МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____ Скарга-Бандурова І.С.
«____»_____ 20__ р.

ДИПЛОМНИЙ ПРОЕКТ (РОБОТА) БАКАЛАВРА

ПОЯСНЮВАЛЬНА ЗАПИСКА

НА ТЕМУ:

Програмне та апаратне забезпечення системи регіонального екологічного моніторингу

Освітньо-кваліфікаційний рівень "бакалавр"
Напрям підготовки 6.050102 – "комп'ютерна інженерія"

Керівник проекту:	_____ (підпис)	Рязанцев О.І. (ініціали, прізвище)
Консультант з охорони праці:	_____ (підпис)	Критська Я.О. (ініціали, прізвище)
Здобувач вищої освіти:	_____ (підпис)	Безкоровайний Р.В. (ініціали, прізвище)
Група:		КІ-14з

Сєвєродонецьк 2018

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітньо-кваліфікаційний рівень бакалавр
Напрям підготовки 6.050102 – "комп'ютерна інженерія"
(шифр і назва)
Спеціальність
(шифр і назва)

ЗАТВЕРДЖУЮ:

Завідувач кафедри КНІ
І.С. Скарга-Бандурова
«_____» _____20____р.

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) БАКАЛАВРА

Безкоровайному Руслану Володимировичу
(прізвище, ім'я, по батькові)

1. Тема роботи Програмне та апаратне забезпечення системи регіонального екологічного моніторингу

керівник проекту (роботи) Рязанцев Олександр Іванович, д.т.н., проф.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від " " 201_ р. № ____
2. Термін подання студентом роботи 16.06.2018
3. Вихідні дані до роботи Методи аналізу зображень, теоретичні відомості про методи ущільнення зображень, середа розробки Microsoft Visual Studio 2013

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Аналіз і постановка технічного завдання, вибір мови програмування, опис програми, охорона праці

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Електронні плакати

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	ст. викл. Критська Я.О.		

7. Дата видачі завдання 30.04.2018

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз завдання та робота з літературою	05.05.2018 - 13.05.2018	
2	Розробка програмного забезпечення	14.05.2018 - 22.05.2018	
3	Тестування програмного забезпечення	22.05.2018 - 02.06.2018	
4	Розробка розділу «Охорона праці»	02.06 .2018- 11.06.2018	
5	Оформлення пояснювальної записки та	11.06.2018 - 16.06.2018	
	електронних плакатів		

Здобувач вищої освіти _____ Безкоровайний Р.В.
(підпис) (прізвище та ініціали)

Керівник _____ Рязанцев О.І.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту: 92с., рис. 23, таблиць 13, джерел 29, додаток1.

Галузь роботи: розробка розподіленої бази даних регіональної системи екологічного моніторингу.

Проаналізовано такі методи розробки програмного забезпечення: C++, C#, DELPHI, Java, PHP, Python, MySQL. Проаналізовано такі технології розробки програмного забезпечення: Apache, MySQL, PHP.

Мета розробки даної системи:

– створення програмного забезпечення, що здійснює збір інформації, дозволяє отримати об'єктивну оцінку про стан повітря міста у будь-який час;

– аналіз моніторингової інформації і генерація звітів.

У розробленому програмному забезпеченні розподіленої бази даних регіональної системи екологічного моніторингу виконані наступні вимоги:

– забезпечення безпечних методів відправки і здобуття моніторингової інформації ручним введенням і автоматизованим;

– незалежність від числа постачальників моніторинговій інформації і вигляду її вистави, а також від часу оновлення;

– розумні вимоги до ресурсів, що дозволяють всій системі функціонувати в реальному режимі часу навіть при інтенсивному потоці заявок до підсистеми;

– забезпечення необхідної частоти перевірки на наявність оновлень моніторингової інформації з аналізом результату і генерацією звіту.

Ключові слова: екологічний моніторинг, методи розрахунків, мережа, потік інформації, PHP, APACHE, MySQL, автоматизована система.

Умови одержання дипломного проекту: СНУ ім. В. Даля, пр. Центральний 59-А, м. Сєвєродонецьк, 93400.

ЗМІСТ

ВСТУП

Усебічний аналіз довкілля передбачає оцінку його екологічного стану і вплив на нього природних і антропогенних дій. Характер цих дій дуже специфічний. Лімітуючим показником рівня природних і антропогенних дій є гранично-допустиме екологічне навантаження (ГДЕН), яке в багатьох країнах встановлене у зв'язку з тим, що нормальне функціонування і стійкість екосистем і біосфери можливі при неперевищенні певних граничних навантажень на них.

Стан біосфери, що безперервно змінюється під впливом природних чинників, зазвичай повертається в первинний. Наприклад, зміни температури і тиску, вологості повітря і грунту відбуваються в межах деяких постійних середніх значень. Як правило, великі екосистеми під впливом природних процесів змінюються надзвичайно повільно. Існуючі у світі екологічні служби (гідрометеорологічна, сейсмічна, іоносферна та ін.) проводять контроль за зміною цих процесів.

Зміна стану біосфери під впливом антропогенних чинників відбувається в коротші тимчасові терміни. Тому з метою виміру, оцінки і прогнозу антропогенних змін абіотичної складової біосфери (в першу чергу забруднень) і реакції у відповідь біоти на ці зміни, а також подальших змін в екосистемах в результаті антропогенних дій створена інформаційна система екологічного моніторингу.

Надзвичайно важливим чинником, що істотно впливає на територіальну організацію усього соціально-економічного життя і ефективність виробництва, є екологічна обстановка. У останні десятиліття в Україні вона істотно погіршала. Одним з основних чинників, що вплинули на екологічну обстановку, є розвиток добувної і переробної промисловості при застарілих технологіях і пов'язана з цим надмірна урбанізація багатьох районів, перш за все усього Донбасу.

Донбас – це великий промисловий регіон України, в якому налічується декілька тисяч великих промислових підприємств, виробничо-промислових

об'єднань і підприємств паливно-енергетичного комплексу, гірничодобувної, металургійної, хімічної промисловості, важкого машинобудування, будівельної галузі, а також агропромислового комплексу. Донбас забезпечує велику частину промислового виробництва України, причому в найбільш екологічно небезпечних галузях.

Висока концентрація промислового і сільськогосподарського виробництва, транспортної інфраструктури, у поєднанні з високою щільністю населення, створили надзвичайно високе техногенне і антропогенне навантаження на біосферу – найвищу в Україні і Європі. Сумарне техногенне навантаження на одиницю території регіону в 4 рази вище середнього по Україні. Донбас має запаси майже усіх хімічних елементів. Головним природним багатством регіону є родовища кам'яного вугілля. Його запаси тільки в Донецькій області оцінюються в 25 млрд. т, що може задовольнити потреби України не на одне десятиліття вперед.

Незважаючи на спад виробництва, в результаті якого загальна кількість викидів і скидань істотно зменшилася, навантаження на біосферу Донбасу як і раніше залишається одним з найбільших в Європі. Підприємства регіону викидають біля третини сумарного об'єму забруднюючих речовин на Україні. Високі швидкості і масштаби техногенних процесів, величезні переміщення гірських мас обумовлюють великі об'єми розсіювання багатьох хімічних елементів (раніше усього вуглецю і важких металів), викликають накопичення в довкіллі з'єднань хімічних елементів в невластивих природі поєднаннях.

З вищесказаного видно, що Донбас відноситься до найбільш критичних по екологічній обстановці регіонів України. Найгострішими проблемами регіону є: забруднення атмосферного повітря, водного басейну і грунтів.

Місто Сєвєродонецьк є територіально-виробничим комплексом (ТВК), де найважливішою галуззю є хімічна промисловість, разом з якою досить розвинені теплоенергетика, автотранспорт, будівництво та інші. Більшість розташованих у місті підприємств застосовують складні технології, що супроводжується виділенням в довкілля хімічних речовин і з'єднань, багато яких відноситься до 1 і 2 класів небезпеки, що, поза сумнівом, робить

негативний вплив на екологічну ситуацію в місті. Тому для даного міста найбільш актуальний є екологічний моніторинг забруднення атмосферного повітря, оскільки на території міста зосереджено 26 великих промислових підприємств, крім того, чинять вплив підприємства сусідніх міст (Лисичанська, Рубіжного). Стан повітряного басейну міста визначає соціально-промисловий комплекс, що склався на території міста і характеризується певними викидами отруйних хімічних речовин (ОХР) в атмосферу, а також метеорологічні умови, що істотно впливають на перенесення і розсіювання ОХВ. Даній проблемі присвячені численні дослідження, які породжують конкретну зацікавленість у вивченні способів зниження викидів шкідливих речовин в атмосферу, як від пересувних джерел забруднень, так і від стаціонарних. Дослідження цих способів є найбільш природним завданням, рішення якого дозволить забезпечити екологічну безпеку для населення, поліпшити стан довкілля, зменшити захворюваність людей.

1 АНАЛІЗ І ПОСТАНОВКА ТЕХНІЧНОГО ЗАВДАННЯ

1.1 Поняття екологічного моніторингу і його завдання. Класифікація моніторингу

Термін "моніторинг" утворений від латинського "монітор" – "спостерігає", "застережний". Існує декілька сучасних формулювань визначення моніторингу. Деякі дослідники під моніторингом розуміють систему повторних спостережень за станом об'єктів довкілля в просторі і в часі відповідно до заздалегідь підготовленої програми. Конкретніше формулювання визначення моніторингу запропоноване академіком Ю.А. Ізраєлем в 1974 р., відповідно до якої під моніторингом стану природного середовища, і в першу чергу забруднень і ефектів, що викликаються ними в біосфері, мають на увазі комплексну систему спостережень, оцінки і прогнозу змін стану біосфери або її окремих елементів під впливом антропогенних дій.

Програма ЮНЕСКО від 1974 р. визначає моніторинг як систему регулярних тривалих спостережень в просторі і в часі, інформацію, що дає, про минуле і сьогоднішнє станах довкілля, що дозволяє прогнозувати майбутню зміну її параметрів, що мають особливе значення для людства

Екологічний моніторинг є комплексним моніторингом біосфери. Він включає контроль змін стану довкілля під впливом як природних, так і антропогенних чинників.

Основні завдання екологічного моніторингу антропогенних дій:

- спостереження за джерелами антропогенної дії;

- спостереження за чинниками антропогенної дії;

- спостереження за станом природного середовища і процесами, що відбуваються в ній, під впливом чинників антропогенної дії;

- оцінка фізичного стану природного середовища;

- прогноз зміни стану природного середовища під впливом чинників антропогенної дії і оцінка прогнозованого стану природного середовища.

Система моніторингу може охоплювати як локальні райони, так і Земну кулю цілому.

На Першій міжурядовій нараді по моніторингу, скликаній в Найробі (Кенія, лютий 1974г.) радою керівників Програми ООН по проблемах навколишнього середовища було прийнято рішення про створення Глобальної системи моніторингу навколишнього середовища (ГСМНС).

Там були визначені основні положення і цілі програми ГСМНС, в яких було висловлено попередження про зміни стану природного середовища, пов'язані із забрудненням, а також попередження про загрозу здоров'ю людини, загрозу стихійних лих.

Національним моніторингом зазвичай називають систему моніторингу у рамках держави. Така система відрізняється від глобального моніторингу не лише масштабами, але і тим, що основним завданням національного моніторингу є отримання інформації і оцінка стану навколишнього середовища в національних інтересах. Так, підвищення рівня забруднення атмосфери в окремих містах або промислових районах може і не мати істотного значення для оцінки стану біосфери в глобальному масштабі, але представляється важливим для вживання заходів на національному рівні.

В межах системи національного моніторингу існують системи регіонального моніторингу, які, як правило, підрозділяються за адміністративною ознакою і рідше за географічною ознакою.

Моніторинг включає наступні основні практичні напрями:

- спостереження за станом довкілля і чинниками, що впливають на неї;

- оцінку фактичного стану довкілля і рівня її забруднення;

- прогноз стану довкілля в результаті можливих забруднень і оцінку цього стану.

Інформаційна система моніторингу розглянута на рис.1.1.

Об'єктами моніторингу є атмосфера (моніторинг приземного шару атмосфери і верхньої атмосфери); атмосферні опади (моніторинг атмосферних опадів); поверхневі води суші, океани і моря, підземні води (моніторинг гідросфери); кріосфера (моніторинг складових кліматичної системи).

По об'єктах спостереження розрізняють: атмосферний, повітряний, водний, грунтовий, кліматичний моніторинг, моніторинг рослинності, тваринного світу, здоров'я населення і так далі.

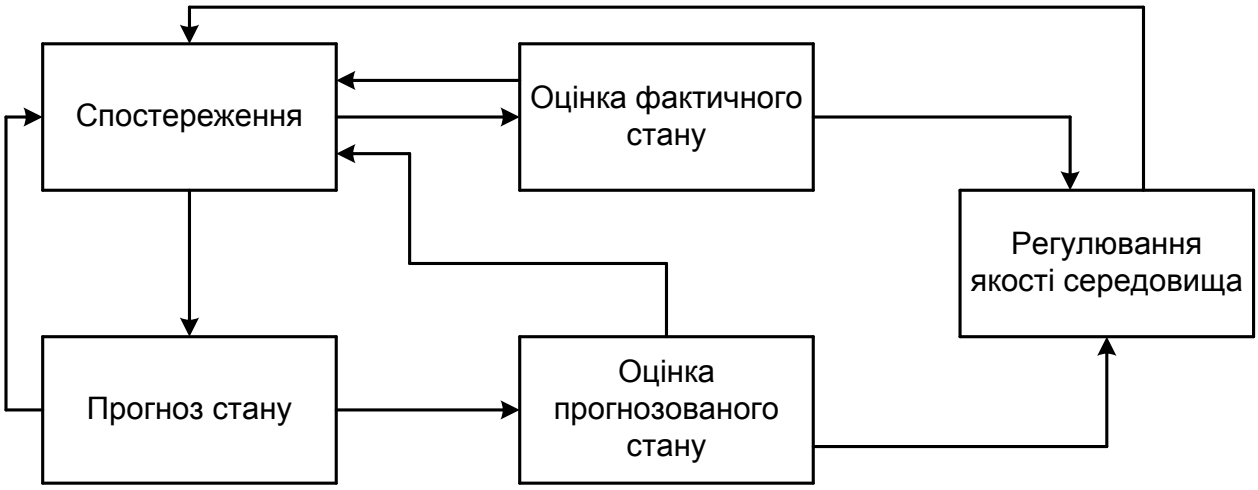


Рисунок 1.1 – Схема моніторингу

Існує класифікація систем моніторингу за чинниками, джерелами і масштабами дії (рисунок 1.2 і таблиця 1.1).

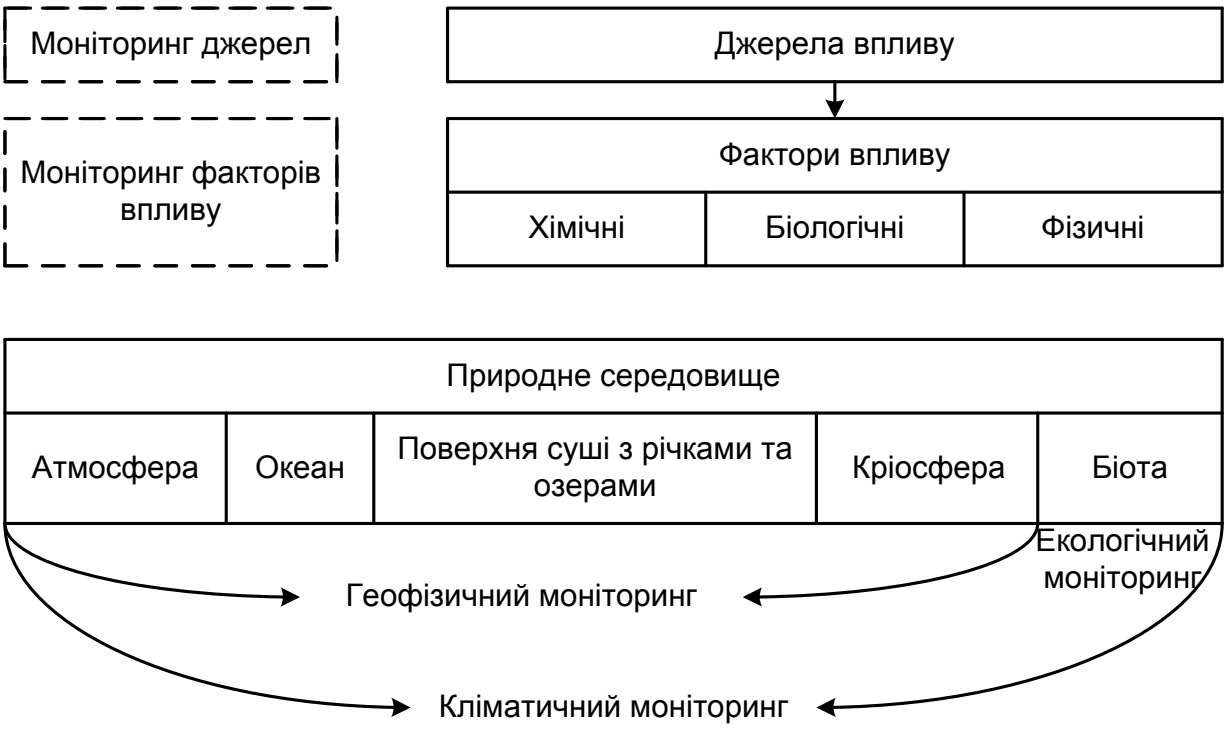


Рисунок 1.2 – Блок-схема системи моніторингу

Моніторинг чинників дії – моніторинг різних хімічних забрудників (моніторинг інгредієнта) і різноманітних природних і фізичних чинників дії (електромагнітне випромінювання, сонячна радіація, шумові вібрації).

Моніторинг джерел забруднень – моніторинг точкових стаціонарних джерел (заводські труби), точкових рухливих (транспорт), просторових (міста, поля з внесеними хімічними речовинами) джерел.

По масштабах дії моніторинг буває просторовим і тимчасовим.

По характеру узагальнення інформації розрізняють наступні системи моніторингу:

– глобальний – стеження за загальносвітовими процесами і явищами в біосфері Землі, включаючи усі її екологічні компоненти, і попередження про виникаючі екстремальні ситуації;

– базовий (фоновий) – стеження за загальнобіосферними, в основному природними, явищами без накладення на них регіональних антропогенних впливів;

– національний – моніторинг в масштабах країни;

– регіональний – стеження за процесами і явищами в межах якогось регіону, де ці процеси і явища можуть відрізнятися і за природним характером, і по антропогенних діях від базового фону, характерного для усієї біосфери;

– локальний – моніторинг дії конкретного антропогенного джерела;

– імпактний – моніторинг регіональних і локальних антропогенних дій в особливо небезпечних зонах і місцях.

Класифікація систем моніторингу може грунтуватися і на методах спостереження (моніторинг за фізико-хімічними і біологічними показниками, дистанційний моніторинг).

Хімічний моніторинг – це система спостережень за хімічним складом (природного і антропогенного походження атмосфери, опадів, поверхневих і підземних вод, вод океанів і морів, грунтів, донних відкладень, рослинності, тварин і контроль за динамікою поширення хімічних забруднюючих речовин. Глобальним завданням хімічного моніторингу є визначення фактичного рівня

забруднень довкілля пріоритетними високотоксичними інгредієнтами, представленими в таблиці 1.2.

