТКЕ виставляє рахунки;

– квартиронаймач одержує послуги;

– РТКЕ надає послуги

– РТКЕ опалює квартиру;

– квартиронаймач має квартиру.

Тісно пов'язане з попереднє третє важливе поняття, як “атрибут”. Атрибут - це властивість сутності. Наприклад, атрибутами сутності квартиронаймач є: ПІБ, рік народження, домашня адреса, категорія населення та інші.

Визначення сутності, зв'язку й атрибута не відрізняються особливою конкретністю, однак є прийнятними для використання в тих цілях, на яких вони розраховані. Зв'язки, що існують між двома сутностями, можуть бути представлені декількома способами. Рисунок 2.2 ілюструє використання діаграми ER-Екземплярів.

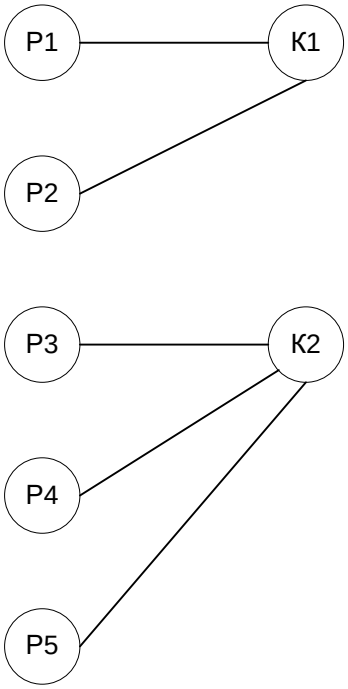

Рисунок 2.2 - Приклад діаграми ER - екземпляру

На цій діаграмі назви всіх сутностей поміщені над екземплярами цих сутностей і в них використані прописні букви, у той час як кожний екземпляр сутності ідентифікується значенням атрибута. Так РАХУНОК є сутністю, а Р1 – конкретним екземпляром сутності. Зв'язок також іменується, і його назва, складена із прописних букв, розміщається над екземплярами зв'язку, при цьому екземпляр кожного окремого зв'язку визначається стрічкою між тими двома екземплярами сутностей, які цей зв'язок з'єднує. Атрибут, або набір атрибутів, що використовується для ідентифікації екземпляра сутності, називається ключем сутності. Кожний екземпляр зв'язку однозначно визначається набором ключів сутностей, що з'єднуються цим зв'язком.

На діаграмах ER - типу, показаної на рисунку 2.3, сутності представляються у вигляді прямокутників, а зв'язки - у вигляді ромбів.

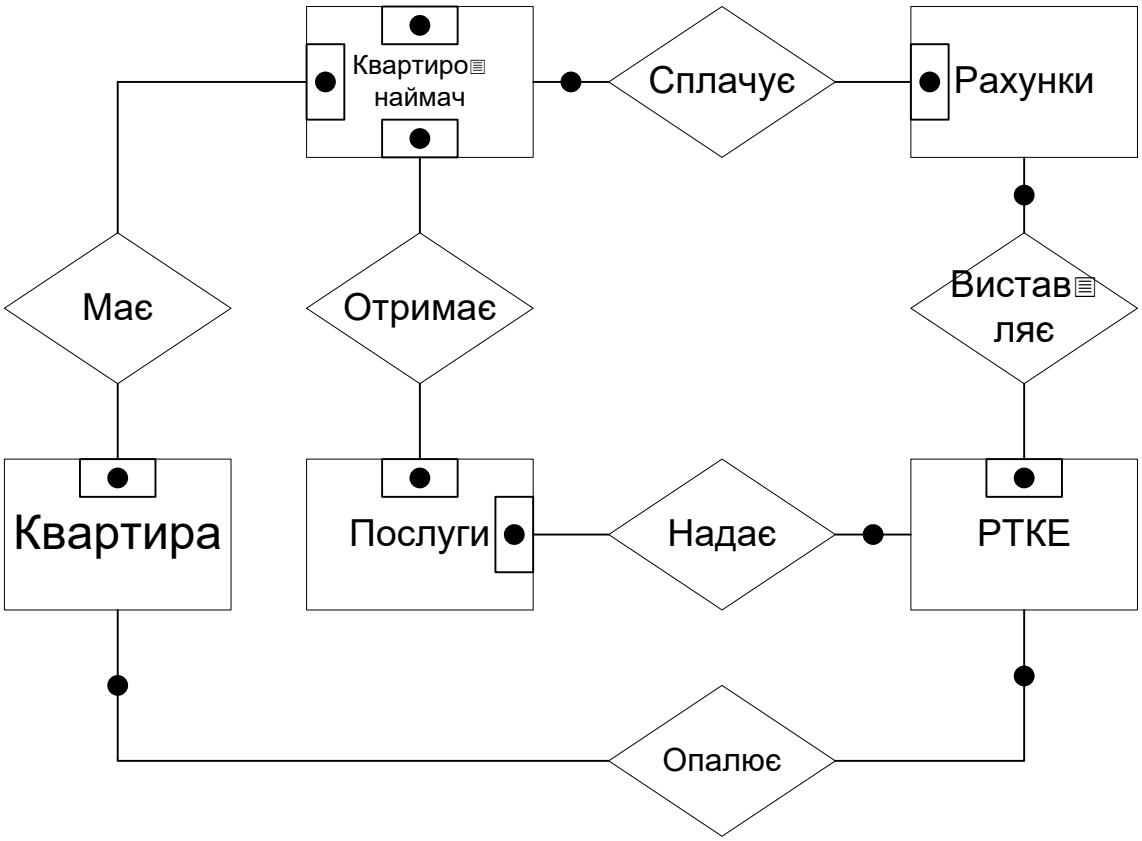

Рисунок 2.3 - Діаграма розробленої моделі “об'єкт/відношення”

У більшості випадків для визначення набору відносин проектованих БД використовуються діаграми ER - типу, а не діаграми екземплярів.

Важливою характеристикою зв'язку між двома (і більше) сутностями є ступінь зв'язку. Ступінь зв'язку показує, з якою максимальною кількістю екземплярів іншої сутності зв'язаний екземпляр даної сутності. Розрізняють наступні основні ступені зв'язку:

– один до одному (1:1);

– один до багатьох (1 : N);

– багато хто до багатьох (M : N).

Існує два класи приналежності: обов'язковий і необов'язковий. Якщо кожний екземпляр сутності пов'язаний з одним або більше екземплярами іншої сутності, то дана сутність має обов'язковий клас приналежності. Інакше якщо в сутності є екземпляри не пов'язані з іншими екземплярами, то клас приналежності необов'язковий. Класи приналежності відображені на діаграмі символом крапка.

На основі діаграми “об'єкт/відношення” складемо схему даних, де для кожної сутності задамо набір атрибутів, ступінь зв’язку та первинні ключі для забезпечення цілісності даних (рис. 2.4).

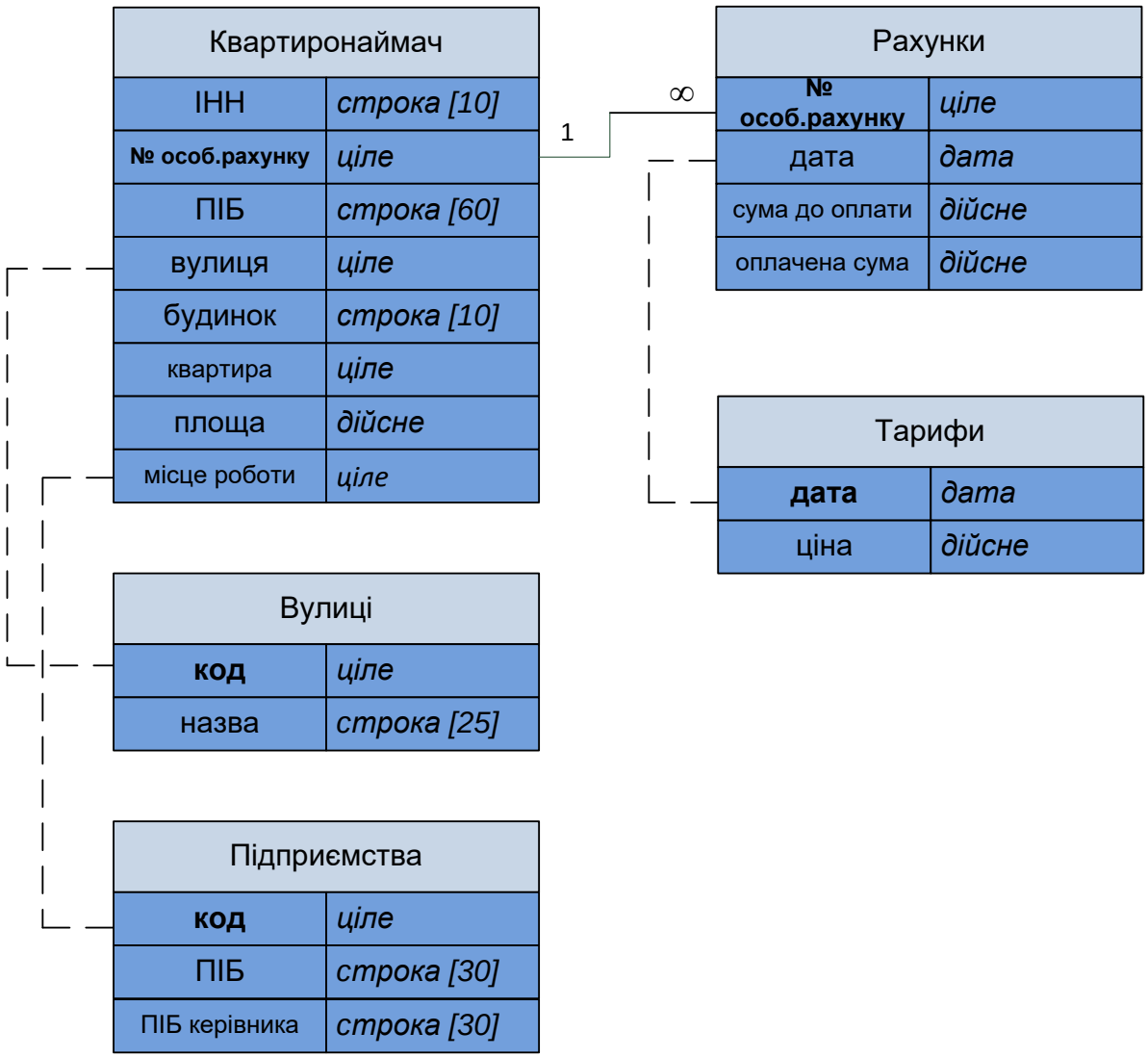

Рисунок 2.4 – Схема даних БД

Цілісність даних - одне з найважливіших вимог, пропонованих до баз даних. Для визначення цілісності даних у базі даних проекту можуть бути використані встановлені відносини між таблицями [5]. Перевірка цілісності даних може здійснюватися як програмними засобами, так і засобами бази даних. Для організації цілісності та унікальності зберігання даних використовуються первинні ключі. Під первинним ключем розуміють поле або набір полів, однозначно ідентифікуючий запис. Значення первинного ключа в таблиці БД повинне бути унікальним, тобто в таблиці не повинне існувати двох або більше записів з однаковим

значенням первинного ключа. Первинний ключ повинен бути мінімально достатнім: у ньому не повинне бути полів, видалення яких з первинного ключа не відіб'ється на його унікальності.

Після етапу проектування сутностей і визначення зв'язків між ними перейдемо до етапу розробки алгоритму роботи з даними, що зберігаються в сутностях.

2.2 Розробка алгоритму програми

Роботу програми “ Підсистема розрахунку витрат теплоенергії споживачами РТКЕ ” починається перевіркою цілісності використовуваної СУБД (блок 2 схеми алгоритму, наведеної на рис 2.5). Як показник працездатності можна використовувати звертання до системи драйверів використовуваного формату файлів БД. При виявленні помилки в СУБД програма виконує екстрене завершення роботи без збереження всіх даних (блок 3).

При успішному звертанні до системних драйверів СУБД підсистема ініціалізації виконує перевірку наявності всіх файлів даних. Список цих файлів визначається розроблювачем у процесі проектування системи й складається тільки із самих файлів даних. Таким чином,. індексні файли перевірці на повну наявність не підлягають. Навпроти, щораз при запуску програми планується проводити повну переіндексацію даних. Це пов'язане з тим, що якщо використовувати файл даних у відкритому стані й невідповідний до нього індексний, то відбудеться збій СУДБ, тому для забезпечення цілісного використання даних передбачає переіндексація даних.

Якщо підсистема ініціалізації виявила відсутність фалів даних, то виконується автоматичне створення нового файлу даних певної структури

(блоки 4-6). Далі відбувається створення індексних файлів (блоки 7-9). Первинні ключі використовуються для автоматичної перевірки на дублювання даних, а вторинні індекси необхідні для сортування даних під час роботи з ними. При успішному виконанні описаних дій ініціалізації, програма переходить до вибору режиму роботи (блоки 11-14).

Закриття програми супроводжується примусовим закриттям всіх активних таблиць даних (блоки 15, 16).

Елементи підсистеми «Довідники» містять дані, які використовуються при роботі з підсистемою «Картотека». До таких даних належать назви улиць, підприємств, номери мікрорайонів та таке інше. Робота з довідником зводиться до наступного алгоритму (рис.2.6).

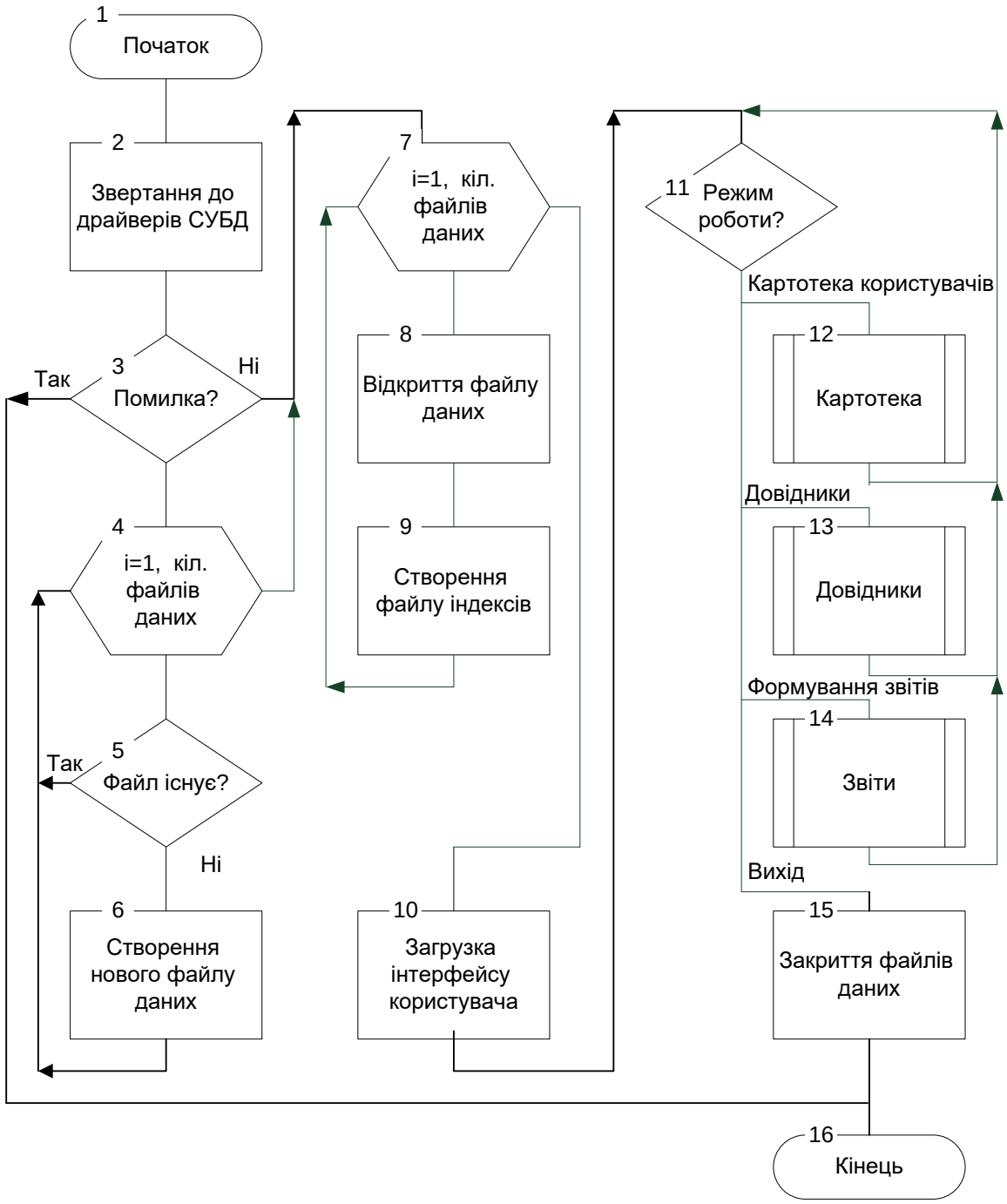

Рисунок 2.5 – Схема алгоритму основної програми

При бажанні користувача працювати з довідковою таблицею, організовується показ змісту всієї таблиці (блок 2) і стандартні засоби навігації (блоки 9, 10), такі як переміщення на перший, останній, наступний і попередній записи. Реалізуємо основні режими роботи з даними: засоби додавання нового порожнього запису (блоки 5, 6), засоби редагування обраного поля поточного запису (блоки 13, 14), видалень одного поточного або декількох відзначених записів (блоки 7,8). Також

передбачені засоби попереднього перегляду та друку змісту довідникової таблиці (блоки 11, 12). Ці методи відображення даних схожі між собою. Різниця полягає лише в пристрої відображення. У першому випадку таким пристроєм є віконна форма, а у другому випадку – пристрій друку.

