МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«РОЗРОБКА ВІДЖЕТУ МОНІТОРИНГУ СТАНУ АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ ПЕРСОНАЛЬНОГО КОМП`ЮТЕРА»**

Виконав: студент 2 курсу, групи 8.1228–з
спеціальності 122 комп'ютерні науки
(шифр і назва спеціальності)
освітньої програми комп'ютерні науки
І.О. Дудко
(ініціали та прізвище)
Керівник доцент кафедри комп'ютерних наук, доцент, к.т.н. Решевська К.С.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)
Рецензент декан математичного факультету, професор, д.т.н С.І. Гоменюк
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра комп'ютерні науки

Рівень вищої освіти магістр

Спеціальність 122 комп'ютерні науки

(шифр і назва)

Освітня програма комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри комп`ютерних наук, к.т.н., доцент
_____ Борю С. Ю.
(підпис)

_____ » _____2020 р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТА

Дудку Ігорю Олександровичу

(прізвище, ім'я та по–батькові)

1. Тема роботи (проекту) Розробка віджету моніторингу стану апаратного забезпечення персонального комп`ютера

керівник роботи (проекту) Решевська Катерина Сергіївна, к.т.н., доцент
(прізвище, ім'я та по–батькові, науковий ступінь, вчене звання)
затверджені наказом ЗНУ від « 29 » 05 2019 року № 812–С

2. Строк подання студентом роботи 27.12.2019

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково–пояснювальної записки (перелік питань, які потрібно розробити)
1. Призначення механізму самоконтролю, аналізу та звітності
2. Середовище реалізації програмного продукту
3. Реалізація вимог програмного продукту

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

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	03.09.19	
2.	Збір вихідних даних.	15.09.19	
3.	Обробка методичних та теоретичних	02.10.19	
	джерел.		
4.	Розробка першого розділу.	13.10.19	
5.	Розробка другого розділу.	05.11.19	
6.	Розробка третього розділу.	16.11.19	
7.	Оформлення та нормо контроль	02.12.19	
	кваліфікаційної роботи.		
8.	Попередній захист кваліфікаційної роботи.	17.12.19	

Студент _____ І. О. Дудко
(підпис) (ініціали та прізвище)

Керівник роботи _____ К.С. Решевська
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О. Г. Спиця
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра «Розробка віджету моніторингу стану апаратного забезпечення персонального комп`ютера»: 78 с., 15 рис., 1 табл., 12 джерел, 4 додатки.

МОНІТОРИНГ, ПРОГРАМА, ТЕСТ АПАРАТНОГО СТАНУ, HDD, S. M. A. R. T.

Об'єкт дослідження – отримання S.M.A.R.T. даних з HDD в комп'ютерних класах ЗНУ.

Мета роботи: розробка віджету моніторингу стану апаратного забезпечення персонального комп'ютера.

Для досягнення мети дипломної роботи поставлено такі завдання:

– розглянути методи аналізу стану ПК;

– розглянути поняття S.M.A.R.T. та методи отримання даних;

– проаналізувати роботу S.M.A.R.T. системи.

На основі виконаних завдань розробити програмний продукт, який дозволить оптимізувати процес моніторингу стану системи персонального комп'ютера.

Методи дослідження – аналітичний.

SUMMARY

Master's Qualification Thesis "Development of a widget for monitoring the state of hardware of a personal computer": 78 page, 15 fig., 1 table., 12 sources, 4 applications.

MONITORING, PROGRAM, HARDWARE TEST, HDD, S. M. A. R. T.

Object of study – obtaining S.M.A.R.T. data from HDD in ZNU computer classes.

Purpos – Develop a PC hardware monitoring widget.

To achieve the goal of the thesis is assigned the following tasks:

– consider methods for PC analysis;

– consider the concept of S.M.A.R.T. and methods for obtaining data;

– analyze the work of S.M.A.R.T. systems.

Based on the completed tasks, develop a software product that will optimize the process of monitoring the status of the personal computer system.

Research Methods – analytical.

ЗМІСТ

ВСТУП

На сьогоднішній день, коли кожен має можливість використовувати комп`ютер виникає необхідність у легкому та швидкому аналізу апаратної частини ПК. Таким чином у таких організаціях як університет постає питання про покращення процесу підтримання комп`ютерів у робочому стані, а насам перед у збереженні інформації. У зв`язку з цим виникло питання про створення програмного забезпечення саме для навчальної установи (університет ЗНУ).

Актуальність теми дипломної роботи полягає в тому, що з постійним збільшенням кількості комп’ютерної техніки виникає необхідність у моніторингу стану компонентів ПК для підтримання всієї системи у належному, робочому стані

Предмет дослідження – спрощення та покращення процесу моніторингу стану ПК в комп’ютерних класах ЗНУ

Об’єкт дослідження – отримання S.M.A.R.T. даних з HDD в комп’ютерних класах ЗНУ.

Мета дипломної роботи: розробка віджету моніторингу стану апаратного забезпечення персонального комп’ютера.

Для досягнення мети дипломної роботи поставлено такі **завдання:**

– розглянути методи аналізу стану ПК;

– розглянути поняття S.M.A.R.T. та методи отримання даних;

– проаналізувати роботу S.M.A.R.T. системи;

на основі виконаних завдань розробити програмний продукт, який дозволить оптимізувати процес моніторингу стану системи персонального комп’ютера.

Новизна полягає у запропонованому методі автоматизації діагностики апаратного забезпечення. Розроблено зручний інтерфейс перевірки та виявлення несправностей комп’ютерних ресурсів. Представлено графічну

оболонку використання методів доступу до системної інформації про стан апаратного забезпечення мови C#

Отримані результати було апробовано в доповіді в конференції «актуальні проблеми математики та інформатики».

1 ПРИЗНАЧЕННЯ МЕХАНІЗМУ САМОКОНТРОЛЮ, АНАЛІЗУ ТА ЗВІТНОСТІ

Велика частина дискових накопичувачів останніх років, функціонує з використанням технології S.M.A.R.T. Скорочення розшифровується як self–monitoring, analysis and reporting technology, що українською звучить як механізм самоконтролю, аналізу та звітності. Її перші розробки побачили світ в 1995 році і з тих пір технологія постійно вдосконалюється. [1]

З моменту виробництва дисковий накопичувач починає зчитувати свій поточний стан, визначаючи його за допомогою спеціальних параметрів або атрибутів. Вони розташовуються в службовій зоні накопичувача, доступ до якої має лише вбудована програма. Переглянути параметри дозволяє окреме ПО, найчастіше представлене утилітами від розробників конкретного жорсткого диска. Через них в накопичувач подаються вводні, після чого в журналі статистики з'явиться інформація про поточний стан диска.

В процесі експлуатації накопичувача, дані представлені в рамках параметрів значення постійно змінюються. Параметри проходять шлях з максимальних показників, які гарантують високу продуктивність і ефективність до мінімальних значень, пов'язаних з високою ймовірністю виходу накопичувача з ладу.

1.1 Програми для перевірки жорсткого диска

Утиліти які можуть стати в нагоді під час аналізу стану жорстких дисків:

а) **Victoria** – одна з найпопулярніших програм для тестування, сервісного обслуговування та допомоги при відновленні інформації з

жорстких дисків. Є версія під DOS і під Windows. Рекомендую використовувати DOS–версію [2];

б) **MHDD** – по суті має майже самі можливості що і victoria, навіть інтерфейс сильно схожий. Запуск з під DOS [3];

в) **HDDScan** – відображення детальної інформації про жорсткий диск, S.M.A.R.T, тестування поверхні. Windows [4];

г) **Scanner** – аналіз використовуваного простору, просто і наочно показує які папки та файли з'їли ваш простір. Windows [5];

д) **CrystalDiskInfo** – програма для відображення і спостереження за показниками S.M.A.R.T жорсткого диска. Windows [6]

е) **HD Tune** – у безкоштовній версії: тестування швидкості читання (розмір блоків можна вказувати), відображення S.M.A.R.T. і деякою інформацією, тестування поверхні. Windows [7].

Функціонал програми Victoria:

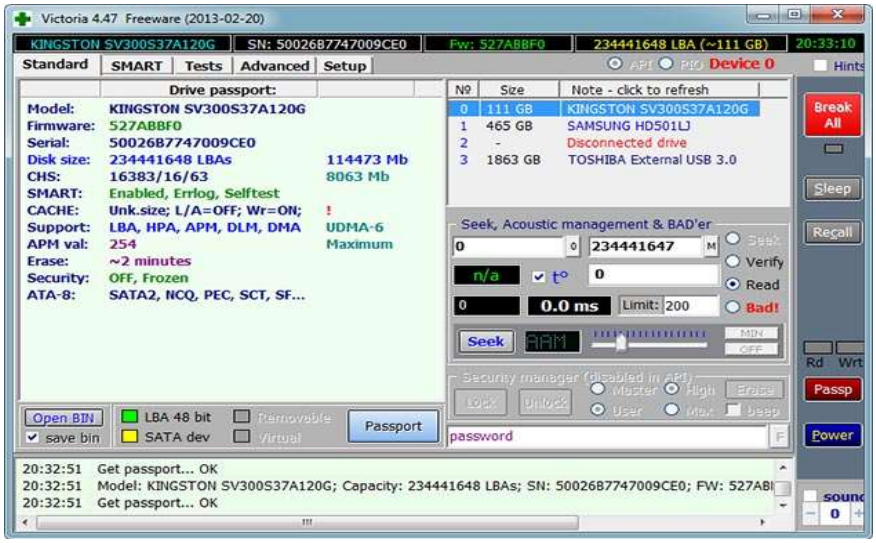


Рисунок 1.1 – Victoria головна вкладка

– прискорити роботу ПК;

– відновити стабільну роботу операційної системи і окремих програм.

Для досягнення таких значних результатів утиліта позитивно впливає на HDD і забезпечує:

– заміну дефектних секторів вінчестера резервними;
– усунення битих секторів;
– ремонт пошкоджених ділянок.

Таким чином, скориставшись цим програмним забезпеченням, можна відновити працездатність накопичувача і продовжити термін його служби.