Таблиця 1.1 – Класифікація систем (підсистем) моніторингу

Принцип класифікації	Існуючі та системи (підсистеми), що розробляються
Універсальні системи	Глобальний моніторинг (базовий, регіональний, імпактний рівні), включаючи фоновий і палеомоніторінг. Національний моніторинг (наприклад, Загальнодержавна служба спостереження і контролю за рівнем забруднення навколишнього середовища). Міжнаціональний моніторинг (наприклад, моніторинг транскордонного переносу забруднюючих речовин)
Реакція основних складових біосфери	Геофізичний моніторинг. Біологічний моніторинг, включаючи генетичний. Екологічний моніторинг (включаючи вищеназвані)
Різні середовища	Моніторинг антропогенних змін (включаючи забруднення та реакцію на нього) в атмосфері, гідросфері, грунті, кріосфері і біоті
Фактори та джерела впливу	Моніторинг джерел забруднення. Інгредієнтний моніторинг (наприклад, окремих забруднюючих речовин, радіоактивних випромінювань, шумів і т.д.)
Гострота і глобальність проблеми	Моніторинг океану. Моніторинг озоносфери
Методи спостереження	Моніторинг по фізичним, хімічним та біологічним показниками Супутниковий моніторинг (дистанційні методи)
Системний підхід	Медико-біологічний (стану здоров'я) моніторинг. Екологічний моніторинг. Кліматичний моніторинг. Варіант: біоекологічний, геоекологічний, біосферний моніторинг

Таблиця 1.2 – Класифікація пріоритетних забруднюючих речовин та контроль за їх утриманням у різних середовищах

Клас пріоритетності	Забруднюючі речовини	Середовище	Тип програми вимірювань
I	Діоксид сірки і зважені частинки	Повітря	I, Р, Б, Г
	Радіонукліди (Sr-90, Cs-197)	Їжа	I, Р
II	Озон	Повітря	I, Б
	Хлорорганічні сполуки	Біота, людина	I, Р
	Кадмій та його сполуки	Їжа, людина, вода	I
III	Нітрати, нітрити	Питна вода, їжа	I
	Оксиди азоту	Повітря	I
IV	Ртуть та її сполуки	Їжа, повітря	I, Р
	Свинець	Повітря, їжа	I
	Діоксид вуглецю	Повітря	Б
V	Оксид вуглецю	Повітря	I
	Нафтоуглеводороди	Морська вода	Р, Б
VI	Фтористі сполуки	Питна вода	I
VII	Азбест	Повітря	I
	Миш'як	Питна вода	I
VIII	Мікротоксини	Їжа	I, Р
	Мікробіологічне зараження	Їжа	I, Р
	Реактивні вуглеводні	Повітря	I

Примітка: I – імпактний, Р – регіональний, Б – базовий, Г – глобальний.

Фізичний моніторинг – система спостережень за впливом фізичних процесів і явищ на довкілля (повені, вулканізм, землетруси, цунамі, посухи, ерозія грунтів і так далі).

Біологічний моніторинг – моніторинг, здійснюваний за допомогою біоіндикаторів (тобто таких організмів, по наявності, стану і поведінці яких судять про зміни в середовищі).

Екобіохімічний моніторинг – моніторинг, що базується на оцінці двох складових довкілля (хімічною і біологічною).

Дистанційний моніторинг – в основному, авіаційний, космічний моніторинг із застосуванням літальних апаратів, оснащених радіометричною апаратурою, здатною здійснювати активне зондування об'єктів, що вивчаються, і реєстрацію досвідчених даних.

Найбільш універсальним є комплексний екологічний моніторинг довкілля.

Комплексний екологічний моніторинг довкілля – це організація системи спостережень за станом об'єктів природного довкілля для оцінки їх фактичного рівня забруднення і попередження про критичні ситуації, що створюються, шкідливі для здоров'я людей і інших живих організмів. Розрізняють моніторинг локальний, регіональний і фоновий.

При проведенні комплексного екологічного моніторингу довкілля :

– проводиться постійна оцінка екологічних умов місця існування людини і біологічних об'єктів (рослин, тварин, мікроорганізмів і так далі), а також оцінка стану і функціональної цілісності екосистем;

– створюються умови для визначення дій, що коригують, в тих випадках, коли цільові показники екологічних умов не досягаються.

Система комплексного екологічного моніторингу передбачає:

– виділення об'єкту спостереження;

– обстеження виділеного об'єкту спостереження;

– складання для об'єкту спостереження інформаційної моделі;

– планування вимірів;

– оцінку стану об'єкту спостереження і ідентифікацію його інформаційної моделі;

– прогнозування зміни стану об'єкту спостереження;

– представлення інформації в зручній для використання формі і доведення її до споживача.

Основні цілі комплексного екологічного моніторингу полягають в тому, щоб на підставі отриманої інформації:

– оцінити показники стану і функціональної цілісності екосистем і місця існування людини (тобто провести оцінку дотримання екологічних нормативів);

– виявити причини зміни цих показників і оцінити наслідки таких змін, а також визначити заходи, що коригують, в тих випадках, коли цільові показники екологічних умов не досягаються (тобто провести діагностику стану екосистем і місця існування);

– створити передумови для визначення заходів по виправленню виникаючих негативних ситуацій до того, як буде завданий збиток, тобто забезпечити завчасне попередження негативних ситуацій.

1.2 Розгляд існуючих варіантів екологічного моніторингу

1.2.1 Державний екологічний моніторинг в Москві

Визначення екологічного моніторингу або моніторингу навколишнього середовища законодавчо визначено у Федеральному законі від 10 січня 2002 року № 7-ФЗ «Про охорону навколишнього середовища» і Федеральному законі від 19 июля1998 року N 113-ФЗ «Про гідрометеорологічну службу».

Повноваження по здійсненню екологічного моніторингу покладено як на органи державної влади Російської Федерації, так і на суб'єкти Федерації (Федеральним законом від 10 січня 2002 року № 7-ФЗ «Про охорону навколишнього середовища»).

З боку федеральних органів виконавчої влади роботи з екологічного моніторингу здійснює територіальний орган Росгідромету – Державна установа «Московський Центр з гідрометеорології та моніторингу навколишнього середовища з Регіональними Функціями» (ДУ «МосЦГНС»).

Відповідно до наданих повноважень в Москві створена регіональна система екологічного моніторингу, яка включає в себе спостереження за станом атмосферного повітря, поверхневих водних об'єктів, грунтів, зелених

насаджень, рівнів шуму, небезпечних геоекологічних процесів. Спеціалізованої уповноваженою організацією по здійсненню цих робіт є «Мосекомоніторінг».

З 2004 року роботи з екологічного моніторингу в Москві регулюються спеціальним Законом міста Москви від 20 листопада 2004 року № 65 «Про екологічний моніторинг у місті Москві», що дозволило встановити цілі екологічного моніторингу, порядок здійснення робіт та основні напрямки використання даних.

В таблиці 1.3 коротко представлені характеристики основних підсистем Єдиної системи екологічного моніторингу міста Москви із зазначенням кількості постійних пунктів спостереження, періодичності вимірювань і кількості вимірюваних показників.

На територіях, «не охоплених» постійними пунктами спостереження проводяться спеціалізовані екологічні обстеження:

– обстеження атмосферного повітря за зверненнями жителів та заявками міських організацій з використанням пересувної екологічної лабораторії, оснащеної автоматичним газоаналітичних обладнанням;

– дослідження забруднення поверхневих водних об'єктів автоматичним аналітичним комплексом, встановленим на спеціалізованому теплоході «Екопатруль» (в навігаційний період);

– обстеження рівнів шуму від автотранспорту, залізничного транспорту, промислових підприємств та інших об'єктів господарської діяльності, будівельних робіт (більш ніж 150 територій міста щорічно).

З 2007 року за рішенням Уряду Москви здійснюються автоматичні вимірювання викидів забруднюючих речовин в атмосферне повітря найбільш великими промисловими підприємствами міста.

Єдина система екологічного моніторингу міста Москви забезпечена сучасним аналітичним обладнанням кращих вітчизняних і зарубіжних виробників. У роботі системи широко використовуються автоматизовані методи та інформаційні технології збору, передачі та зберігання даних (рисунок 1.3). Всі результати вимірювань передаються до Єдиного міського фонду даних екологічного моніторингу, що є складовою частиною Комплексної

автоматизованої системи в галузі охорони навколишнього середовища та природокористування міста Москви.

Таблиця 1.3 – Характеристики основних підсистем Єдиної системи екологічного моніторингу міста Москви

Підсистема Єдиної системи екологічного моніторингу міста Москви	Кількість постійних пунктів спостереження	Режим і періодичність надходження інформації	Вимірювані показники
1	2	3	4
Моніторинг атмосферного повітря	43 автоматичні станції контролю забруднення атмосфери (АСКЗА) (З них 3 висотні станції, розташовані на Останкінській телевежі і 2 станції за межею міста Москви)	Цілодобово в режимі реального часу 1 раз на 20 хвилин	На житлових територіях: оксид вуглецю, оксиди азоту, аміак, діоксид сірки, сірководень, озон, вуглеводні, метан, зважені частки з розміром менше 10 мкм. Поблизу автотрас додатково вимірюються: бензол, толуал, формальдегід, метаксілол, параксілол, етилбензол, фенол, стирол, нафталін, метеопараметри
Моніторинг поверхневих водних об'єктів	27 затверджених контрольних створу (З них 14 на річці Москві, 13 у гирлах малих річок-приток ріки Москви) (3 автоматичні станції контролю забруднення вод (у стадії створення))	Щомісяця	Прозорість, завислі речовини, сухий залишок, розчинений кисень, хлориди, сульфати, фосфати, іон амонію, нітрит-іон, нітрат-іон, залізо загальне, марганець, мідь, цинк, свинець, хром, алюміній, нікель, кадмій, кобальт , сульфіди, нафтопродукти, фенол, формальдегід

Продовження таблиці 1.3

1	2	3	4
Моніторинг грунтів	253 постійних майданчика спостереження	1 раз на рік	Вміст гумусу, % свинцю, цинку, міді, нікель, кадмій, марганець, ртуть і миш'яку (валові і рухомі форми), нафтопродуктів, величина pH рідкої фази грунту, макроелементів живлення (N, P, K), склад обмінних катіонів, електропровідність грунтового розчину
Моніторинг зелених насаджень	494 постійні майданчика спостереження	1 раз на рік	Дендрологічні, ентомофітопатологічні, геохімічні обстеження. Оцінка стану і приживлюваності молодих посадок
Моніторинг підтоплення та якості підземних вод	170 свердловин 50 джерел	2 рази на рік	Рівень грунтових вод, температура, вміст хімічних речовин
Моніторинг зсувних процесів	14 постійних ділянок спостереження	1 раз на рік, оперативна інформація про негативні процеси	
Моніторинг рівнів шуму	1 автоматична станція контролю авіашума (в стадії створення)		

Рисунок 1.3 – Автоматизовані засоби екологічного моніторингу міста Москви

За рішенням Уряду Москви «Мосекомоніторінг» є оператором Єдиного міського фонду даних екологічного моніторингу, який включає в себе бази даних по забрудненню атмосферного повітря, водних об'єктів, грунтів, станом зелених насаджень, рівнів шуму, здійснює ведення кадастру відходів міста Москви, реєстру зелених насаджень. В таблиці 1.4 представлені основні інформаційні ресурси, ведення яких здійснює «Мосекомоніторінг».

Таблиця 1.4 – Інформаційні ресурси міста Москви в області природокористування та охорони навколишнього середовища

Найменування інформаційного ресурсу	Склад інформації
1	2
База даних про забруднення атмосферного повітря за результатами роботи автоматичних станцій контролю забруднення атмосфери (АСКЗА)	20-ти хвилинні значення концентрацій 23-х пріоритетних забруднюючих речовин на 43-х АСКЗА
База даних метеорологічних спостережень	Метеопараметри на приземному рівні і в повітряному потоці, що натікає на місто. Висотні профілі температури повітря (за даними температурних профілемеров і содаров)
База даних про забруднення водних об'єктів	Зміст 27-ми забруднюючих речовин у 14-ти контрольних створах на річці Москві і 13 контрольних створах в гирлах малих річок.
База даних про забруднення атмосферного повітря за результатами рейдів пересувної екологічної лабораторії	Концентрації 10-15 забруднюючих речовин в атмосферному повітрі на більш ніж 300 територій міста Москви (щороку обстежується більше 100 територій при різних метеоумовах).
База даних про рівні шуму на територіях міста Москви (за результатами відпрацювання скарг, заявок префектур і моніторингу рівня шуму від будівельних майданчиків)	Рівні шуму на 257 територіях, прилеглих до промислових підприємств, автотрас, залізничним лініях і т.д. і на 150 територіях, прилеглих до будівельних майданчиків
База даних про забруднення грунтів міста Москви	Здійснюється на 912 постійних майданчиках спостереження. Моніторинг грунтів проводиться по 18 показникам забруднення
База даних про стан зелених насаджень міста Москви	Результати дендрологічних, геохімічних, фітопатологічних досліджень зелених насаджень на 425 майданчиках постійного

	спостереження

Продовження таблиці 1.4

1	2
Реєстр зелених насаджень	Кількість і видовий склад дерев і чагарників, площі газонів, квітників, дорожньо-тропіночної мережі на озеленених територіях першої та другої категорії
Кадастр відходів	Обсяги утворення відходів розрізі окремих видів на 4000 підприємств міста Москви
Екологічна карта міста Москви	111 тематичних картографічних шарів по природному комплексу Москви, забруднення атмосферного повітря, водних об'єктів і т.д.
База даних за результатами прямих інструментальних вимірювань викидів промислових підприємств	20-ти хвилинні значення вмісту забруднюючих речовин у викидах 41 об'єкта теплоенергетики та 1 промислового підприємства. Виміри здійснюються цілодобово.

З 2004 року в місті Москві здійснюється ведення Екологічної карти міста Москви. Екологічна карта міста Москви представляє собою інформаційний ресурс, що включає тематичні електронні карти з різних напрямком охорони навколишнього середовища. В даний час у складі Екологічної карти більше 110 тематичних електронних карт. Основні напрямки тематичних карт представлені у таблиці 1.5.

Таблиця 1.5 – Застосування геоінформаційних технологій для вирішення завдань охорони навколишнього середовища міста Москви

Тематичні шари	Завдання, які вирішуються
1	2
Забруднення атмосферного повітря	Інформування населення про забруднення атмосферного повітря. Виявлення територій міста, схильних до наднормативних забруднень повітря. Аналіз поточної ситуації по забрудненню атмосферного повітря при проведенні експертизи містобудівних проектів

Продовження таблиці 1.5

1	2
Джерела негативного впливу на навколишнє середовище (промислові підприємства, ТЕЦ, РТС, КТС, ринки)	Планування рейдів пересувної екологічної лабораторії за скаргами мешканців та заявками префектур. Виявлення переліку підприємств для позапланових перевірок державного екологічного контролю за скаргами мешканців та заявками префектур.
Несанкціоновані звалища	Експертиза проектів територіального планування та архітектурно-будівельного проектування. Планування обсягів робіт з рекультивації несанкціонованих звалищ при територіальному плануванні та архітектурно-будівельному проектуванні.
Колекторно-річкова мережа	Планування перевірок державного екологічного контролю підприємств-спецводокористувачей. Інформування населення про забруднення водних об'єктів

«Мосекомоніторінг» створено в 2002 році за рішенням Уряду Москви і є структурним підрозділом Департаменту природокористування і охорони навколишнього середовища міста Москви.

«Мосекомоніторінг» є спеціально уповноваженою організацією міста Москви щодо здійснення державного екологічного моніторингу і надає широкий спектр природоохоронних послуг для юридичних і фізичних осіб.

Основними напрямками діяльності «Мосекомоніторінг» є:

– організація регулярних спостережень, за станом атмосферного повітря, поверхневих водних об'єктів, грунтів, зелених насаджень, рівнів шуму, небезпечних геоекологічних процесів;

– виконання функцій замовника по експлуатації автоматичних станцій контролю забруднення атмосфери, пересувний екологічної лабораторії, теплохода «Екопатруль», аналітичної лабораторії Департаменту природокористування і охорони навколишнього середовища;

– організація екологічних обстежень територій міста за скаргами населення, зверненнями міських організацій і т.д.;

– організація прямих інструментальних спостережень за викидами забруднюючих речовин в атмосферне повітря та скидами забруднюючих речовин у поверхневі водні об'єкти міськими підприємствами – суб'єктами локального екологічного моніторингу (посилання на глосарій), затвердженими Урядом Москви;

– обробка та аналіз даних про стан природних середовищ в місті Москві;

– надання екологічної інформації різним категоріям користувачів і населенню;

– ведення Єдиного міського фонду даних екологічного моніторингу;

– ведення Екологічної карти міста Москви.

За дорученням Уряду Москви «Мосекомоніторінг» також здійснює:

– ведення Реєстру зелених насаджень міста Москви;

– ведення Кадастру відходів міста Москви;

– виконання функцій замовника по заходах ДЦП «Електронна Москва» в частині інформатизації в галузі природокористування та охорони навколишнього середовища міста Москви.

За 6 років з моменту утворення «Мосекомоніторінг» досягнуто суттєвих результатів у галузі розвитку державного екологічного моніторингу міста Москви: кількість автоматичних станцій контролю забруднення атмосфери зросла з 6-ти до 43-х; організований єдиний в РФ багаторівневий пункт контролю забруднення атмосфери (на Останкінській вежі), створено систему моніторингу забруднення грунтів, регіональна мережа моніторингу за небезпечними геоекологічними процесами, на 42-х підприємствах організовані прямі інструментальні вимірювання викидів забруднюючих речовин в атмосферне повітря з передачею даних в режимі реального часу в єдиний інформаційно-аналітичний центр, суттєво розширені канали надання екологічної інформації населенню.

Автоматизована система моніторингу атмосферного повітря Москви визнана фахівцями європейських та міжнародних організацій повністю відповідної вимогою Європейського союзу, що дозволило «Мосекомоніторінг» приймати широку участь у міжнародних проектах в порівнянні забруднення атмосферного повітря у великих містах і оцінки впливу забруднення атмосфери на здоров'я населення.

«Мосекомоніторінг» – динамічно розвинута природоохоронна установа, оснащена сучасним аналітичним обладнанням та передовими інформаційними технологіями збору, обробки, зберігання та надання даних користувачам.

В даний час «Мосекомоніторінг» надає наступні послуги організаціям:

- розробка програм екологічного моніторингу (схеми розміщення постів спостереження, перелік контрольованих показників, підбір аналітичних методів та рекомендації щодо приладового оснащення);

- проведення екологічних обстежень міських територій (атмосферне повітря, водні об'єкти, грунти, рівні шуму);

- надання екологічної інформації проектним організаціям;

- організація та ведення моніторингу природних середовищ в період будівництва;

- надання за індивідуальним форматом екологічної і метеорологічної інформації ЗМІ, інформаційним агентствам і т.д. [1].

1.2.2 Система екологічного моніторингу міста Донецьк

Важливим напрямком діяльності Донецької міської Ради є отримання достовірної екологічної інформації про стан навколишнього середовища та надання цієї інформації населенню. Без аналізу стану забруднення природних середовищ неможливо прийняття ефективних рішень на міському рівні в галузі екологічної безпеки, крім того, отримання такої інформації є конституційним правом громадян.

З метою збору даних про поточну екологічну обстановку в місті Донецьку міською Радою було прийнято рішення про створення муніципальної системи екологічного моніторингу навколишнього природного середовища.

У цьому плані створення системи моніторингу забруднення атмосферного повітря є першочерговим завданням. Статистичні дані за 2004 – 2005 рр. вказують на те, що в окремі дні або тижні в місті можуть виникати ситуації з доволі значним підвищенням концентрацій шкідливих речовин в атмосферному повітрі.

На першому етапі, основне завдання системи моніторингу атмосфери полягає в контролі небезпечних рівнів забруднення повітря та виявленні причин виникнення таких ситуацій. В умовах, що система моніторингу є важливою складовою контролю якості навколишнього природного середовища в місті.

В даний час моніторинг атмосферного повітря являє собою складний процес, який включає в себе безперервний контроль вмісту найбільш значущих забруднюючих речовин на території міста, оперативну обробку даних і регулярне інформування органів виконавчої влади, контролюючих організацій, громадськості і населення.

У 2004 – 2005 роках за завданням Донецької міської Ради ДонНТУ і СКТБ «Турбулентність» був розроблений і випробуваний експериментальний зразок аппаратно-програмного комплексу екологічного моніторингу (АКІАМ). Комплекс АКІАМ призначений для автоматизації контролю забруднення атмосфери, а також подання, обробки, передачі, зберігання та аналізу інформації про забруднення приземного шару атмосферного повітря.

Комплекс АКІАМ дозволяє:

– контролювати показники забруднення атмосферного повітря за пріоритетними шкідливими речовинами на стаціонарних постах в автоматичному режимі;

– здійснювати контроль метеорологічних параметрів повітря на стаціонарних постах;

– вести бази даних показників забруднення атмосферного повітря та метеопараметрів на міському рівні;

– підвищити ефективність екологічного контролю за рахунок автоматизації процесу реєстрації інформації, її передачі, обробки та аналізу;

– контролювати забруднення атмосфери в санітарнозахисних зонах і параметри технологічних викидів підприємств в реальному часі;

– отримувати інформацію для ідентифікації джерел викидів забруднюючих речовин на території міста, що дозволить виявляти порушників природоохоронного законодавства до яких будуть використані штрафні санкції.