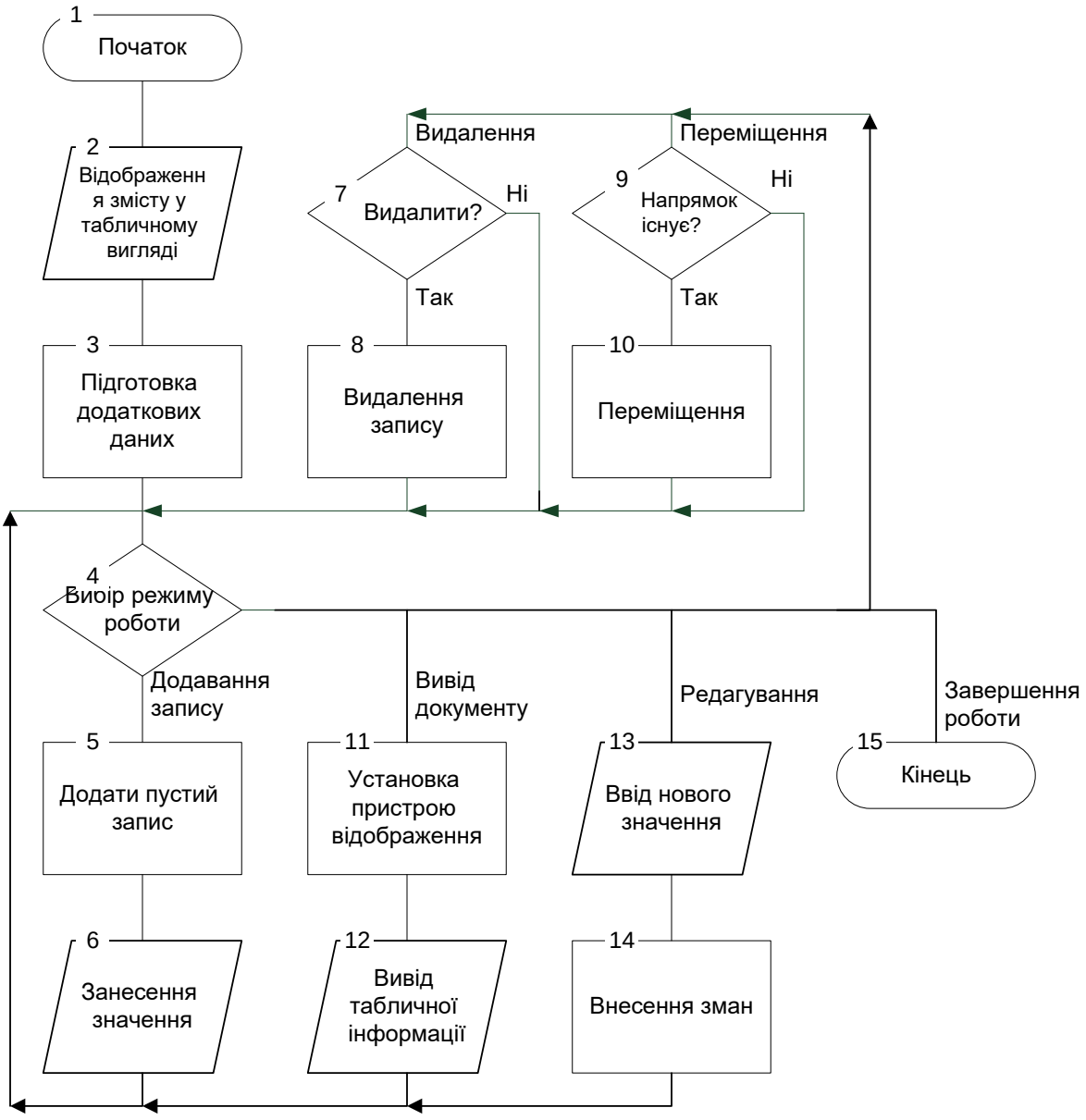

Рисунок 2.6 – Схема алгоритму підсистеми «Довідники»

Робота з основною підсистемою «Картотека» наведена на рис.2.7.-2.8. На початку роботи підсистеми виконується відображення змісту картотеки користувачів теплової енергії в табличному вигляді (блок 2 схеми алгоритму, наведеної на рис.2.7). Далі, в залежності от обраного

користувачем режиму роботи (блок 3), програма переходить до однієї з можливих гілок.

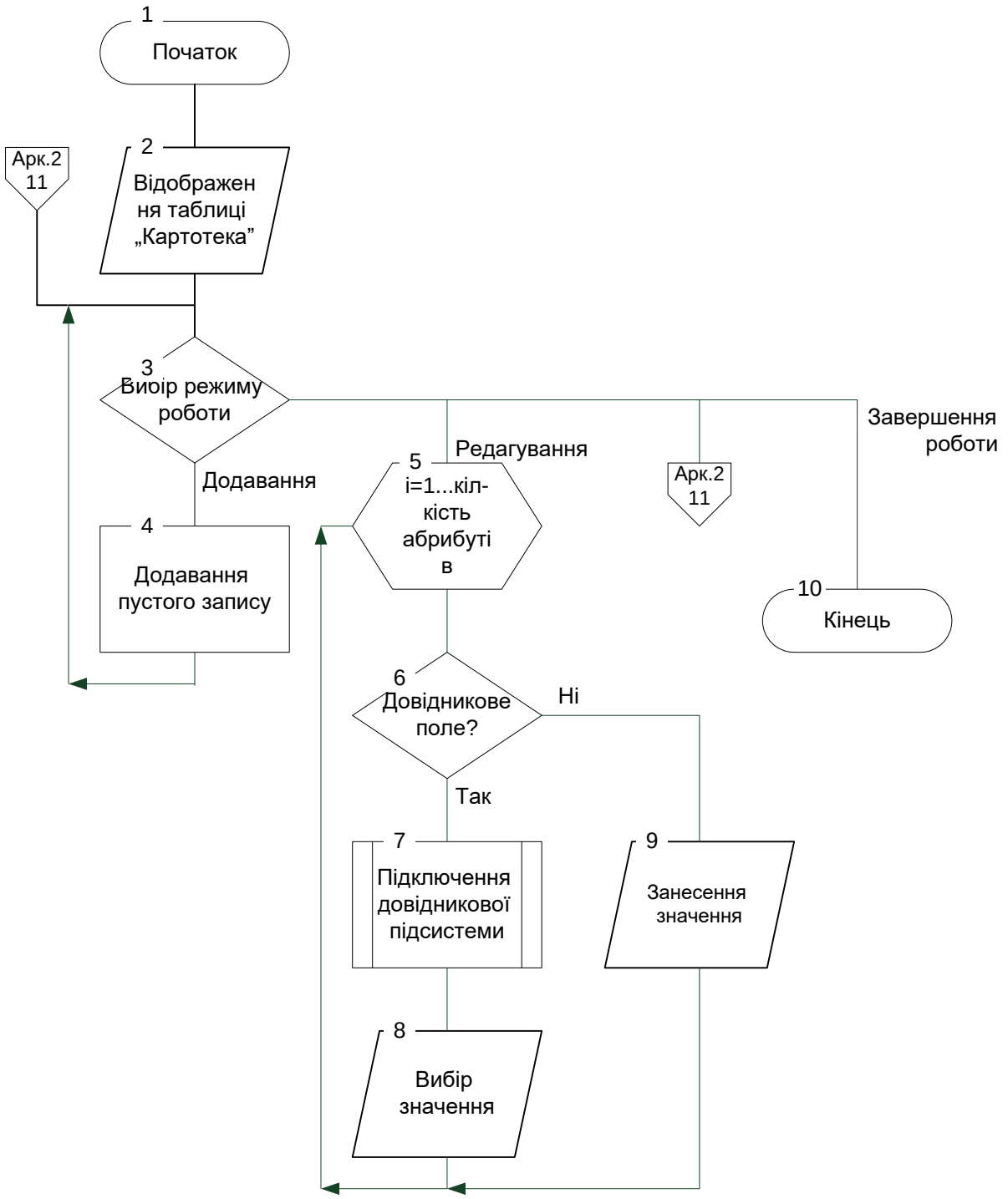

Рисунок 2.7 – Схема алгоритму підсистеми «Картотека». Аркуш 1

Додавання нового запису (блок 4) складається з безпосереднього додавання нового запису до таблиці та редагуванню цього рядка (блоки 5-10). В залежності від того, чи пов’язан атрибут з елементом довідникової

підсистеми чи ні вводимо два режиму редагування. Перший складається зі звертання до таблиці довідникової підсистеми, вибору значення з неї та занесенню обраного значення в атрибут таблиці «Картотека». Других засіб складається з безпосереднього занесення значення до таблиці.

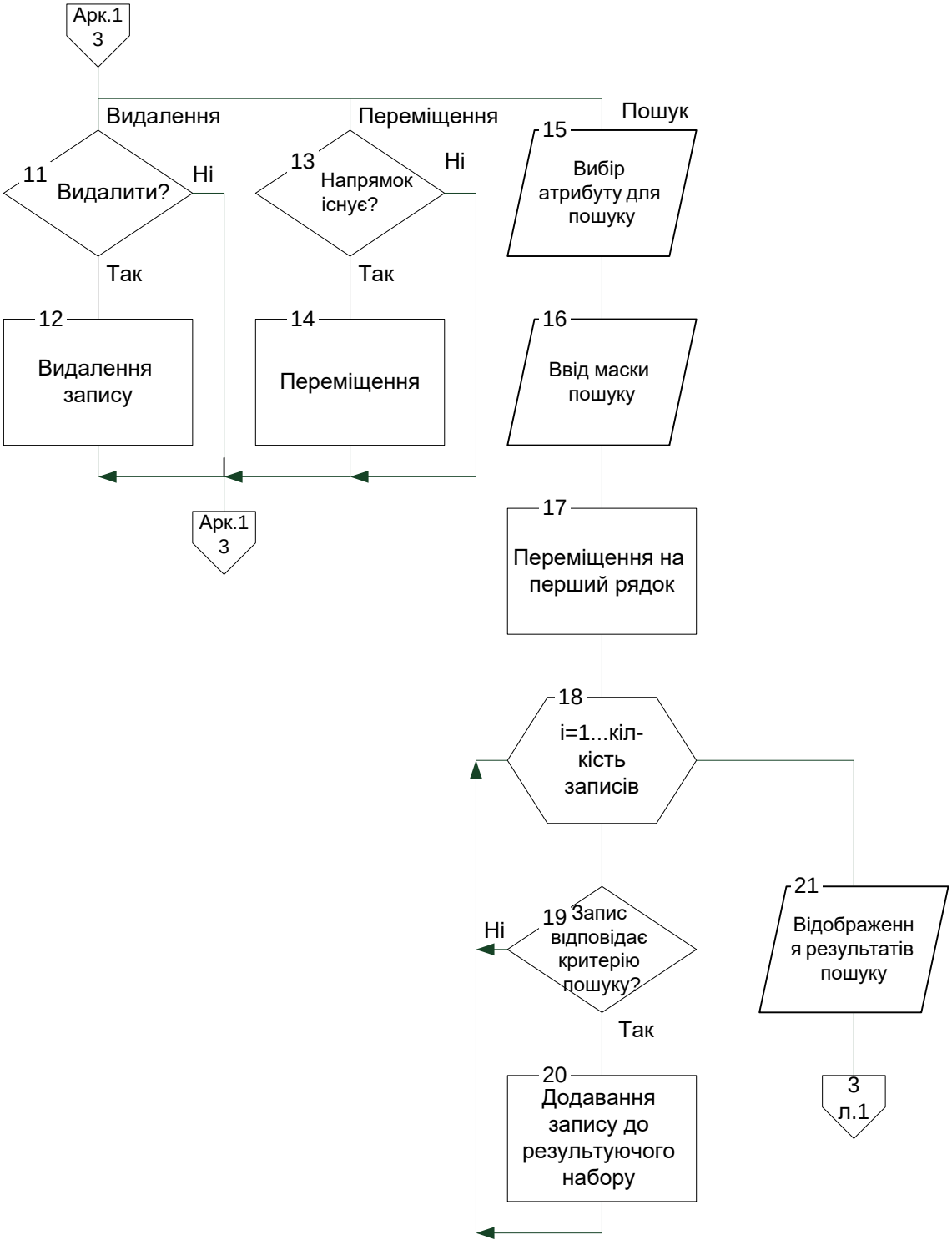

Рисунок 2.8 – Схема алгоритму підсистеми «Картотека». Аркуш 2

Режиму видалення поточного запису та переміщення по таблиці реалізовані в блоках 11-14 схеми алгоритму. Важливим режимом роботи підсистеми є пошук конкретного споживача теплової енергії за декількома признаками. Механізм пошуку реалізовано у блоках 15-21 схеми алгоритму. На першому кроці вибирається атрибут таблиці, по значенням котрого буде проводитися пошук. Далі вводиться маска пошуку. Потім при повному переборі записів таблиці «Картотека» виконується порівняння значення обраного атрибуту пошуку з маскою пошуку. При збігу цих значень, поточний рядок заноситься до результуючого набору даних. Після повного перебору таблиці «Картотека», отриманий набір даних виводиться оператору на екран.

Важливим етапом роботи з картотекою споживачів теплової енергії є формування звітів для аналізу стани розрахунків. Для цього до складу програмного забезпечення вводимо підсистему «Звіти». Схема алгоритму зазначеної підсистеми наведена на рис.2.9.

Робота підсистеми полягає в наступному. На першому етапі виконується вибір типу звіту (блок 2). Керівництвом РТКЕ поставлена задача формувати звіти по приналежності користувачів до певних мікрорайонів, підприємств, категорій платників. Перед формуванням звіту очищується таблиця звіту від попередніх даних (блок 3). Далі відбувається ввід умов формування звіту. До таких умов належить часовий інтервал, за який формується звіт, категорія споживачів, та таке інше в залежності від типу звіту. Далі у циклі до досягнення кінця таблиці «Картотека» перевіряються значення ключових атрибутів зі значенням умов формування звіту. Якщо атрибути задовольняють умовам пошуку, то поточний запис додається до таблиці «Звіти». Після проходження циклу відбувається відображення звіту або на принтер, або у вікно попереднього перегляду (блоки 11-14).