Функціонал програми MHDD:

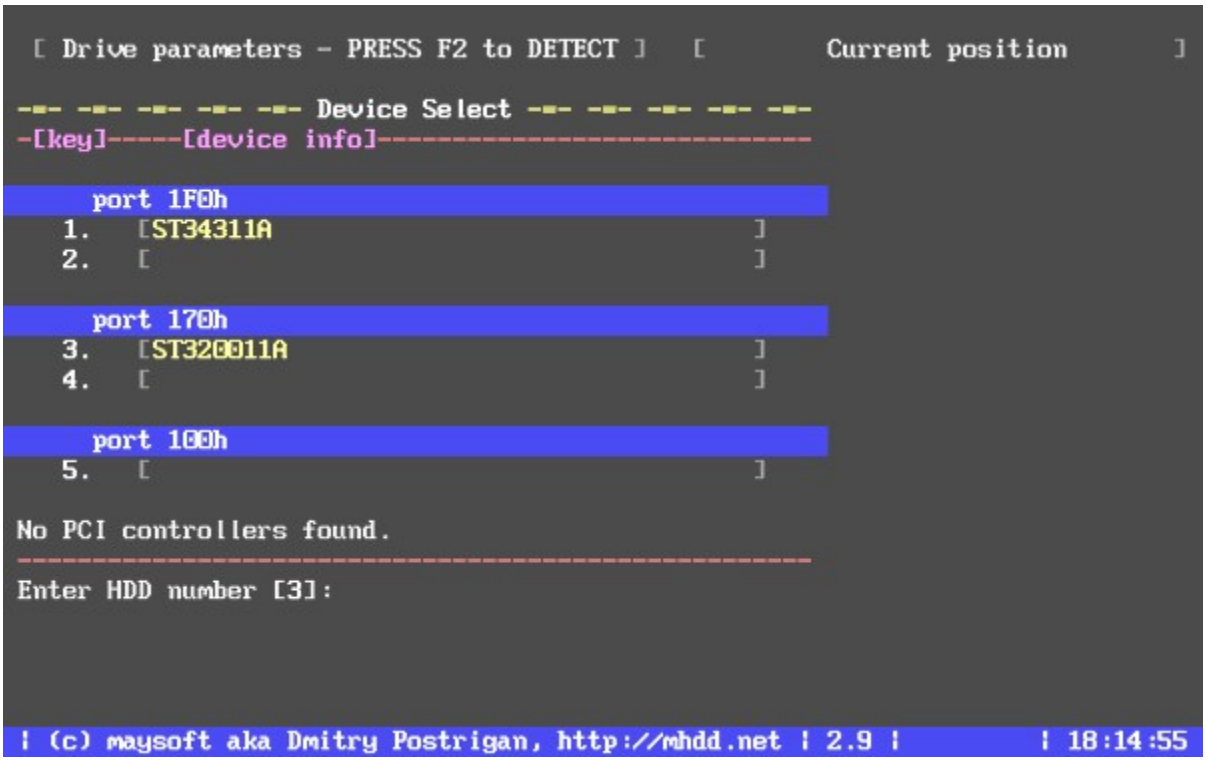


Рисунок 1.2 – Робоче поле MHDD

– точна діагностика механічної поверхні диска;
– моніторинг регістрів IDE контролера;
– якісне використання функції HPA (зменшення обсягу вінчестера);
– регулювання шуму, видаваного жорстким диском;
– ведення журналу помилок;
– наявність функції повного знищення даних на вінчестері без можливості відновлення;
– тестування HDD в екстремальних умов;

– можливість одночасного тестування декількох вінчестерів;

– робота з системою паролів на диску.

Функціонал програми HDDScan:

Рисунок 1.3 – HDDScan головна панель

– підтримуються жорсткі диски з інтерфейсом IDE / SATA / SCSI, RAID–масиви, зовнішні USB / FireWire–накопичувачі, SSD і флеш–карти;

– 4 режими тестування накопичувачів (лінійної верифікації, лінійного читання, лінійного запису, читання Butterfly – штучний тест випадкового читання);

– читання і аналіз S.M.A.R.T. – параметрів;

– запуск S.M.A.R.T.–тестів на накопичувачах з інтерфейсом ATA / SATA / USB / FireWire;

– моніторинг температури на накопичувачах з інтерфейсом ATA / SATA / USB / FireWire / SCSI;

– зміна спеціальних налаштувань (система розподілу, старт / стоп шпинделя, регулювання акустичного режиму і т. П.);

– збереження і друк звітів.

Функціонал програми Skanner:

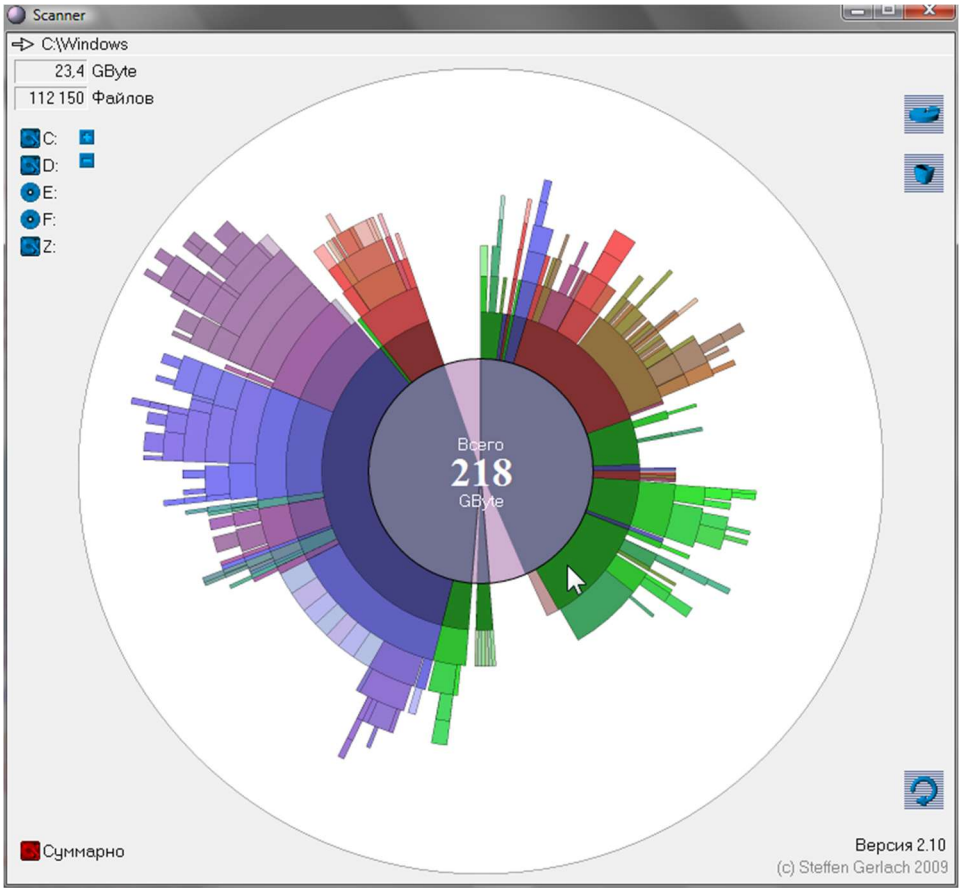


Рисунок 1.4 – Інтерфейс Skanner

– аналіз вмісту жорстких дисків;
– наочна діаграма всіх папок і файлів, що знаходяться в корені аналізованого диска;
– швидке видалення виявлених непотрібних файлів і папок;
– працює без інсталяції.

Функціонал програми Crystaldiskinfo

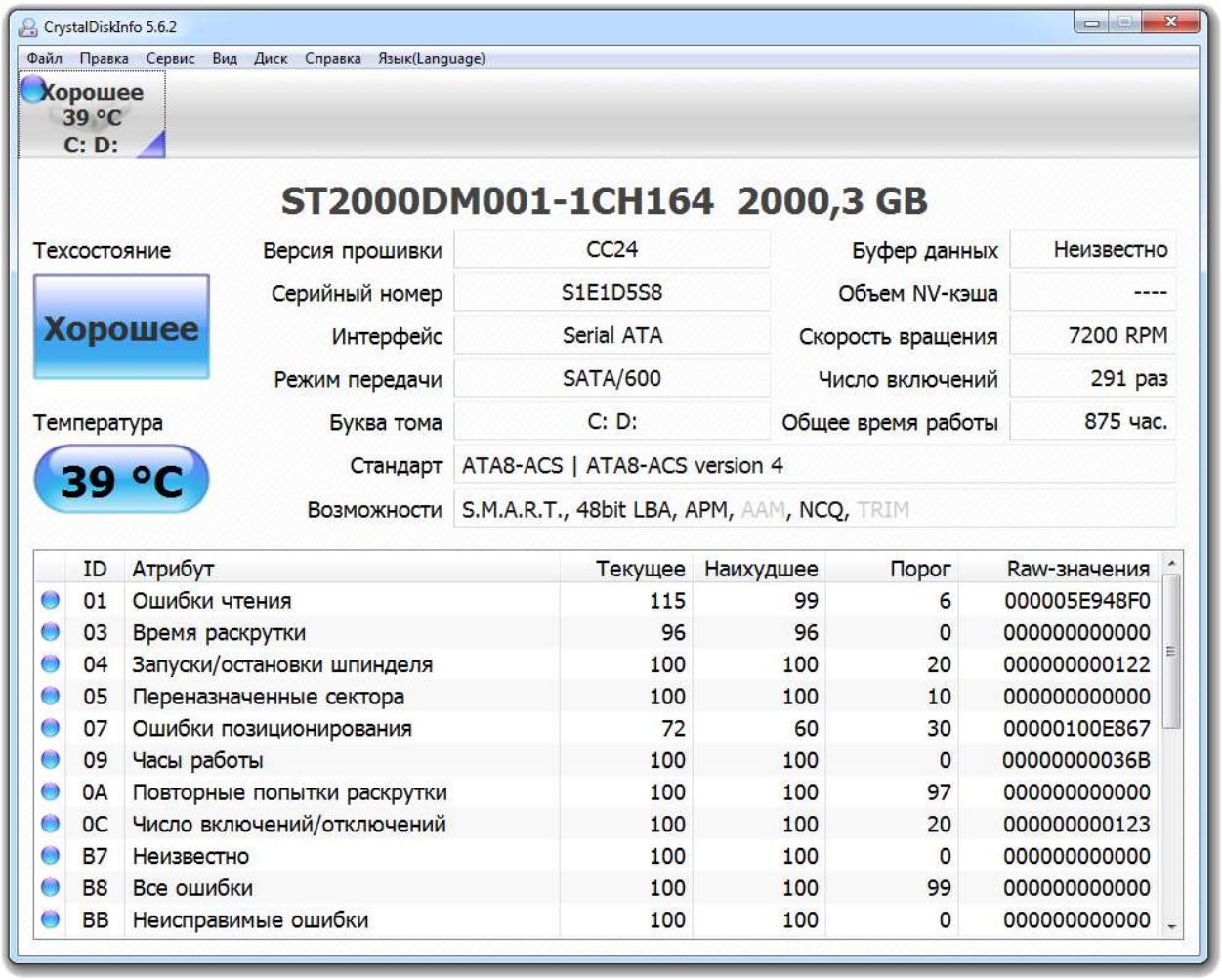


Рисунок. 1.5 – Інтерфейс Crystaldiskinfo

– підтримка RAID;

– робота із зовнішніми USB–дисками;

– побудова графіків роботи дисків;

– моніторинг температури і стану S.M.A.R.T;

– налаштування енергозбереження (APM) і шумозаглушення (AAM);

– повідомлення за допомогою звукових сигналів або по електронній пошті.