До складу апаратно-програмного комплексу АКІАМ входять інформаційний сервер збору та обробки даних, автоматичні пости контролю забруднення атмосфери й автоматизовані робочі місця суб'єктів моніторингу. Зв'язок між інформаційним сервером, автоматичними постами контролю та суб'єктами моніторингу проводиться по радіоканалу або через мережу Інтернет.

Кожен автоматичний пост контролю включає в себе блок газоаналізаторів для контролю концентрацій шкідливих речовин в атмосферному повітрі, метеостацію, контролер для первинної обробки даних та систему передачі інформації на диспетчерський пункт.

На основі комплексу АКІАМ в даний час створюється міська система екологічного моніторингу, перша черга якої була введена в експлуатацію до кінці 2006 року. Загальна схема міської системи екологічного моніторингу наведена на рис.1.4. З початку 2007 року в місті працюють два автоматизовані пости контролю забруднення атмосферного повітря, а до кінця 2007 року їх кількість буде збільшена до трьох-чотирьох.

Важливим результатом функціонування міської системи моніторингу атмосфери є створений ДонНТУ при розробці комплексу АКІАМ банк даних, в якому зберігаються метеопараметри і значення концентрацій шкідливих речовин на території міста з 2000 року. Накопичена кількість даних перевищує 1 мільйон спостережень. Зібрані дані дозволяють аналізувати тенденції змін

екологічних параметрів якості атмосферного повітря, здійснювати прогнози та обгрунтовувати природоохоронні заходи.

Комплекс АКІАМ стане основою для розвитку міської системи екологічного моніторингу навколишнього природного середовища. В даний час створено технічне завдання та проведені роботи з розробки проекту міської системи екологічного моніторингу.

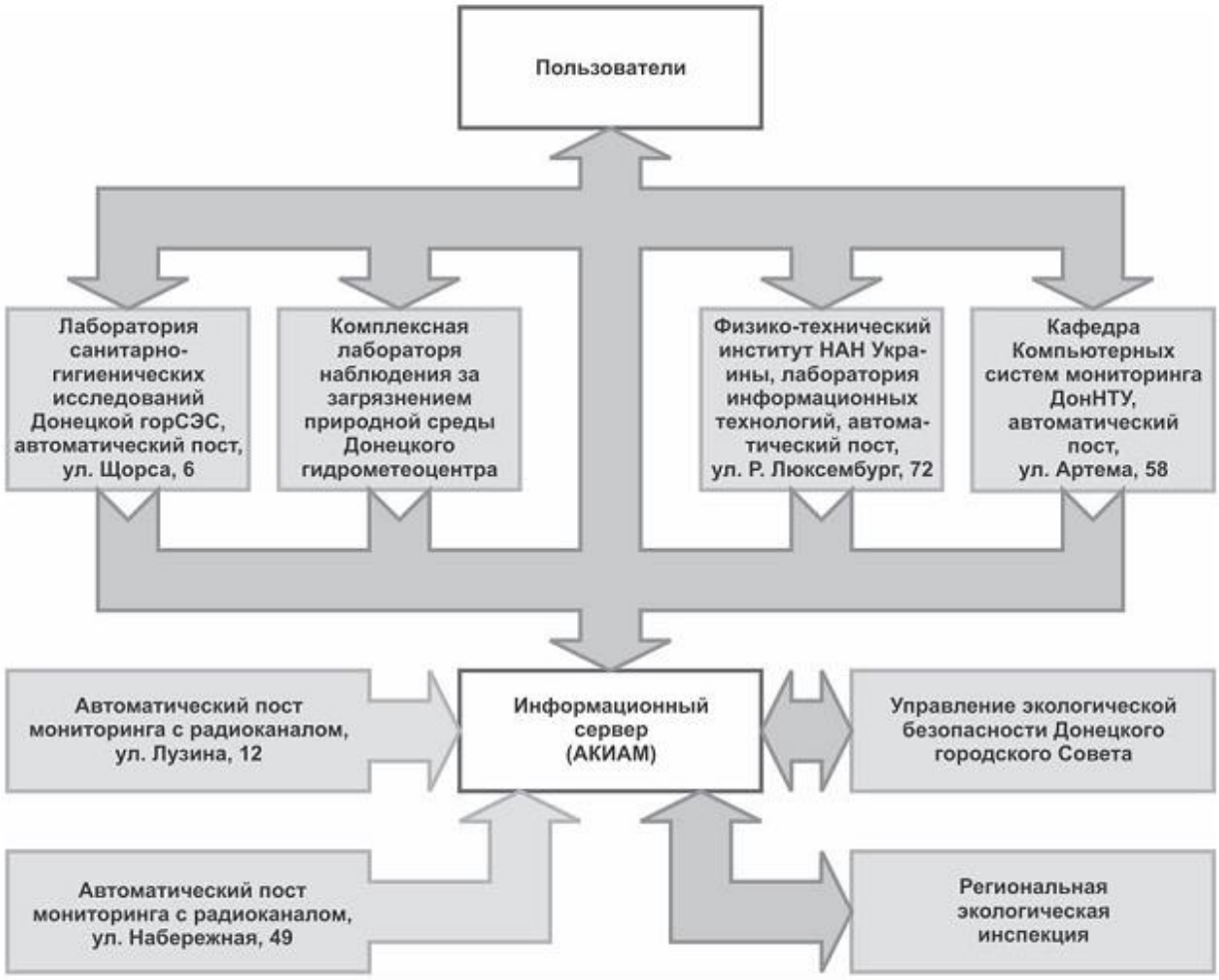


Рисунок 1.4 – Схема міської системи екологічного моніторингу

Після введення в 2007 році системи моніторингу в опитнопромислову експлуатацію передбачається, що жителі Донецька можуть отримати щоденну інформацію про рівень забруднення атмосфери міста в Інтернеті на громадському інформаційному сайті (http://www.doneco.org.ua). Для інформування населення інформація про стан навколишнього середовища буде виводитися на електронне рекламне табло в центральній частині міста,

зведення про забруднення повітряного басейну будуть озвучуватися на радіостанціях і муніципальному телевізійному каналі [2].

1.3 Технічні вимоги до розробки

Необхідно розробити розподілену базу даних регіональної системи екологічного моніторингу.

Призначення: збір оперативної інформації про стан атмосферного повітря в місті і відображення її в зручному вигляді; підтримка статистичної, математичної, спеціалізованої обробки.

Мета розробки даної системи:

– створення програмного забезпечення, що здійснює збір інформації, дозволяє отримати об'єктивну оцінку про стан повітря міста у будь-який час;

– аналіз моніторингової інформації і генерація звітів.

Для створення програмного забезпечення необхідно використовувати Web-сервер. Apache – найпоширеніший у світі Web-сервер. Будучи безкоштовною відкритою програмою, Apache по функціональним можливостям і надійності не поступається комерційним серверам, а широкі можливості конфігурації дозволяють налаштувати його для роботи практично з будь-якою конкретною системою.

MySQL це реляційна база даних, робота з даними в якій здійснюється за допомогою SQL запитів. Основними перевагами цього типу БД є швидкість і простота у використанні. Саме з цих причин MySQL обрана для створення БД екологічного моніторингу.

Visual Studio 2017 – інтегроване середовище, що спрощує створення, налагодження та розгортання додатків. Visual Studio включає декілька компонентів: Visual Basic .NET, Visual C++, Visual C#. Для написання програмного забезпечення використовується мова програмування Visual C#.

Встановлення зв'язку між БД MySQL та програмним забезпеченням Visual C# виконується за допомогою mysql-connector-net-5.2.7. Даний конектор дозволяє у Visual C# виконувати звичайні SQL запити.

Апаратне забезпечення необхідне для установки операційної системи Windows XP(Vista, 7, 10), а також Web-сервера Apache. Достатньо наступних параметрів:

- процесор класу Pentium DualCore з частотою процесора 2 ГГц;
- жорсткий диск об’ємом 100 Гб;
- оперативна пам'ять 4 Гб.

2 ВИБІР МОВИ ПРОГРАМУВАННЯ

2.1 Загальні положення

Мова програмування – формальна знакова система, призначена для запису комп'ютерних програм. Мову програмування визначає набір лексичних, синтаксичних і семантичних правил, задаючих зовнішній вигляд програми і дії, які виконає виконавець (комп'ютер) під її управлінням.

Створити мову, зручну для написання програм, недостатньо. Для кожної мови потрібний свій перекладач. Такими перекладачами є спеціальні програми-транслятори.

Транслятор – це програма, призначена для переведення програми, написаної на одній мові програмування, в програму на іншій мові програмування. Процес переведення називається трансляцією. Тексти початкової і результуючої програм знаходяться в пам'яті комп'ютера. Прикладом транслятора є компілятор.

Компілятор – це програма, призначена для переведення програми, написаної на якій-небудь мові, в програму в машинних кодах. Процес такого переведення називається компіляцією.

Компілятор створює закінчений результат – програму в машинних кодах. Потім ця програма виконується. Варіант початкової програми, що відкомпілювався, можна зберегти на диску. Для повторного виконання початкової програми компілятор вже не потрібний. Досить завантажити з диска в пам'ять комп'ютера варіант, що відкомпілювався в попередній раз, і виконати його.

Існує інший спосіб поєднання процесів трансляції і виконання програми. Він називається інтерпретацією. Суть процесу інтерпретації полягає в наступному. Спочатку переводиться в машинні коди, а потім виконується перший рядок програми. Коли виконання першого рядка закінчене, починається переведення другого рядка, який потім виконується і так далі. Управляє цим процесом програма-інтерпретатор.

Інтерпретатор – це програма, призначена для відрядкової трансляції і виконання початкової програми. Такий процес називається інтерпретацією.

У процес трансляції входить перевірка початкової програми на відповідність правилам використовуваної в ній мови. Якщо в програмі виявлені помилки, транслятор вводить повідомлення про них на пристрій виводу (зазвичай, на екран дисплея) [3].

Інтерпретатор повідомляє про знайдені їм помилки після трансляції кожного рядка програми. Це значно полегшує процес пошуку і виправлення помилок в програмі, проте істотно збільшує час трансляції. Компілятор транслює програму набагато швидше, ніж інтерпретатор, але повідомляє про знайдені їм помилки після завершення компіляції усієї програми. Знайти і виправити помилки в цьому випадку важче. Тому інтерпретатори розраховані, в основному, на мови, призначені для навчання програмуванню, і використовуються початкуючими програмістами. Більшість сучасних мов призначені для розробки складних пакетів програм і розраховані на компіляцію.

Об'єктно-орієнтоване програмування – це метод програмування, спосіб написання "гарних" програм для безлічі завдань. Якщо цей термін має якийсь сенс, то він повинен мати на увазі: така мова програмування, яка надає хороші можливості для об'єктно-орієнтованого стилю програмування. Тут слід вказати на важливі відмінності. Говорять, що мова підтримує деякий стиль програмування, якщо в нім є такі можливості, які роблять програмування в цьому стилі зручним (досить простим, надійним і ефективним). Мова не підтримує деякий стиль програмування, якщо вимагаються великі зусилля або навіть мистецтво, щоб написати програму в цьому стилі. Проте це не означає, що мова забороняє писати програми в цьому стилі. Дійсно, можна писати структурні програми на Фортрані і об'єктно-орієнтовані програми, але це буде марною тратою сил, оскільки ці мови не підтримують вказаних стилів програмування.

Підтримка мовою певної парадигми (стилю) програмування явно проявляється в конкретних мовних конструкціях, розрахованих на ній. Але

вона може проявлятися в тоншій, прихованій формі, коли відхилення від парадигми діагностується на стадії трансляції або виконання програми. Найочевидніший приклад – це контроль типів. Крім того, мовна підтримка парадигми може доповнюватися перевіркою на однозначність і динамічним контролем. Підтримка може надаватися і окрім самої мови, наприклад, стандартними бібліотеками або середовищем програмування.

Не можна сказати, що одна мова краще за іншу тільки тому, що в неї є можливості, які в іншій відсутні. Часто буває якраз навпаки. Тут важливіше не те, які можливості має мова, а те, наскільки наявні в ній можливості підтримують обраний стиль програмування для певного круга завдань. Тому можна сформулювати наступні вимоги до мови:

– усі конструкції мови повинні природно і елегантно визначатися в ній;

– для вирішення певного завдання має бути можливість використовувати поєднання конструкцій, щоб уникнути необхідності вводити для цієї мети нову конструкцію;

– має бути мінімальне число неочевидних конструкцій спеціального призначення;

– конструкція повинна допускати таку реалізацію, щоб в програмі, що не використовує її, не виникло додаткових витрат;

– користувачеві досить знати тільки ту безліч конструкцій, яка безпосередньо використовується в його програмі [4].

2.2 Опис мов програмування

2.2.1 Язик програмування C++

C++ - компільована статично мова програмування загального призначення, що типізується. Підтримує різні парадигми програмування, але, порівняно з його попередником — мовою Сі, — найбільша увага приділена підтримці об'єктно-орієнтованого і узагальненого програмування. Назва «C++» походить від Сі (C), в якому унарний оператор ++ позначає інкремент змінної. У 1990-х роках мова стала однією з найширше вживаних мов програмування загального призначення. При створенні C++ прагнули зберегти сумісність з мовою Сі. Більшість програм на Сі справно працюватимуть і з компілятором C++. C++ має синтаксис, заснований на синтаксисі Сі.

Стандарт C++ на 1998 рік складається з двох основних частин: ядра мови і стандартної бібліотеки. Стандартна бібліотека C++ увібрала в себе бібліотеку шаблонів STL, що розроблялась одночасно із стандартом. Зараз назва STL офіційно не вживається, проте в кругах програмістів на C++ ця назва використовується для позначення частини стандартної бібліотеки, що містить визначення шаблонів контейнерів, ітераторів, алгоритмів.

Стандарт C++ містить нормативне посилання на стандарт Сі від 1990 року і не визначає самостійно ті функції стандартної бібліотеки, які запозичуються із стандартної бібліотеки Сі. Крім того, існує величезна кількість бібліотек C++, що не входять в стандарт. У програмах на C++ можна використовувати багато бібліотек Сі. Стандартизація визначила мову програмування C++, проте за цією назвою можуть ховатися також неповні, обмежені варіанти мови.

Спочатку мова розвивалася поза формальними рамками, спонтанно, у міру завдань, що ставилися перед ним. Розвитку мови супроводив розвиток кросу-компілятора cfront. Новини в мові відбивалися в зміні номера версії кросу-компілятора. Ці номери версій кросу-компілятора поширювалися і на саму мову, але стосовно теперішнього часу мову про версії мови C++ не ведуть.

Нововведеннями C++ порівняно з Сі є:

– підтримка об'єктно-орієнтованого програмування;
– підтримка узагальненого програмування через шаблони;
– додаткові типи даних;
– виключення;
– простори імен;
– вбудовані функції;
– перевантаження операторів;
– перевантаження імен функцій;
– посилання і оператори управління вільно розподілюваною пам'яттю;
– доповнення до стандартної бібліотеки.

Мова C++ багато в чому є надмножиною Сі. Нові можливості C++ включають оголошення у вигляді виразів, перетворення типів у вигляді функцій, оператори new і delete, тип bool, заслання, розширене поняття константної, підставлювані функції, аргументи за умовчанням, перевизначення, простори імен, класи (включаючи і всі пов'язані з класами можливості, такі як спадкоємство, функції-члени, віртуальні функції, абстрактні класи і конструктори), перевизначення операторів, шаблони, оператор ::, обробку виключень, динамічну ідентифікацію і багато що інше. Мова C++ також у багатьох випадках строго відноситься до перевірки типів, чим Сі. У C++ з'явилися коментарі у вигляді подвійної косої межі (//), які були в попереднику Сі — мові BCPL.

Переваги C++: надзвичайно потужна мова, що містить засоби створення ефективних програм практично будь-якого призначення, від низькорівневих утиліт і драйверів до складних програмних комплексів самого різного призначення. Зокрема:

Підтримуються різні стилі і технології програмування, включаючи традиційне директивне програмування, ООП, узагальнене програмування, метапрограмування (шаблони, макроси).

Передбачене виконання програм є важливою перевагою для побудови систем реального часу. Код, що неявно генерується компілятором для реалізації

мовних можливостей (наприклад, при перетворенні змінної до іншого типа), визначений в стандарті. Також строго визначені місця програми, в яких цей код виконується. Це дає можливість заміряти або розраховувати час реакції програми на зовнішню подію.

Автоматичний виклик деструкцій об'єктів при їх знищенні, причому в порядку, зворотному виклику конструкторів. Це спрощує (досить оголосити змінну) і робить надійнішим звільнення ресурсів (пам'ять, файли, семафори і т. п.), а також дозволяє гарантовано виконувати переходи станів програми, не обов'язково пов'язані із звільненням ресурсів (наприклад, запис в журнал).

Призначені для користувача функції-оператори дозволяють коротко і ємко записувати вирази над призначеними для користувача типами в природній формі алгебри.

Мова підтримує поняття фізичної (const) і логічної (mutable) константної. Це робить програму надійніше, оскільки дозволяє компілятору, наприклад, діагностувати помилкові спроби зміни значення змінній.

Використовуючи шаблони, можливо створювати узагальнені контейнери і алгоритми для різних типів даних, а також спеціалізувати і обчислювати на етапі компіляції.

Можливість імітації розширення мови для підтримки парадигм, які не підтримуються компіляторами безпосередньо. Наприклад, бібліотека Boost.Bind дозволяє зв'язувати аргументи функцій.

Можливість створення вбудованих наочно-орієнтованих мов програмування. Такий підхід використовує, наприклад бібліотека Boost.Spirit, що дозволяє задавати ebnf-граматику парсерів прямо в коді C++.

Використовуючи шаблони і множинне наслідування, можна імітувати класи-домішки і комбінаторну параметризацію бібліотек. Такий підхід застосований в бібліотеці Loki, клас Smartprt якої дозволяє, управляючи всього декількома параметрами часу компіляції, згенерувати близько 300 видів «розумних покажників» для управління ресурсами.

Кроссплатформенность: стандарт мови накладає мінімальні вимоги на ЕОМ для запуску скомпільованих програм. Для визначення реальних

властивостей системи виконання в стандартній бібліотеці присутні відповідні можливості. Доступні компілятори для великої кількості платформ, на мові C++ розробляють програми для самих різних платформ і систем.

Ефективність. Мова спроектована так, щоб дати програмістові максимальний контроль над всіма аспектами структури і порядку виконання програми. Жодна з мовних можливостей, що приводить до додаткових накладних витрат, не є обов'язковою для використання — при необхідності мова дозволяє забезпечити максимальну ефективність програми.

Недоліки:

Складність і надмірність, із-за яких C++ важко вивчати, а побудова компілятора зв'язана з великою кількістю проблем. Підтримка множинного спадкоємства викликає цілий ряд логічних проблем, а також створює додаткові труднощі в реалізації компілятора. Наприклад, вказівник на клас, що має декілька батьків, більше не може розглядатися (з використанням застарілого приведення типа в стилі Сі) як вказівник на одного зі своїх предків, оскільки батьківська частина об'єкту може бути розташована з деяким зсувом відносно початку об'єкту (тобто значення вказівника). З цієї ж причини не можна приводити вказівник на батьківський клас до вказівника на похідний без використання операторів приведення C++ (dynamic_cast). Інколи шаблони приводять до породження коду дуже великого об'єму. Для зниження розміру машинного коду можна спеціальним чином готувати вихідний код. Іншим рішенням є стандартизована ще в 1998 році можливість експорту шаблонів.

Метапрограмування на основі шаблонів C++ складне і при цьому обмежене в можливостях. Воно полягає в реалізації засобами шаблонів C++ інтерпретатора примітивної функціональної мови програмування, що виконується під час компіляції. Сама по собі дана можливість дуже приваблива, але такий код дуже важко сприймати і налагоджувати. Менш поширені мови (Lisp/scheme, Nemerle) мають потужніші і одночасно простіші для сприйняття підсистеми метапрограмування.

Явна підтримка функціонального програмування присутня лише в майбутньому стандарті c++0x. Даний недолік усувається різними бібліотеками

(Loki, Boost), що використовують засоби метапрограмування для розширення мови функціональними конструкціями (наприклад, підтримкою лямбд/анонімних методів), але якість подібних рішень значно поступається якості вбудованих у функціональні мови рішень. Такі можливості функціональних мов, як зіставлення із зразком, взагалі складно емулювати засобами метапрограмування.

Деякі вважають недоліком мови C++ відсутність вбудованої системи збірки сміття. З іншого боку, засоби C++ дозволяють реалізувати збірку сміття на рівні бібліотеки. Противники збірки сміття вважають, що RAII є більш гідною альтернативою. C++ дозволяє користувачеві самому обирати стратегію управління ресурсами [5].