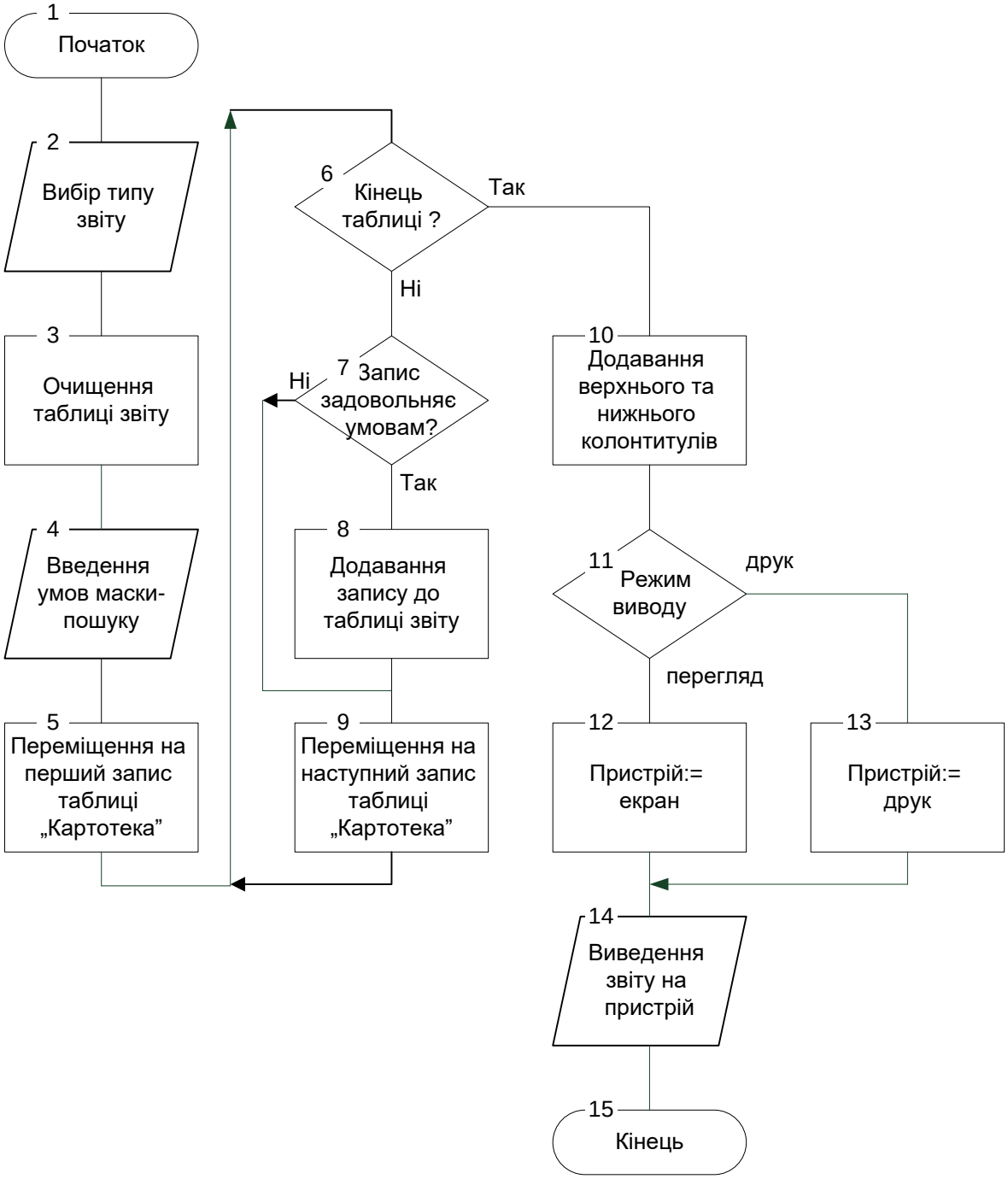

Рисунок 2.9 – Схема алгоритму підсистеми «Звіти»

2.3 Вибір і обґрунтування мови програмування

При виборі інструментальної мови для проектування програмного забезпечення "Підсистема розрахунку витрат теплоенергії споживачами РТКЕ" зробимо порівняльний аналіз сучасних СУБД.

2.3.1 Мова програмування ASSEMBLER

Це яскравий представник мов низького рівня, набір понять якого заснований на апаратній реалізації. Це засіб автоматизації для програмування безпосередньо в кодах процесора. Машинні команди описуються у вигляді мнемонічних операцій, що дозволяє домогтися досить високого рівня модіфікації коду. Оскільки набір команд на різних процесорах різний, то й про сумісність говорити не доводиться. Мова Assembler характеризується наданням повного контролю над ресурсами ПЕОМ. При максимальній швидкодії модуль, що виконується, має мінімальний розмір, однак обсяг вихідного коду перевищує обсяги вихідного коду мов високого рівня в кілька разів. Налагодження програм утруднене, а функції роботи з масивами пам'яті представлені в мінімальному обсязі.

Використання асемблера доцільно у випадках, коли необхідно прямо взаємодіяти з обладнанням, або одержати більшу ефективність для деякої частини програми за рахунок більше високого контролю над генерацією коду.

2.3.2 СУБД Microsoft Visual FoxPro

Microsoft Visual FoxPro – об’єктно-орієнтоване середовище для проектування баз даних і розробки додатків. Її засобами можна створювати об'єкти й класи, що володіють повними властивостями спадкування, інкапсуляції й інших відмінних рис об’єктно-орієнтоване проектування [5].

До складу Visual FoxPro входять всі необхідні інструменти для створення таблиць, виконання запитів, побудови інтегрованих систем управління реляційними базами даних і додатків для управління даними. Основа Visual FoxPro - повноцінне ядро сервера реляційної БД, оптимізоване для роботи з більшими наборами даних.

Система Microsoft Visual FoxPro на прикладі версії 6.0 містить всі необхідні засоби для створення й управління високопродуктивними 32-х розрядними додатками й компонентами баз даних. Надійні інструментальні засоби та об’єктно-орієнтована мова, спеціалізована для роботи з даними, ідеально підходять для створення сучасних багаторівневих додатків, інтегрувальних в архітектуру клієнт/сервер та інтернет. Microsoft Visual FoxPro володіє простим засобам створення компонентів для багаторазового використання в додатках. Це спрощує освоєння системи для розроблювачів. Є можливість створення за допомогою системи Visual FoxPro компонентів, інтегрувальних в архітектуру клієнт/сервер, а також у середовище мережі Інтернет. Удосконалення середовища розроблювача й набору інструментальних засобів забезпечують для розроблювача на Visual FoxPro гарну гнучкість настроювання й продуктивність. Підтримка технології Foundation Classes надає в розпорядження розроблювача готові бібліотеки класів для багаторазового використання, що дозволяє легко включати в додатки такі стандартні функції, як обробка даних, виявлення конфліктів відновлення, а також пошук і вибірку даних. Автоматизація створення додатків Майстер додатків і засіб створення додатків Application Builder надають просту у використанні об’єктно-орієнтовану структуру для створення додатків. Інструмент для налагодження Coverage Profiler - це вдосконалення процесу тестування й налагодження з використанням. Coverage Profiler перевіряє виконувані в сучасний момент рядки програмного коду й визначає час виконання кожного рядка. Застосування бібліотеки компонентів Component Gallery для створення й організації каталогів багаторазового

використовуваних об'єктів. Функціональні можливості додатків розширюються простим перетаскуванням об'єктів з бібліотеки Component Gallery у проект. Використання системи Microsoft Transaction Server для автоматичного керування, розміщення й масштабування компонентів COM системи Visual FoxPro [5].

Для розширення можливостей додатків Visual FoxPro є більше 6000 елементів управління ActiveX, розроблених незалежними компаніями. Для компонентів є вдосконалені бібліотеки типів, що поліпшує інтеграцію з іншими додатками й інструментальними засобами.

2.3.3 СУБД Microsoft Access

Програмний продукт Access фірми Microsoft - це, насамперед, набір інструментів кінцевого користувача для управління базами даних, що призначений для зберігання й одержання даних, подання їх у зручному вигляді, автоматизації часто виконуваних операцій і складань звітів [2]. Система Access не є “повноцінною” мовою програмування, за допомогою якого можна швидко й легко створити користувальницький інтерфейс, зручний і компактний додаток. Однак при необхідності можна писати різні макроси для автоматизації задач, тим самим знаходити нестандартні й громіздкі рішення стандартних ситуацій. Для згладжування цих недоліків і розширення можливості Access роботи з формами й звітами в систему уведена мова Visual Basic for Application (у ранніх версіях Access він називався Access Basic), що є псевдооб’єктно-орієнтованою. VBA використовує об'єкти й методи, але не підтримує основні концепції об’єктно-орієнтованого програмування, таких як інкапсуляція, спадкування й поліморфізм.

2.3.4 Огляд середовища розробки Delphi

Програмний продукт фірми Inprise Borland Delphi - це середовище розробки, що включає редактор, компілятор, налагодник, візуальні методи створення інтерфейсу програм і автоматичну генерацію відповідного програмного коду. Палітра компонентів містить всі необхідні елементи для візуального управління прикладною програмою [3]. Модель наборів даних зв'язує додаток-клієнт із базою даних через систему доступу до БД фірми Borland (Borland Database Engine, або BDE), що означає включення в додаток деякого коду, призначеного для взаємодії з повною системою BDE.

Система BDE підтримує, як власні, два типи локальних таблиць - Paradox і dBase. Кожний з них має свої переваги й недоліки.

Таблиці Paradox постійно розвиваються й надають багато можливостей. Таблиці Paradox підтримують більше 15 типів полів: від числових до текстових з фіксованою й змінною довжиною й полів, що дозволяють зберігати безпосередньо двійкові дані. Такий великий набір типів дозволяє гнучко вибирати параметри проекту бази даних точно представляти зберігається інформацію. Концепція цілісності даних забезпечують правильність посилань між таблицями. Для таблиць Paradox система BDE автоматично здійснює механізм каскадного видалення, що виключає можливість посилання на неіснуюче значення поля [3].

Більшість значень полів укладено в певному діапазоні, що дозволяє виключити ситуацію уведення помилкового значення. Для кожного поля можна визначити мінімальне й максимальне значення, а також значення за замовчуванням. Мінімальне й максимальне значення містять уміст поля в певний діапазон. Якщо ввести значення, що виходить за межі діапазону, Delphi згенерує виняткову ситуацію й відкине неправильне значення [4].

Значення за замовчуванням автоматично привласнюються полям при додаванні нових записів.

Формат таблиць dBase - один з перших форматів PC-таблиць, і тому він підтримується практично всіма додатками, які пов'язані з даними, що мають формат таблиць. Використання формату файлів dBase надає кілька додаткових переваг. Таблиці dBASE не підтримують цілісності посилань, вірогідності даних і захисту даних. Імена полів можуть складатися не більш ніж з восьми символів і не допускають використання пробілів. У таблицях dBASE підтримується лише вісім типів полів - символьні, числа із плаваючої коми, числові, дати, логічні, Мемо, OLE- і двійкові поля. Цей обмежений набір означає, що не можна використовувати автоматичне форматування значень полів при відновленні даних з таблиці. Як приклад можна привести відсутність грошового типу, що змушує в додатку спеціально форматувати числове поле dBASE, у той час як в Paradox для грошового типу це форматування виконується автоматично.

Перевірка цілісності посилань, вірогідності даних, завдання значень за замовчуванням, парольний захист - все це підтримується в Paradox і в SQL-Базах даних. Такі загальні атрибути спрощують переміщення з локальної бази даних Paradox на SQL-Сервер. Бази даних Paradox трохи перевершують бази даних dBASE по продуктивності. Вставка й зміна записів, як і пошук по індексованих і неіндексованих полях, у базах даних Paradox виконується швидше.

Висновок: Зараз поширюється безкоштовний аналог Borland Delphi під назвою Lazarus. Зовнішнє ця середа програмування дуже схожа на Borland Delphi, але у своєму складі має меншу кількість компонентів, та не має власного ядра СУБД. Це приводить до того, що при написанні програми обробки БД у середовищі Lazarus потрібно шукати і застосовувати СУБД інших розроблювачів (більшість з котрих платна). Тому Lazarus не підлягає розгляду у якості середовища програмування даної задачи.

З вище описаних порівняльних характеристик робимо вивід про доцільність використовувати як мову програмування мову Object Pascal середовища розробки Delphi 7 і формату баз даних Paradox. Крім того, вибір середовища розробки Delphi ще обумовлений прийнятністю з раніше розробленими підсистемами. ПЗ «Бухгалтерія. Заробітна плата», «Бухгалтерія. Облік матеріальних цінностей», яке зараз використовується в РТКЕ, розроблено кільками роками раніше за допомогою Borland Delphi 7. Ліцензійна копія Borland Delphi 7 є у наявності в технічному підрозділі РТКЕ.

3 РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ «ПІДСИСТЕМА РОЗРАХУНКУ ВИТРАТ ТЕПЛОЕНЕРГІЇ СПОЖИВАЧАМ»

Початкове завантаження програми супроводжується появою на екрані інформаційного вікна, наведеного на рисунку 3.1.

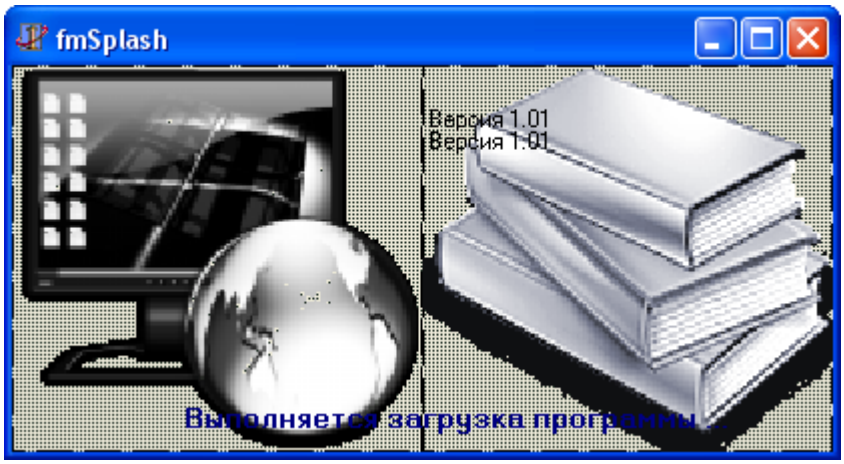

Рисунок 3.1 – Інформаційне форма-заставка

Під час відображення вікна-заставки відбувається перевірка наявності робочих таблиць БД. При виявленні відсутності робочого файлу БД відбувається автоматичне створення порожнього «еталонного» файлу БД заданої структури (рядки 24-141 лістингу програми, наведеного в додатку А) та із заданими властивостями. На наступному етапі відбувається підключення до робочих файлів БД (рядки 143-237 лістингу програми).

Хід етапів процесу ініціалізації відбивається в реальному режимі часу на елементі відображення строкової інформації, розташованому в нижній частині вікна. Крім цього, вікно-заставка містить інформацію про версію програми й даті внесення останньої зміни в неї. Ця інформація розташована у верхній частині вікна.

У випадку успішного виконання описаних етапів перевірки виконується відновлення особистих настроювань користувача, які

зберігаються в однойменному INI-Файлі, розташованому в робочому каталозі операційної системи WINDOWS.

Головна форма підсистеми містить закладки, на яких розташовані візуальні елементи та елементи управління (рисунок 3.2).