Функціонал програми HD Tune:

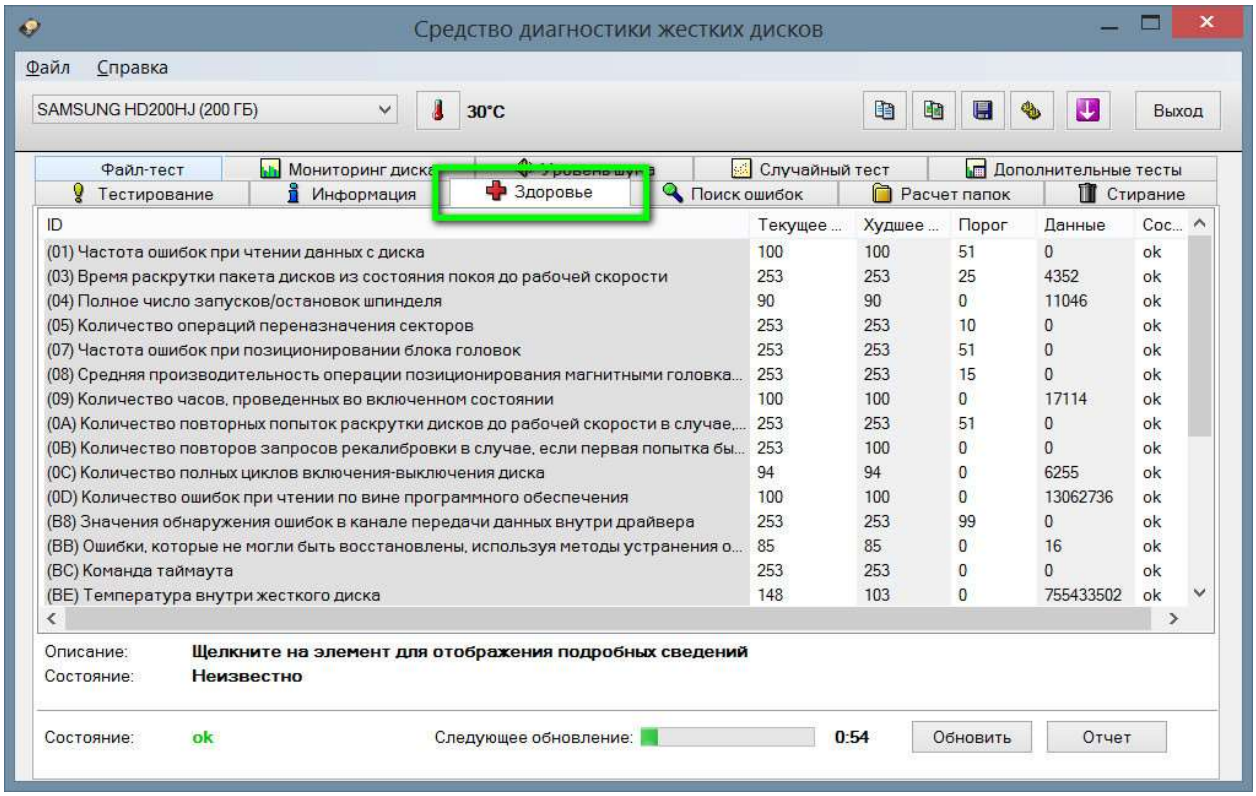


Рисунок 1.6 – Інтерфейс HD Tune

– тест продуктивності накопичувача інформації;
– тест кешу;
– відображення детальної інформації протестованого пристрою;
– сканування поверхні жорсткого диска на наявність помилок;
– перевірка стану жорсткого диска з використанням технології S.M.A.R.T;
– відображення інформації про розділи накопичувача.

1.2 Спостереження за апаратною складовою жорсткого диску програмним шляхом

Надійність жорстких дисків далека від ідеалу – рано чи пізно вони виходять з ладу, що може спричинити за собою серйозні проблеми, аж до повної втрати даних, які зберігаються на них. Звичайно, в більшості випадків інформацію (повністю або частково) на зовні «мертвих» жорстких дисках можна відновити, але краще спробувати запобігти виникненню подібних ситуацій.

У цьому плані найбільш надійним є регулярне резервне копіювання інформації. Однак навіть при скрупульозному підході до задачі резервування можна не встигнути скопіювати важливу інформацію, адже вихід диска з ладу завжди відбувається несподівано. Разом з тим уникнути небезпеки втрати даних можна, якщо контролювати стан жорсткого диска за допомогою спеціалізованої утиліти S.M.A.R.T.–моніторингу. Кілька подібних програм ми і розглянемо в далі

1.3 Механізму самоконтролю, аналізу та звітності –моніторинг

Сьогодні всі сучасні HDD–диски підтримують технологію самодіагностики дисків – S.M.A.R.T. (Self–Monitoring Analysis and Reporting Technolodgy), яка була спеціально розроблена для своєчасного виявлення майбутнього виходу накопичувача з ладу. В основу цієї технології покладено безперервний моніторинг показань спеціальних сенсорів. Дані сенсори відображають поточні значення S.M.A.R.T.–параметрів, кожен з яких показує стан певної життєво важливої частини жорсткого диска (кількість помилок читання або запису, температуру, час роботи диска, продуктивність, швидкість пошуку інформації тощо). Значення параметрів при нормальній роботі диска можуть варіюватися в тих чи інших інтервалах. При цьому для

будь–якого параметра виробником визначено якесь порогове значення – мінімальне безпечне значення, яке не може бути перевищений при нормальних умовах експлуатації [8].

Утиліти S.M.A.R.T.–моніторингу регулярно сканують жорсткі диски, витягають S.M.A.R.T.–інформацію з сенсорів і термодатчиків (датчиків температури, якими оснащені всі сучасні жорсткі диски), аналізують її і стежать за зміною стану всіх атрибутів. При виявленні критичних змін, які вказують на істотне падіння надійності диска, програми інформують користувача про те, що зберігання даних на жорсткому диску стало небезпечним. За запевненнями ряду розробників, це відбувається не пізніше, ніж за день–два до виходу з ладу жорсткогодиску, що забезпечує користувачеві деякий резерв часу, за який можна встигнути зробити копії всієї інформації, а може бути, навіть замінити жорсткий диск.

1.4 Признаки виходу жорсткого диску з ладу

Велика частина жорстких дисків виходить з ладу протягом 8–10 років, а з першими проблемами в їх роботі доводиться стикатися після чотирьох років експлуатації. Втім, термін служби HDD залежить від безлічі факторів: кількості циклів перезапису, ступеня фрагментації, умов експлуатації, наявності виробничих дефектів і так далі. Так чи інакше, але рано чи пізно з ладу повинен вийти всякий жорсткий диск, навіть той, який використовується виключно для зберігання даних.

Головне не чекати до останнього, а при наближенні вінчестера до свого кінця перенести з нього всі важливі дані на новий і здоровий диск. Тільки ось як визначити, коли настане цей самий кінець? Передбачити його з точністю до хвилини навряд чи можливо, але є явні і непрямі ознаки, що вказують на близьку поломку HDD [10].

Поява сторонніх шумів:

Поява в роботі жорсткого диска незвичних звуків є найбільш явною ознакою його найближчого виходу з ладу. Стук, клацання і клацання зазвичай вказують на поганий стан зчитують головок – грізний симптом, що вимагає невідкладного звернення до сервісного центру. Менш імовірною причиною клацають звуків може бути несправність елементів керуючої плати, що так само не віщує нічого доброго.

Найбільшу небезпеку несе пошкодження механіки, яке здатне привести до зіткнення зчитуючої головки з поверхнею магнітного шару. У цьому випадку на поверхні диска утворюються подряпини різної глибини з подальшим руйнуванням секторів і втратою даних. Сторонні звуки теж можуть вказувати на серйозні проблеми з магнітною поверхнею, так, при проходженні голівки над скупченнями «бедів» можна чути дряпаючі і скрегочучи звуки.

Зниження швидкості читання і запису:

Швидко прогресуюче зниження швидкості запису (див. рис. 1.7.) відноситься до явних ознак руйнування магнітного шару.

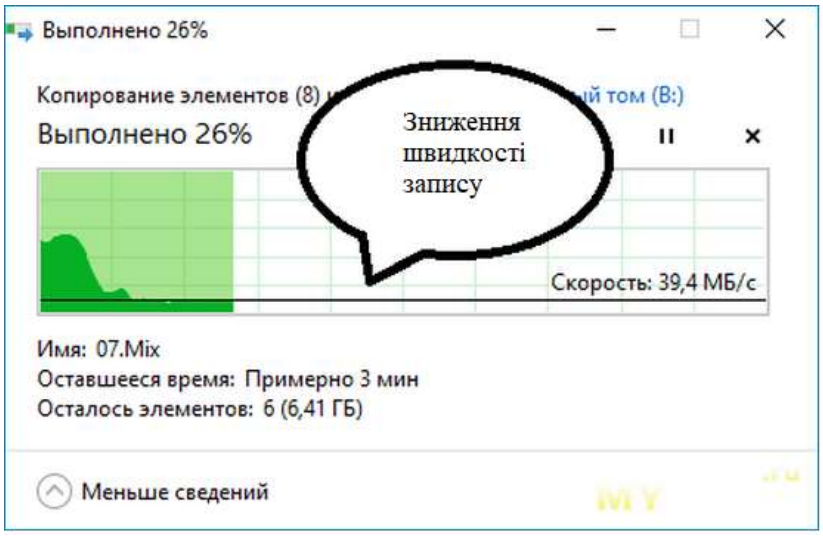


Рисунок 1.7 – Зниження швидкості запису

Таке зниження особливо добре помітно при копіюванні великих обсягів даних, швидкість то падає, то зростає, час від часу комп'ютер починає

зависати без видимих на те причин, спостерігається різке падіння продуктивності і зниження чутливості інтерфейсу при дефрагментації, перевірки «chkdsk», відтворенні фільмів, запуску віртуальних машин, розпакуванню об'ємних архівів.

На серйозні проблеми з диском можуть вказувати повільне видалення файлів, неможливість видалити, записати або прочитати дані. При аналізі диска з поганим станом магнітного шару утилітами HDDScan або Victoria можна бачити як різко знижується графік швидкості читання (Рисунок 1.8).

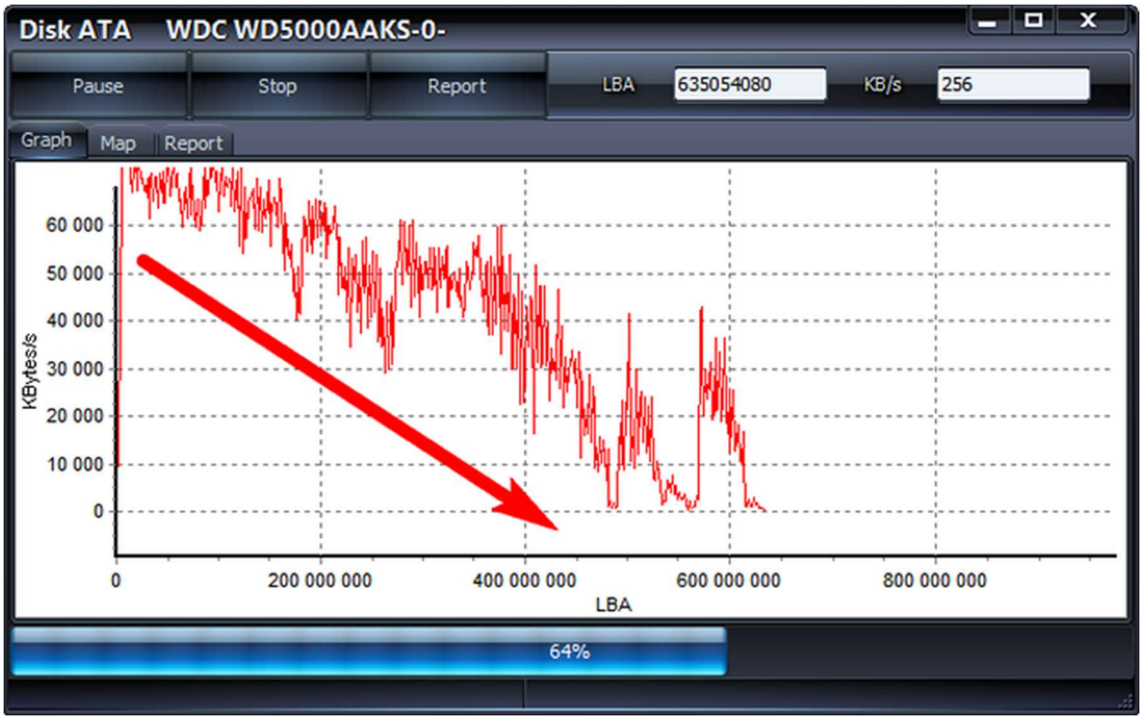


Рисунок 1.8 – Графік зниження швидкості читання

Тривожні показники S.M.A.R.T.