2.2.2 Язик програмування C#

C# - об'єктно-орієнтована мова програмування. Розроблена в 1998—2001 роках групою інженерів під керівництвом Андерса Хейлсберга в компанії Microsoft як основна мова розробки додатків для платформи Microsoft .NET. Компілятор з C# входить в стандартну установку самої .NET, тому програми на ньому можна створювати і компілювати навіть без інструментальних засобів. C# відноситься до сім'ї мов з c-подібнім синтаксисом, з них його синтаксис найбільш близький до C++ і Java.

Мова має статичну типізацію, підтримує поліморфізм, перевантаження операторів (у тому числі операторів явного і неявного приведення типів), делегати, атрибути, події, властивості, узагальнені типи і методи, ітератори, анонімні функції з підтримкою замикань, LINQ, виключення, коментарі у форматі XML. Перейнявши багато що від своїх попередників (мов C++, Java, Delphi, Модула і Smalltalk) C#, спираючись на практику їх використання, виключає деякі моделі, що зарекомендували себе як проблематичні при

розробці програмних систем: так, C# не підтримує множинне наслідування класів (на відміну від C++).

C# розроблявся як мова програмування прикладного рівня для CLR і залежить, перш за все, від можливостей самої CLR. Це стосується, насамперед, системи типів C#, яка відображає BCL. Присутність або відсутність тих або інших особливостей мови диктується тим, чи може конкретна мовна особливість бути трансльована у відповідні конструкції CLR. Так, з розвитком CLR від версії 1.1 до 2.0 значно збагатився і сам C#; подібної взаємодії слід чекати і надалі. Проте ця закономірність була порушена з виходом C# 3.0, що є розширеннями мови, платформами, що не спираються на розширення .NET. CLR надає C#, як і всім іншим .NET-орієнтованим мовам, багато можливостей, яких позбавлені «класичні» мови програмування. Наприклад, збірка сміття не реалізована в самому C#, а виробляється CLR для програм, написаних на C# точно так, як і це робиться для програм на VB.NET, J# і ін.

C# стандартизований в ECMA і ISO. Відомо, як мінімум, про дві незалежні реалізації C#, що базуються на цій специфікації і знаходяться в даний час на різних стадіях розробки:

Mono, почата компанією Ximian і продовжена її покупцем і наступником Novell.

dotgnu і Portable.NET, Free Software Foundation, що розробляються.

Проект C# був початий в грудні 1998 і отримав кодову назву COOL (C-style Object Oriented Language). Версія 1.0 була анонсована разом з платформою .NET у червні 2000 року, тоді ж з'явилася і перша загальнодоступна бета-версія; C# 1.0 остаточно вийшов разом з Microsoft Visual Studio .NET у лютому 2002 року. Перша версія C# нагадувала по своїх можливостях Java 1.4, трохи їх розширюючи: так, в C# були властивості (що виглядають в коді як поля об'єкту, але що на ділі викликають при зверненні методи класу), індексатори (подібні до властивостей, але що приймають параметр як індекс масиву), події, делегати, цикли foreach, структури, що передається за значенням, автоматичне перетворення вбудованих типів в об'єкти при необхідності (boxing), атрибути, вбудовані засоби взаємодії з некерованим кодом (DLL, COM) і інше. Крім

того, в C# вирішено було перенести деякі можливості C++, відсутні в Java: беззнакові типи, перевантаження операторів (з деякими обмеженнями, на відміну від C++), передача параметрів в метод за посиланням, методи із змінним числом параметрів, оператор goto (з обмеженнями). Також в C# залишили обмежену можливість роботи з вказівниками - в місцях коду, спеціально позначених словом unsafe і при вказівці спеціальної опції компілятору.

Проект специфікації C# 2.0 вперше був опублікований Microsoft в жовтні 2003 року; у 2004 році виходили бета-версії (проект з кодовою назвою Whidbey), C# 2.0 остаточно вийшов 7 листопада 2005 року разом з Visual Studio 2005 і .NET 2.0.

Нові можливості у версії 2.0:

Часткові типи (розділення реалізації класу більш ніж на один файл).

Узагальнені типи (generics). На відміну від шаблонів C++, вони підтримують деякі додаткові можливості і працюють на рівні віртуальної машини. В той же час, параметрами узагальненого типа не можуть бути вирази, вони не можуть бути повністю або частково спеціалізовані, не підтримують шаблонних параметрів за умовчанням, від шаблонного параметра не можна успадковуватися, і так далі.

Нова форма ітератора, що дозволяє створювати співпрограми за допомогою ключового слова yield, подібно Python і Ruby.

Анонімні методи, що забезпечують функціональність замикання.

Можливість створювати процедури, що зберігаються, тригери і навіть типи даних в .Net мовах (у тому числі і на C#).

Підтримка 64-розрядних обчислень, що крім усього іншого, дозволяє збільшити адресний простір і використовувати 64-розрядних примітивних типів даних.

У червні 2004 року Андерс Хейлсберг вперше розповів на сайті Microsoft про плановані розширення мови в C#3.0. У вересні 2005 року вийшли проект специфікації C# 3.0 і бета-версія C# 3.0, що встановлюється у вигляді

доповнення до існуючих Visual Studio 2005 і .NET 2.0. Остаточно ця версія мови увійшла в Visual Studio 2008 і .NET 3.5.

В C# 3.0 з'явилися наступні радикальні додавання до мови:

– ключові слова select, from, where, що дозволяють робити запити з SQL, XML, колекцій і тому подібне (запит, інтегрований в мову, Language Integrated Query, або LINQ);

– ініціалізація об'єкту разом з його властивостями;

– дерева виразів: лямбда-вирази тепер можуть представлятися у вигляді структури даних, доступної для обходу під час виконання, тим самим дозволяючи транслювати строго C# - вирази, що типізуються, в інші домени (наприклад, вирази SQL);

– виведення типів локальної змінної;

– методи-розширення - додавання методу в існуючий клас за допомогою ключового слова this при першому параметрі статичної функції;

Прев'ю C# 4.0 було представлене в кінці 2016 року, разом з ctp-версією Visual Studio 7 Visual Basic 10.0 і C# 4.0 були випущені в квітні 2017 року, одночасно з випуском Visual Studio 2017 [6].

2.2.3 Язик програмування Java

Java - об'єктно-орієнтована мова програмування, розроблена компанією Sun Microsystems. Додатки Java зазвичай компілюються в спеціальний байт-код, тому вони можуть працювати на будь-якій віртуальній java-машині (JVM) незалежно від комп'ютерної архітектури. Дата офіційного випуску — 23 травня 1995 року.

Іншою важливою особливістю технології Java є гнучка система безпеки завдяки тому, що виконання програми повністю контролюється віртуальною машиною. Будь-які операції, які перевищують встановлені повноваження

програми (наприклад, спроба несанкціонованого доступу до даних або з'єднання з іншим комп'ютером) викликають негайне переривання.

Часто до недоліків концепції віртуальної машини відносять те, що виконання байта-коду віртуальною машиною може знижувати продуктивність програм і алгоритмів, реалізованих на мові Java. Дане твердження було справедливе для перших версій віртуальної машини Java, проте останнім часом воно практично втратило актуальність.

Цьому сприяв ряд удосконалень:

– вживання технології трансляції байта-коду в машинний код безпосередньо під час роботи програми (Jit-технологія) з можливістю збереження версій класу в машинному коді;

– широке використання платформено-орієнтованого коду (native-код) в стандартних бібліотеках;

– апаратні засоби, що забезпечують прискорену обробку байта-коду (наприклад, технологія Jazelle, підтримувана деякими процесорами фірми ARM).

Основні можливості мови:

– автоматичне управління пам'яттю;

– розширені можливості обробки виняткових ситуацій;

– багатий набір засобів фільтрації введення/виводу;

– набір стандартних колекцій, таких як масив, список, стік і т. п.;

– наявність простих засобів створення мережевих застосувань (у тому числі з використанням протоколу RMI);

– наявність класів, що дозволяють виконувати http-запити і обробляти відповіді;

– вбудовані в мову засобу створення багатопотокових додатків;

– уніфікований доступ до баз даних;

– підтримка шаблонів (починаючи з версії 1.5);

– паралельне виконання програм.

Засоби розробки ПО:

– JDK - окрім набору бібліотек для платформ Java SE і Java EE містить компілятор командного рядка javac і набір утиліт, що також працюють в режимі командного рядка.

– Netbeans IDE - вільне інтегроване середовище розробки для всіх платформ Java - Java ME, Java SE і Java EE. Пропонується Sun Microsystems, розробником Java, як базовий засіб для розробки ПО на мові Java і інших мовах (C, C++, Ruby, php, Fortran і ін.).

– Eclipse IDE - вільне інтегроване середовище розробки для Java SE і Java EE. Ведуться роботи по підтримці в Eclipse платформи Java ME. Пропонується IBM, одним з найважливіших розробників корпоративного ПО, як базовий засіб для розробки ПО на мові Java і інших мовах (C, C++, Ruby, Fortran і ін.)

– INTELLIJ IDEA - середовище розробки для платформ Java SE, Java EE і Java ME. Поширюється в двох версіях: вільною безкоштовною (Community Edition) і комерційною (Ultimate Edition).

Наступні успішні проекти реалізовані з використанням Java - технологій: Runescape, Amazon, ebay, Yandex, Linkedin, Yahoo!.

Наступні компанії в основному фокусуються на Java - технологіях: SAP, IBM, Oracle. Зокрема, СУБД Oracle включає JVM як свою складову частину, що забезпечує можливість безпосереднього програмування СУБД на мові Java.

JavaScript - об'єктно-орієнтована скріптова мова програмування. Є діалектом мови Ecmascript. Javascript зазвичай використовується як вбудована мова для програмного доступу до об'єктів додатків. Найбільш широке вживання знаходить в браузерах як мова сценаріїв для додання інтерактивності веб-сторінкам. Основні архітектурні риси: динамічна типізація, слабка типізація, автоматичне управління пам'яттю, прототипне програмування, функції як об'єкти першого класу.

На Javascript вплинуло багато мов; при розробці була поставлена мета зробити мову схожою на Java, але при цьому легкою для використання не програмістами. Мовою Javascript не володіє яка-небудь компанія або

організація, що відрізняє її від ряду мов програмування, використовуваних у веб-розробці.

У 1996 році компанія Microsoft випустила аналог мови Javascript, названий Jscript. Анонсована ця мова була 18 липня 1996 року. Першим браузером, що підтримував цю реалізацію, був Internet Explorer 3.0. За ініціативою компанії Netscape була проведена стандартизація мови асоціацією ECMA. Стандартизована версія має назву Ecmascript, що описується стандартом Ecma-262. Першій версії специфікації відповідав Javascript версії 1.1, а також мови Jscript і Scripteasy.

Javascript володіє рядом властивостей об'єктно-орієнтованої мови, але реалізоване в мові прототипування приводить відмінності в роботі з об'єктами в порівнянні з традиційними об'єктно-орієнтованими мовами. Крім того, Javascript має ряд властивостей, властивих функціональним мовам, - функції як об'єкти першого класу, об'єкти як списки, каррінг, анонімні функції, замикання, що додає мові додаткову гнучкість.

Синтаксис мови Javascript багато в чому нагадує синтаксис Сі і Java, семантично ж мова набагато ближче до Self, Smalltalk або навіть Ліспу. Структурно Javascript можна представити у вигляді об'єднання трьох чітко розділених одна від другої частин:

– ядро (Ecmascript);
– об'єктна модель браузера (Browser Object Model або BOM);
– об'єктна модель документа (Document Object Model або DOM).

Якщо розглядати Javascript у відмінних від браузера оточеннях, то об'єктна модель браузера і об'єктна модель документа можуть не підтримуватися.

Середа застосування:

– веб-додатки;
– AJAX - популярний підхід до побудови інтерактивних, призначених для користувача, інтерфейсів веб-додатків, що полягає в «фоновому» асинхронному обміні даними браузера з веб-сервером;
– браузерні операційні системи;

– скрипти користувача в браузері;

– серверні додатки;

– мобільні додатки;

– прикладне програмне забезпечення.

У Javascript доступ до відладчиків стає особливо корисним при розробці крупних нетривіальних програм із-за відмінностей в реалізаціях різних браузерів (зокрема, відносно об'єктної моделі документа). Корисно мати доступ до відладчика для кожного з браузерів, в яких працюватиме веб-додаток.

За станом на листопад 2009 року, Internet Explorer, Firefox, Safari, Google Chrome, а також Opera мають відладчики сценаріїв. Internet Explorer має три відладчики: Microsoft Visual Studio - найповніший з них, слідом за ним слідує Microsoft Script Editor, і, нарешті, вільний Microsoft Script Debugger, який набагато простіший, ніж два попередніх. Безкоштовний Microsoft Visual Web Developer Express надає обмежену версію з налагоджувальною функцією Javascript в Microsoft Visual Studio. У восьмій версії в IE разом з інструментами для розробників з'явився вбудований відладчик. У Opera також є власний відладчик — Opera Dragonfly. У Safari входить відладчик Javascript Webkit Web Inspector. Цей же відладчик доступний і в інших браузерах, використовуючих Webkit: Google Chrome, Arora, Rekonq, Midori і ін.

У деяких мовах програмування існують засоби підтримки взаємодії з javascript-кодом:

– Для PHP є пакет Html_javascript, що надає інтерфейс створення простих javascript-програм.

– Відповідний пакет для TCL називається ::javascript. Він надає команди генерації коди HTML і Javascript.

– Пакет для Perl Data::javascript дозволяє переносити структури даних Perl в javascript-код [7][8].

2.2.4 Язик програмування PHP

PHP - скріптова мова програмування загального призначення, що інтенсивно застосовується для розробки веб-додатків. В даний час підтримується переважною більшістю хостингу провайдерів і є одним з лідерів серед мов програмування динамічних веб-сайтів.

Популярність в області побудови веб-сайтів визначається наявністю великого набору вбудованих засобів для розробки веб-додатків. Основні з них:

– автоматичне вилучення POST і get-параметрів, а також змінних оточення веб-сервера в зумовлені масиви;

– файлові функції успішно обробляють як локальні, так і видалені файли;

– автоматична відправка http-заголовків;

– робота з cookies і сесіями;

– обробка файлів, що завантажуються на сервер;

– робота з HTTP заголовками і HTTP авторизацією;

– робота з Xforms;

– робота з видаленими файлами і сокетами.

На сьогоднішній день PHP використовується сотнями тисяч розробників. Згідно з рейтингом Tiobe, що базується на даних пошукових систем, в грудні 2009 року PHP знаходиться на 3 місці серед мов програмування (поступаючись Java і C), піднявшись за рік на дві позиції. До найбільших сайтів, використовуючим PHP, відносяться Facebook, В контакті, Wikipedia, Youtube.

Входить в LAMP — поширений набір для створення веб-сайтів (Linux, Apache, MYSQL, PHP).

З використанням PHP розроблено безліч додатків, які широко використовуються на різних сайтах, форумах і блогах:

– Drupal, Typo3, Php-nuke, Mambo, Joomla — системи управління вмістом (CMS);

– Magento, oscommerce — системи для інтернет-комерції;

– 4images — галереї зображень;

– Mediawiki, Dokuwiki — вікі-движки;

– phpbb, SMF, vbulletin, Invision Power Board — форумні движки;

– phpmyadmin, phppgadmin — утиліти адміністрування СУБД;

– Wordpress — движок для побудови блог-сайтів;

– eyeos — видалена операційна система, заснована на принципі Desktop Operating System.

Для швидкої розробки додатків на PHP було створено безліч фреймворков, найбільш популярними з яких є Zend Framework, CAKEPHP, Symfony і Codeigniter.

2.2.5 Язик програмування Python

Python - високорівнева мова програмування загального призначення з акцентом на продуктивність розробника. Стандартна бібліотека включає великий об'єм корисних функцій. Python підтримує декілька парадигм програмування, у тому числі структурне, об'єктно-орієнтоване, функціональне, імперативне і аспектно-орієнтоване. Основні архітектурні риси - динамічна типізація, автоматичне управління пам'яттю, повна інтроспективна, механізм обробки виключень, підтримка багатопотокових обчислень і зручні високорівневі структури даних. Код в Python організовується у функції і класи, які можуть об'єднуватися в модулі (які у свою чергу можуть бути об'єднані в пакети).

Еталонною реалізацією Python є інтерпретатор Cpython, що підтримує більшість активно використовуваємих платформ. Він поширюється вільно під дуже ліберальною ліцензією, що дозволяє використовувати його без обмежень в будь-яких застосуваннях. Є реалізації інтерпретаторів для JVM (з можливістю компіляції), MSIL (з можливістю компіляції), LLVM і інших.

Python - мова програмування, що активно розвивається, нові версії (з додаванням/зміною мовних властивостей) виходять приблизно раз в два з половиною роки. Внаслідок цього і деяких інших причин на Python відсутні ANSI, ISO або інші офіційні стандарти, їх роль виконує Cpython.

З'явившись порівняно пізно, Python створювався під впливом безлічі мов програмування:

– ABC - відступи для угрупування операторів, високорівневі структури даних (фактично, Python створювався як спроба виправити помилки, допущені при проектуванні ABC);

– Modula-3 — пакети, модулі, використання else спільно з try і except, іменовані аргументи функцій (на це також вплинув Common Lisp);

– Ci, C++ — деякі синтаксичні конструкції

– Smalltalk — об'єктно-орієнтоване програмування;

– Lisp — окремі риси функціонального програмування (lambda, map, reduce, filter та інші);

– Fortran — зрізи масивів, комплексна арифметика;

– Miranda — облікові вирази;

– Java — модулі logging, unittest, threading (частина можливостей оригінального модуля не реалізована), xml.sax стандартної бібліотеки, спільне використання finally і except при обробці виключень, використання @ для декораторів;

– Icon — генератори.

Велика частина інших можливостей Python (наприклад, байт-компіляція вихідної коди) також була реалізована раніше в інших мовах.

Python — стабільна і поширена мова. Він використовується в багатьох проектах і в різних якостях: як основна мова програмування або для створення розширень і інтеграції додатків. На Python реалізована велика кількість проектів, також він активно використовується для створення прототипів майбутніх програм.

2.3 Порівняльна характеристика мов програмування

Для того, щоб обрати з вище наведених мов програмування найбільш доцільну, проведемо порівняння їх переваг та недоліків.

1) C# и Java. Обидві мови реалізують одну модель роботи з динамічними даними: об'єкти створюються динамічно за допомогою конструкції new, середовище виконання відстежує наявність посилань на них, а збірник сміття періодично очищає пам'ять від об'єктів, посилань на яких немає.

У Java модифікатор protected в описі, окрім доступу з класів-нащадків, дозволяє доступ зі всіх класів, що входять в той же пакет, що і клас-власник. У C# для об'єктів, які мають бути видні в межах збірки (зразковий аналог пакету Java) введений окремий модифікатор internal, а protected зберігає свій початковий сенс, узятий з C++, - доступ лише з класів-нащадків. Допускається комбінувати internal і protected — тоді вийде область доступу, відповідна protected в Java.

У C# примітивні типи (byte, int, double, float, bool і ін.) і структури (struct) передаються за значенням (т.з. значимі типи), останні типи передаються по посиланню (т.з. посилальні типи). C# також підтримує явний опис передачі параметрів по посиланню (ключові слова ref і out). При використанні out компілятор контролює наявність в методі надання значення. У Java параметри методу передаються лише за значенням, але оскільки для екземплярів класів передається посилання, ніщо не заважає змінити в методі екземпляр, переданий через параметр.

У Java для того, щоб від класу не можна було успадковуватися, його можна оголосити фінальним final, тим самим отримавши частковий аналог конструкції struct (копіювання за значенням при цьому підтримуватися не буде). У C# для тих же цілей використовується модифікатор sealed.

У Java можуть бути оголошені, строго кажучи, лише одновимірні масиви. Багатовимірний масив в Java — масив масивів. У C# є як справжні багатовимірні масиви, так і масиви масивів, які в C# зазвичай називаються «нерівними», або «ступінчастими» (jagged). Багатовимірні масиви завжди «прямокутні» (кажучи в двовимірній термінології), тоді як масиви масивів можуть зберігати рядки різної довжини (знову-таки в двовимірному випадку, в

багатовимірному аналогічно). Багатовимірні масиви прискорюють доступ до пам'яті, а нерівні масиви працюють повільніше, але економлять пам'ять, коли не всі рядки заповнені. Багатовимірні масиви вимагають для свого створення лише один виклик оператора new, а ступінчасті вимагають явно виділяти пам'ять в циклі для кожного рядка.