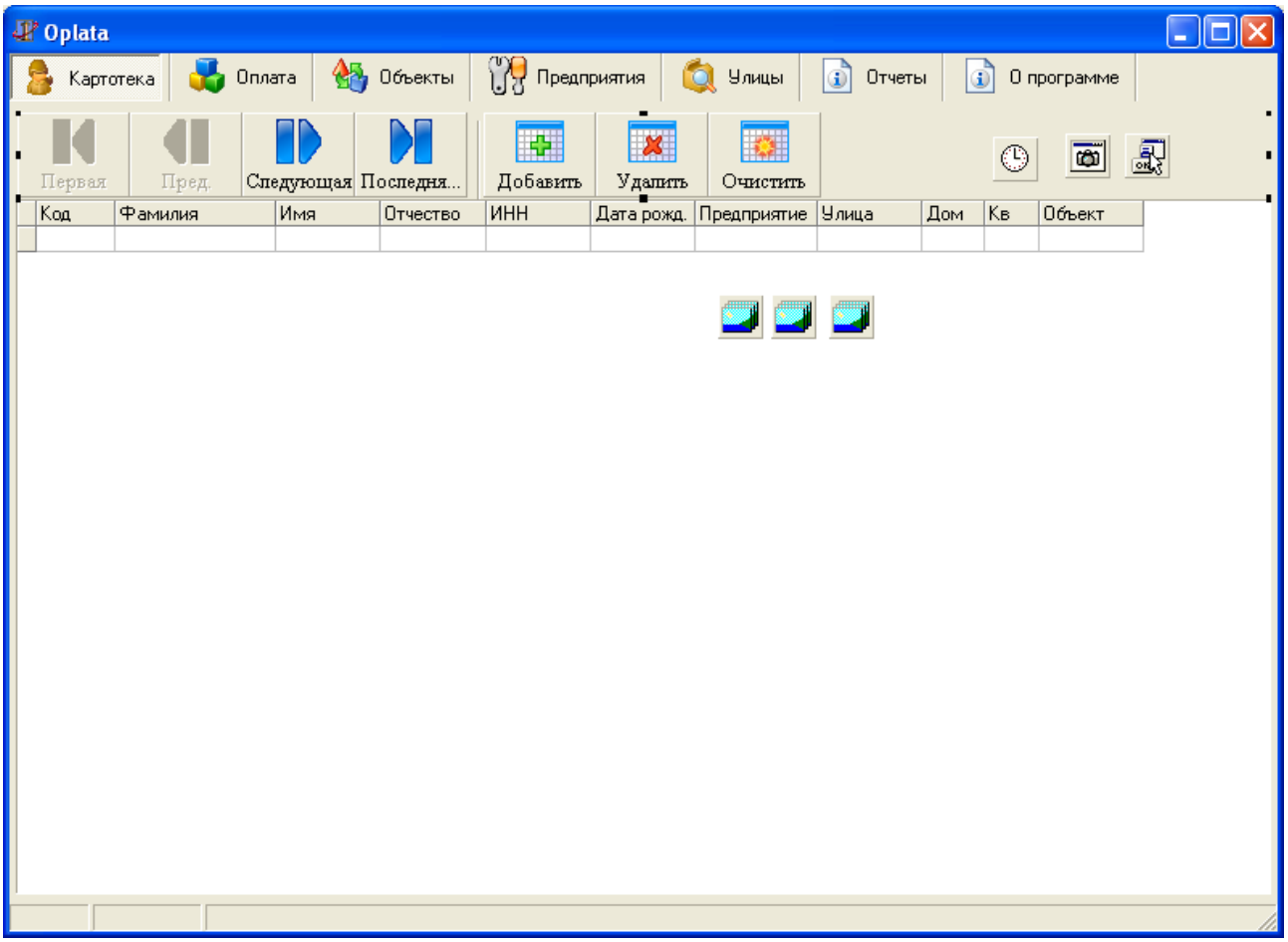

Рисунок 3.2 – Вікно додатка

Розбивка програмного пакета «Підсистема розрахунку витрат теплоенергії споживачами РТКЕ» на окремі зв'язні частини «Довідники», «Картотека», «Звіти» надає гнучкість у використанні, надійність у зберіганні та безпеці даних. Застосувавши один з існуючих криптографічних методів або методів операційної системи платформи Windows NT підвищить надійність, вірогідність і безпеку зберігання всієї інформації.

3.1 Підсистема «Довідники»

Для поліпшення роботи оператора з розроблювальним програмним забезпеченням та зменшення ймовірності помилок при наборі даних, до складу програмного забезпечення «Підсистема розрахунку витрат теплоенергії споживачами РТКЕ» вводиться підсистема «Довідники». На початковому етапі розробки та впровадження нового програмного забезпечення виділимо довідники «Вулиці», «Підприємства» та «Об'єкти». Довідник «Вулиці», розробка форми якого наведена на рисунку 3.3, будить містити перелік вулиці міста. Цей перелік призначено для завдання адреси споживачів теплової енергії. Тому операторові не треба писати літерами повністю назву той чи іншої вулиці, а треба буде лише вибрати певну вулиці із переліку. Це зменшить ймовірність неправильного задавання адреси із-за орфографічної помилки.

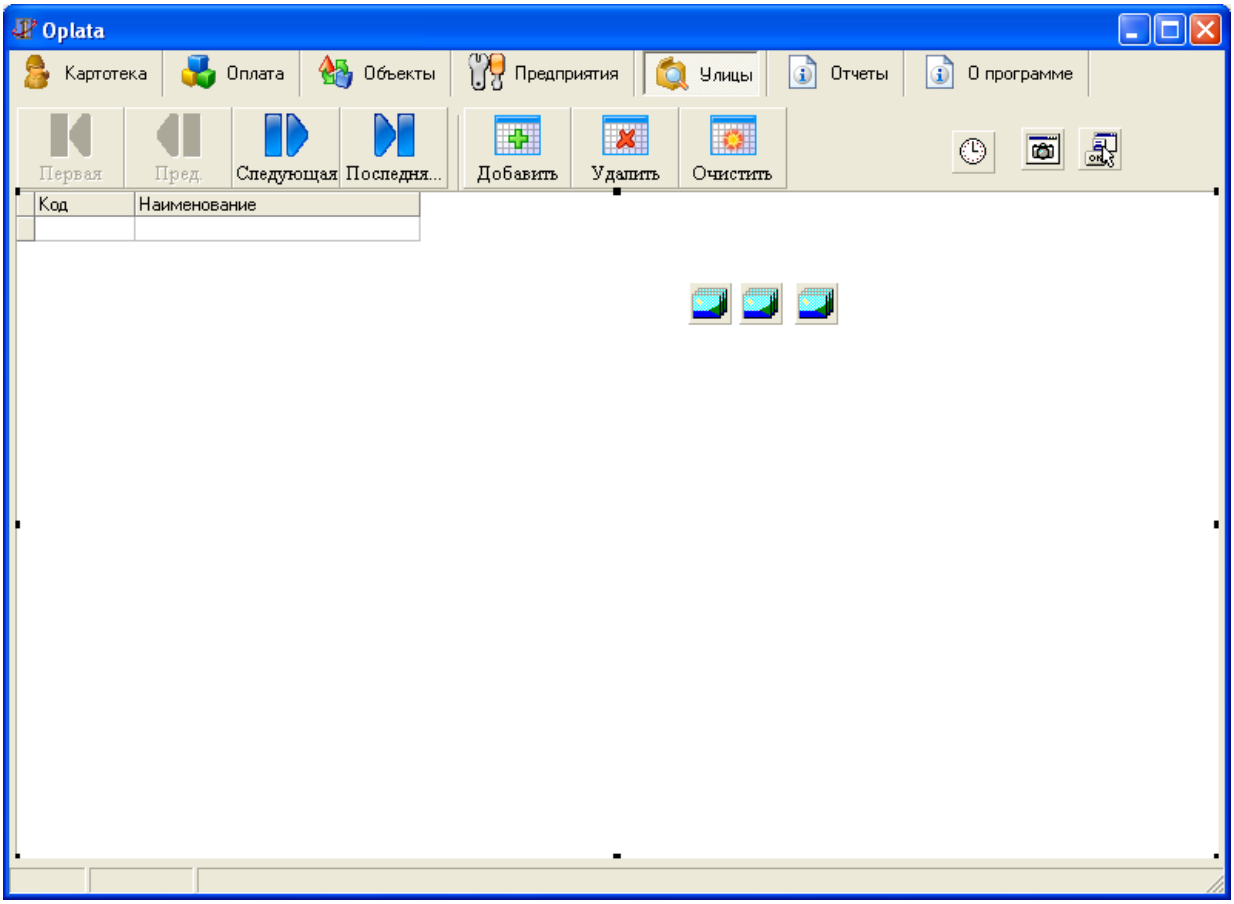

Рисунок 3.3 – Закладка «Улицы»

Закладка містить панель кнопок класу TSpeedBar та таблицю для відображення даних TDBView. Панель кнопок містить сім кнопок класу TSpeedItem: siFirstUl, siPrevUl, siNextUl, siLastUl, siInsertUl, siDeleteUl, siClearUl (рядки 313-319 лістингу програми). Обробники натискання на перші чотири кнопки містять програмні засоби переміщення по строкам таблиці. Наступні дві кнопки запрограмовані на додавання та видалення рядків таблиці відповідно. Остання кнопка виконую повне видалення всіх рядків таблиці. На кожній кнопці зображена графічна піктограма на текстовий напис, які асоціюються з призначенням кнопки. Поєднання графічних та текстових пояснень сприятиме поліпшенню звикання оператора до програми.

За таким принципом побудовані інші довідники «Объекты» (рядки 265-271, 442-494 лістингу програми) та «Предприятия» (рядки 291-298, 580-682 лістингу програми). Довідник «Объекты» введений для реалізації автоматичного перерахунку тарифів за використану теплоенергію. Об’єктами виступають житлові будинки, або під’їзди з завданням тарифу на опалення одного квадратного метру житлової площини. Цей механізм використовується тоді, коли треба перерахувати суму сплати наданих послуг для усіх користувачів. Про цей режим роботи програми буде сказано при описанні підсистеми «Картотека». Довідник «Предприятия» містить перелік підприємств з зазначенням назви, адреси, ПІБ керівника. Ці дані будуть використовуватися в роботі підсистем «Картотека» та «Звіти».

3.2 Підсистема «Картотека»

Підсистема «Картотека» містить основні дані програми «Підсистема розрахунку витрат теплоенергії споживачами РТКЕ». Складається з двох таблиць «Картотека» та «Оплата». Перша таблиця містить паспортні дані квартиро найманців та технічні дані їх квартир. Друга таблиця – перелік платежів. На рис. 3.4 зображена закладка «Картотека».

Таблиця «Картотека» містить одне ключове поле «Код», яке є унікальним і не може повторюватися для будь яких записів. Це зроблено при описі полів таблиці за допомогою признаку первинного ключа [ixPrimary, ixUnique]. На практиці це означає що не може бути двоє споживачів з однаковим унікальним номером. В якості такого номера виступає номер особистого рахунку.

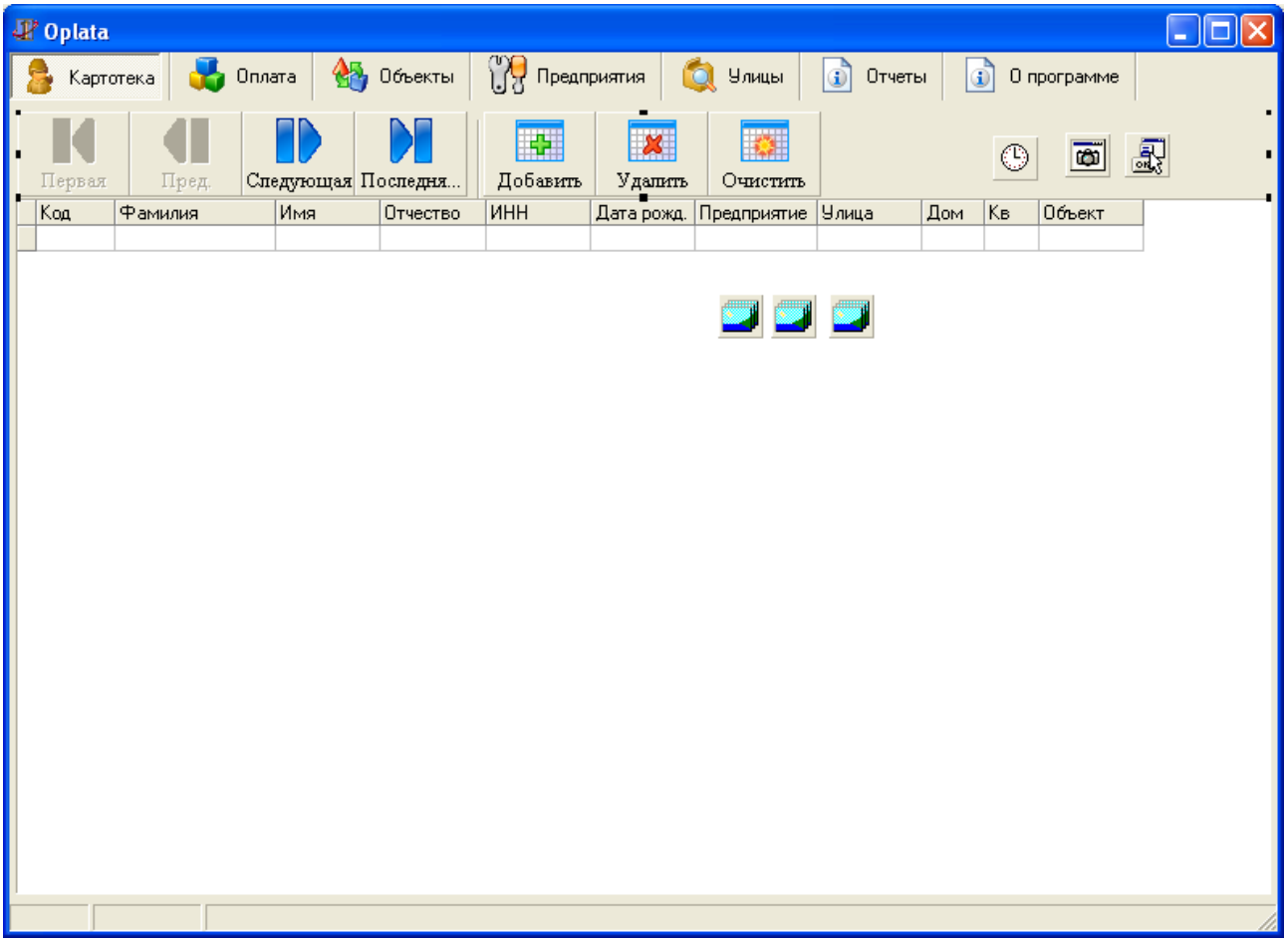

Рисунок 3.4 – Закладка «Картотека»

Поля «Улица», «Предприятие» та «Объекты» є довідниковими, тобто для завдання значення треба використовувати перелік допустимих значень, які зберігаються у відповідному довіднику (рядки 510-579- лістингу програми).

Для звертання до відповідного довідника розроблюється спеціальна форма, яка наведена на рис.3.5. Форма містить табличне уявлення складу довідника та засоби пошуку інформації за назвою (рядки 654-752 лістингу програми).

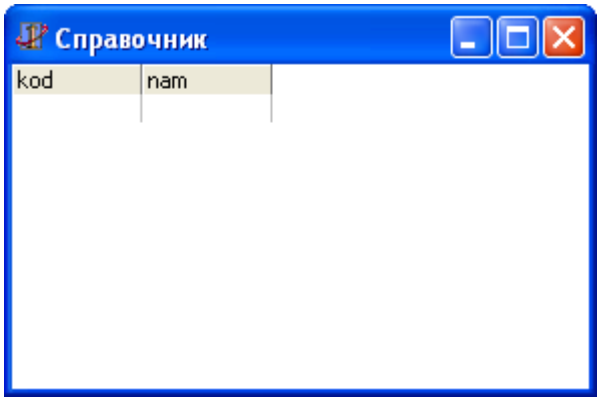

Рисунок 3.5 – Форма відображення довідника

Різниця цієї форми від відповідної закладки полягає у тому, що дані відображуються у режимі «ReadOnly» і форма буде виникати у певному місті вікна не закриваючи собою таблицю «Картотека». У той час, як відповідна закладка довідникової підсистеми розташовується на весь екран і користувачеві не має можливості бачити редагування таблиці «Картотека».

Згідно поставленої задачі та схеми даних (рис.2.3) для кожного споживача теплоенергії потрібно заносити та зберігати дані про сплату рахунків. Для цього і введена закладка «Оплата», яка пов’язана з відповідною таблицею даних (рис.3.7). Таблиця містить два ключових поля: «Код» та «Период платежа». Поле «Код» не відображується при роботі з даними, бо завдяки зв’язку «один до багатьох», таблиця «Оплата» буде містити при роботі лише записи, які відносяться до поточного запису

таблиці «Картотека». Друге ключове поле потрібно для упорядкування даних в таблиці «Оплата».

Заповнення таблиці «Оплата» відбуваються в автоматизованому режимі. Якщо в таблиці «Картотека» для поточного запису заповнено поле «Объект» і у відповідному довіднику зазначено тариф сплати опалювання одного квадратного метру площини, то програма автоматично проведе розрахунок поля «Сумма к оплате» і занесе отримане значення до таблиці.