Особливу увагу слід приділити звітам S.M.A.R.T., на що наближається кінець вінчестера вказує неухильне зростання наступних показників (рисунок 1.9.):

– Reallocated Sector Count (ID 05);
– Reported Uncorrected Sector Count (ID 187);
– Current Pending Sector Count (ID 197);

– Uncorrectable Sector Count (ID 198);

– End–to–End Error (ID 184);

– Write Error Rate (ID 200)

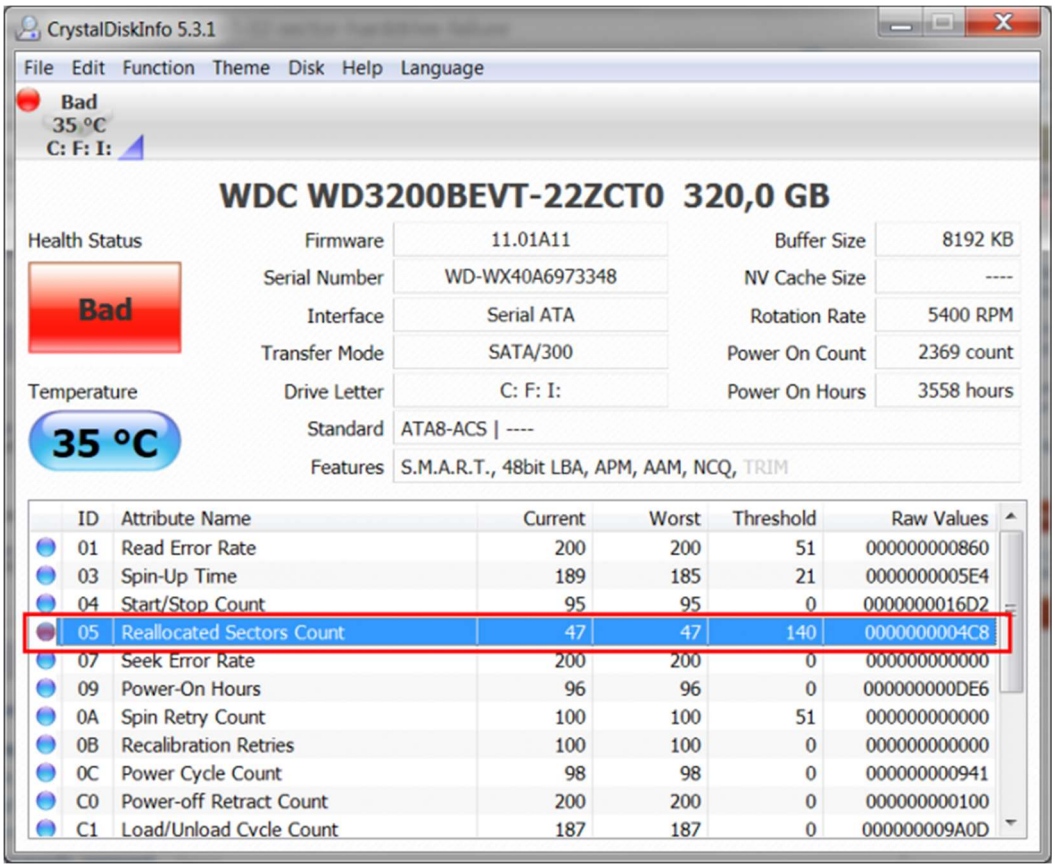


Рисунок 1.9 – Пошкоджений сектор (Reallocated Sector Count)

На жаль, показники S.M.A.R.T. не завжди відображають справжній стан здоров'я дисків. Наприклад, сектора, помічені як нестабільні (Current Pending Sector Count), можуть перебувати як в більш–менш хорошому стані, так і на межі повного руйнування. Важко сказати, скільки протримається диск, на якому почали з'являтися проблемні сектора. Це може бути рік, півроку або місяць.

Інші ознаки:

Крім явних, існує цілий ряд непрямих ознак, які можуть «натякати» на не найкращий стан HDD. Є над чим замислитися, якщо:

– диск часом не визначається BIOS;

– диск визначається як невідформатований;
– з незрозумілих причин то зникають, то з'являються папки і файли;
– часто доводиться стикатися з різними помилками читання / Запису;
– система зависає відразу після завантаження або не завантажується з першого разу.

Таблиця 1.1 – Порівняння програмних забезпечень, які дозволяють проводити моніторинг стану HDD

Програмне забезпечення	S.M.A.R.T. – аналіз	Система повідомлень	Виправлення помилок	DOS–версія
Crystal Disk Mark	Ні	Ні	Ні	Ні
HDDScan	Так	Ні	Ні	Ні
DiskCheckup	Так	Так	Ні	Ні
CrystalDiskInfo	Так	Так	Ні	Ні
Western Digital Data Lifeguard Diagnostic	Так	Ні	Так	Ні
MHDD	Так	Ні	Так	Так
HDD Regenerator	Так	Так	Так	Так
Victoria HDD	Так	Ні	Так	Так
HD Tune	Так	Ні	Ні	Ні
Win_APP	Так	Так	Ні	Ні

1.5 Висновок до розділу 1

Проаналізувавши програмний ринок з моніторингу стану HDD було виділено низку переваг деяких ПЗ на які саме і буде зосереджено акцент в подальшій розробці власного програмно забезпечення, що реалізує потреби для навчальної установи, такої як Запорізький Національний Університет.

Надалі розглянемо інструментарій за допомогою якого і буде розроблено такий програмний продукт

2 СЕРЕДОВИЩЕ РЕАЛІЗАЦІЇ ПРОГРАМНОГО ПРОДУКТУ

2.1 Використання мови програмування C#

Мову C # було розроблено в 2000–му році корпорацією Microsoft, основною аксіомою цієї мови є вислів: «будь–яка сутність є об'єкт». Мову засновано на суворій компонентній архітектурі, а також реалізовано передові механізми забезпечення безпеки коду [9].

Мова програмування C # відобразила у собі кращі риси цілого ряду своїх попередників. Крім мови C ++, необхідно вказати ще кілька знакових для теперішнього часу мов програмування, а саме, Java і Visual Basic.

Провівши аналіз з основних особливостей даної мови програмування, сформулюємо найбільш помітні переваги даної мови програмування.

C # – мова для практичного реалізування компонентно–орієнтованого підходу до програмування, який сприяє меншій машинно–архітектурної залежності результуючого програмного коду, більшої гнучкості, переносимості і легкості повторного використання (фрагментів) програм. Принципово важливою відмінністю від попередників є початкова орієнтація на безпеку коду (що особливо помітно в порівнянні з мовами C і C ++), дана безпека перш за все дотримується в самій платформі .NET. Розширена підтримка подієво–орієнтованого програмування вигідно відрізняє мова програмування C # від цілого ряду попередників. Об'єднання кращих ідей сучасних мов програмування (Java, C ++, Visual Basic та ін.) робить мову C # не просто сумою їх переваг, а мовою програмування нового покоління [10].

Незважаючи на значну кількість принципових переваг в порівнянні з існуючими аналогами, мова програмування C # не позбавлений і окремих недоліків, які носять суб'єктивний, локальний, тимчасовий характер.

До теперішнього часу компілятор і середовище розробки програмного забезпечення, що підтримують мову C #, мають відносно невисокою

продуктивністю (тобто код програми на мові C # компілюється і виконується приблизно в 100 разів повільніше, ніж на мові C). Справедливості заради потрібно відзначити, що продуктивність програм на C # цілком порівнянна з тим же показником для мови Java [11].

Перевагами мови програмування C # є:

а) компонентно–орієнтоване програмування;

б) безпечний (в порівнянні з мовами C і C ++) код;

в) уніфікована система типізації;

г) підтримка подієво–орієнтованого програмування;

д) «рідна» мова для створення додатків в середовищі .NET;

е) об'єднання кращих ідей сучасних мов програмування: Java, C ++, Visual Basic та ін.

Недоліки мови програмування C #:

а) досить складний синтаксис (75% з Java, 10% з C ++, 5% з Visual Basic);

б) що до трохи свіжих концептуальних ідей (ймовірно, менш ніж 10% конструкцій мови);

в) відносно невисока продуктивність (~ в 100 разів повільніше, ніж мова C, хоча і порівнянна з мовою Java).

2.2 Особливості платформи .NET. Переваги та недоліки

Microsoft .Net Framework являється так званою програмною платформою. У загальних рисах можна провести аналогію з відео файлами, які не будуть відтворюватися якщо в системі не встановлений потрібний кодек. В даному випадку відео файл – це програма, написана з використанням технології .Net, а кодек – це сама платформа Microsoft .Net Framework. Причому для роботи програми, написаної на конкретній версії .Net Framework, потрібна установка саме цієї версії.

Зроблено це для того, щоб розробник міг максимально абстрагуватися від системного оточення на комп'ютері користувача. Його не повинно хвилювати, яка операційна система встановлена, яка розрядність у процесора – 32–х або 64–бітна, яка у нього архітектура і т.д. Для запуску програми досить щоб під цю систему було встановлена відповідна версія .Net Framework. Для операційних систем Windows розробкою платформи займається її творець, компанія Microsoft. Існують також незалежні реалізації, перш за все це Mono і Portable.NET, що дозволяють запускати програми .Net на інших операційних системах, наприклад на Linux.

Переваги платформи .NET [12]:

а) платформа .NET заснована на єдиній об'єктно–орієнтованій моделі; всі сервіси, що надаються програмістові платформою, оформлені у вигляді єдиної ієрархії класів. Це вирішує багато проблем програмування на платформі Win32.

б) завдяки тому, що проміжне уявлення .NET не прив'язане до будь–якої платформи, додатки, які створені в архітектурі .NET, є багатоплатформним набором інструментів.

в) платформа .NET надає автоматичне керування ресурсами. Це вирішує багато поширених проблеми, такі як витоку пам'яті, повторне звільнення ресурсу і т.п.

г) код, згенерований для .NET, може бути перевірений на безпеку. Це гарантує, що програма не може нашкодити користувачеві або порушити функціонування операційної системи (так звана "модель пісочниці"). Таким чином, додатки для .NET можуть бути сертифіковані на безпеку.

д) міжмовна взаємодія (language interoperability). Це єдина модель, що дозволяє на рівних користуватися різними мовами для створення додатків. Так як MSIL не залежить від вихідної мови програмування, або від цільової платформи, в рамках .NET стає можливим розвивати нові програми на базі старих програм – причому і перша, і друга мови програмування не так вже й важливі!

Недоліки платформи .NET:

1. Найвідчутнішим недоліком є істотне уповільнення виконання програм.

2.3 Постановка вимог до програмного продукту

Програмний продукт повинен виконувати наступні пункти:

– збирати інформацію про ПК (операційна система, процесор, материнська плата, і т. д.);

– отримувати S. M. A. R. T. Атрибут;

– моніторинг процесів в реальному часі;

– відстежувати служби що запущено, та стислу інформацію за що відповідають;

– обслуговування (відстеження попередніх несправностей, мір що було прийнято для вирішення тих чи інших проблем).

Для більш детального розуміння програмного продукту розглянемо діаграма прецидентов, яка повною мірою відображає взаємодію користувача з програмою (див. додаток 1).

Мінімальні вимоги до ПК:

- **ОС:** Windows XP, 7, 8, 8.1, 10, відповідно Windows NT 5.1, Windows NT 6.2, Windows NT 6.3, Windows NT 10;
- **процесор:** 1 гігагерц (ГГц) або швидший процесор чи SoC;
- **оперативна пам’ять:** 1 гігабайт (ГБ) і більше;
- **обсяг дискового простору:**16 ГБ для 32–розрядної ОС або 20 ГБ для 64–розрядної ОС;
- **версія .**NET Framework 4 та вище.