C# включає перевантаження операцій і приведення типів, що задається користувачем. Java не включає перевантаження операцій, щоб уникнути зловживань нею і для підтримки простоти мови.

C#, на відміну від Java, підтримує умовну компіляцію з використанням директив препроцесора. У ньому також є атрибут Conditional, що означає, що вказаний метод викликається лише тоді, коли визначена дана константа компіляції. Таким чином можна вставляти в код, наприклад, перевірки допущень (assertion checks), які працюватимуть лише в налагоджувальній версії, коли визначена константа DEBUG. У стандартній бібліотеці .NET такий метод Debug.Assert().

Java Native Interface (JNI) дозволяє програмам викликати з Java низькорівневі, системно-залежні функції (наприклад, бібліотека winapi). C# також дозволяє програмістові відключити нормальну перевірку типів і інші можливості безпеки CLR, вирішуючи використання змінних-покажчиків за умови вживання ключового слова unsafe.

Java підтримує виключення, що перевіряються (checked): програміст повинен явно вказати для кожного методу типів виключень, які метод може викинути, ці типи перераховують в оголошенні методу після ключового слова throws. Якщо метод використовує методи, що викидають виключення, що перевіряються, він повинен або явно перехоплювати всі ці виключення, або містити їх у власному описі. Таким чином, код явно містить перелік виключень, які можуть бути викинуті кожним методом. C# виключення, що перевіряються, не підтримує. Їх відсутність є свідомим вибором розробників. Андерс Хейлсберг, головний архітектор C#, вважає, що в Java вони були якоюсь мірою експериментом і себе не виправдали.

2) JavaScript та мови Сі. Не дивлячись на схожий з Сі синтаксис, Javascript в порівнянні з мовою Сі має корінні відмінності:

– об'єкти з інтроспективною можливістю;

– функції як об'єкти першого класу;

– автоматичне приведення типів;

– автоматична збірка сміття;

– анонімні функції.

3) JavaScript та Java. Загальною помилкою є те, що Javascript аналогічний або тісно пов'язаний з Java, але це не зовсім так. Обидві мови мають С-подібний синтаксис, є об'єктно-орієнтованими і, як правило, широко використовуються в клієнтських веб-застосуваннях, та на цьому їх схожість закінчується.

Відмінності між JavaScript та Java:

– Java реалізує ООП підхід, заснований на класах, а Javascript - на прототипах;

– Java має статичну типізацію, Javascript - динамічну типізацію;

– Java завантажується із скомпільованого байт-коду, Javascript інтерпретується безпосередньо з файлу (але часто з непомітною jit-компіляцією).

В якості мови програмування обрано Visual C#, який має ряд переваг порівняно з іншими мовами програмування. C# відноситься до мов з C – подібним синтаксисом, з яких його синтаксис найбільш подібний до C++ та Java. Мова C# має статичну типізацію, підтримує поліморфізм, перевантаження операторів (в тому числі операторів явного та неявного приведення типу), делегати, атрибути, події, властивості, узагальнені типи та методи, ітератори, виключення, коментарі в форматі XML. Також C# виключає деякі моделі, які зарекомендували себе як проблематичні при розробці програмного забезпечення, наприклад, численне наслідування[9].

3 ОПИС ПРОГРАМИ

3.1 Огляд програмного забезпечення

Стосовно технічного завдання необхідно розробити розподілену базу даних регіональної системи екологічного моніторингу.

База даних призначена для збору оперативної інформації про стан атмосферного повітря в місті і відображення її в зручному вигляді, підтримки статистичної, математичної, спеціалізованої інформації.

Розроблювана система припускає наступні етапи:

– створення програмного забезпечення, що здійснює збір інформації, дозволяє отримати об'єктивну оцінку про стан повітря міста у будь-який час;

– аналіз моніторингової інформації і генерація звітів.

Для реалізації програмного забезпечення використовується http-сервер Apache.

Apache є кросплатформним ПЗ, який підтримує операційні системи GNU/Linux, BSD, Mac OS, Microsoft Windows, Novell NetWare, BeOS.

Основними перевагами Apache вважаються надійність і гнучкість конфігурації. Він дозволяє підключати зовнішні модулі для надання даних, використовувати СУБД для аутентифікації користувачів, модифікувати повідомлення про помилки і т.д., підтримувати протокол IPv6.

Для створення і зберігання бази даних будемо використовувати систему управління базами даних MySQL.

MySQL – вільна система управління базами даних (СУБД). MySQL є власністю компанії Oracle Corporation, що одержала її разом з поглиненою Sun Microsystems, що здійснює розробку і підтримку додатку. MySQL розповсюджується під GNU General Public License і підвладна комерційній ліцензії. Крім цього розробники створюють функціональність за замовленням ліцензійних користувачів, саме завдяки такому замовленню майже в найраніших версіях з'явився механізм реплікації.

MySQL є рішенням для малих і середніх додатків. Входить в LAMP. Зазвичай MySQL використовується як сервер, до якого звертаються локальні або видалені клієнти, проте в дистрибутив входить бібліотека внутрішнього сервера, що дозволяє включати MySQL в автономні програми.

Гнучкість СУБД MySQL забезпечується підтримкою великої кількості типів таблиць: користувачі можуть вибрати як таблиці типу MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції на рівні окремих записів. Більш того, СУБД MySQL поставляється із спеціальним типом таблиць EXAMPLE, що демонструє принципи створення нових типів таблиць. Завдяки відкритій архітектурі і GPL–ліцензуванню, в СУБД MySQL постійно з'являються нові типи таблиць.

3.2 Опис бази даних

3.2.1 Створення бази даних

Для створення бази даних в MySQL необхідно використовувати наступну команду

CREATE DATABASE [IF NOT EXISTS] db_name [CHARACTER SET charset] [COLLATE collation];

db_name – Ім'я, яке буде присвоєно створюваної базі даних.

IF NOT EXISTS – Якщо не вказати цей параметр, то при спробі створення бази даних з вже існуючим ім'ям, виникне помилка виконання команди.

CHARACTER SET, COLLATE – використовується для завдання стандартної кодування таблиці і порядку сортування.

Якщо при створенні таблиці ці параметри не вказуються, то кодування і порядок сортування новостворюваної таблиці беруться з значень, зазначених для всієї бази даних. Якщо встановлено параметр CHARACTER SET, але не заданий параметр COLLATE, то використовується стандартний порядок сортування. Якщо встановлено параметр COLLATE, але не заданий

CHARACTER SET, то кодування визначає перша частина імені порядку сортування в COLLATE.

Кодування, задана в CHARACTER SET, повинна підтримуватися сервером (latin1 або sjis), а порядок сортування повинен бути допустимим для поточного кодування.

Створення бази даних Atmosfera приведено на рис.3.1.

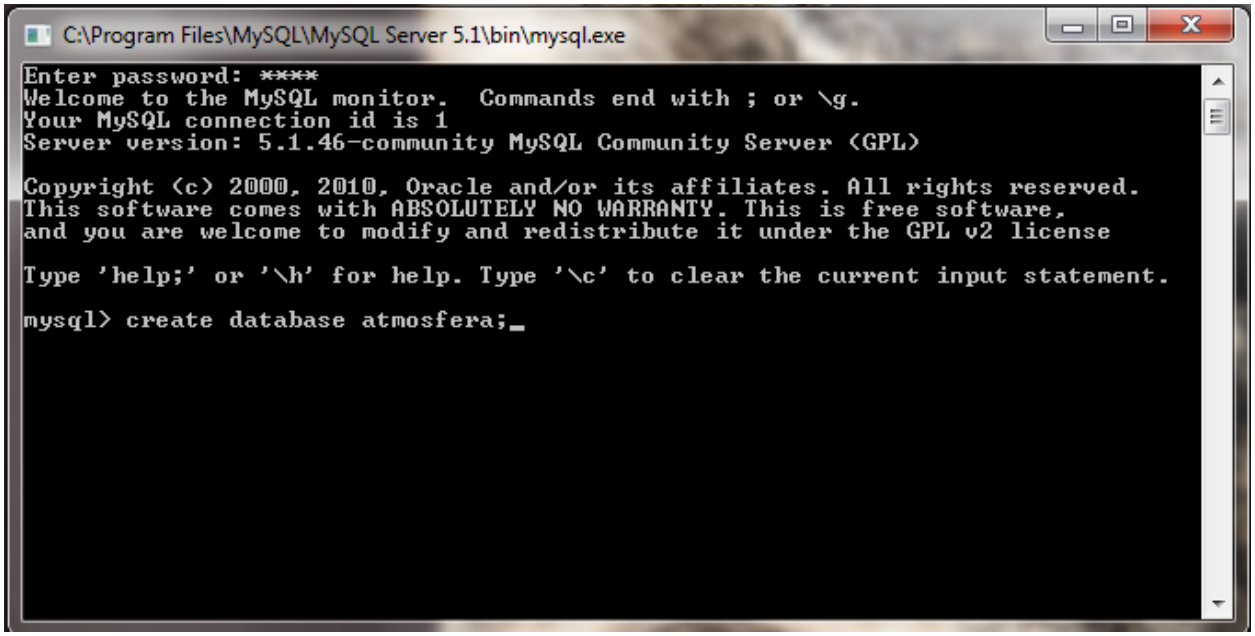


Рисунок 3.1 – Створення бази даних

3.2.2 Створення таблиць бази даних

Для створення таблиць в базах даних MySQL виконується запит за допомогою команди:

CREATE [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name [(create_definition,...)]

[table_options] [select_statement]

tbl_name – зазначає ім'я таблиці, яка буде створена в поточній базі даних. Якщо ніяка база даних на момент виклику команди CREATE TABLE не була прийнята поточної, то виникне помилка виконання команди. Починаючи з

MySQL 3.22 запроваджена можливість явно вказати базу даних, в якій буде створена нова таблиця, за допомогою синтаксису db_name.tbl_name.

TEMPORARY – цей параметр використовується для створення тимчасової таблиці з ім'ям tbl_name протягом лише поточного сценарію. По закінченню виконання сценарію створена таблиця видаляється. Дана можливість з'явилася в MySQL 3.23. В MySQL 4.0.2 для створення тимчасових таблиць потрібні привілеї CREATE TEMPORARY TABLES.

IF NOT EXISTS – якщо вказаний цей параметр і проводиться спроба створити таблицю з дублюючим ім'ям (тобто таблиця з таким ім'ям в поточної БД вже є), то таблиця створена не буде і повідомлення про помилку не з'явиться. В іншому випадку таблиця також створена не буде, але команда викличе помилку. Слід зазначити, що при створенні порівнюються тільки імена таблиць. Внутрішні структури не порівнюються.

create_definition – визначає внутрішню структуру створюваної таблиці (назви і типи полів, ключі, індекси і т.д.). Можливий синтаксис create_definition:

col_name type [NOT NULL | NULL] [DEFAULT default_value] [AUTO_INCREMENT] [PRIMARY KEY] [reference_definition] або PRIMARY KEY (index_col_name ,...) або KEY [index_name] (index_col_name ,...) або INDEX [index_name] (index_col_name ,...) або UNIQUE [INDEX] [index_name] (index_col_name ,...) або FULLTEXT [INDEX] [index_name] (index_col_name ,...) або [CONSTRAINT symbol] FOREIGN KEY [index_name] (index_col_name ,...) [reference_definition] або CHECK (expr) col_name. Зазначає ім'я стовпця в створюваній таблиці.

type – Визначає тип даних для стовпця col_name.

[NOT NULL | NULL] – вказує, чи може даний стовпець міститиме значення NULL чи ні. Якщо не вказано, то за замовчуванням приймається NULL (тобто може містити NULL).

[DEFAULT default_value] – задає значення за замовчуванням для даного стовпця. При вставці нового запису в таблицю командою INSERT, якщо

значення для поля col_name явно вказано не було, то встановлюється значення default_value.

[AUTO_INCREMENT] – при вставці нового запису в таблицю поле з цим атрибутом автоматично отримає числове значення, на 1 більше самого великого значення для цього поля в поточний момент часу. Дана можливість звичайно використовується для генерування унікальних ідентифікаторів рядків. Стовпець, для якого застосовується атрибут AUTO_INCREMENT, повинен мати цілочисельний тип. У таблиці може бути тільки один стовпець з атрибутом AUTO_INCREMENT. Так само цей стовпець повинен бути проіндексований. Відлік послідовності чисел для AUTO_INCREMENT починається з 1. Це можуть бути тільки позитивні числа.

Створення таблиці admt приведено на рис.3.2

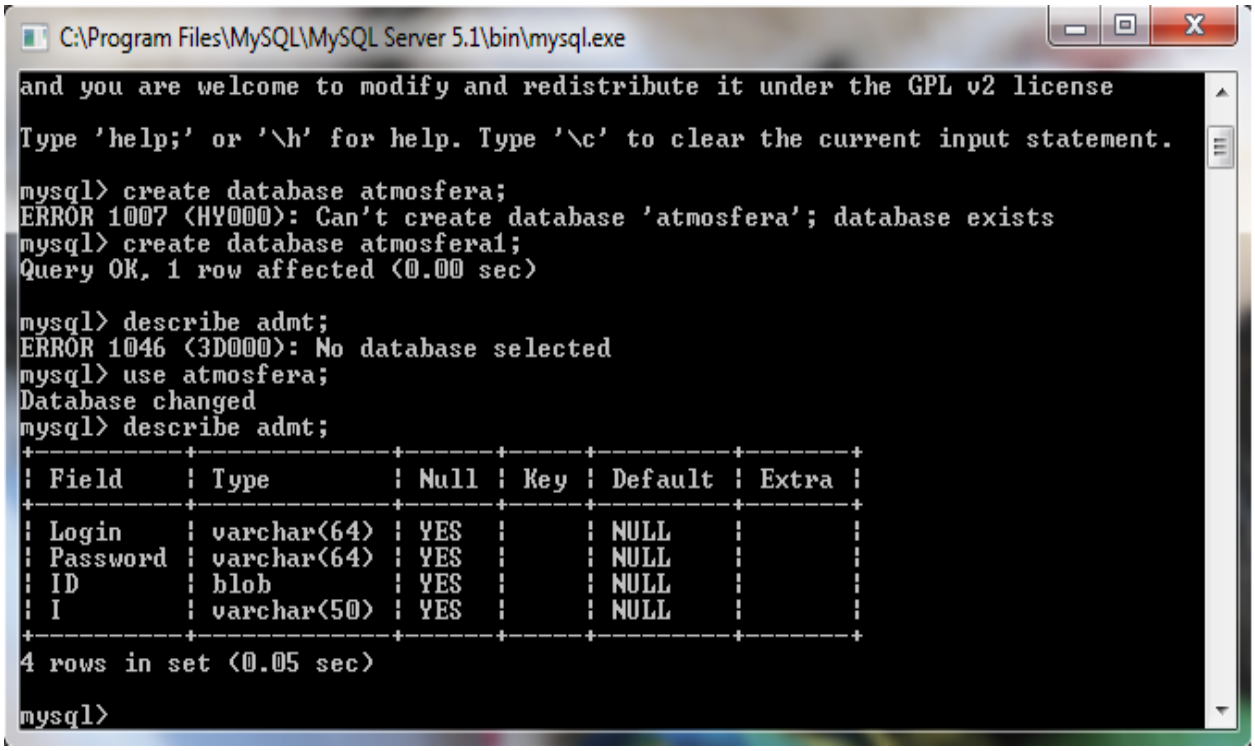


Рисунок 3.2 – Створення таблиці admt

Створення таблиці Users приведено на рис.3.3.

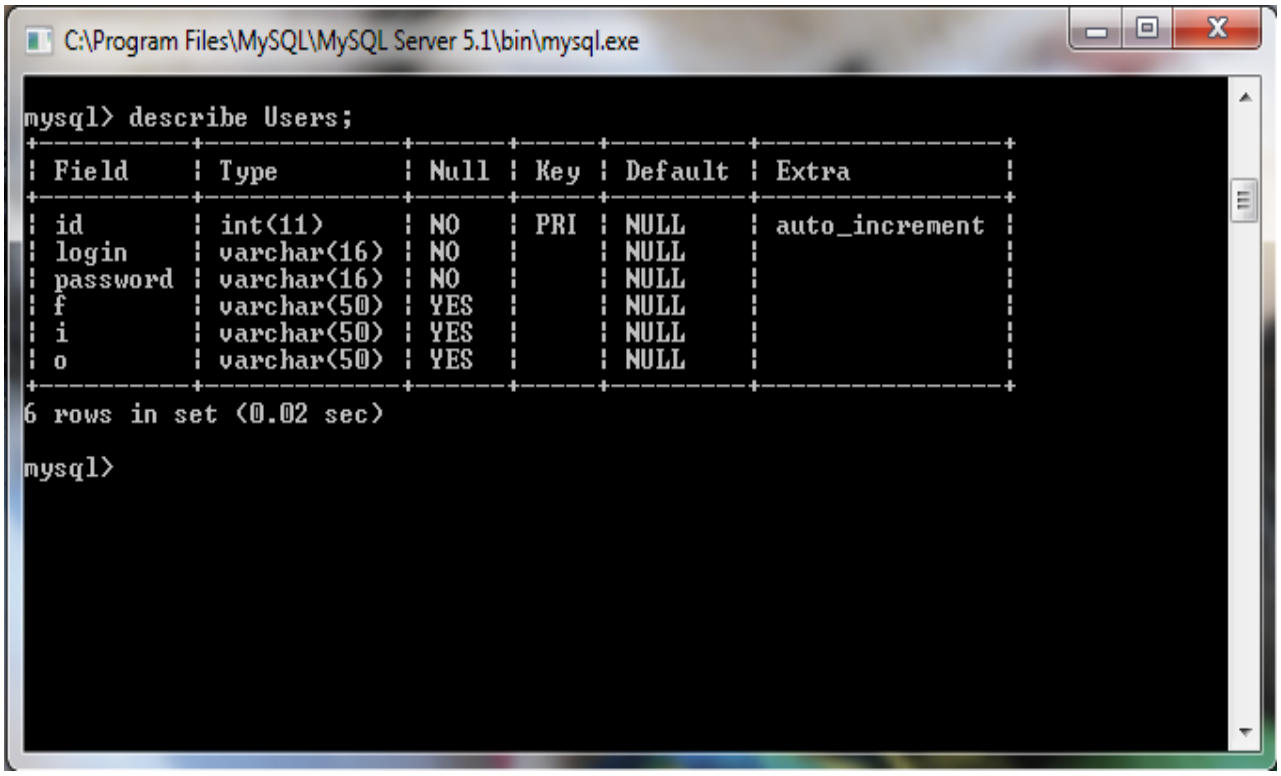


Рисунок 3.3 – Створення таблиці Users

Створення таблиці atmos приведено на рис.3.4.

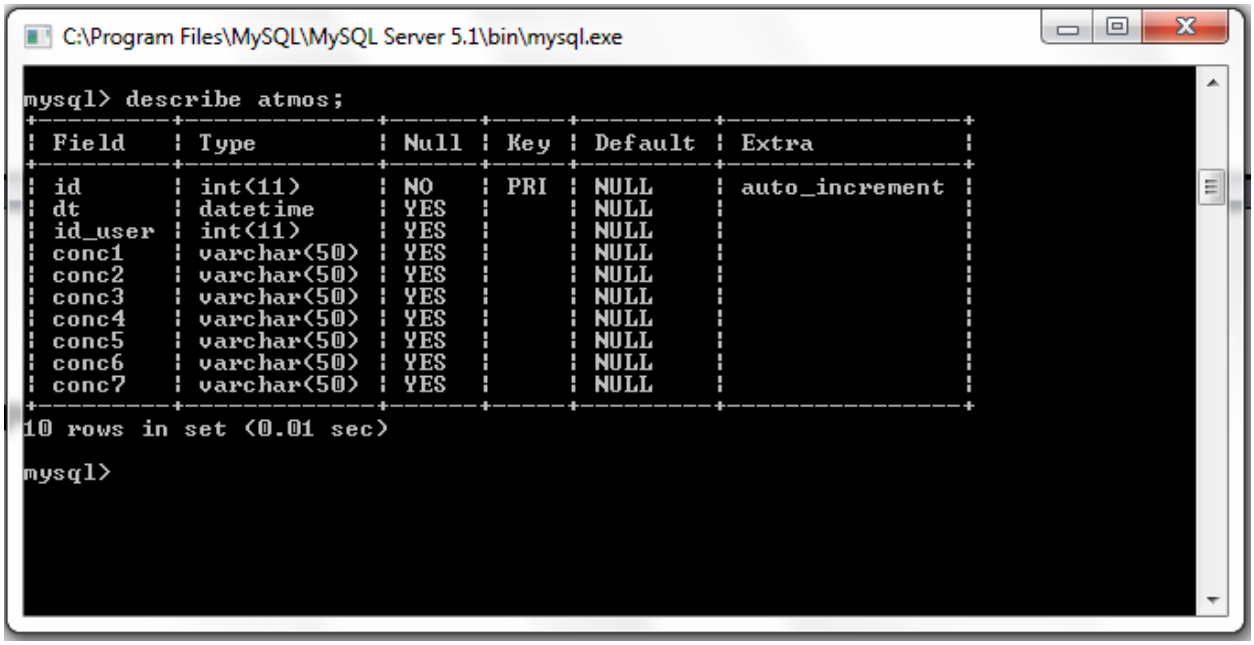


Рисунок 3.4 – Створення таблиці atmos

3.2.3 Інтерфейс програми

Для реалізації програмного забезпечення використовується WindowsFormsApplication язику програмування C#. Для аутентифікації користувачів виконується форма зображена на рис.3.5.