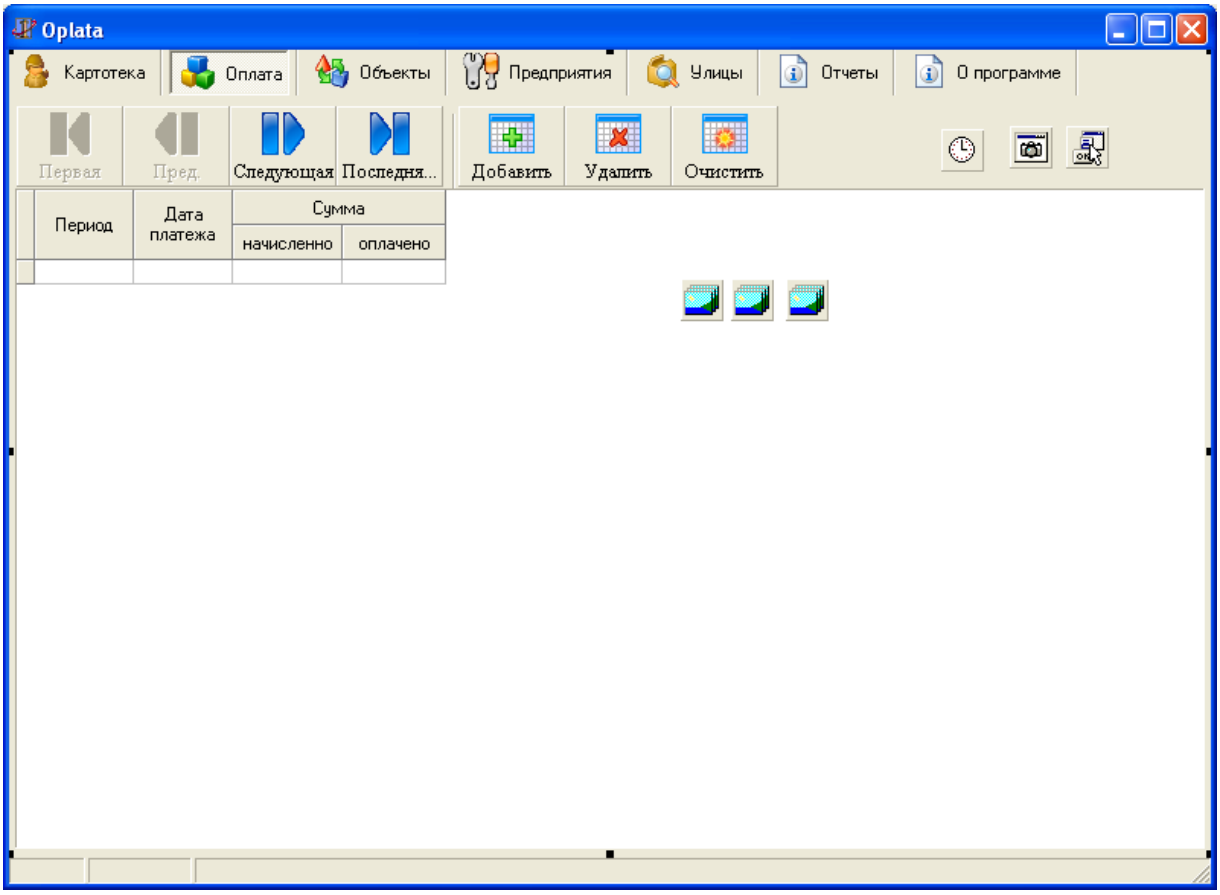

Рисунок 3.6 – Закладка «Оплата»

Дані таблиці «Оплата» безпосередньо використовуються в підсистемі «Звіти» при формуванні звітної документації.

4 ОХОРОНА ПРАЦІ

4.1 Аналіз потенційних небезпечних і шкідливих виробничих чинників проектованого об’єкту, що мають вплив на персонал

У даному дипломному проекті розробляється програмне забезпечення.

Розроблене програмне забезпечення орієнтоване на роботу з персональним комп'ютером. Експлуатовані для вирішення внутрішньовиробничих завдань ПЕОМ типу IBM PC мають наступні характеристики:

споживана потужність	220 Вт;
робоча напруга	220 В;
напруга джерел живлення	+12 В; - 12 В; +5 В;
робоча частота	50 Гц.

Виходячи з приведених характеристик, вочевидь, що для людини існує небезпека поразки електричним струмом, унаслідок недбалого поводження з комп'ютером і порушення правил експлуатації, залишення частин ПЕОМ, що знаходяться під напругою, відкритими або знятих для ремонту вузлів.

Відповідно до ГОСТ 12.1.005-88 до легкої фізичної роботи відносяться всі види діяльності, виконувані сидячи і ті, що не потребують фізичної напруги. Робота користувача ПК відноситься до категорії 1а.

При роботі на ПЕОМ користувач піддається ряду потенційних небезпек. Унаслідок недотримання правив техніки безпеки при роботі з машиною(невиконання огляду відкритих частин ПЕОМ, що знаходяться під напругою або знятих для ремонту вузлів) для користувача існує небезпека поразки електричним струмом.

Джерелами підвищеної небезпеки можуть служити наступні елементи:

– розподільний щит;

– джерела живлення;

– блоки ПЕОМ і друку, що знаходяться в ремонті.

Ще одна проблема полягає у тому, що спектр випромінювання комп'ютерного монітора включає рентгенівську, ультрафіолетову і інфрачервону області, а також широкий діапазон хвиль інших частот. Небезпека рентгенівського проміння мала, оскільки цей вид випромінювання поглинається речовиною екрану. Проте велику увагу слід приділяти біологічним ефектам низькочастотних електромагнітних полів(аж до порушення ДНК).

Відповідно до ГОСТ 12.1.003-74, при обслуговуванні ПЕОМ мають місце фізичні і психофізичні небезпечні, а також шкідливі виробничі чинники:

– підвищене значення напруги в електричному ланцюзі, замикання якої може відбутися через тіло людини;

– підвищений рівень статичної електрики;

– підвищений рівень електромагнітних випромінювань;

– підвищена або знижена температура повітря робочої зони;

– підвищений або знижений рух повітря;

– підвищена або знижена вологість повітря;

– відсутність або недостатність природного світла;

– підвищена пульсація світлового потоку;

– недостатня освітленість робочого місця;

– підвищений рівень шуму на робочому місці;

– розумове перенапруження;

– емоційні навантаження;

– монотонність праці.

4.2 Заходи щодо техніки безпеки

Основним небезпечним чинником при роботі з ЕОМ є небезпека поразки людини електричним струмом, яка посилюється тим, що органи чуття людини не можуть на відстані знайти наявності електричної напруги на устаткуванні.

Проходячи через тіло людини, електричний струм чинить на нього складну дію, що є сукупністю термічної(нагрів тканин і біологічних середовищ), електролітичної(розкладання крові і плазми) і біологічної(роздратування і збудження нервових волокон і інших органів тканин організму) дій.

Тяжкість поразки людини електричним струмом залежить від цілого ряду чинників:

– значення сили струму;

– електричного опору тіла людини і тривалості протікання через нього струму;

– роду і частоти струму;

– індивідуальних властивостей людини і навколишнього середовища.

Розроблений дипломний проект передбачає наступні технічні способи і засоби, що застерігають людину від ураження електричним струмом:

– заземлення електроустановок;

– занулення;

– захисне відключення;

– електричне розділення ятерів;

– використовування малої напруги;

– ізоляція частин, що проводять струм;

– огорожа електроустановок.

Занулення зменшує напругу дотику і обмежує година, протягом якого людина, ткнувшись до корпусу, може потрапити під дію напруги.

Струм однофазного короткого замикання визначається по наближеній формулі:

$$I_K = \frac{\text{Uф}}{\text{Zп} + \frac{\text{Zт}}{3}}, \qquad (4.1)$$

де Uф - номінальна фазна напруга мережі, В;

Zп - повний опір петлі, створене фазними і нульовими дротами, Ом;

Z_T - повний опір струму короткого замикання на корпус, Ом.

Згідно таблиці 4 ГОСТ 12.1.009-76: $Z_T/3 = 0{,}1$ Ом.

Для провідників і жил кабелю для розрахунку повного опору петлі використовуємо формулу(4.2.) :

$$Z\text{п} = \sqrt{\text{Rп}^2 + \text{Xп}^2}, \qquad (5.2)$$

де Rп = Rф + Ro - сумарний активний опір фазного Rф і нульового Ro дротів, Ом;

Xп - індуктивний опір паяння дротів, Ом.

Перетин 1 км мідного дроту S = 2.5 мм, тоді згідно таблицям 5 і 6 ГОСТ 12.1.009-76 [13], має такий опір:

Xп = 0,11 Ом;

Rф = 7,55 Ом;

Ro = 7,55 Ом.

Отже, Rп = 7,55 + 7,55 = 15,1 Ом.

Тоді по формулі (4.2) знаходимо повний опір петлі :

$$Zп = \sqrt{15,1^2 + 0,11^2} \approx 15,1 \text{ (Ом).}$$

Струм однофазного короткого замикання рівний:

$$Iк = \frac{220}{15,1 + 0,1} = 14,47 \text{ (А).}$$

Дія плавкої вставки на ПЕОМ забезпечується, якщо виконується співвідношення:

$$Iк \geq k * Iн, \quad (4.3)$$

де Ін - номінальний струм спрацьовування плавкої вставки, А;

k - коефіцієнт кратності нелінійного струму Ін, А.

Коефіцієнт кратності нелінійного струму Ін розраховується по формулі (4.4.) :

$$Iн = P / U, \quad (4.4)$$

де P = 220 Вт - споживана потужність;

U = 220 В - робоча напруга;

k = 3 А - для плавких вставок.

Отже, Ін = 220 / 220 = 1 А.

Підставивши значення у вираз (4.3), одержимо:

$$14,47 > 3*1.$$

Таким чином, доведено, що апарат забезпечить спрацьовування(і захист) при підвищенні номінального струму.

4.3 Заходи, що забезпечують виробничу санітарію і гігієну праці

Вимоги до виробничих приміщень встановлюються СН 245-71, СНіП, відповідними ГОСТами і ОСТамі з урахуванням небезпечних і шкідливих чинників, що утворюються в процесі експлуатації електроустаткування.

Підвищення працездатності людини і збереження її здоров'я забезпечується стабільними метеорологічними умовами.

Мікроклімат виробничих приміщень визначається діючими на організм людини поєднаннями температури, вологості і швидкості руху повітря, а також температури навколишніх поверхонь. Значне коливання параметрів мікроклімату приводити до порушення систем кровообігу, нервової і потовидільної, що може викликати підвищення або пониження температури тіла, слабкість, запаморочення і навіть непритомність.

Відповідно до ГОСТ 12.1.005-88 встановлюють оптимальну і допустиму температуру, відносну вологість і швидкість руху повітря в робочій зоні . За відсутності надмірного тепла, вологи, шкідливих речовин в приміщенні досить природної вентиляції.

У приміщенні для виконання робіт операторського типу(категорія 1а), пов'язаних з нервово-емоційною напругою, проектом передбачається дотримання наступних нормованих величин параметрів мікроклімату (табл. 4.1).

Таблиця 4.1 - Санітарні норми мікроклімату робочої зони приміщень для робіт категорії 1а.

Пора року	Температура, С	Відносна вологість, %	Швидкість руху повітря, м/с
Холодна	22…24	40…60	0,1
Тепло	23…25	40…60	0,1

У приміщенні, де знаходиться ПЕОМ, повітрообмін реалізується за допомогою природної організованої вентиляції(з пристроєм вентиляційних каналів в перекриттях будівлі і вертикальних шахт) й установленого промислового кондиціонера фірми Mitsubishi, який дозволяє вирішити переважну більшість завдань по створінню та підтримці необхідних параметрів повітряного середовища. Цей метод забезпечує приток потрібної кількості свіжого повітря, визначеного в СНіП (30 $м^3$ в годину на одного працівника).

Шум на виробництві має шкідливу дію на організм людини. Стомлення операторів через шум збільшує число помилок при роботі, призводить до виникнення травм. Для оператора ПЕОМ джерелом шуму є робота принтера. Щоб усунути це джерело шуму, використовують наступні методи. При покупці принтера слід вибирати найбільш шумозахисні матричні принтери або з великою швидкістю роботи(струменеві, лазерні). Рекомендується принтер поміщати в найбільш віддалене місце від персоналу, або застосувати звукоізоляцію та звукопоглинання(під принтер підкладають демпфуючі підкладки з пористих звукопоглинальних матеріалів з листів тонкої повсті, поролону, пеноплену).

При роботі на ПЕОМ, проектом передбачені наступні методи захисту від електромагнітного випромінювання : обмеження часом, відстанню, властивостями екрану.

Обмеження годині роботи на ПЕОМ складає 3,5-4,5 години. Захист відстанню передбачає розміщення монітора на відстані 0,4-0,5 м від оператора. Передбачений монітор 20" TFT, Samsung 2043BW відповідає вимогам стандарту TCO'03.

TCO'03 пред’являє жорсткі вимоги в таких областях: ергономіка(фізична, візуальна і зручність користування), енергія, випромінювання(електричних і магнітних полів), навколишнє середовище

і екологія, а також пожежна та електрична безпека, які відповідають всім вимогам ДСанПіН 3.3.2.007-98.

Для зниження стомлюваності та підвищення продуктивності праці обслуговуючого персоналу в колірній композиції інтер'єру приміщень для ПЕОМ дипломним проектом пропонується використовувати спокійні колірні поєднання і покриття, що не дають відблисків.

У проекті передбачається використання сумісного освітлення. У світлий час доби приміщення освітлюватиметься через віконні отвори, в решту часу використовуватиметься штучне освітлення.

Як штучне освітлення необхідно використовувати штучне робоче загальне освітлення. Для загального освітлення необхідно використовувати люмінесцентні лампи. Вони володіють наступними перевагами: високою світловою віддачею, тривалим терміном служби, хоча мають і недоліки: високу пульсацію світлового потоку.

При експлуатації ПЕОМ виробляється зорова робота. Відповідно до СніП II - 4-79 ця робота відноситься до розряду 5а. При цьому нормоване освітлення на робочому місці(Ен) при загальному освітленні рівна 200 лк.

Приміщення завдовжки 12 м, шириною 10 м, заввишки 4 м обладнується світильниками типу ЛП02П, оснащеними лампами типу ЛБ зі світловим потоком 3120 лм кожна.

Виконаємо розрахунок кількості світильників в робочому приміщенні завдовжки a=12 м, шириною b=10 м, заввишки з=4 м, використовуючи формулу (4.5) розрахунку штучного освітлення при горизонтальній робочій поверхні методом світлового потоку:

$$n = (E \cdot S \cdot Z \cdot k)/(F \cdot U \cdot M), \qquad (4.5)$$

де F - світловий потік = 3120 лм;

E - максимально допустима освітленість робочих поверхонь = 200 лк;

S - площа підлоги = 120 м2;

Z - поправочний коефіцієнт світильника = 1,2;

k - коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації світильників = 1,5;

n - кількість світильників;

U - коефіцієнт використовування освітлювальної установки = 0,6;

M - кількість ламп у світильнику =2.

З формули (4.5) виразимо n (4.6) і визначимо кількість світильників для даного приміщення:

$$n = (E \cdot S \cdot Z \cdot k)/(F \cdot U \cdot M), \quad (4.6)$$

Отже, n = (200·120·1,2·1,5)/(3120·0,6·2) = 12 .

Виходячи з цього, рекомендується використовувати 12 світильників. Світильники слід розміщувати рядами, бажано паралельно стіні з вікнами. Схема розташування світильників зображена на рис. 5.1.

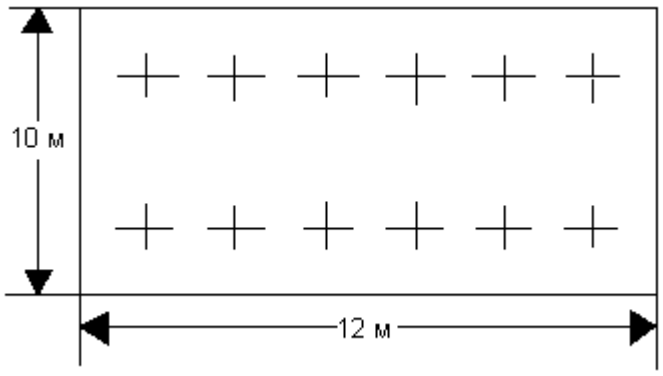

Рисунок 4.1 - Схема розташування світильників

4.4 Рекомендації по пожежній безпеці

Пожежі в приміщеннях, де встановлена обчислювальна техніка, представляють небезпеку для життя людини. Пожежі також пов’язані як з матеріальними втратами, так і з відмовою засобів обчислювальної техніки, що у свою чергу спричиняє за собою порушення ходу технологічного процесу.