2.4 Модулі та їх призначення

В программі були використані такі модулі:

– System.drawing – Простір імен System.Drawing забезпечує доступ до основних графічних функцій GDI +;

– linq – простір імен System.Linq надає класи і інтерфейси, які підтримують запити, що використовують вбудований в мову запит (LINQ);

– xml linq\LINQ to XML – це інтерфейс програмування в пам'яті, який дозволяє ефективно і легко змінювати XML–документи;

– serviceprocess – простір імен містить класи, які дозволяють реалізувати, установки і управління додатками, а саме службами Windows;

– io – простір імен System.IO містить типи, які дозволяють читати і записувати файли і потоки даних, а також типи, що забезпечують базову підтримку файлів і каталогів;

– diagnostics – простір імен System.Diagnostics надає класи, які дозволяють взаємодіяти з системними процесами, журналами подій і лічильниками продуктивності;

– management – надає доступ до великого набору керуючої інформації і подій управління системою, пристроями і додатками, оснащеними інфраструктурою інструментарію управління Windows (WMI);

– windows.forms – простір імен System.Windows.Forms містить класи для створення додатків на базі Windows, які в повній мірі використовують можливості багатофункціонального призначеного для користувача інтерфейсу, які надаються операційною системою Microsoft Windows;

– system – простір імен System містить фундаментальні класи і базові класи, які визначають за допомогою яких традиційно значення і еталонні типи даних, події і обробники подій, інтерфейси, атрибути і виключення обробки.

В коді программи вони були відображені таки чином:

using System;

```
using System.Drawing;
using System.Windows.Forms;
using System.Management;
using System.Diagnostics;
using System.IO;
using System.Xml.Linq;
using System.ServiceProcess;
using System.Linq.
```

2.5 Висновок до розділу 2

Дослідження предметної області дозволило виявити вимоги, які пред`явлено до програмного забезпечення з діагностики апаратних ресурсів персонального комп`ютера.

Спроектовано алгоритм роботи програмного забезпечення, що розробляється у кваліфікаційній роботі, який представлено у вигляді UML діаграмі прецедентів та діаграмі взаємодії методів бібліотек System C#.

3 РЕАЛІЗАЦІЯ ВИМОГ ПРОГРАМНОГО ПРОДУКТУ

3.1 Отримання інформації про складову ПК

а) Інформація про операційну систему, а саме поверхневі відомості що до версії Windows, та користувача:

```
void CreateNodes(){// створення інформаційних записів
                    treeView1.Nodes.Clear();
                    treeView1.Nodes.Add("System", "Операційна
система");
                    treeView1.Nodes.Add("CPU","Процесор");
                    treeView1.Nodes.Add("MainBoard","Материнська
плата");
                    treeView1.Nodes.Add("RAM","Оперативна  пам'ять");
                    treeView1.Nodes.Add("GPU","Відеокарта");
                    treeView1.Nodes.Add("HDD","Жорсткий диск");
                    treeView1.Nodes["System"].Nodes.Add("OSVersion",
"Операционная система: " + Environment.OSVersion);
                treeView1.Nodes["System"].Nodes.Add("SystemDirectory",
"Системна директорія: " + Environment.SystemDirectory);
                treeView1.Nodes["System"].Nodes.Add("UserName", "Ім'я
користувача: "+ Environment.UserName);

        treeView1.Nodes["System"].Nodes.Add("HostName","Мережеве
ім'я: "+System.Net.Dns.GetHostName());
                }
```

б) Отримання повноцінних даних про процесор (виробник, кількість ядер):

```
void GetProcessorInfo(){ // інформація про процесор
```

```
                ManagementObjectSearcher searcher = new
ManagementObjectSearcher("SELECT * FROM Win32_Processor");
                foreach (ManagementObject wmi in searcher.Get())
                {
                 try
                        {

        treeView1.Nodes["CPU"].Nodes.Add("Name", "Процесор: " +
wmi.GetPropertyValue("Name").ToString());

        treeView1.Nodes["CPU"].Nodes.Add("Manufacturer", "Виробник: "
+ wmi.GetPropertyValue("Manufacturer").ToString());

        treeView1.Nodes["CPU"].Nodes.Add("Count", "Кількість ядер: " +
Environment.ProcessorCount);

        treeView1.Nodes["CPU"].Nodes.Add("Description", "Опис: " +
wmi.GetPropertyValue("Description").ToString());
                        }
                  catch { }
                }
            }
```

в) Отримання даних про материнську плату а саме про виробника, модель, серійний номер та версію:

```
void GetMotherBoardInfo(){ // інформація про материнську плату
                ManagementObjectSearcher searcher = new
ManagementObjectSearcher("SELECT Manufacturer,Product,
SerialNumber,Version FROM Win32_BaseBoard");
                ManagementObjectCollection information =
searcher.Get();
```

```
                    foreach (ManagementObject obj in information)
                    {

        treeView1.Nodes["MainBoard"].Nodes.Add("Manufacturer","Виробн
    ик: "+ obj.GetPropertyValue("Manufacturer").ToString());

        treeView1.Nodes["MainBoard"].Nodes.Add("Product","Модель: "+
    obj.GetPropertyValue("Product").ToString());

        treeView1.Nodes["MainBoard"].Nodes.Add("SerialNumber","Серійн
    ий номер: "+ obj.GetPropertyValue("SerialNumber").ToString());

        treeView1.Nodes["MainBoard"].Nodes.Add("Version","Версія: "+
    obj.GetPropertyValue("Version").ToString());
                    }
                }
```

г) Отримання інформації о ОЗУ (к–ть планок ОЗУ, зайняті слоти, об`єм кожної планки, виробник):

```
    void GetRAMInfo(){ // инфа о озу
                    ManagementObjectSearcher searcher = new
    ManagementObjectSearcher("SELECT * FROM
    Win32_PhysicalMemory");
                    ManagementObjectCollection ram = searcher.Get();
                    string[]
    strFormFactor={"Unknown","Other","SIP","DIP","ZIP","SOJ","Proprietary
    ","SIMM","DIMM","TSOP","PGA","RIMM","SODIMM","SRIMM","SM
    D","SSMP","QFP","TQFP","SOIC","LCC","PLCC","BGA","FPBGA","LG
    A"};
                    string[]
    strMemoryType={"Unknown","Other","DRAM","Synchronous
```

```
DRAM","Cache
DRAM","EDO","EDRAM","VRAM","SRAM","RAM","ROM","Flash","E
EPROM","FEPROM","EPROM","CDRAM","3DRAM","SDRAM","SGRA
M","RDRAM","DDR","DDR2","DDR2 FB–DIMM","","DDR3","FBD2"};
                    int indexRam=0;
                    foreach (ManagementObject qwe in ram)
                    {

                        try{

treeView1.Nodes["RAM"].Nodes.Add("DeviceLocator","Слот: "+
qwe.GetPropertyValue("DeviceLocator").ToString());

treeView1.Nodes["RAM"].Nodes[indexRam].Nodes.Add("Manufacturer","
Виробник: "+ qwe.GetPropertyValue("Manufacturer").ToString());

        treeView1.Nodes["RAM"].Nodes[indexRam].Nodes.Add("MemoryT
ype","Тип пам'яті: "+
strMemoryType[Convert.ToInt32(qwe.GetPropertyValue("MemoryType"))]
);

        treeView1.Nodes["RAM"].Nodes[indexRam].Nodes.Add("Formfactor
","Форм фактор: "+
strFormFactor[Convert.ToInt32(qwe.GetPropertyValue("FormFactor"))]);

        treeView1.Nodes["RAM"].Nodes[indexRam].Nodes.Add("Capasity",
"Об'єм: "+
Convert.ToDouble(qwe.GetPropertyValue("Capacity"))/1024/1024/1024 + "
Гб");
```

```
        treeView1.Nodes["RAM"].Nodes[indexRam].Nodes.Add("Speed","
Швидкість: "+ qwe.GetPropertyValue("Speed").ToString());

        treeView1.Nodes["RAM"].Nodes[indexRam].Nodes.Add("SerialNum
ber","Серійний номер: "+
qwe.GetPropertyValue("SerialNumber").ToString());
                        indexRam++;
                        }
                        catch{};
                }
        }
```

д) Отримання даних про відео адаптер (виробник, об`єм):

```
void GetVideoInfo(){ // інформація про відео
                ManagementObjectSearcher searcher = new
ManagementObjectSearcher("SELECT * FROM Win32_VideoController");
                        int indexVideo=0;
                        foreach (ManagementObject wmi in
searcher.Get())
                {
                 try
                        {

        treeView1.Nodes["GPU"].Nodes.Add("Name", "Назва: " +
wmi.GetPropertyValue("Name").ToString());

        treeView1.Nodes["GPU"].Nodes[indexVideo].Nodes.Add("VideoPro
cessor", "Відео процесор: " +
wmi.GetPropertyValue("VideoProcessor").ToString());
```

```
            treeView1.Nodes["GPU"].Nodes[indexVideo].Nodes.Add("DriverVer
        sion", "Версія драйверу: " +
        wmi.GetPropertyValue("DriverVersion").ToString());
                                        double
        AdapterRAM=Convert.ToDouble(wmi.GetPropertyValue("AdapterRAM"))/
        1024/1024/1024;

            treeView1.Nodes["GPU"].Nodes[indexVideo].Nodes.Add("AdapterR
        AM", "Об'єм пам'яті: " + AdapterRAM.ToString("F"+ 2) + " Гб");
                                        indexVideo++;
                                    }
                                catch { }
                            }
                        }
```

е) Отримання інформації про стан жорсткого диску, я саме S. M. A. R. T.: Інформація що до реалізації отримання S. M. A. R. T. даних див. додаток Б.

ж) Створення записів що до технічного обслуговування реалізовано у додатку В.

3.2 Користувацька інструкція з використання програмного

Для початку необхідно запустити ПЗ WIN_INFO_APP.exe, для цього подвійним натисканням лівої кнопки миші по програмі WIN_INFO_APP.exe відкривається інформаційне вікно, яке дозволяє дізнатися час роботи ПК, ім`я користувача, мережеве ім`я, використання ЦП, вільну ОЗУ.

Рисунок 3.1 – Головне меню ПЗ WIN_INFO_APP

Натискаючи кнопку детальніше у нас з`являється шість вкладок:

– Інформація
– S.M.A.R.T. HDD
– Процеси
– Служби
– Обслуговування
– Автор

а) графічний вигляд інформаційної вкладки (див. рис. 3.2):

З цього меню ми легко одержуємо основну інформацію, що до складової ПК, а саме данні про операційну систему, процесор, виробника та модель материнської плати, інформацію про оперативну пам`ять її об`єм, та в які слоти встановлено модулі, виробника відеокарти об`єм пам`яті.