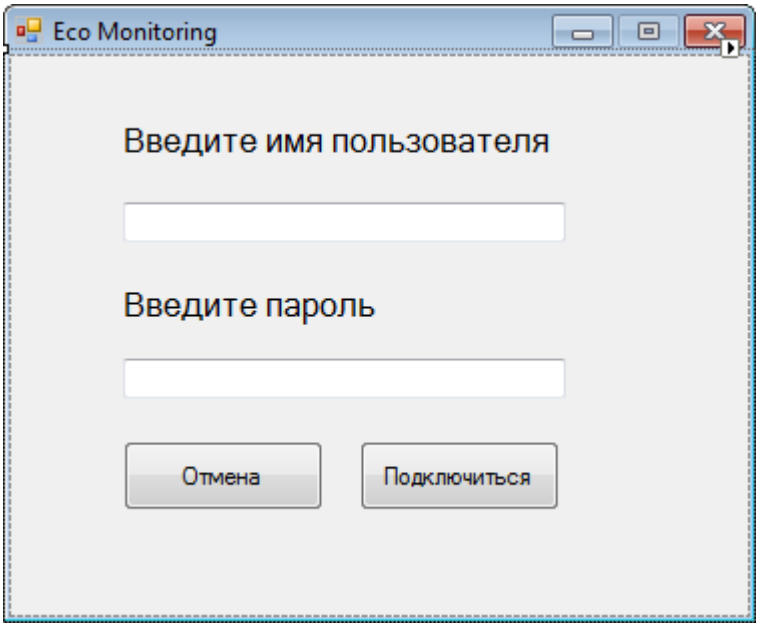


Рисунок 3.5 – Форма аутентифікації користувачів

Алгоритм аутентифікації користувачів зображений на рис.3.6-3.7

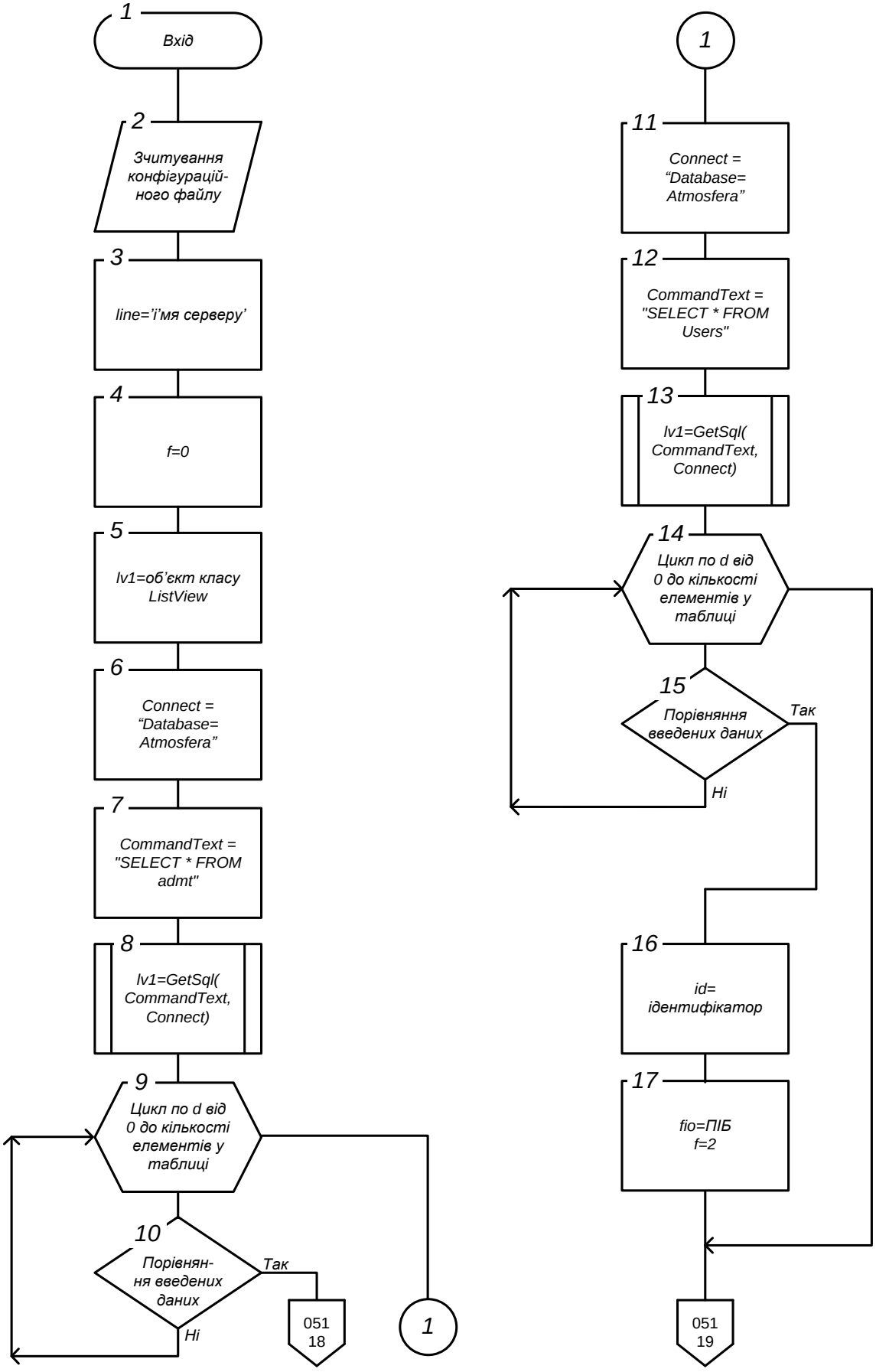
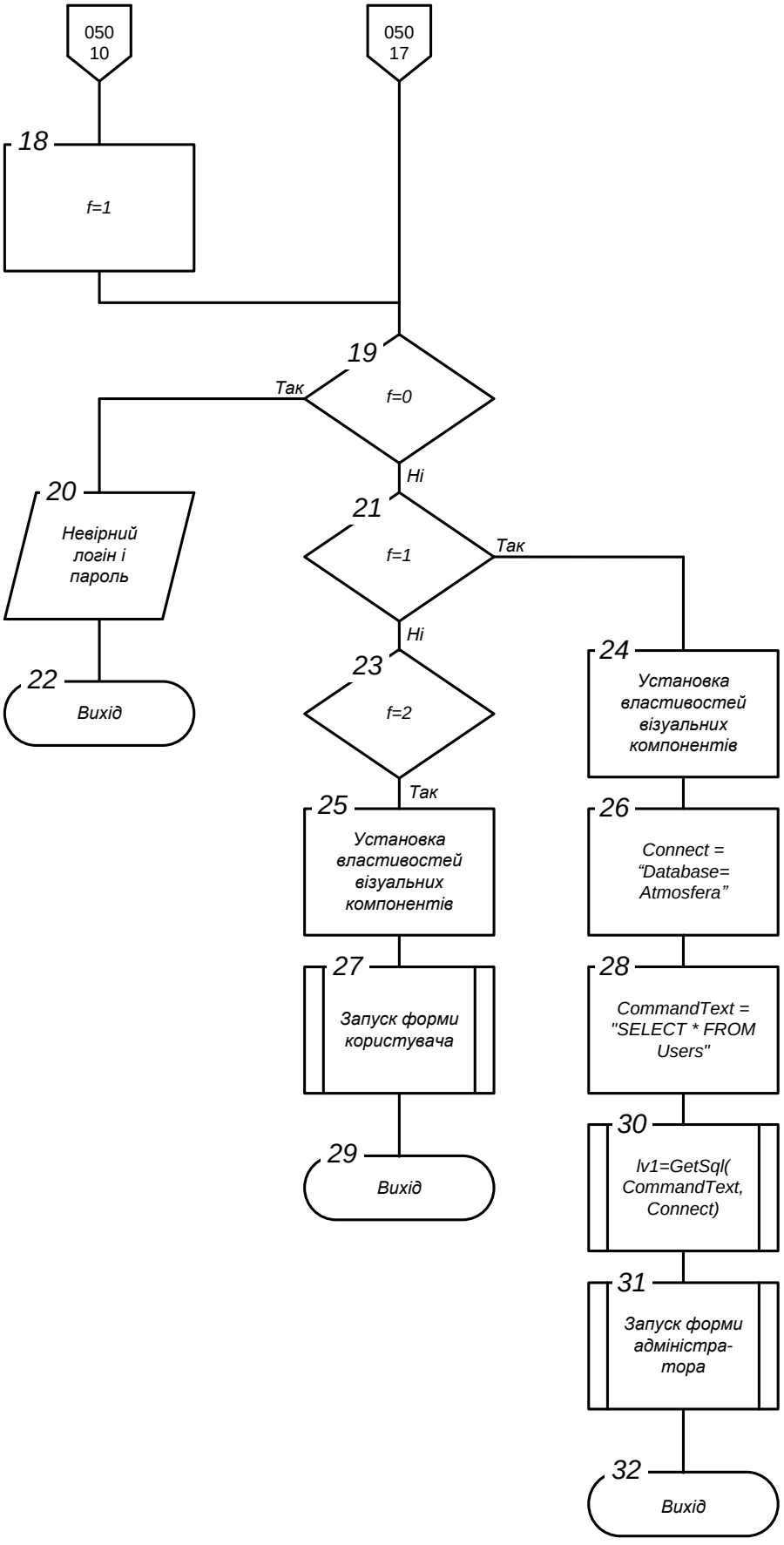


Рисунок 3.6 – Алгоритм аутентифікації користувачів



,

Рисунок 3.7 – Алгоритм аутентифікації користувачів

Форма адміністратора для редагування користувачів зображена на рис.3.8.

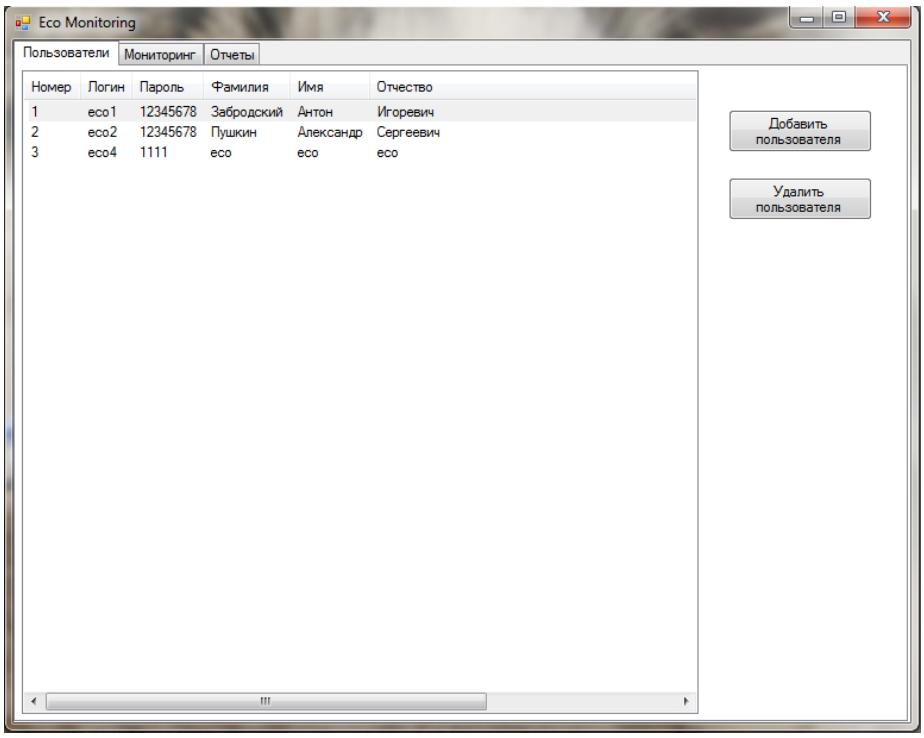


Рисунок 3.8 – Форма адміністратора для редагування користувачів

Форма адміністратора для редагування бази даних зображена на рис.3.9

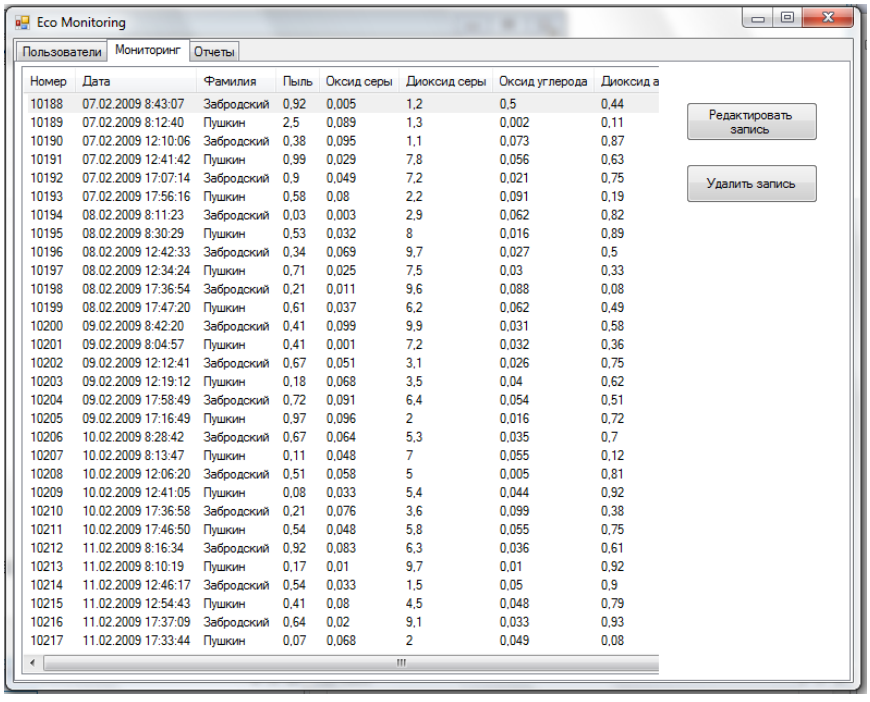


Рисунок 3.9 – Форма адміністратора для редагування бази даних

Форма адміністратора для виконання звітів зображена на рис.3.10.

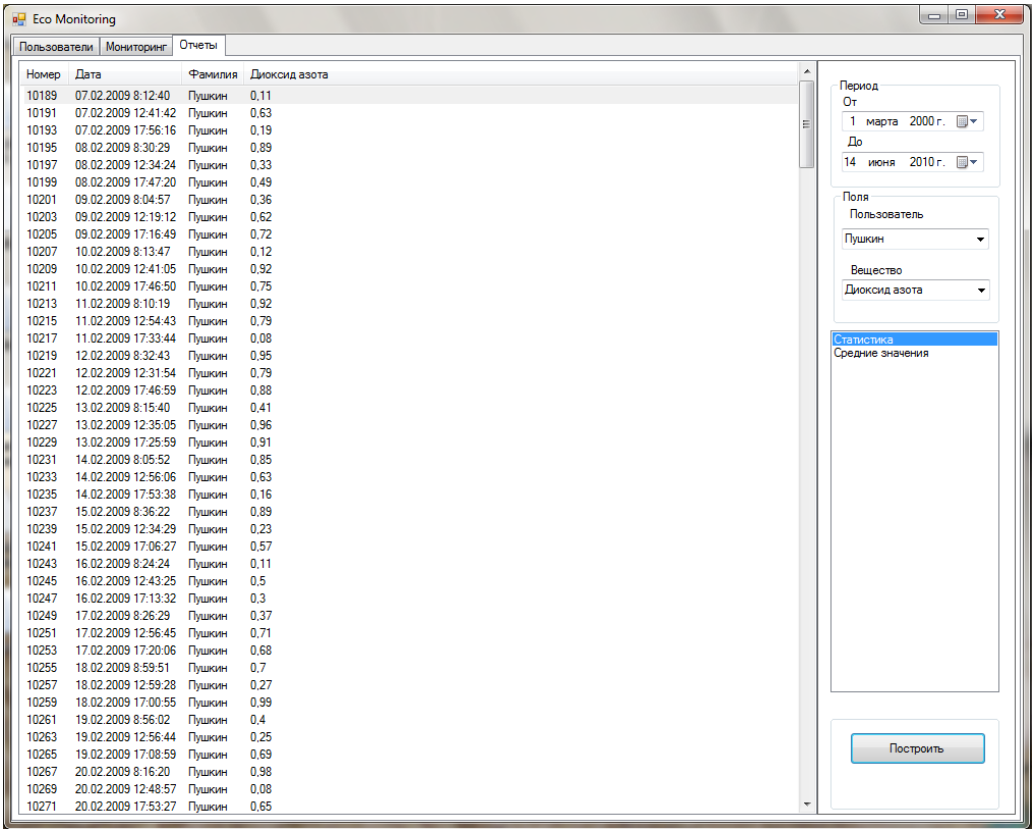


Рисунок 3.10 Форма адміністратора для виконання звітів

Форма користувача для введення даних зображена на рис.3.11.

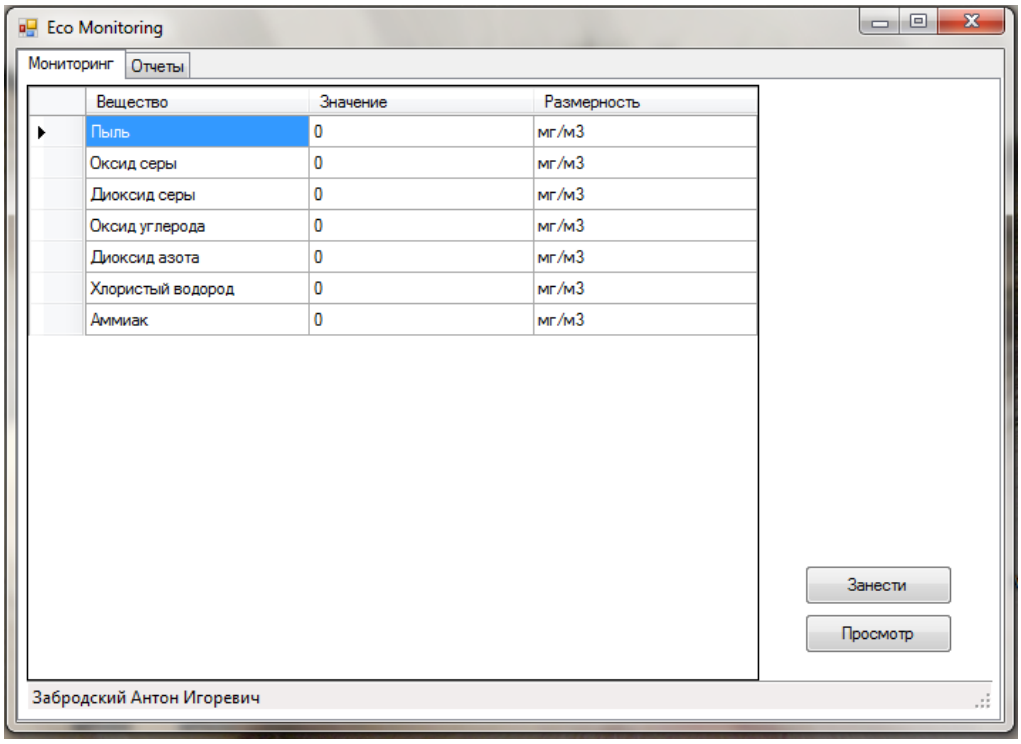


Рисунок 3.11 – Форма користувача для введення даних

Форма користувача для виконання звітів зображена на рис.3.12.