Пожежа може виникнути при наявності горючої речовини та внесення джерела запалювання в горюче середовище. Пальними матеріалами в приміщеннях, де розташовані ПЕОМ, є:

– поліамід - матеріал корпусу мікросхеми, горюча речовина, температура самозаймання аерогелю 420 З ;

– полівінілхлорид - ізоляційний матеріал, горюча речовина, температура запалювання 335 З, температура самозаймання 530 З, кількість енергії, що виділяється при згоранні - 18000 - 20700 кДж/кг;

– стеклотекстоліт ДЦ - матеріал друкарських плат, важкозаймистий матеріал, показник горючості 1.74, не схильний до температурного самозаймання;

– пластика кабельний №489 - матеріал ізоляції кабелю, горючий матеріал, показник горючості більш 2.1;

– деревина - будівельний і обробний матеріал, матеріал з якого виготовлені меблі, горючий матеріал, показник горючості більше 2.1, теплота згорання 18731 - 20853 кДж/кг, температура запалювання 399 З, схильна до самозаймання [17].

Згідно НАПБ.Б. 03.002-2012 приміщення відносяться до категорії В(пожежовибухонебезпечним) і згідно правилам побудови електроустановок простір усередині приміщення відноситься до вогненебезпечної зони класу П - IIа (зони, розташовані в приміщеннях, в яких зберігаються тверді горючі речовини).

Потенційними джерелами запалення при роботі ПЕОМ є:

– іскри при замиканні і розмиканні ланцюгів;

– іскри і дуги коротких замикань;

– перегріви від тривалого перевантаження і наявності перехідного опору.

Продуктами згорання, що виділяються при пожежі, є : оксид вуглецю, сірчистий газ, оксид азоту, синильна кислота, акропеін, фосген, хлор та ін. При горінні пластмас, окрім звичайних продуктів згорання, виділяються різні продукти термічного розкладання: хлорангідрідні кислоти, формальдегіди, хлористий водень, фосген, синильна кислота, аміак, фенол, ацетон, стирол та ін., що шкідливо впливають на організм людини.

Для захисту персоналу від дії небезпечних і шкідливих чинників пожежі проектом передбачається застосування промислового протигаза з коробкою марки В(жовта).

Пожежна безпека об’єктів народного господарства регламентується ГОСТ 12.1.004-91 і забезпечується системами запобігання пожежам і протипожежному захисту. Для успішного гасіння пожеж вирішальне значення має швидке виявлення пожежі і своєчасний виклик пожежних підрозділів до місця пожежі.

Зменшити горюче навантаження не представляється можливим, тому проектом передбачається застосувати наступні способи і їх комбінації для запобігання утворенню(внесення) джерел запалення :

– застосування устаткування, що задовольняє вимогам електростатичної безпеки;

– застосування в конструкції швидкодіючих засобів захисного відключення можливих джерел запалення;

– виключення можливості появи іскрового заряду статичної електрики в горючому середовищі з енергією, рівної і вище мінімальної енергії запалення;

– заміна небезпечних технологічних операцій більш безпечними;

– ізольоване розташування небезпечних технологічних установок і устаткування;

– зменшення кількості пальних і вибухонебезпечних речовин, що знаходяться у виробничих приміщеннях;

– запобігання можливості утворення пальних сумішей на лінії, вентиляційних системах і ін.;

– механізація, автоматизація та справність(потокова) виробництва;

– суворе дотримання стандартів і точне виконання встановленого технологічного режиму;

– запобігання можливості появи в небезпечних місцях джерел запалення;

– запобігання розповсюдженню пожеж і вибухів;

– використовування устаткування і пристроїв, при роботі яких не виникає джерел запалення;

– виконання вимог сумісного зберігання речовин і матеріалів;

– наявність громовідводу;

– організація автоматичного контролю параметрів, що визначають джерела запалення;

– ліквідація можливості самозаймання речовин і матеріалів .

Для запобігання пожежі в обчислювальних центрах проектом пропонується виконання наступних вимог :

– електроживлення ЕОМ повинно мати автоматичне блокування відключення електроенергії на випадок зупинки системи охолоджування і кондиціонування;

– система вентиляції обчислювальних центрів повинна бути обладнана блокуючими пристроями, що забезпечують її відключення на випадок пожежі;

– робочі місця повинні бути оснащені пожежними щитами, сигналізацією, засобами для сповіщення про пожежну небезпеку

(телефонами), медичними аптечками для надання першої медичної допомоги, розробленим планом евакуації.

Для зниження пожежної небезпеки в приміщеннях використовуються первинні засоби гасіння пожеж, а також система автоматичної пожежної сигналізації, яка дозволяє знайти початкову стадію загоряння, швидко і точно оповістити службу пожежної охорони про час і місце виникнення пожежі.

Відповідно до правил пожежної безпеки для промислових підприємств приміщення категорії В підлягають устаткуванню системами автоматичної пожежної сигналізації. Проектом передбачається застосування датчика типу ІДФ - 1(димовий фотоелектричний датчик), оскільки специфікою пожеж обчислювальної техніки і радіоапаратури є, в першу чергу, виділення диму, а потім - підвищення температури.

При виникненні пожежі в робочому приміщенні обслуговуючий персонал зобов'язаний негайно вжити заходи по ліквідації пожежі. Для ліквідації пожежі використовують вогнегасники (хімічно-пінні, пінні для повітря ОП-5, ОП-6, ОП-9, вуглекислотні ОУ-5), пісок, пожежний інвентар(сокири, ломи, багри, шерстяну або азбестову ковдри). Як засіб індивідуального захисту проектом передбачається використовування промислового протигаза з маскою, фільтруючої коробки В.

В якості організаційно-технічних заходів рекомендується проводити навчання робочого персоналу правилам пожежної безпеки.

У розділі «Охорона праці» виконано аналіз потенційних небезпек при роботі із засобами обчислювальної техніки і механізмами, розроблені заходи щодо техніки безпеки, заходи, які забезпечують виробничу санітарію і гігієну праці, розраховане штучне освітлення, виконані рекомендації по пожежній безпеці.

ВИСНОВКИ

У дипломному проекті був проведений аналіз задачі проектування інформаційної системи муніципального підприємства. Інформаційна система призначається для автоматизації роботи по обліку платежів, контролю за обсягом оплати, автоматичному перерахуванню при змін тарифів, заповненню звітів.

Розроблено алгоритми вводу/виводу даних підсистем, алгоритм обробки даних, механізм попереднього перегляду та друку звітних документів. Програмне забезпечення реалізоване мовою Object Pascal середовища програмування Delphi 7 з використанням наявних у нього можливостей.

ПЗ “ Підсистема розрахунку витрат теплоенергії споживачами РТКЕ ” призначена для роботи на персональному комп'ютері тину IBM PC під керуванням операційної системи сімейства Windows.

У розділі «Охорона праці" виконаний аналіз небезпечних і шкідливих виробничих факторів, причин пожеж. На основі аналізу розроблені заходи щодо техніки безпеки і рекомендації з пожежної профілактики. Виконаний розрахунки захисного заземлення, розрахунки кількості світильників у приміщенні, імовірності виникнення пожежі при виникненні короткого замикання від транзистора блоку живлення монітора.

Розроблена система задовольняє всім вимогам технічного завдання.

ПЕРЕЛІК ПОСИЛАНЬ

1) Стан та перспективи реформування житлово-комунального господарства України / Аналітичне дослідження. — К.: Лабораторія законодавчих актів, 2014.-100 с.

2) Праг К., Ирвин М. Библия пользователя Access для Windows 95, 3-е издание. – К.: Диалектика, 1996 – 576с., ил.

3) Сван Т. Секреты 32-разрядного программирования в Delphi.– К.: Диалектика, 1998. – 480 с.: ил.

4) Дейт К. Дж. Введение в системы баз данных.: Пер. с. англ. – 6–е изд..– К.: Диалектика, 1998. – 784 с.: ил.

5) Джексон Г. Проектирование реляционных БД для использования с микроЭВМ.–М.: МИР, 1991.– 252 с.

6) С компьютером на “ты”: Методическое пособие по информационным технологиям и Интернет.Вып.3/ Ред.сост. Л. А. Казаченкова.– М.: Либерея, 2000.– 112 с.

7) Методические указания по выполнению раздела "Охрана труда" в дипломном проекте.

8) Долин П.А. Справочник по технике безопасности. – М.:Энергоатомиздат,1985.

9) ССБТ 12.1.005–88. Воздух рабочей зоны. Общие санитарно–гигиенические требования.

10) СНиП 11–4–79. Нормы проектирования. – М.:Стройиздат, 1980. – 29 с.

11) Пожаровзрывоопасность веществ и материалов и средства их тушения./Справочник/ – М.: Статистика, 1989.

12) ССБТ 12.1.004-91. Пожарная безопасность. Общие требования.

13) ССБТ 12.1.018–91. Статическое электричество. Искробезопасность.

14) ССБТ 12.1.013–78. Электробезопасность. Общие требования.

15) ОНТП 24–86. Определение категорий помещений и зданий.

16) ССБТ 12.1.003–74. Опасные и вредные производственные факторы. Классификация.

17) Добровольский А.А. Переслыцких Ф.Ф. Пожарная техника. /Справочник/ - М.: Статистика, 1981.