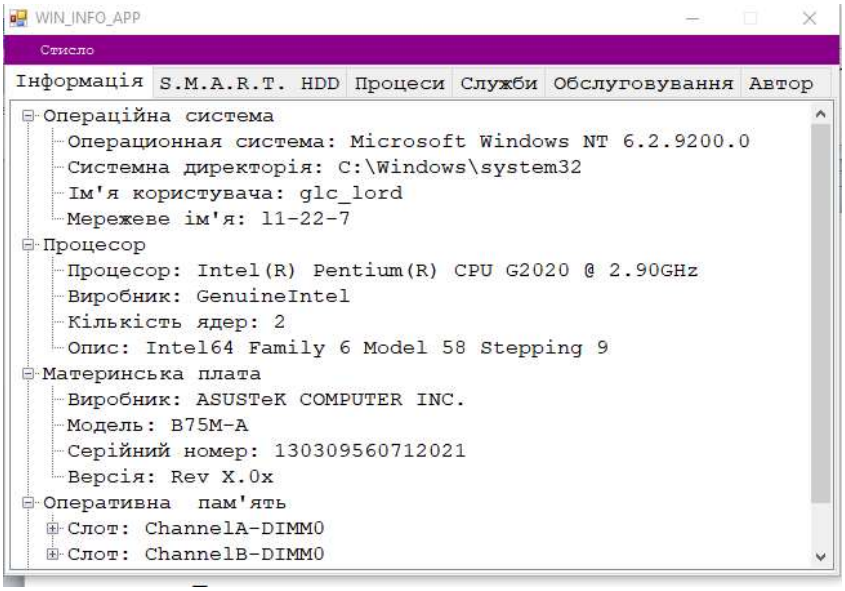


Рисунок 3.2 – Вкладка інформація

б) графічний вигляд S.M.A.R.T. HDD:

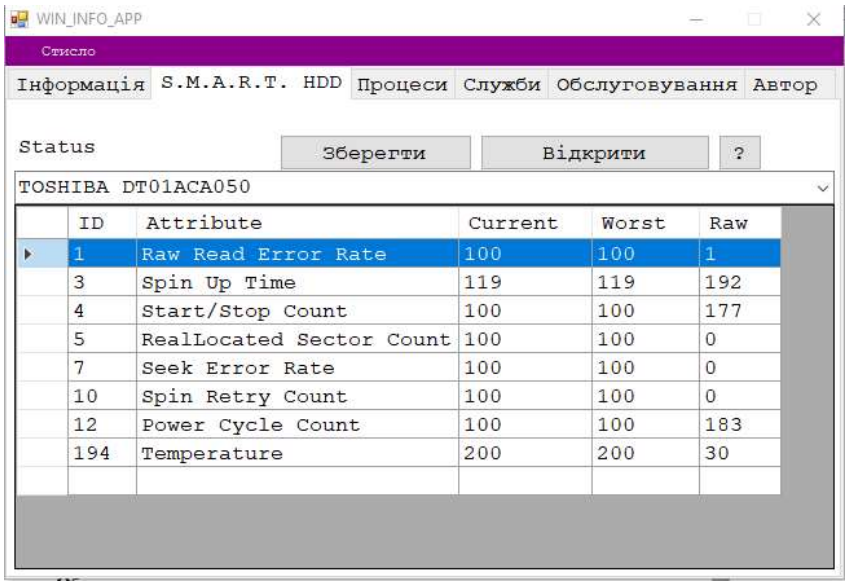


Рисунок 3.3 — Вкладка S.M.A.R.T. HDD

В даному меню зібрана інформація про стан жорсткого диску, тобто S.M.A.R.T., де відображено наступні параметри:

– Raw Read Error Rate – містить інформацію про частоту виникнення помилок при зчитуванні даних з жорсткого магнітного диску;

– Spin–Up Time – містить дані про час, за який шпиндель диску в останній раз розігнався зі стану спокою до номінальної швидкості. Може вимірюватися в мілісекундах, десятках мілісекунд і т.д. в залежності від виробника і моделі диску;

– Number of Spin–Up Times (Start/Stop Count) – данні про кількість запуску диску;

– Reallocated Sector Count – інформація про кількість секторів, перепризначених вінчестером в резервну область. Майже ключовий параметр в оцінці стану диску;

– Seek Error Rate – частота виникнення помилок при позиціонуванні блоку магнітних головок (БМГ);

– Spin Retry Count – містить кількість повторів запуску шпінделя, якщо перша спроба виявилась невдалою;

– Power Cycle Count – містить кількість повних циклів "включення–виключення" диску;

– Temperature – містить поточну інформацію про температуру диску.

в) графічний вигляд вкладки процеси:

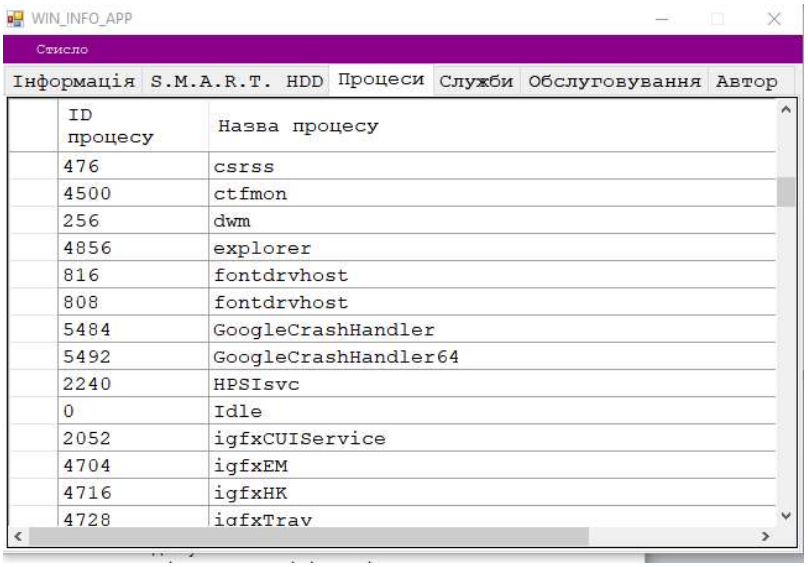


Рисунок 3.4 – Вкладка процеси

Іноді постає питання про перегляд процесів що запущено на ПК без використання стандартного диспетчера задач.

г) графічний вигляд вкладки служби:

Служби ОС Windows – додатки, що автоматично (якщо є відповідне налаштування) запускаються системою при запуску Windows і виконуються в фоновому режимі.

Існує кілька режимів для служб:

– заборонений до запуску;

– ручний запуск (за запитом);

– автоматичний запуск при завантаженні комп'ютера ;

– автоматичний (відкладений) запуск (введений в Windows Vista і Windows Server 2008) ;

– обов'язкова служба/драйвер (автоматичний запуск і неможливість (для користувача) зупинити службу).

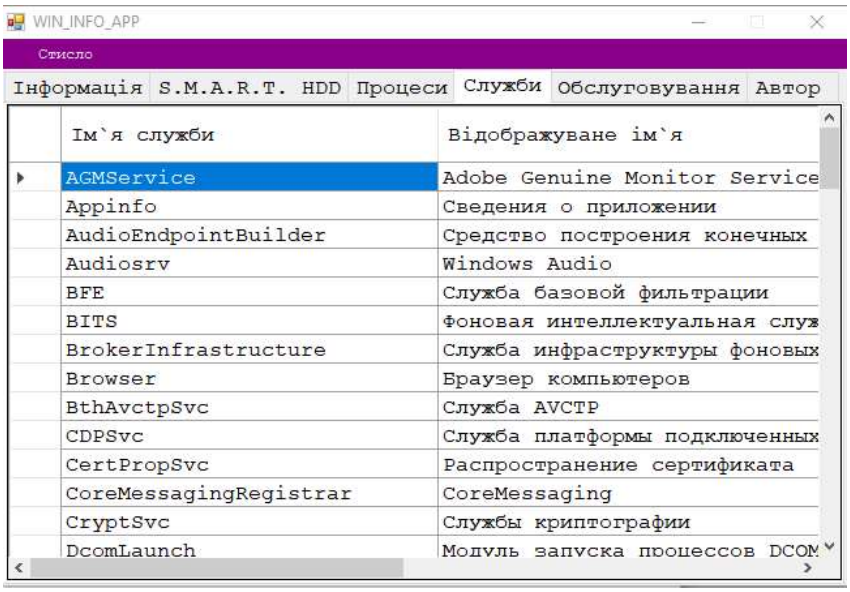


Рисунок. 3.5 – Вкладка служби

д) обслуговування:

Найважливіша вкладка донного ПЗ, так як за допомогою даної вкладки можна вести повний перелік всіх несправностей ПК. Легко відстежити, що й коли вийшло із строю, як було вирішено проблему (див. рис. 3.6).

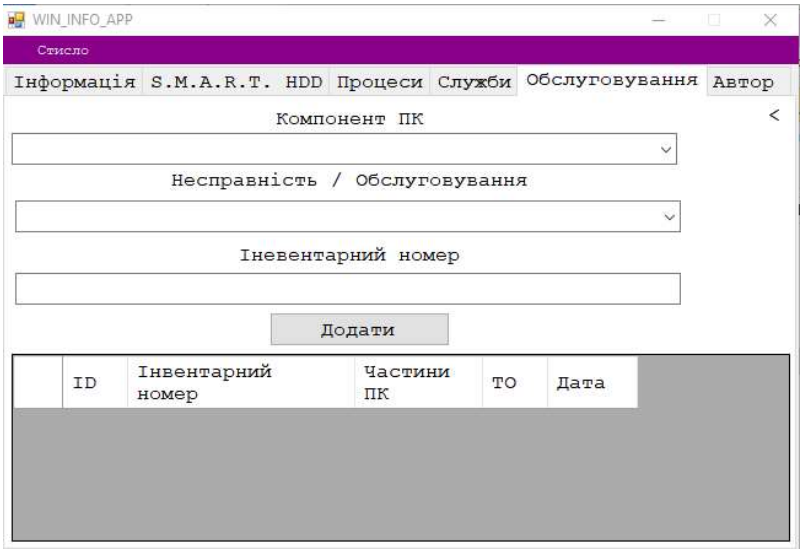


Рисунок 3.6 – Вкладка обслуговування

ВИСНОВКИ

Розроблено віджет моніторингу стану апаратного забезпечення персонального комп’ютера.

Реалізовано наступні задачі:

– розглянуто методи аналізу стану ПК;

– розглянуто поняття S.M.A.R.T. та методика отримання даних;

– проаналізовано роботу S.M.A.R.T. системи.

На основі виконаних завдань розроблено програмний продукт, що дозволяє поліпшити та спростити роботу інженера комп’ютерних класів, а саме поліпшити процес спостереження за станом роботи ПК

ПЕРЕЛІК ПОСИЛАНЬ

1. SMART – як не втратити інформацію на диску. URL: https://datarecovery.org.ua/2019/01/21/smart/ (дата звернення 10.11.2019)

2. Как пользоваться программой Victoria HDD. URL: https://victoria4.ru/kak–polzovatsya–programmoj–victoria–hdd.php (дата звернення 10.11.2019)

3. MHDD. URL: http://ihdd.ru/mhdd (дата звернення 12.11.2019)

4. HDDScan – Free HDD Diagnostic Utility. URL: https://hddscan.com/ (дата звернення 12.11.2019)

5. Scanner. URL: http://www.steffengerlach.de/freeware/ (дата звернення 12.11.2019)

6. CrystalDiskInfo. URL: https://crystalmark.info/en/software/crystaldiskinfo/ (дата звернення 20.11.2019)

7. HD Tune Pro. URL: https://www.hdtune.com/ (дата звернення 22.11.2019)

8. Як перевірити твердотільний жорсткий диск. Як дізнатися, скільки працює SSD диск і оцінити його стан URL: https://floruzor.ru/uk/kak–proverit–tverdotelnyi–zhestkii–disk–kak–uznat–skolko/ (дата звернення 22.11.2019)

9. Основи C#. URL: http://www.znannya.org/?view=csharp–bases (дата звернення 22.11.2019)