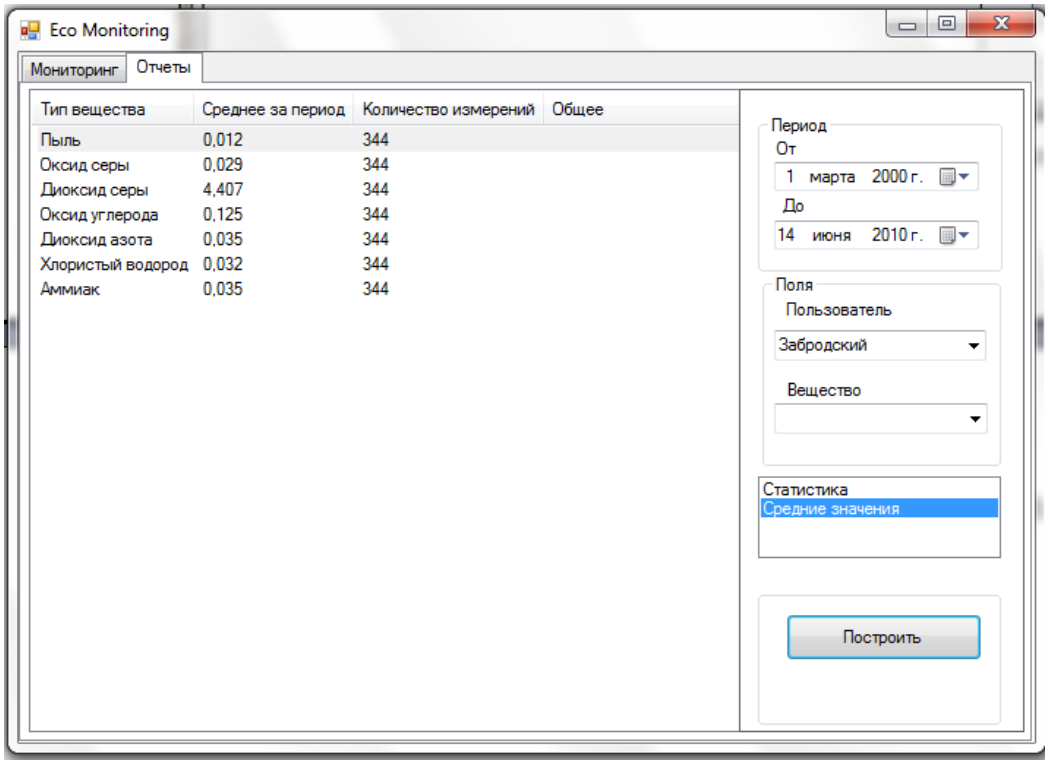


Рисунок 3.12 – Форма користувача для введення даних

Форма користувача для виконання звітів зображена на рис.3.13

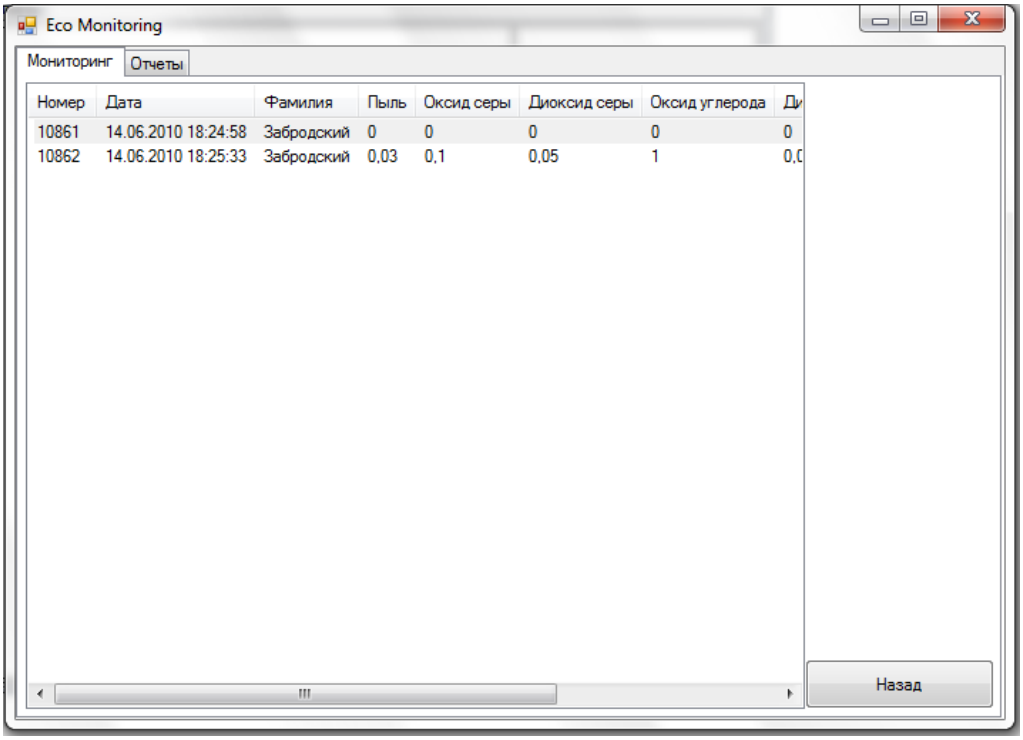


Рисунок 3.13 – Форма користувача для виконання звітів

Алгоритм виконання запиту додавання записів та вибірки користувачів зображений на рис.3.14.

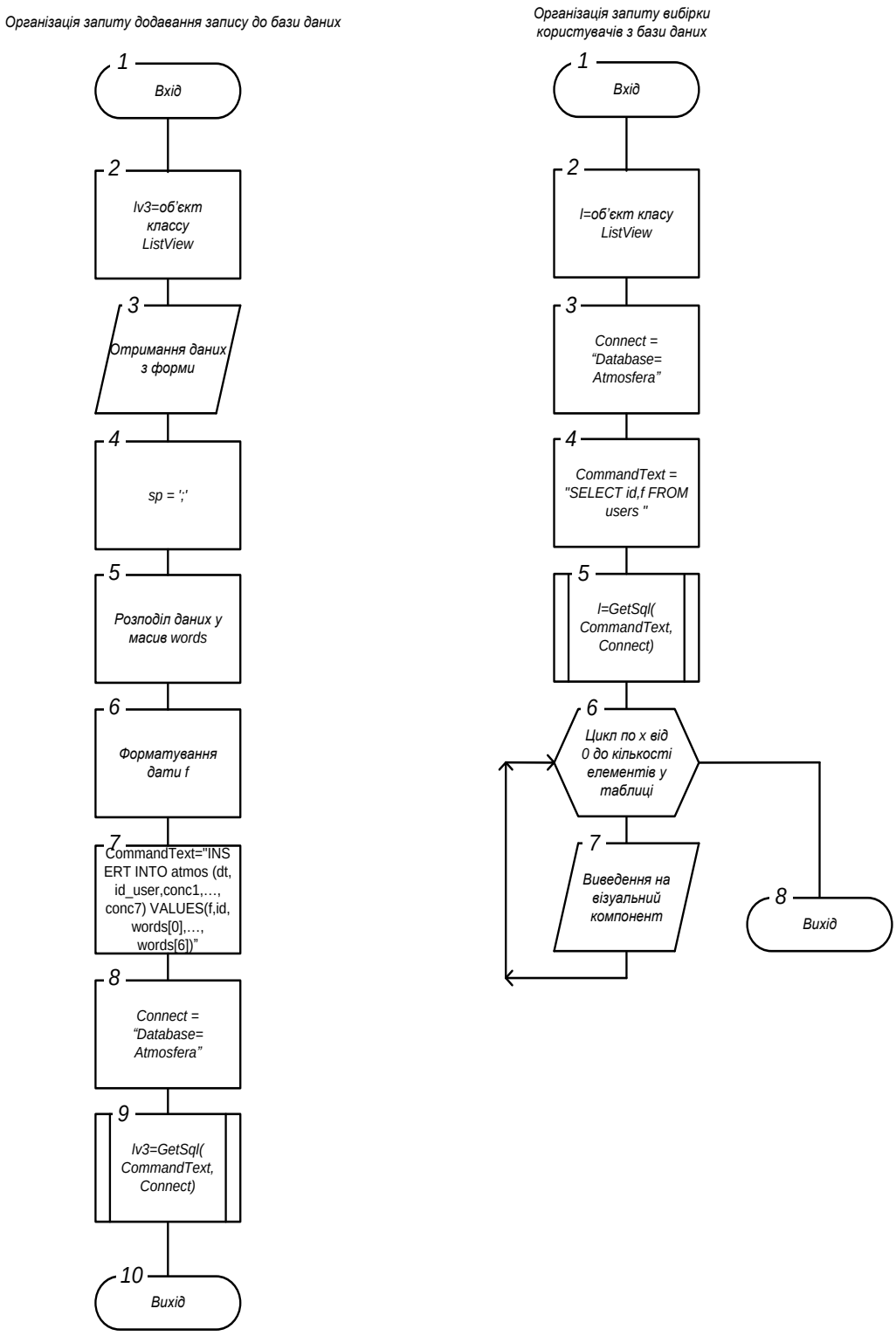


Рисунок 3.14 – Алгоритм виконання запиту додавання записів та вибірки користувачів

Алгоритм з’єднання з базою даних зображений на рис.3.15.

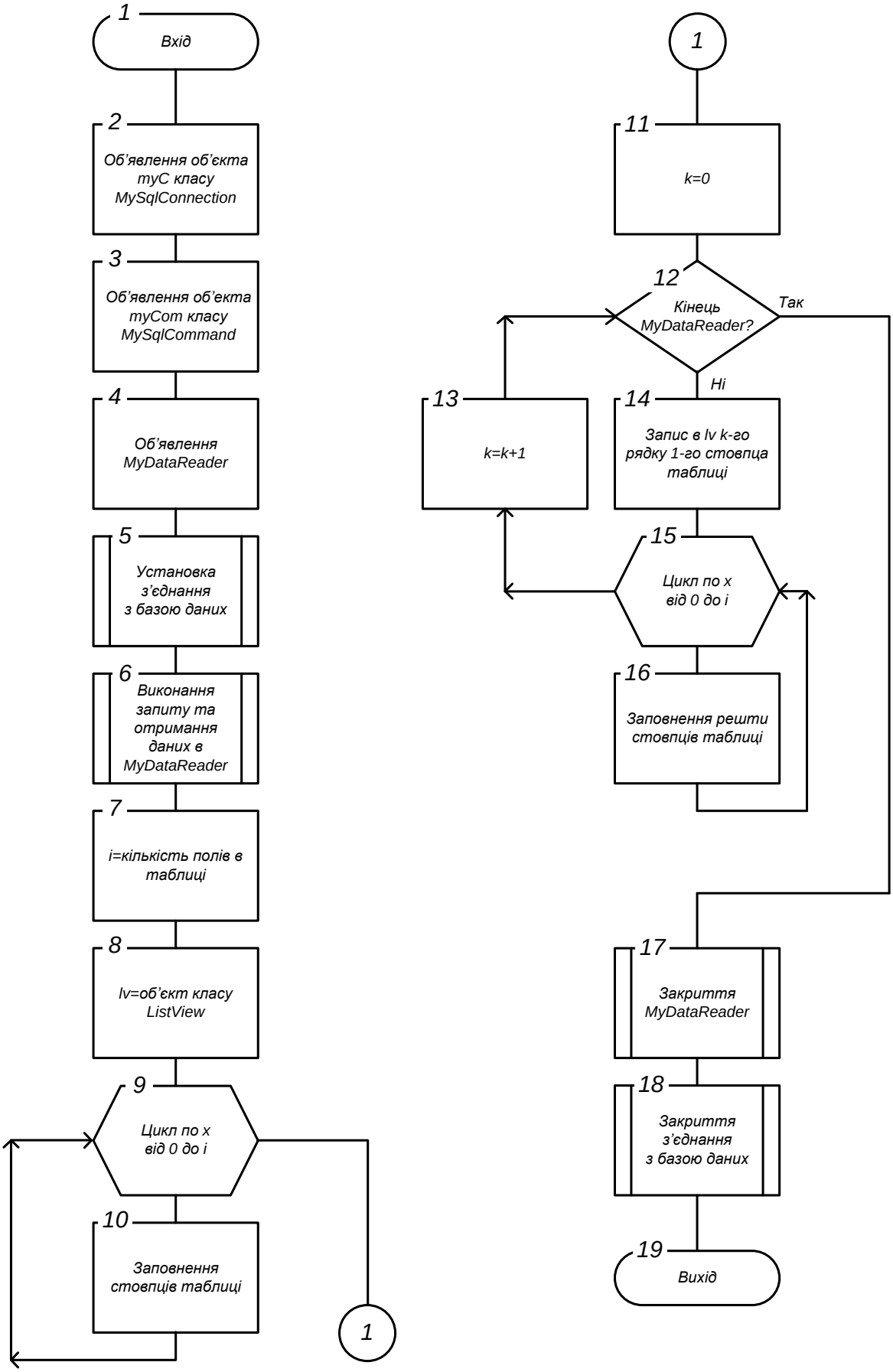


Рисунок 3.15 – Алгоритм з’єднання з базою даних

Форма адміністратора для додавання користувачів зображена на рис.3.16

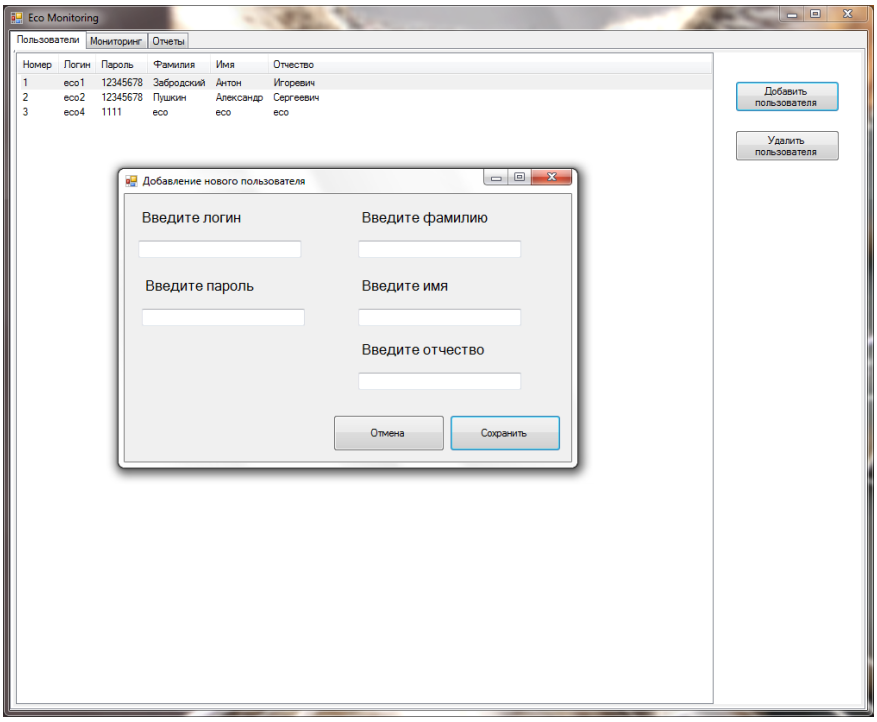


Рисунок 3.16 – Форма адміністратора для додавання користувачів

Форма адміністратора для редагування бази даних зображена на рис.3.17.

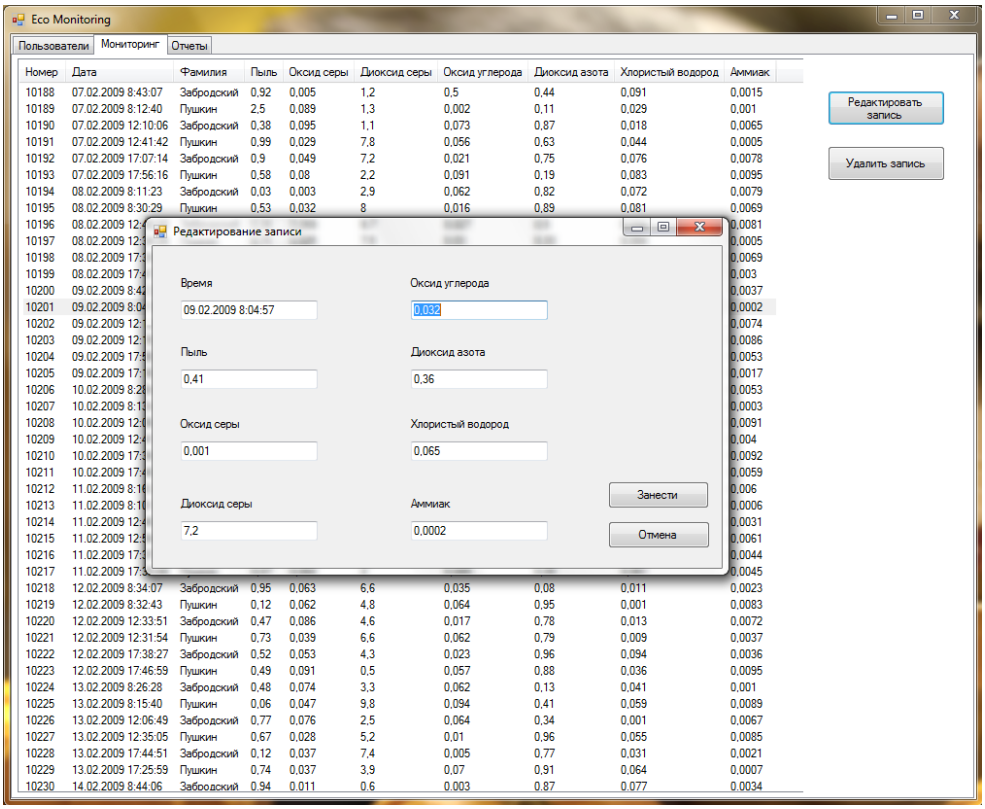


Рисунок 3.17 – Форма адміністратора для редагування бази даних

Алгоритм видалення користувачів з бази даних зображений на рис.3.18.

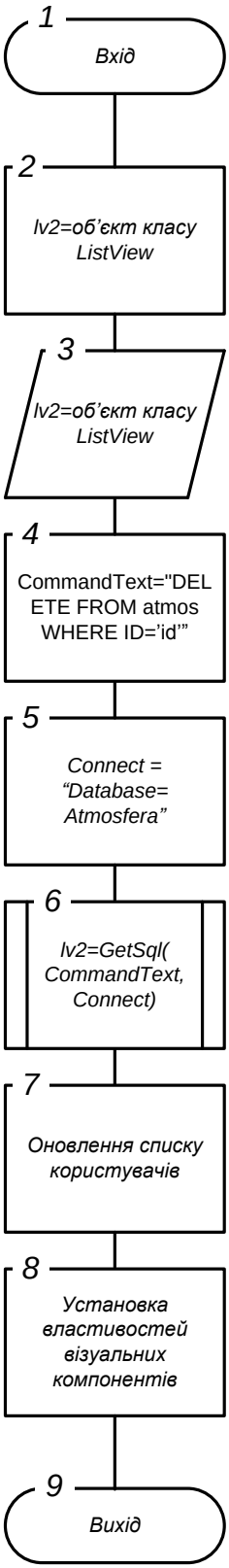


Рисунок 3.18 – Алгоритм видалення користувачів з бази даних

3.2.4 Програмний код загальних функцій системи бази даних екологічного моніторингу

Функція з'єднання з базою даних та виконання запиту GetSql():

```
public ListView GetSql(string CommandText, string Connect)
    {
        MySqlConnection myC = new MySqlConnection(Connect);
        MySqlCommand myCom = new MySqlCommand(CommandText, myC);

        MySqlDataReader MyDataReader;

            myC.Open(); //Устанавливаем соединение с базой данных.


            MyDataReader = myCom.ExecuteReader();

            int i = MyDataReader.FieldCount;

            ListView lv = new ListView();

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

                lv.Columns.Add(MyDataReader.GetName(x));
            };

            int k=0;

            while (MyDataReader.Read())
            {
                lv.Items.Add(MyDataReader.GetValue(0).ToString());
                for (int x=1;x<i;x++){

                    lv.Items[k].SubItems.Add(MyDataReader.GetValue(x).ToStri
                    ng());
                };

                k++;
            };

            MyDataReader.Close();
            myC.Close(); //Обязательно закрываем соединение!

            return lv;
     }
```

Функція додавання користувачів до бази даних:

```
  public void SaveUser(string line1)
  {
      if (line1 != null)
      {
          string s;
          char sp = ';';
          s = line1;
          string[] words = s.Split(sp);
```

```
            CommandText = "INSERT INTO Users (login,password,f,i,o)
            VALUES('" + words[0] + "','" + words[1] + "','" + words[2] +
            "','" + words[3] + "','" + words[4] + "')";
            Connect = "Database=Atmosfera;Data Source=" + line + ";User
            Id=root;Password=test";
            lv1 = GetSql(CommandText, Connect);

            CommandText = "SELECT * FROM Users";
            Connect = "Database=Atmosfera;Data Source=" + line + ";User
             Id=root;Password=test";

            lv1 = GetSql(CommandText, Connect);

            Setlv(ref lv1,ref panel4);
            SetFields(ref lv1, list1, 6);
        }
    }
```

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 [10] 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 правил 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 [11], 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 [12]:

– 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;

$Z_т$ - повний опір струму короткого замикання на корпус, Ом.