ДОДАТОК А
Лістинг програми

```
1)  unit UEtalon;
2)
3)  interface
4)  uses
5)    Windows, Messages, SysUtils, Classes, Graphics, Controls,
Forms, Dialogs,
6)    ExtCtrls, Menus, Printers, VCLUtils, DB, DBTables;
7)
8)  procedure CreateEtalon;
9)  procedure SetIndexFile;
10) procedure CreateMassiv;
11) procedure AppendFrom(taOld, taNew: TTable);
12) procedure UpdateTable;
13) procedure ActiveTable;
14) function GetSoftVersion: string;
15) procedure SetSoftVersion;
16) procedure ConnectTemp(sName1, sName2: string);
17) procedure ActiveOneTable(sTitle, sDir, sTableName: string;
table1: TTable);
18) procedure UpdateOneTable(sTitle, sDir, sTableName: string;
Table1, EtTable: TTable);
19)
20) implementation
21) uses
22)   UData, USplash, UMain, UFunction;
23)
24) procedure CreateEtalon;
25) begin
26) // создание структуры для "параметры"
27)    fmSplash.lbProgress.Caption:= 'Обновление "Параметров"';
28)    fmSplash.lbProgress.Refresh;
29)    dm.taEtSost.Active:= false;
30)    dm.taEtSost.DatabaseName:=sWorkDir+'\info\';
31)    dm.taEtSost.TableName:= 'EtSost.db';
32)    dm.taEtSost.TableType:= ttParadox;
33)    if not FileExists(sWorkDir+ '\info\EtSost.db')then
34)     begin
35)      dm.taEtSost.FieldDefs.Clear;
36)      dm.taEtSost.FieldDefs.Add('ver', ftString, 4, false);
37)      dm.taEtSost.IndexDefs.Clear;
38)      dm.taEtSost.CreateTable;
39)     end;
40)    dm.taEtSost.Active:= true;
41) // создание структуры для "Справочника предприятий"
42)    fmSplash.lbProgress.Caption:= 'Обновление "Справочника
предприятий"';
43)    fmSplash.lbProgress.Refresh;
44)    dm.taEtPred.Active:= false;
45)    dm.taEtPred.DatabaseName:=GetCurrentDir+'\info\';
46)    dm.taEtPred.TableName:= 'EtPred.db';
47)    dm.taEtPred.TableType:= ttParadox;
48)    if not FileExists(GetCurrentDir+ '\info\EtPred.db')then
```

```
49)      begin
50)       dm.taEtPred.FieldDefs.Clear;
51)       dm.taEtPred.FieldDefs.Add('kod', ftInteger, 0, false);
52)       dm.taEtPred.FieldDefs.Add('nam', ftString, 30, false);
53)       dm.taEtPred.FieldDefs.Add('adr', ftString, 30, false);
54)       dm.taEtPred.FieldDefs.Add('tel', ftString, 12, false);
55)       dm.taEtPred.FieldDefs.Add('fio_ruk', ftString, 50, false);
56)       dm.taEtPred.FieldDefs.Add('rs', ftString, 20, false);
57)       dm.taEtPred.IndexDefs.Clear;
58)       dm.taEtPred.CreateTable;
59)      end;
60)     dm.taEtPred.Active:= true;
61)  // создание структуры для "Справочника Объектов"
62)     fmSplash.lbProgress.Caption:= 'Обновление "Справочника
объектов"';
63)     fmSplash.lbProgress.Refresh;
64)     dm.taEtObjekt.Active:= false;
65)     dm.taEtObjekt.DatabaseName:=GetCurrentDir+'\info\';
66)     dm.taEtObjekt.TableName:= 'EtObjekt.db';
67)     dm.taEtObjekt.TableType:= ttParadox;
68)     if not FileExists(GetCurrentDir+ '\info\EtObjekt.db')then
69)      begin
70)       dm.taEtObjekt.FieldDefs.Clear;
71)       dm.taEtObjekt.FieldDefs.Add('kod', ftInteger, 0, false);
72)       dm.taEtObjekt.FieldDefs.Add('nam', ftString, 30, false);
73)       dm.taEtObjekt.FieldDefs.Add('tarif', ftFloat, 0, false);
74)       dm.taEtObjekt.IndexDefs.Clear;
75)       dm.taEtObjekt.CreateTable;
76)      end;
77)     dm.taEtObjekt.Active:= true;
78)  // создание структуры для "Справочника улиц"
79)     fmSplash.lbProgress.Caption:= 'Обновление "Справочника
улиц"';
80)     fmSplash.lbProgress.Refresh;
81)     dm.taEtUl.Active:= false;
82)     dm.taEtUl.DatabaseName:=GetCurrentDir+'\info\';
83)     dm.taEtUl.TableName:= 'EtUl.db';
84)     dm.taEtUl.TableType:= ttParadox;
85)     if not FileExists(GetCurrentDir+ '\info\EtUl.db')then
86)      begin
87)       dm.taEtUl.FieldDefs.Clear;
88)       dm.taEtUl.FieldDefs.Add('kod', ftInteger, 0, false);
89)       dm.taEtUl.FieldDefs.Add('nam', ftString, 30, false);
90)       dm.taEtUl.IndexDefs.Clear;
91)       dm.taEtUl.CreateTable;
92)      end;
93)     dm.taEtUl.Active:= true;
94)  // создание структуры для "Картотеки"
95)     fmSplash.lbProgress.Caption:= 'Обновление "Картотеки"';
96)     fmSplash.lbProgress.Refresh;
97)     dm.taEtKart.Active:= false;
98)     dm.taEtKart.DatabaseName:=GetCurrentDir+'\data\';
99)     dm.taEtKart.TableName:= 'EtKart.db';
100)    dm.taEtKart.TableType:= ttParadox;
101)    if not FileExists(GetCurrentDir+ '\info\EtKart.db')then
102)     begin
103)      dm.taEtKart.FieldDefs.Clear;
104)      dm.taEtKart.FieldDefs.Add('kod', ftString, 15, false);
105)      dm.taEtKart.FieldDefs.Add('fam', ftString, 30, false);
```

```
106)      dm.taEtKart.FieldDefs.Add('ima', ftString, 25, false);
107)      dm.taEtKart.FieldDefs.Add('ots', ftString, 30, false);
108)      dm.taEtKart.FieldDefs.Add('idn', ftString, 10, false);
109)      dm.taEtKart.FieldDefs.Add('dat_r', ftDate, 0, false);
110)      dm.taEtKart.FieldDefs.Add('pred', ftInteger, 0, false);
111)      dm.taEtKart.FieldDefs.Add('ul', ftInteger, 0, false);
112)      dm.taEtKart.FieldDefs.Add('dom', ftString, 10, false);
113)      dm.taEtKart.FieldDefs.Add('kv', ftString, 10, false);
114)      dm.taEtKart.FieldDefs.Add('objekt', ftInteger, 0, false);
115)      dm.taEtKart.FieldDefs.Add('pl_ob', ftFloat, 0, false);
116)      dm.taEtKart.FieldDefs.Add('pl_ot', ftFloat, 0, false);
117)      dm.taEtKart.IndexDefs.Clear;
118)      dm.taEtKart.CreateTable;
119)     end;
120)    dm.taEtKart.Active:= true;
121) // создание структуры для "Оплаты"
122)    fmSplash.lbProgress.Caption:= 'Обновление "Оплаты"';
123)    fmSplash.lbProgress.Refresh;
124)    dm.taEtOplata.Active:= false;
125)    dm.taEtOplata.DatabaseName:=GetCurrentDir+'\data\';
126)    dm.taEtOplata.TableName:= 'EtOplata.db';
127)    dm.taEtOplata.TableType:= ttParadox;
128)    if not FileExists(GetCurrentDir+ '\info\EtOplata.db')then
129)     begin
130)      dm.taEtOplata.FieldDefs.Clear;
131)      dm.taEtOplata.FieldDefs.Add('kod', ftString, 15, false);
132)      dm.taEtOplata.FieldDefs.Add('period', ftDate, 0, false);
133)      dm.taEtOplata.FieldDefs.Add('dat_pl', ftDate, 0, false);
134)      dm.taEtOplata.FieldDefs.Add('summa_n', ftFloat, 0, false);
135)      dm.taEtOplata.FieldDefs.Add('summa_pl', ftFloat, 0, false);
136)      dm.taEtOplata.IndexDefs.Clear;
137)      dm.taEtOplata.CreateTable;
138)     end;
139)    dm.taEtOplata.Active:= true;
140)
141) end;
142)
143) procedure SetIndexFile;
144) begin
145) // Справочник предприятий
146)    fmSplash.lbProgress.Caption:= 'Индексация "Справочника
предприятий"';
147)    fmSplash.lbProgress.Refresh;
148)    dm.taPred.Active:= false;
149)    dm.taPred.AddIndex('I_Pred', 'kod' , [ixPrimary, ixUnique]);
150)    dm.taPred.AddIndex('I_Pred_nam', 'nam' ,
[ixCaseInsensitive]);
151)    dm.taPred.AddIndex('I_Pred_kod', 'kod' ,
[ixCaseInsensitive]);
152)    dm.taPred.IndexName:= 'I_Pred_kod';
153)    dm.taPred.Active:= true;
154) // Справочник улиц
155)    fmSplash.lbProgress.Caption:= 'Индексация "Справочника
улиц"';
156)    fmSplash.lbProgress.Refresh;
157)    dm.taUl.Active:= false;
158)    dm.taUl.AddIndex('I_Ul', 'kod' , [ixPrimary, ixUnique]);
159)    dm.taUl.AddIndex('I_Ul_nam', 'nam' , [ixCaseInsensitive]);
160)    dm.taUl.AddIndex('I_Ul_kod', 'kod' , [ixCaseInsensitive]);
```

```
161)    dm.taUl.IndexName:= 'I_Ul_kod';
162)    dm.taUl.Active:= true;
163) // Справочник Объектов
164)    fmSplash.lbProgress.Caption:= 'Индексация "Справочника
объектов"';
165)    fmSplash.lbProgress.Refresh;
166)    dm.taObjekt.Active:= false;
167)    dm.taObjekt.AddIndex('I_Objekt', 'kod' , [ixPrimary,
ixUnique]);
168)    dm.taObjekt.AddIndex('I_Objekt_nam', 'nam' ,
[ixCaseInsensitive]);
169)    dm.taObjekt.AddIndex('I_Objekt_kod', 'kod' ,
[ixCaseInsensitive]);
170)    dm.taObjekt.IndexName:= 'I_Objekt_kod';
171)    dm.taObjekt.Active:= true;
172) // картотека
173)    fmSplash.lbProgress.Caption:= 'Индексация "Картотеки"';
174)    fmSplash.lbProgress.Refresh;
175)    dm.taKart.Active:= false;
176)    dm.taKart.AddIndex('I_Kart', 'kod' , [ixPrimary, ixUnique]);
177)    dm.taKart.AddIndex('I_Kart_fam', 'fam;ima;ots;kod' ,
[ixCaseInsensitive]);
178)    dm.taKart.IndexName:= 'I_Kart_fam';
179)    dm.taKart.Active:= true;
180) // оплата
181)    fmSplash.lbProgress.Caption:= 'Индексация "Оплаты"';
182)    fmSplash.lbProgress.Refresh;
183)    dm.taOplata.Active:= false;
184)    dm.taOplata.AddIndex('I_Oplata', 'kod;period' , [ixPrimary,
ixUnique]);
185)    dm.taOplata.AddIndex('I_Oplata_kod', 'kod;period' ,
[ixCaseInsensitive]);
186)    dm.taOplata.IndexName:= 'I_Oplata_kod';
187)    dm.taOplata.Active:= true;
188) end;
189)
190) procedure CreateMassiv;
191) begin
192)    fmSplash.lbProgress.Caption:= 'Подготовка вспомогательных
данных';
193)    fmSplash.lbProgress.Refresh;
194) // Справочник предприятий
195)    Create_ms(dm.taPred, msPred, iKolPred, 'nam');
196)    iMaxPred:= iKolPred;
197) // Справочник улиц
198)    Create_ms(dm.taUl, msUl, iKolUl, 'nam');
199)    iMaxUl:= iKolUl;
200) // Справочник объектов
201)    Create_ms(dm.taObjekt, msObjekt, iKolObjekt, 'nam');
202)    iMaxObjekt:= iKolObjekt;
203) end;
204)
205) procedure UpdateTable;
206) begin
207)    UpdateOneTable('Обновление параметров', sWorkDir+'\info\',
'sost.db', dm.taSost, dm.taEtSost);
208)    UpdateOneTable('Обновление "Справочника предприятий"',
sWorkDir+'\info\', 'pred.db', dm.taPred, dm.taEtPred);
```

```
209)    UpdateOneTable('Обновление "Справочника улиц"',
sWorkDir+'\info\', 'Ul.db', dm.taUl, dm.taEtUl);
210)    UpdateOneTable('Обновление "Справочника объектов"',
sWorkDir+'\info\', 'Objekt.db', dm.taObjekt, dm.taEtObjekt);
211)    UpdateOneTable('Обновление "Картотеки"', sWorkDir+'\data\',
'Kart.db', dm.taKart, dm.taEtKart);
212)    UpdateOneTable('Обновление "Оплаты"', sWorkDir+'\data\',
'Oplata.db', dm.taOplata, dm.taEtOplata);
213) end;
214)
215) procedure ActiveTable;
216) begin
217)    ActiveOneTable('Открытие параметров', sWorkDir+'\info\',
'sost.db', dm.taSost);
218)    ActiveOneTable('Открытие "Справочника предприятий"',
sWorkDir+'\info\', 'pred.db', dm.taPred);
219)    ActiveOneTable('Открытие "Справочника предприятий"',
sWorkDir+'\info\', 'ul.db', dm.taUl);
220)    ActiveOneTable('Открытие "Справочника предприятий"',
sWorkDir+'\info\', 'objekt.db', dm.taObjekt);
221)    ActiveOneTable('Открытие "Справочника предприятий"',
sWorkDir+'\data\', 'kart.db', dm.taKart);
222)    ActiveOneTable('Открытие "Справочника предприятий"',
sWorkDir+'\data\', 'oplata.db', dm.taOplata);
223) end;
224)
225) function GetSoftVersion: string;
226) begin
227) //параметры
228)    fmSplash.lbProgress.Caption:= 'Подготовка "Параметров
программы"';
229)    fmSplash.lbProgress.Refresh;
230)    dm.taSost.Active:= false;
231)    dm.taSost.DatabaseName:=sWorkDir+'\info\';
232)    dm.taSost.TableName:= 'Sost.db';
233)    dm.taSost.TableType:= ttParadox;
234)    dm.taSost.Active:= true;
235)    GetSoftVersion:= dm.taSost.FieldByName('ver').AsString;
236)    sOldVer:= dm.taSost.FieldByName('ver').AsString;
237) end;
238)
239) end.
240) // -----------------------------------------
241) unit UMain;


242) interface


243) uses
244) Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs,
245) ComCtrls, RxGrdCpt, ExtCtrls, Placemnt, DBActns, StdActns,
ActnList,
246) ImgList, Menus, RxMenus, RxHints, UFunction, Registry, StdCtrls,
247) JvExComCtrls, JvComCtrls, SpeedBar, Grids, DBGridEh,
JvExControls,
248) JvComponent, JvgCheckBox, JvXPCore, JvXPCheckCtrls,
JvExExtCtrls,
```

```
249) JvNetscapeSplitter, OleCtrls, JvSpeedbar, JvEditorCommon,
JvEditor, ComObj,
250) FileCtrl, ShellAPI, SHDocVw, JvXPButtons, Mask, JvExMask,
JvToolEdit,
251) Buttons, DBTables, JvProgressBar, JvXPProgressBar, RXCtrls,
JvPanel,
252) JvPageList, RXSplit, JvMaskEdit, JvDBControls, DBCtrls,
JvDBImage,
253) JvgPage, ExtDlgs, DB, JvExtComponent;


254) type
255) TfmMain = class(TForm)
256) BaseImage: TImageList;
257) Timer2: TTimer;
258) StatusBar1: TStatusBar;
259) pcUMKD: TJvPageControl;
260) tsOplata: TTabSheet;
261) tsObjekt: TTabSheet;
262) sbObjekt: TSpeedBar;
263) Bevel1: TBevel;
264) SpeedbarSection1: TSpeedbarSection;
265) siFirstObjekt: TSpeedItem;
266) siPrevObjekt: TSpeedItem;
267) siNextObjekt: TSpeedItem;
268) siLastObjekt: TSpeedItem;
269) siInsertObjekt: TSpeedItem;
270) siDeleteObjekt: TSpeedItem;
271) siClearObjekt: TSpeedItem;
272) dbgObjekt: TDBGridEh;
273) im24: TImageList;
274) im32: TImageList;
275) FormStorage1: TFormStorage;
276) tsKart: TTabSheet;
277) tsPred: TTabSheet;
278) sbKart: TSpeedBar;
279) Bevel2: TBevel;
280) SpeedbarSection2: TSpeedbarSection;
281) siFirstKart: TSpeedItem;
282) siPrevKart: TSpeedItem;
283) siNextKart: TSpeedItem;
284) siLastKart: TSpeedItem;
285) siInsertKart: TSpeedItem;
286) siDeleteKart: TSpeedItem;
287) siClearKart: TSpeedItem;
288) dbgKart: TDBGridEh;
289) sbPred: TSpeedBar;
290) Bevel3: TBevel;
291) SpeedbarSection3: TSpeedbarSection;
292) siFirstPred: TSpeedItem;
293) siPrevPred: TSpeedItem;
294) siNextPred: TSpeedItem;
295) siLastPred: TSpeedItem;
296) siInsertPred: TSpeedItem;
297) siDeletePred: TSpeedItem;
298) siClearPred: TSpeedItem;
299) dbgPred: TDBGridEh;
300) tsAbout: TTabSheet;
301) gbAbout: TGroupBox;
```

```
302) imAbout: TImage;
303) lbAbout1: TRxLabel;
304) lbAbout2: TRxLabel;
305) buExit: TJvXPButton;
306) acMain: TActionList;
307) acInsert: TAction;
308) tsOtchet: TTabSheet;
309) tsUl: TTabSheet;
310) SpeedBar1: TSpeedBar;
311) be1: TBevel;
312) SpeedbarSection4: TSpeedbarSection;
313) siFirstUl: TSpeedItem;
314) siPrevUl: TSpeedItem;
315) siNextUl: TSpeedItem;
316) siLastUl: TSpeedItem;
317) siInsertUl: TSpeedItem;
318) siDeleteUl: TSpeedItem;
319) siClesrUl: TSpeedItem;
320) dbgUl: TDBGridEh;
321) SpeedBar2: TSpeedBar;
322) be2: TBevel;
323) SpeedbarSection5: TSpeedbarSection;
324) si1: TSpeedItem;
325) si2: TSpeedItem;
326) si3: TSpeedItem;
327) si4: TSpeedItem;
328) si5: TSpeedItem;
329) si6: TSpeedItem;
330) siClearOplata: TSpeedItem;
331) dbgOplata: TDBGridEh;
332) procedure Timer2Timer(Sender: TObject);
333) procedure FormCreate(Sender: TObject);
334) procedure SetVisualParam;
335) procedure EnabledFirstButton(f: boolean; siFirst, siPrev:
TSpeedItem);
336) procedure EnabledLastButton(f: boolean; siNext, siLast:
TSpeedItem);
337) procedure siFirstObjektClick(Sender: TObject);
338) procedure siPrevObjektClick(Sender: TObject);
339) procedure siNextObjektClick(Sender: TObject);
340) procedure siLastObjektClick(Sender: TObject);
341) procedure siInsertObjektClick(Sender: TObject);
342) procedure siDeleteObjektClick(Sender: TObject);
343) procedure siClearObjektClick(Sender: TObject);
344) procedure dbgObjektKeyDown(Sender: TObject; var Key: Word;
a.   Shift: TShiftState);
345) procedure FormCanResize(Sender: TObject; var NewWidth,
a.   NewHeight: Integer; var Resize: Boolean);
346) procedure dbgKartKeyDown(Sender: TObject; var Key: Word;
a.   Shift: TShiftState);