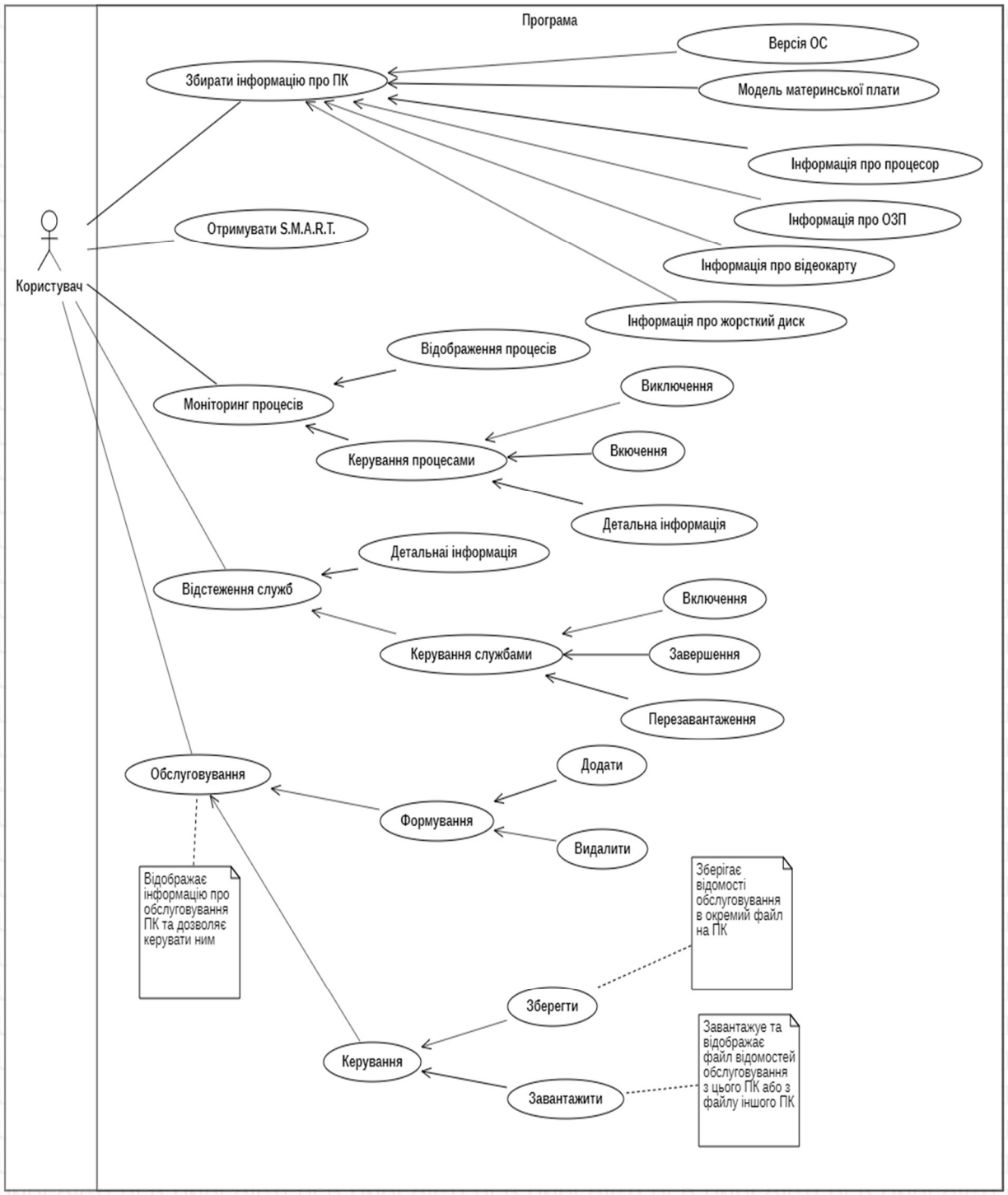
10. Кузьомін О.Я. Від основ С++ до об'єктно–орієнтованого програмування додатків : Навч. посіб. О. Я. Кузьомін; Наук.–метод. центр вищ. освіти. Х., 2000. 124 с.

11. Семантика основних конструкцій мови програмування C#. URL: http://www.znannya.org/?view=csharp–bases (дата звернення 25.11.2019)

12. Переваги Microsoft .NET Framework. URL: https://support.microsoft.com/uk–ua/help/929300/benefits–of–the–microsoft–net–framework (дата звернення 25.11.2019)

ДОДАТОК А

Діаграма прецидентів, що описує взаємодію між користувачем і програмним продуктом



ДОДАТОК Б

Отримання данних механізму самоконтролю, аналізу та звітності

```
void GetSMART(){ // отримання S. M. A. R. T

                            ManagementObjectSearcher        WMISearch        =new
ManagementObjectSearcher("Select * from Win32_DiskDrive");
                                    ManagementObjectCollection       Drives       =
WMISearch.Get();
                                    int ind=comboBox3.SelectedIndex;
                    //Clear();
                    //foreach ( ManagementObject Drive in Drives )
                    //{
                    //
comboBox3.Items.Add(Drive.Properties["Model"].Value.ToString());
                    //}
                                    SMART smartdata;
                                    int IsTemperature=0;
                                    dataGridView1.Rows.Clear();
                                    WMISearch.Scope=new  ManagementScope(@"\root\wmi"
);
                                    WMISearch.Query=new  ObjectQuery("Select  *  from
MSStorageDriver_FailurePredictData");
                                    ManagementObjectCollection     FailDataSet     =
WMISearch.Get();
                                    ManagementObject               mo              =
FailDataSet.OfType<ManagementObject>().ElementAtOrDefault(ind);
                                    try{
                                           Byte[]               data2               =
(Byte[])mo.Properties["VendorSpecific"].Value;
                                           for (int i = 0; i < data2[0] – 1; i++)
                                           {
                                                  int start=i*12;
                                                  switch(data2[start+2])
                                                  {
                                                         case 1:

      smartdata.RawReadErrorRate.value = data2[start + 5];
```

```
        smartdata.RawReadErrorRate.worstvvalue=data2[start+6];

        smartdata.RawReadErrorRate.rawvalue=data2[start+7];

        this.dataGridView1.Rows.Add();

        this.dataGridView1.Rows[0].Cells[0].Value ="1";

        this.dataGridView1.Rows[0].Cells[1].Value ="Raw Read Error Rate";

        this.dataGridView1.Rows[0].Cells[2].Value
=smartdata.RawReadErrorRate.value.ToString();

        this.dataGridView1.Rows[0].Cells[3].Value
=smartdata.RawReadErrorRate.worstvvalue.ToString();

        this.dataGridView1.Rows[0].Cells[4].Value
=smartdata.RawReadErrorRate.rawvalue.ToString();

        if(smartdata.RawReadErrorRate.rawvalue>10)
dataGridView1.Rows[0].Cells[0].Style.BackColor=Color.Yellow;

        if(smartdata.RawReadErrorRate.rawvalue>10)
dataGridView1.Rows[0].Cells[1].Style.BackColor=Color.Yellow;

        if(smartdata.RawReadErrorRate.rawvalue>10)
dataGridView1.Rows[0].Cells[2].Style.BackColor=Color.Yellow;

        if(smartdata.RawReadErrorRate.rawvalue>10)
dataGridView1.Rows[0].Cells[3].Style.BackColor=Color.Yellow;

        if(smartdata.RawReadErrorRate.rawvalue>10)
dataGridView1.Rows[0].Cells[4].Style.BackColor=Color.Yellow;
                                                        break;
                                                        case 3:

        smartdata.SpinUpTime.value = data2[start + 5];

        smartdata.SpinUpTime.worstvvalue=data2[start+6];

        smartdata.SpinUpTime.rawvalue=data2[start+7];
```

```
        this.dataGridView1.Rows.Add();

        this.dataGridView1.Rows[1].Cells[0].Value ="3";

        this.dataGridView1.Rows[1].Cells[1].Value ="Spin Up Time";

        this.dataGridView1.Rows[1].Cells[2].Value
=smartdata.SpinUpTime.value.ToString();

        this.dataGridView1.Rows[1].Cells[3].Value
=smartdata.SpinUpTime.worstvvalue.ToString();

        this.dataGridView1.Rows[1].Cells[4].Value
=smartdata.SpinUpTime.rawvalue.ToString();
                                                        break;
                                                        case 4:

        smartdata.StartStopCount.value = data2[start + 5];

        smartdata.StartStopCount.worstvvalue=data2[start+6];

        smartdata.StartStopCount.rawvalue=data2[start+7];

        this.dataGridView1.Rows.Add();

        this.dataGridView1.Rows[2].Cells[0].Value ="4";

        this.dataGridView1.Rows[2].Cells[1].Value ="Start/Stop Count";

        this.dataGridView1.Rows[2].Cells[2].Value
=smartdata.StartStopCount.value.ToString();

        this.dataGridView1.Rows[2].Cells[3].Value
=smartdata.StartStopCount.worstvvalue.ToString();

        this.dataGridView1.Rows[2].Cells[4].Value
=smartdata.StartStopCount.rawvalue.ToString();
                                                        break;
                                                        case 5:

        smartdata.ReallocatedSectorCount.value = data2[start + 5];
```

```
        smartdata.ReallocatedSectorCount.worstvvalue=data2[start+6];

        smartdata.ReallocatedSectorCount.rawvalue=data2[start+7];

        this.dataGridView1.Rows.Add();

        this.dataGridView1.Rows[3].Cells[0].Value ="5";

        this.dataGridView1.Rows[3].Cells[1].Value ="RealLocated Sector Count";

        this.dataGridView1.Rows[3].Cells[2].Value
=smartdata.ReallocatedSectorCount.value.ToString();

        this.dataGridView1.Rows[3].Cells[3].Value
=smartdata.ReallocatedSectorCount.worstvvalue.ToString();

        this.dataGridView1.Rows[3].Cells[4].Value
=smartdata.ReallocatedSectorCount.rawvalue.ToString();

        if(smartdata.ReallocatedSectorCount.rawvalue>5)
dataGridView1.Rows[3].Cells[0].Style.BackColor=Color.Red;

        if(smartdata.ReallocatedSectorCount.rawvalue>5)
dataGridView1.Rows[3].Cells[1].Style.BackColor=Color.Red;

        if(smartdata.ReallocatedSectorCount.rawvalue>5)
dataGridView1.Rows[3].Cells[2].Style.BackColor=Color.Red;

        if(smartdata.ReallocatedSectorCount.rawvalue>5)
dataGridView1.Rows[3].Cells[3].Style.BackColor=Color.Red;

        if(smartdata.ReallocatedSectorCount.rawvalue>5)
dataGridView1.Rows[3].Cells[4].Style.BackColor=Color.Red;
                                                        break;
                                                        case 7:

        smartdata.SeekErrorRate.value = data2[start + 5];

        smartdata.SeekErrorRate.worstvvalue=data2[start+6];

        smartdata.SeekErrorRate.rawvalue=data2[start+7];
```

```
        this.dataGridView1.Rows.Add();

        this.dataGridView1.Rows[4].Cells[0].Value ="7";

        this.dataGridView1.Rows[4].Cells[1].Value ="Seek Error Rate";

        this.dataGridView1.Rows[4].Cells[2].Value
=smartdata.SeekErrorRate.value.ToString();

        this.dataGridView1.Rows[4].Cells[3].Value
=smartdata.SeekErrorRate.worstvvalue.ToString();

        this.dataGridView1.Rows[4].Cells[4].Value
=smartdata.SeekErrorRate.rawvalue.ToString();
                                                        break;
                                                        case 10:

        smartdata.SpinRetryCount.value = data2[start + 5];

        smartdata.SpinRetryCount.worstvvalue=data2[start+6];

        smartdata.SpinRetryCount.rawvalue=data2[start+7];

        this.dataGridView1.Rows.Add();

        this.dataGridView1.Rows[5].Cells[0].Value ="10";

        this.dataGridView1.Rows[5].Cells[1].Value ="Spin Retry Count";

        this.dataGridView1.Rows[5].Cells[2].Value
=smartdata.SpinRetryCount.value.ToString();

        this.dataGridView1.Rows[5].Cells[3].Value
=smartdata.SpinRetryCount.worstvvalue.ToString();

        this.dataGridView1.Rows[5].Cells[4].Value
=smartdata.SpinRetryCount.rawvalue.ToString();
                                                        break;
                                                        case 12:

        smartdata.PowerCycleCount.value = data2[start + 5];
```

```
        smartdata.PowerCycleCount.worstvvalue=data2[start+6];

        smartdata.PowerCycleCount.rawvalue=data2[start+7];

        this.dataGridView1.Rows.Add();

        this.dataGridView1.Rows[6].Cells[0].Value ="12";

        this.dataGridView1.Rows[6].Cells[1].Value ="Power Cycle Count";

        this.dataGridView1.Rows[6].Cells[2].Value
=smartdata.PowerCycleCount.value.ToString();

        this.dataGridView1.Rows[6].Cells[3].Value
=smartdata.PowerCycleCount.worstvvalue.ToString();

        this.dataGridView1.Rows[6].Cells[4].Value
=smartdata.PowerCycleCount.rawvalue.ToString();
                                                    break;
                                                    //   отримуємо   температуру,
якщо диск має стандартний S. M. A. R. T.
                                                    case 194:

        smartdata.Temperature.value = data2[start+5];

        smartdata.Temperature.worstvvalue=data2[start+6];

        smartdata.Temperature.rawvalue=data2[start+7];

        this.dataGridView1.Rows.Add();

        this.dataGridView1.Rows[7].Cells[0].Value ="194";

        this.dataGridView1.Rows[7].Cells[1].Value ="Temperature";

        this.dataGridView1.Rows[7].Cells[2].Value
=smartdata.Temperature.value.ToString();

        this.dataGridView1.Rows[7].Cells[3].Value
=smartdata.Temperature.worstvvalue.ToString();
```

```
       this.dataGridView1.Rows[7].Cells[4].Value
=smartdata.Temperature.rawvalue.ToString();
                                                              IsTemperature=1;
                                                      break;


                                                }
                                                // отримуємо температуру з записів,
якщо  S. M. A. R. T не типовий
                                                //  не  виводимо  значення  доки  не
виведено 5-го попереднього рядка
                                                if
((IsTemperature==0)&&(dataGridView1.Rows.Count==7)){
                                                      smartdata.Temperature.value
= data2[173];

       smartdata.Temperature.worstvvalue=data2[174];

       smartdata.Temperature.rawvalue=data2[175];
                                                      dataGridView1.Rows.Add();

       this.dataGridView1.Rows[7].Cells[0].Value ="194";

       this.dataGridView1.Rows[7].Cells[1].Value ="Temperature";

       this.dataGridView1.Rows[7].Cells[2].Value
=smartdata.Temperature.value.ToString();

       this.dataGridView1.Rows[7].Cells[3].Value
=smartdata.Temperature.worstvvalue.ToString();

       this.dataGridView1.Rows[7].Cells[4].Value
=smartdata.Temperature.rawvalue.ToString();
                                                }

                                          }
                                          }
                                          catch{
                                                MessageBox.Show("Не          можливо
отримати значення");
                                          }
```

```
        }

        void GetProcess(){ // получение списка процесов
            int j=0;
                dataGridView2.Rows.Clear();
                foreach(Process process in Process.GetProcesses())
                {
                    dataGridView2.Rows.Add();
                // выводим id и имя процесса
                dataGridView2.Rows[j].Cells[0].Value = process.Id.ToString();
                dataGridView2.Rows[j].Cells[1].Value = process.ProcessName.ToString();
                j++;
                }
```