Згідно таблиці 4 [13]: $Z_т/3 = 0,1$ Ом.

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

$$Zп = \sqrt{Rп^2 + Xп^2}, \quad (4.2)$$

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 [18], 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 Iн - номінальний струм спрацьовування плавкої вставки, А;

k - коефіцієнт кратності нелінійного струму Iн, А.

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

$$Ін = P / U, \quad (4.4)$$

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 [14], 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 [15] 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 [16] 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 [17].

Стандарт [17] 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 [18].

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 [24] 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:

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

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, $n = (200 \cdot 120 \cdot 1{,}2 \cdot 1{,}5)/(3120 \cdot 0{,}6 \cdot 2) = 12$.

Виходячи з цього, рекомендується використовувати 12 світильників. Світильники слід розміщувати рядами, бажано паралельно стіні з вікнами. Схема розташування світильників зображена на рис. 4.1.

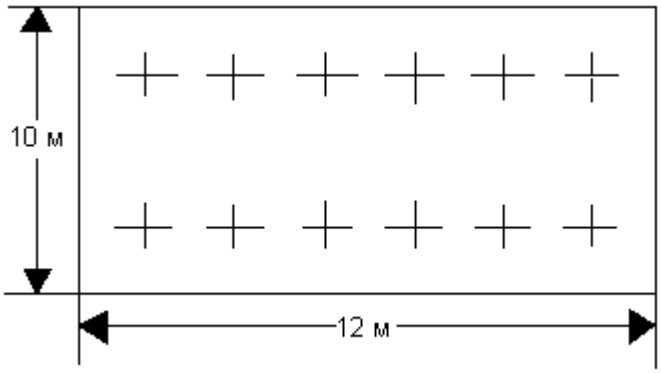


Рисунок 4.1 - Схема розташування світильників

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 [11].

Згідно [12] 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 [18] 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;

– підтримка температури нагріву поверхні машин, механізмів, устаткування, пристроїв, речовин і матеріалів, які можуть увійти до контакту з пальним середовищем, нижче гранично допустимої, становить 80% якнайменшої температури самозаймання пального.

– 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 [19] 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) [20]. 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.

ВИСНОВКИ

У дипломній роботі було досліджено існуючі методи та технології розробки програмного забезпечення регіональної системи екологічного моніторингу.

В роботі проведений аналіз існуючих технологій розробки програмних систем і розробка по декількох з них, де враховувалися різні параметри: тип інформації, безпека, особливості розробки програмного забезпечення, надійність і здатність відповідати необхідним характеристикам. Розроблено ряд алгоритмів для забезпечення безпечної передачі інформації в системі, алгоритми статистичного та екологічного аналізу, підсистеми ухвалення рішень. Для реалізації програмного забезпечення використано Microsoft Visual Studio 2017 та бібліотеки ATL і STL.

У першому та другому розділі відображено короткий екологічний огляд, загальні визначення, розрахунки основних показників, обернена увага на різні особливості і аспекти побудови і аналізу систем екологічного моніторингу, обґрунтовано технічне завдання і показані можливі варіанти його вирішення.

У третьому розділі досліджені існуючі методи та технології розробки програмного забезпечення, а також недоліки і переваги кожного з них.

Програмне забезпечення було спроектовано за допомогою мови C#, далі обґрунтовано та обрано структуру бази даних з можливих варіантів її реалізації. Програмне забезпечення розроблено на основі технологій .NET та MySql. Проведено порівняльний аналіз отриманих результатів та зроблено висновки.

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.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1) http://www.mosecom.ru/about/mosecom/

2) http://doneco.org.ua/showwork.php?id=4

3) Пітрек М. Секрети системного програмування в Windows - К.: Діалектика, 1996. - 448 с., мул.

4) Кейт Г. Іспользованіє Visual C#. Спеціальне видання.: Пер. з англ. - К.: Діалектика 1997, - 816 с.: мул.

5) Бьерн Страуструп. Мова програмування C#. Третє видання./Пер. з английского:- Спб.;- М.: «Невський Діалект»-«іздательство БІНОМ»-2000 р. - 991 с.

6) Майкл Дж. Янг: Visual C#. Повне керівництво: Пер. з англ. - К.: Видавнича група BHV, 2000 р. - 1056 с.

7) Дональд Бокс: Суть языка Java/css . Бібліотека програміста. - Спб.: Пітер, 2001. - 400 с.: мул.

8) С. Маклін, Дж. Нафтел, К. Уїльямс : Microsoft .NET Framework 2.0. «Русская редакция», 2003 - 456с.: мул.

9) Щербаков Е.В., Щербакова М.Е., Охрамовіч В.К.: Автоматизированное проектирование ППО КСУ на базе пакета программ “КВАРЦ” «Наукове видання»,2003-200с.:ил.

10) Державний стандарт України. ГОСТ 12.1.005-88. Общие санитарно-гигиенические требования к воздуху рабочей зоны.

11) Державний стандарт України. ГОСТ 12.1.005-88. Общие санитарно-гигиенические требования к воздуху рабочей зоны.

12) Державний стандарт України. ГОСТ 12.0.003-74 Опасные и вредные производственные факторы. Классификация.

13) Нормативно-правові акти з охорони праці. НПАОП 40.1-1.21-98. Правила безпечної експлуатації електроустановок споживачів

14) Державний стандарт України. ГОСТ 12.1.009-76. ССБТ. Электробезопасность. Термины и определения.

15) Державні санітарні норми України. ДСП 173-96. Державні санітарні правила планування та забудови населених пунктів.

16) Державні санітарні норми України. ДСН 3.3.6.042-99 Санітарні норми мікроклімату виробничих приміщень.

17) Державний стандарт України. ГОСТ 12.1.005-88. Система стандартов безопасности труда. Общие санитарно-гигиенические требования к воздуху рабочей зоны.

18) TCO’ 07 Certified Displays. © 2007 Copyright TCO Development AB

19) Державні санітарні норми і правила. ДСанПіН 3.3.2.007-98, Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин.

20) Державні будівельні норми ДБН В.2.5-28-2006. Природне і штучне освітлення

21) Державний стандарт України. ГОСТ 12.1.044-89 Система стандартов безопасности труда. Пожаровзрывоопасность веществ и материалов. Номенклатура показателей и методы их определения.

22) Нормативні акти пожежної безпеки. НАПБ Б.03.002-2007. Норми визначення категорій приміщень, будинків та зовнішніх установок за вибухопожежною та пожежною небезпекою.

23) Державний стандарт України. ГОСТ 12.1.004-91. "Система стандартов безопасности труда. Пожарная безопасность. Общие требования".

24) Нормативні акти пожежної безпеки. НАПБ А.01.001-2014 “Правила пожежної безпеки в Україні”

25) Нормативні акти пожежної безпеки. НАПБ Б.03.001-2004. Про затвердження Типових норм належності вогнегасників.

ДОДАТОК А.

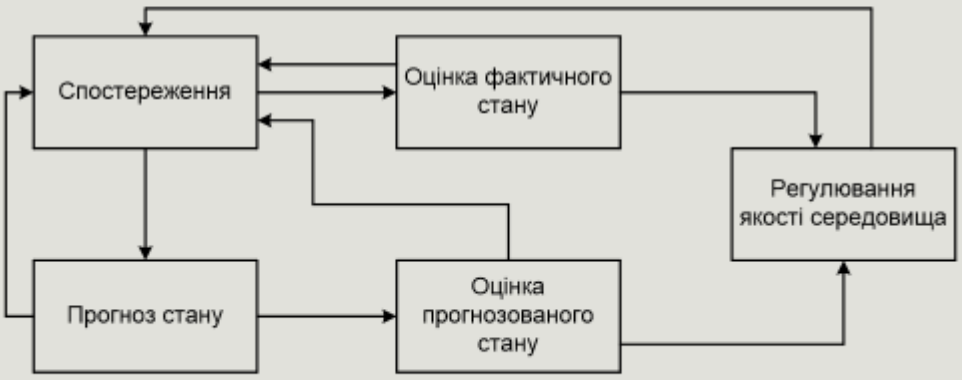
Електронні плакати

Міністерство освіти і науки України
Східноукраїнський національний університет імені Володимира Даля
Кафедра комп'ютерних наук та інженерії

ПРОГРАМНЕ ТА АПАРАТНЕ ЗАБЕЗПЕЧЕННЯ СИСТЕМИ РЕГІОНАЛЬНОГО ЕКОЛОГІЧНОГО МОНІТОРИНГУ

Безкоровайний Р.В. Рязанцев О.І.

Схема моніторингу



Галузь та мета розробки

Галузь роботи: розробка розподіленої бази даних регіональної системи екологічного моніторингу.

Мета розробки даної системи:

- створення програмного забезпечення, що здійснює збір інформації, дозволяє отримати об'єктивну оцінку про стан повітря міста у будь-який час;
- аналіз моніторингової інформації і генерація звітів.

Завдання

Необхідно розробити розподілену базу даних регіональної системи екологічного моніторингу.

Призначення: збір оперативної інформації про стан атмосферного повітря в місті і відображення її в зручному вигляді; підтримка статистичної, математичної, спеціалізованої обробки.

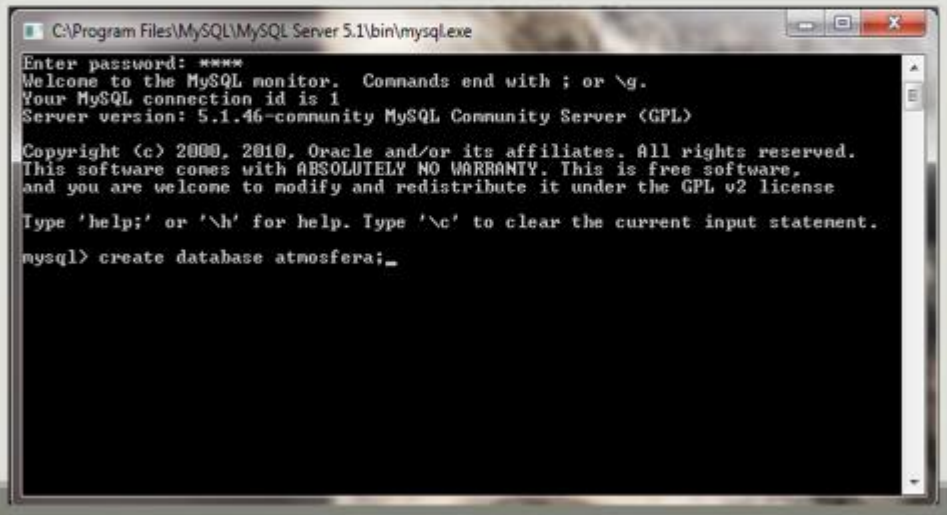
Вибір засобів розробки

Microsoft Visual C#

APACHE HTTP SERVER

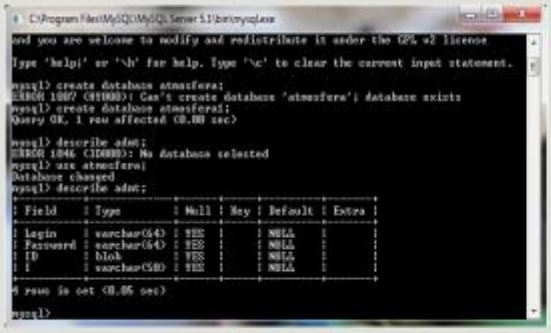
MySQL®

Створення БД

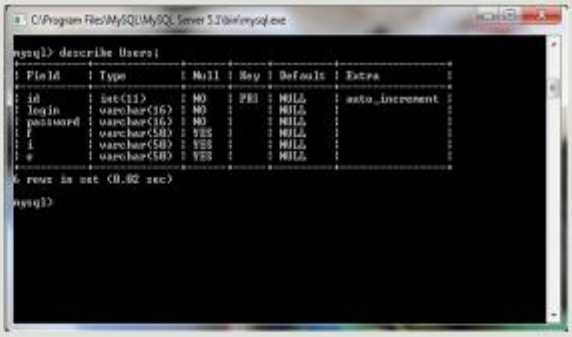


Створення таблиць

Створення таблиці admt

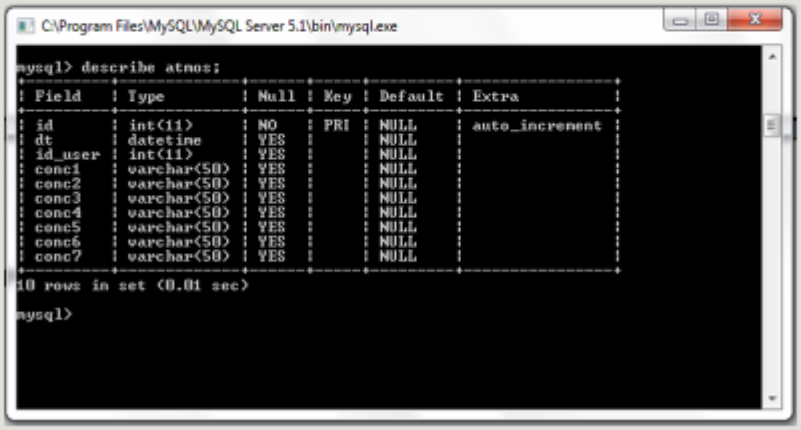


Створення таблиці Users



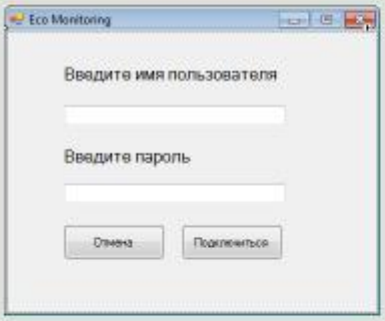
Створення таблиць

Створення таблиці atmos

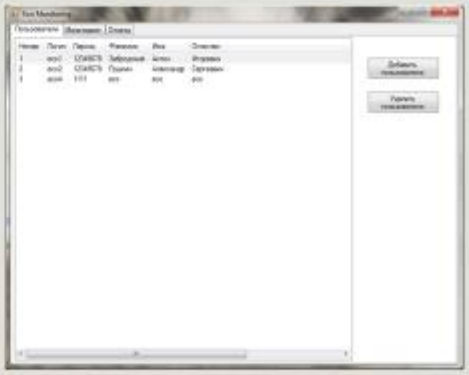


Інтерфейс програми

Форма аутентифікації користувачів

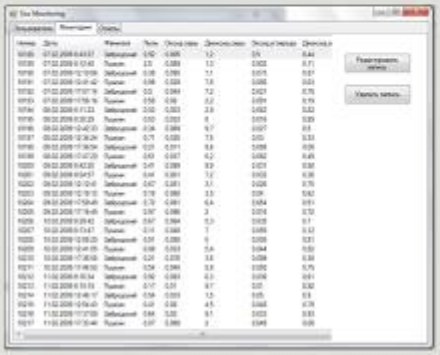


Форма адміністратора для редагування користувачів

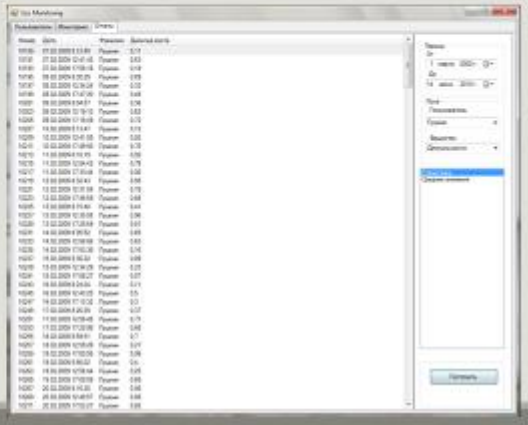


Інтерфейс програми

Форма адміністратора для редагування бази даних

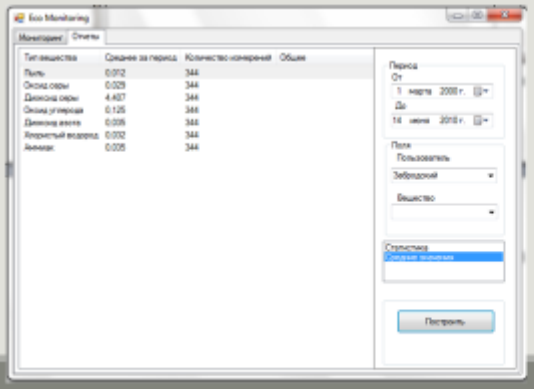


Форма користувача для введення даних

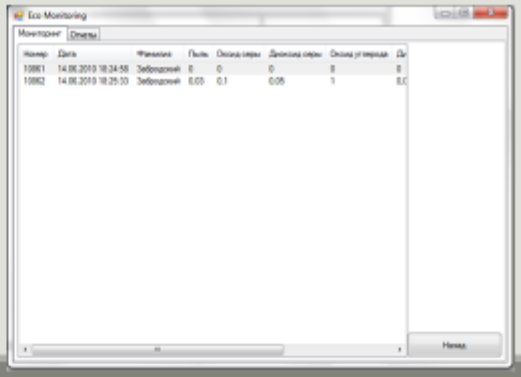


Інтерфейс програми

Форма користувача для введення даних

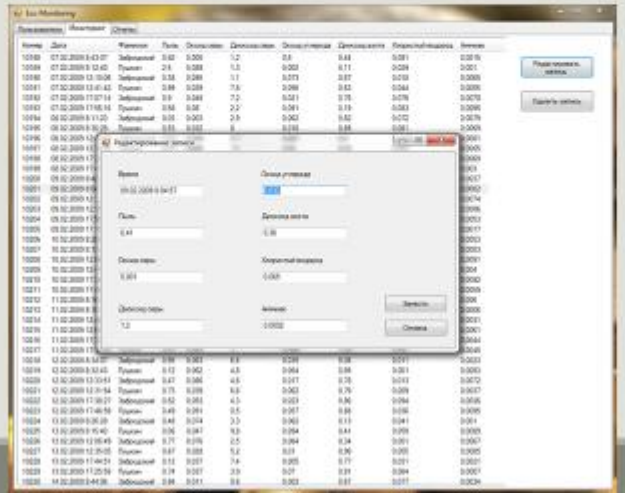


Форма користувача для виконання звітів



Інтерфейс програми

Форма адміністратора для редагування бази даних



ВИСНОВКИ

У дипломній роботі було досліджено існуючі методи та технології розробки програмного забезпечення регіональної системи екологічного моніторингу.

В роботі проведений аналіз існуючих технологій розробки програмних систем і розробка по декількох з них, де враховувалися різні параметри: тип інформації, безпека, особливості розробки програмного забезпечення, надійність і здатність відповідати необхідним характеристикам. Розроблено ряд алгоритмів для забезпечення безпечної передачі інформації в системі, алгоритми статистичного та екологічного аналізу, підсистеми ухвалення рішень. Для реалізації програмного забезпечення використано Microsoft Visual Studio 2017 та бібліотеки ATL і STL.

ВИСНОВКИ

У першому та другому розділі відображено короткий екологічний огляд, загальні визначення, розрахунки основних показників, обернена увага на різні особливості і аспекти побудови і аналізу систем екологічного моніторингу, обґрунтовано технічне завдання і показані можливі варіанти його вирішення.

У третьому розділі досліджені існуючі методи та технології розробки програмного забезпечення, а також недоліки і переваги кожного з них.

Програмне забезпечення було спроектовано за допомогою мови C#, далі обґрунтовано та обрано структуру бази даних з можливих варіантів її реалізації. Програмне забезпечення розроблено на основі технологій .NET та MySql. Проведено порівняльний аналіз отриманих результатів та зроблено висновки.

ВИСНОВКИ

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.