347) procedure dbgKartCellClick(Column: TColumnEh);
348) procedure dbgObjektCellClick(Column: TColumnEh);
349) procedure siFirstKartClick(Sender: TObject);
350) procedure siPrevKartClick(Sender: TObject);
351) procedure siNextKartClick(Sender: TObject);
352) procedure siLastKartClick(Sender: TObject);
353) procedure siInsertKartClick(Sender: TObject);
354) procedure siDeleteKartClick(Sender: TObject);
355) procedure siClearKartClick(Sender: TObject);
```

```
356) procedure siFirstPredClick(Sender: TObject);
357) procedure siPrevPredClick(Sender: TObject);
358) procedure siNextPredClick(Sender: TObject);
359) procedure siLastPredClick(Sender: TObject);
360) procedure siInsertPredClick(Sender: TObject);
361) procedure siDeletePredClick(Sender: TObject);
362) procedure siClearPredClick(Sender: TObject);
363) procedure dbgPredCellClick(Column: TColumnEh);
a.   procedure dbgPredKeyDown(Sender: TObject; var Key: Word;Shift:
TShiftState);
364) procedure buExitClick(Sender: TObject);


365) private
366) { Private declarations }
367) public
368) { Public declarations }
369) end;


370) const
371) indexLng:integer=0;
372) sVer: string='1.00';
373) sDat: string='02.10.2013';
374) msFileList: array[1..6]of string=('\info\sost.db',
'\info\pred.db', '\info\ul.db',
'\info\objekt.db', '\data\kart.db',

'\data\oplata.db');

375) iFileCount: integer= 6;
376) sSearch: string='';
377) var
378) fmMain: TfmMain;
379) msPred, msUl, msObjekt: din_ar;
380) sOldVer: string;
381) flUpdateMode, flInsertMode: boolean;


382) implementation


383) uses UData, UEtalon, USplash, UInfo, UddSpr;


384) {$R *.DFM}


385) procedure TfmMain.Timer2Timer(Sender: TObject);
386) begin
387) StatusBar1.Panels[0].Text:= TimeToStr(Time);
388) StatusBar1.Panels[1].Text:=FormatDateTime('d.mm.yyyy',Date)
389) end;


390) procedure TfmMain.FormCreate(Sender: TObject);
391) var
392) i: integer;
393) fl: boolean;
```

```
394) begin
395) sWorkDir:= GetCurrentDir;
396) SetHintStyle(hsRectangle, 2, True, taCenter);
397) fl:= false;
398) for i:= 1 to iFileCount do
399) if not FileExists(sWorkDir+msFileList[i])then
400) begin
401) fl:= true;
402) break;
403) end;
404) if fl then
405) // не хватает файла
406) begin
407) CreateEtalon;
408) UpdateTable;
409) ActiveTable;
410) SetSoftVersion;
411) SetIndexFile;
412) CreateMassiv;
413) end
414) else
415) // все файлы
416) begin
417) if sVer<>GetSoftVersion then
418) // не та версия
begin

CreateEtalon;

UpdateTable;

ActiveTable;

SetSoftVersion;

SetIndexFile;

CreateMassiv;

end

419) else
420) // та версия
begin

ActiveTable;

SetIndexFile;

CreateMassiv;

end;

421) end;
422) fmSplash.lbProgress.Caption:= 'Подготовка данных';
423) fmSplash.lbProgress.Refresh;
424) pcUMKD.ActivePageIndex:= 0;
425) // параметры
426) SetVisualParam;
```

```
427) end;


428) procedure TfmMain.SetVisualParam;
429) begin
430) dm.taOplata.MasterSource:= dm.dsKart;
431) dm.taOplata.MasterFields:= 'kod';
432) end;


433) procedure TfmMain.EnabledFirstButton(f: boolean; siFirst,
siPrev: TSpeedItem);
434) begin
435) siFirst.Enabled:=f;
436) siPrev.Enabled:=f;
437) end;


438) procedure TfmMain.EnabledLastButton(f: boolean; siNext, siLast:
TSpeedItem);
439) begin
440) siLast.Enabled:=f;
441) siNext.Enabled:=f;
442) end;


443) procedure TfmMain.siFirstObjektClick(Sender: TObject);
444) begin
445) dm.taObjekt.First;
446) EnabledLastButton(true, siPrevObjekt, siLastObjekt);
447) EnabledFirstButton(false, siFirstObjekt, siNextObjekt);
448) end;


449) procedure TfmMain.siPrevObjektClick(Sender: TObject);
450) begin
451) if dm.taObjekt.RecNo>1 then dm.taObjekt.MoveBy(-1);
452) EnabledFirstButton(dm.taObjekt.RecNo>1, siFirstObjekt,
siNextObjekt);
453) EnabledLastButton(true, siPrevObjekt, siLastObjekt);
454) end;


455) procedure TfmMain.siNextObjektClick(Sender: TObject);
456) begin
457) if dm.taObjekt.RecNo<=dm.taObjekt.RecordCount-1 then
dm.taObjekt.MoveBy(1);
458) EnabledLastButton(dm.taObjekt.RecNo<=dm.taObjekt.RecordCount-1,
siPrevObjekt, siLastObjekt);
459) EnabledFirstButton(true, siFirstObjekt, siNextObjekt);
460) end;


461) procedure TfmMain.siLastObjektClick(Sender: TObject);
462) begin
463) dm.taObjekt.Last;
464) EnabledLastButton(false, siPrevObjekt, siLastObjekt);
465) EnabledFirstButton(true, siFirstObjekt, siNextObjekt);
466) end;
```

```
467) procedure TfmMain.siInsertObjektClick(Sender: TObject);
468) begin
469) dm.taObjekt.Append;
470) inc(iMaxObjekt);
471) dm.taObjekt.FieldByName('kod').AsInteger:= iMaxObjekt;
472) EnabledLastButton(false, siPrevObjekt, siLastObjekt);
473) EnabledFirstButton(true, siFirstObjekt, siNextObjekt);
474) end;


475) procedure TfmMain.siDeleteObjektClick(Sender: TObject);
476) begin
477) dm.taObjekt.Delete;
478) siDeleteObjekt.Enabled:= not dm.taObjekt.IsEmpty;
479) EnabledLastButton(dm.taObjekt.RecNo<=dm.taObjekt.RecordCount-1,
siPrevObjekt, siLastObjekt);
480) EnabledFirstButton(dm.taObjekt.RecNo>1, siFirstObjekt,
siNextObjekt);
481) end;


482) procedure TfmMain.siClearObjektClick(Sender: TObject);
483) begin
484) if MessageDlg('Очистить "Справочник объектов"?', mtConfirmation,
[mbYes, mbNo],0)=idYes then
485) begin
486) dm.taObjekt.Active:= false;
487) dm.taObjekt.EmptyTable;
488) dm.taObjekt.Active:= true;
489) siDeleteObjekt.Enabled:= false;
490) siInsertObjekt.Enabled:= true;
491) EnabledLastButton(false, siFirstObjekt, siNextObjekt);
492) EnabledFirstButton(false, siPrevObjekt, siLastObjekt);
493) iMaxObjekt:= 0;
494) end;
495) end;


496) procedure TfmMain.dbgObjektKeyDown(Sender: TObject; var Key:
Word;
497) Shift: TShiftState);
498) begin
499) if (Key=vk_Up)or(Key=vk_Down)or(Key=34)or(Key=33)then
500) begin
501) EnabledFirstButton(dm.taObjekt.RecNo>1, siFirstObjekt,
siNextObjekt);
502) EnabledLastButton(dm.taObjekt.RecNo<=dm.taObjekt.RecordCount-1,
siPrevObjekt, siLastObjekt);
503) end;
504) if Key=13 then dm.taObjekt.Refresh;
505) end;


506) procedure TfmMain.FormCanResize(Sender: TObject; var NewWidth,
507) NewHeight: Integer; var Resize: Boolean);
508) begin
```

```
509) //   dbgObjekt.Columns[1].Width:= dbgObjekt.Width-
dbgObjekt.Columns[0].Width-40;
510) end;


511) procedure TfmMain.dbgKartKeyDown(Sender: TObject; var Key: Word;
512) Shift: TShiftState);
513) begin
514) if (Key=vk_Up)or(Key=vk_Down)or(Key=34)or(Key=33)then
515) begin
516) EnabledFirstButton(dm.taKart.RecNo>1, siFirstKart, siPrevKart);
517) EnabledLastButton(dm.taKart.RecNo<=dm.taKart.RecordCount-1,
siNextKart, siLastKart);
518) end;
519) if Key=13 then dm.taKart.Refresh;
520) end;


521) procedure TfmMain.dbgKartCellClick(Column: TColumnEh);
522) begin
523) EnabledFirstButton(dm.taKart.RecNo>1, siFirstKart, siPrevKart);
524) EnabledLastButton(dm.taKart.RecNo<=dm.taKart.RecordCount-1,
siNextKart, siLastKart);
525) end;


526) procedure TfmMain.dbgObjektCellClick(Column: TColumnEh);
527) begin
528) EnabledFirstButton(dm.taObjekt.RecNo>1, siFirstObjekt,
siNextObjekt);
529) EnabledLastButton(dm.taObjekt.RecNo<=dm.taObjekt.RecordCount-1,
siPrevObjekt, siLastObjekt);
530) end;


531) procedure TfmMain.siFirstKartClick(Sender: TObject);
532) begin
533) dm.taKart.First;
534) EnabledLastButton(true, siNextKart, siLastKart);
535) EnabledFirstButton(false, siFirstKart, siPrevKart);
536) end;


537) procedure TfmMain.siPrevKartClick(Sender: TObject);
538) begin
539) if dm.taKart.RecNo>1 then dm.taKart.MoveBy(-1);
540) EnabledFirstButton(dm.taKart.RecNo>1, siFirstKart, siPrevKart);
541) EnabledLastButton(true, siNextKart, siLastKart);
542) end;


543) procedure TfmMain.siNextKartClick(Sender: TObject);
544) begin
545) if dm.taKart.RecNo<=dm.taKart.RecordCount-1 then
dm.taKart.MoveBy(1);
546) EnabledLastButton(dm.taKart.RecNo<=dm.taKart.RecordCount-1,
siNextKart, siLastKart);
547) EnabledFirstButton(true, siFirstKart, siPrevKart);
548) end;
```

```
549) procedure TfmMain.siLastKartClick(Sender: TObject);
550) begin
551) dm.taKart.Last;
552) EnabledLastButton(false, siNextKart, siLastKart);
553) EnabledFirstButton(true, siFirstKart, siPrevKart);
554) end;


555) procedure TfmMain.siInsertKartClick(Sender: TObject);
556) begin
557) dm.taKart.Append;
558) EnabledLastButton(false, siNextKart, siLastKart);
559) EnabledFirstButton(true, siFirstKart, siPrevKart);
560) end;


561) procedure TfmMain.siDeleteKartClick(Sender: TObject);
562) begin
563) dm.taKart.Delete;
564) siDeleteKart.Enabled:= not dm.taKart.IsEmpty;
565) EnabledLastButton(dm.taKart.RecNo<=dm.taKart.RecordCount-1,
siNextKart, siLastKart);
566) EnabledFirstButton(dm.taKart.RecNo>1, siFirstKart, siPrevKart);
567) end;


568) procedure TfmMain.siClearKartClick(Sender: TObject);
569) begin
570) if MessageDlg('Очистить картотеку ?',
1.   mtConfirmation, [mbYes, mbNo],0)=idYes then
571) begin
572) dm.taKart.Active:= false;
573) dm.taKart.EmptyTable;
574) dm.taKart.Active:= true;
575) siDeleteKart.Enabled:= false;
576) siInsertKart.Enabled:= true;
577) EnabledLastButton(false, siNextKart, siLastKart);
578) EnabledFirstButton(false, siFirstKart, siPrevKart);
579) end;
580) end;


581) procedure TfmMain.siFirstPredClick(Sender: TObject);
582) begin
583) dm.taPred.First;
584) EnabledLastButton(true, siNextPred, siLastPred);
585) EnabledFirstButton(false, siFirstPred, siPrevPred);
586) end;


587) procedure TfmMain.siPrevPredClick(Sender: TObject);
588) begin
589) if dm.taPred.RecNo>1 then dm.taPred.MoveBy(-1);
590) EnabledFirstButton(dm.taPred.RecNo>1, siFirstPred, siPrevPred);
591) EnabledLastButton(true, siNextPred, siLastPred);
592) end;
```

```
593) procedure TfmMain.siNextPredClick(Sender: TObject);
594) begin
595) if dm.taPred.RecNo<=dm.taPred.RecordCount-1 then
dm.taPred.MoveBy(1);
596) EnabledLastButton(dm.taPred.RecNo<=dm.taPred.RecordCount-1,
siNextPred, siLastPred);
597) EnabledFirstButton(true, siFirstPred, siPrevPred);
598) end;


599) procedure TfmMain.siLastPredClick(Sender: TObject);
600) begin
601) dm.taPred.Last;
602) EnabledLastButton(false, siNextPred, siLastPred);
603) EnabledFirstButton(true, siFirstPred, siPrevPred);
604) end;


605) procedure TfmMain.siInsertPredClick(Sender: TObject);
606) begin
607) dm.taPred.Append;
608) inc(iMaxPred);
609) dm.taPred.FieldByName('kod').AsInteger:= iMaxPred;
610) EnabledLastButton(false, siNextPred, siLastPred);
611) EnabledFirstButton(true, siFirstPred, siPrevPred);
612) end;


613) procedure TfmMain.siDeletePredClick(Sender: TObject);
614) begin
615) dm.taPred.Delete;
616) siDeletePred.Enabled:= not dm.taPred.IsEmpty;
617) EnabledLastButton(dm.taPred.RecNo<=dm.taPred.RecordCount-1,
siNextPred, siLastPred);
618) EnabledFirstButton(dm.taPred.RecNo>1, siFirstPred, siPrevPred);
619) end;


620) procedure TfmMain.siClearPredClick(Sender: TObject);
621) begin
622) if MessageDlg('Очистить "Справочник предприятий"?',
mtConfirmation, [mbYes, mbNo],0)=idYes then
623) begin
624) dm.taPred.Active:= false;
625) dm.taPred.EmptyTable;
626) dm.taPred.Active:= true;
627) siDeletePred.Enabled:= false;
628) siInsertPred.Enabled:= true;
629) EnabledLastButton(false, siNextPred, siLastPred);
630) EnabledFirstButton(false, siFirstPred, siPrevPred);
631) iMaxPred:= 0;
632) end;
633) end;


634) procedure TfmMain.dbgPredCellClick(Column: TColumnEh);
635) begin
636) EnabledFirstButton(dm.taPred.RecNo>1, siFirstPred, siPrevPred);
```

```
637) EnabledLastButton(dm.taPred.RecNo<=dm.taPred.RecordCount-1,
siNextPred, siLastPred);
638) end;


639) procedure TfmMain.dbgPredKeyDown(Sender: TObject; var Key: Word;
640) Shift: TShiftState);
641) begin
642) if (Key=vk_Up)or(Key=vk_Down)or(Key=34)or(Key=33)then
643) begin
644) EnabledFirstButton(dm.taPred.RecNo>1, siFirstPred, siPrevPred);
645) EnabledLastButton(dm.taPred.RecNo<=dm.taPred.RecordCount-1,
siNextPred, siLastPred);
646) end;
647) if Key=13 then dm.taPred.Refresh;
648) end;


649) procedure TfmMain.buExitClick(Sender: TObject);
650) begin
651) close;
652) end;


653) end.

654) //---------------------------------------

655) unit UddSpr;
656)
657) interface
658)
659) uses
660)   Windows, Messages, SysUtils, Classes, Graphics, Controls,
Forms, Dialogs,
661)   Grids, DBGridEh, DB, DBTables;
662)
663) type
664)   TfmddSpr = class(TForm)
665)     dbgSpr: TDBGridEh;
666)     procedure dbgSprDblClick(Sender: TObject);
667)     procedure dbgSprKeyUp(Sender: TObject; var Key: Word;
668)       Shift: TShiftState);
669)     procedure dbgSprKeyPress(Sender: TObject; var Key: Char);
670)   private
671)     procedure OnActivate(var msg: TWMActivate); message
WM_ACTIVATE;
672)     { Private declarations }
673)   public
674)     function Execute(HostControl: TControl; WorkTable: TTable;
var iKodSpr: integer; var sNamSpr: String):Boolean;
675)     { Public declarations }
676)   end;
677)
678) var
679)   fmddSpr: TfmddSpr;
680)   sSearch: string;
681)   myTable: TTable;
682)
```

```
683) implementation
684)
685) uses UData, UMain, UFunction;
686)
687) {$R *.DFM}
688)
689) function TfmddSpr.Execute(HostControl: TControl; WorkTable:
TTable; var iKodSpr: integer; var sNamSpr: String):Boolean;
690) begin
691)    sSearch:= '';
692)    if iKodSpr <> 0 then WorkTable.Locate('kod',iKodSpr,[]);
693)    myTable:= WorkTable;
694)    AdjustDropDownForm(Self,HostControl);
695)    Visible:= True;
696)    ModalResult:= mrCancel;
697)    while (Visible) do Application.ProcessMessages;
698)    Result:= False;
699)    if ModalResult= mrOk then
700)    begin
701)     iKodSpr:= WorkTable.FieldByName('kod').AsInteger;
702)     sNamSpr:= WorkTable.FieldByName('nam').AsString;
703)     Result:= True;
704)    end;
705) end;
706)
707) procedure TfmddSpr.dbgSprDblClick(Sender: TObject);
708) begin
709)    ModalResult:= mrOk;
710)    close;
711) end;
712)
713) procedure TfmddSpr.OnActivate(var msg: TWMActivate);
714) begin
715)   inherited;
716)   if (msg.Active=WA_INACTIVE) then
717)    Close;
718) end;
719)
720) procedure TfmddSpr.dbgSprKeyUp(Sender: TObject; var Key: Word;
721)   Shift: TShiftState);
722) begin
723)    if Key= vk_Escape then close;
724)    if Key= 13 then dbgSprDblClick(Sender);
725) end;
726)
727) procedure TfmddSpr.dbgSprKeyPress(Sender: TObject; var Key:
Char);
728) begin
729)    if ((key>='A')and(key<='я'))or(key='I')or(key='i')
730)       or(key='Є')or(key='є')or
731)        (key='Ї')or(key='ї')or
732)        ((key>='0')and(key<='9'))or
733)        (key='-')then
734)     begin
735) // поиск по названию
736)      sSearch:= sSearch+key;
737) //   gb.Caption:= 'Поиск: '+sSearch;
738)      myTable.DisableControls;
739)      myTable.First;
```

```
740)      while not myTable.Eof do
741)       begin
742)        if pos(BigChar(sSearch),
BigChar(myTable.FieldByName('nam').AsString))=1 then
743)         break;
744)        myTable.Next;
745)       end;
746)      myTable.EnableControls;
747)      key:= chr(0);
748)     end
749)    else
750)     sSearch:= '';
751) end;
752)
753) end.
```