ДОДАТОК В

Вкладка обслуговування заповнення інформацій, що до виходу зі строю тієї чи іншої складової ПК

```
void Button5Click(object sender, EventArgs e) // заповнення ТО
              {
                     if((comboBox1.Text!="")&&(comboBox2.Text!="")){
                     int count=0;

                     XDocument xdoc = XDocument.Load("Maintenance.xml");
                     foreach (XElement Element in
xdoc.Element("Maintenances").Elements("maintenance"))
                     {
                            count++;
                     }
                     XElement root = xdoc.Element("Maintenances");
                     root.Add(
                                   new XElement("maintenance",
                                   new XAttribute("id",count+1),
                                   new XElement("inventory", textBox1.Text),
                                   new XElement("Part", comboBox1.Text),
                                   new XElement("Crash", comboBox2.Text),
                                   new XElement("date", DateTime.Today)
                            )

                     );
                     xdoc.Save("Maintenance.xml");
                     }
                     else{
                            MessageBox.Show("Заповніть всі неодхіні поля");
                     }
                     MaintenanceLoad();

              }

              void MaintenanceLoad(){ // загрузка програмного файлу ТО до таблиці
                     int j=0;
                     dataGridView3.Rows.Clear();
                     XDocument xdoc = XDocument.Load("Maintenance.xml");
                     foreach (XElement BDelement in
xdoc.Element("Maintenances").Elements("maintenance"))
                     {
                            dataGridView3.Rows.Add();
                            dataGridView3.Rows[j].Cells[0].Value =
BDelement.Attribute("id").Value;
                     dataGridView3.Rows[j].Cells[1].Value =
BDelement.Element("inventory").Value;
                     dataGridView3.Rows[j].Cells[2].Value =
BDelement.Element("Part").Value;
                     dataGridView3.Rows[j].Cells[3].Value =
BDelement.Element("Crash").Value;
                     dataGridView3.Rows[j].Cells[4].Value =
BDelement.Element("date").Value;
                     j++;
                     }
```

```
        }

        void ПопередняВерсіяФайлууЗвітуЗЦьогоПКToolStripMenuItemClick(object
sender, EventArgs e) // загрузка файлу обраного користувачем та об`єднання з програмним
файлом ТО
        {
                int j=0;
                if (openFileDialog1.ShowDialog() == DialogResult.Cancel)
            return;
                XDocument xdoc = XDocument.Load(openFileDialog1.FileName);
                foreach (XElement BDelement in
xdoc.Element("Maintenances").Elements("maintenance"))
                {
                        dataGridView3.Rows.Add();
                        dataGridView3.Rows[j].Cells[0].Value =
BDelement.Attribute("id").Value;
                dataGridView3.Rows[j].Cells[1].Value =
BDelement.Element("inventory").Value;
                dataGridView3.Rows[j].Cells[2].Value =
BDelement.Element("Part").Value;
                dataGridView3.Rows[j].Cells[3].Value =
BDelement.Element("Crash").Value;
                dataGridView3.Rows[j].Cells[4].Value =
BDelement.Element("date").Value;
                j++;
                }
                xdoc = XDocument.Load("Maintenance.xml");
                foreach (XElement BDelement in
xdoc.Element("Maintenances").Elements("maintenance"))
                {
                        dataGridView3.Rows.Add();
                        dataGridView3.Rows[j].Cells[0].Value =
BDelement.Attribute("id").Value;
                dataGridView3.Rows[j].Cells[1].Value =
BDelement.Element("inventory").Value;
                dataGridView3.Rows[j].Cells[2].Value =
BDelement.Element("Part").Value;
                dataGridView3.Rows[j].Cells[3].Value =
BDelement.Element("Crash").Value;
                dataGridView3.Rows[j].Cells[4].Value =
BDelement.Element("date").Value;
                j++;
                }
                xdoc.Element("Maintenances").RemoveAll();
                for(int i=0;i<dataGridView3.Rows.Count-1;i++){
                        try{
                        XElement root = xdoc.Element("Maintenances");
                        root.Add(
                                new XElement("maintenance",
                                new XAttribute("id",i),
                                new XElement("inventory", textBox1.Text),
                                new XElement("Part", comboBox1.Text),
                                new XElement("Crash", comboBox2.Text),
                                new XElement("date", DateTime.Today)
                         )

                        );
                        }
                        catch{ };
                xdoc.Save("Maintenance.xml");

                }
        }
```

```
            void Button3Click(object sender, EventArgs e) // збереження файлу ТО з
им`ям указаним користувачем за адресою указаною користувачем
            {
                  if (saveFileDialog1.ShowDialog() == DialogResult.Cancel)
            return;
                   // отримуємо обраний файл
                  string filename = saveFileDialog1.FileName;
                  // створюємо файл і одразу записуємо корневі єлементи з файлу зразка
                  File.Copy("examples\\maintenance.xml", filename);
                  // активируємо корінний вузол XML
                  XDocument xdoc = XDocument.Load(filename);
                  XElement root = xdoc.Element("Maintenances");
                  // виконуємо запис показників S.M.A.R.T.
                  for(int i=0;i<dataGridView3.Rows.Count;i++){
                  root.Add(
                              new XElement("maintenance",
                              new XAttribute("id",i),
                              new XElement("inventory",
dataGridView3.Rows[i].Cells[1].Value),
                              new XElement("Part",
dataGridView3.Rows[i].Cells[2].Value),
                              new XElement("Crash",
dataGridView3.Rows[i].Cells[3].Value),
                              new XElement("date",
dataGridView3.Rows[i].Cells[4].Value)
                               )

                             );
                  }
                  // зберігаємо файл
                  xdoc.Save(filename);
            }

            void Button4Click(object sender, EventArgs e) // расположение выпадающего
списка открытия файла ТО под кнопкой открыть файл
            {
                  Point screenPoint = button4.PointToScreen(new Point(button4.Left,
button4.Bottom));
                  if (screenPoint.Y + contextMenuStrip1.Size.Height >
Screen.PrimaryScreen.WorkingArea.Height) {
                  contextMenuStrip1.Show(button4, new Point(0, -
contextMenuStrip1.Size.Height));
                  }
                  else {
                  contextMenuStrip1.Show(button4, new Point(0, button4.Height));
                  }
            }
```

ДОДАТОК Г

Повний код програмного продукту

```
using System;
using System.Drawing;
using System.Windows.Forms;
using System.Management;
using System.Diagnostics;
using System.IO;
using System.Xml.Linq;
using System.ServiceProcess;
using System.Linq;

namespace WIN_INFO_APP
{
	/// <summary>
	/// Description of Form1.
	/// </summary>
	public partial class Form1 : Form
	{
		public struct SMART
			{
				public SMARTAttribute RawReadErrorRate; //1
				public SMARTAttribute SpinUpTime; //3
				public SMARTAttribute StartStopCount; //4
				public SMARTAttribute ReallocatedSectorCount; //5
				public SMARTAttribute SeekErrorRate; //7
				public SMARTAttribute SpinRetryCount;//10
				public SMARTAttribute Temperature; //194
				public SMARTAttribute PowerCycleCount; //12
			}

		public struct SMARTAttribute
			{
				public int value;
				public int worstvvalue;
				public int rawvalue;
			}
```

```
		public Form1()
		{
			//
			// The InitializeComponent() call is required for Windows Forms designer support.
			//
			InitializeComponent();
			//
			// TODO: Add constructor code after the InitializeComponent() call.
			//
		}
		void Label5Click(object sender, EventArgs e) // возврат в форму краткого отображения информации
		{
			MainForm small = new MainForm();
			small.Show();
			small.Top=this.Top;
			small.Left=this.Left;
			this.Hide();
		}

		void TabControl1SelectedIndexChanged(object sender, EventArgs e) // переключение вкладок
		{
			switch(tabControl1.SelectedIndex){
				case 0:
					GetInfo();
					break;
				case 1:
					panel4.Hide();
					dataGridView1.Rows.Clear();
					GetSMART();
				break;
			case 2: // процессы
		   button7.Hide();
		   GetProcess();
					break;
	    case 3:// службы
		   GetServices();
		   break;
			case 4: //Тех обслуживание
				MaintenanceLoad();
				break;
```

```
                }
            }

        void GetServices()
        {
            ServiceController[] scServices;
            scServices = ServiceController.GetServices();
            int j = 0;
            dataGridView4.Rows.Clear();
            foreach (ServiceController scTemp in scServices)
            {
                if (scTemp.Status == ServiceControllerStatus.Running)
                {
                    // Write the service name and the display name
                    // for each running service.
                    dataGridView4.Rows.Add();
                    // выводим id и имя процесса
                    dataGridView4.Rows[j].Cells[0].Value = scTemp.ServiceName.ToString();
                    dataGridView4.Rows[j].Cells[1].Value = scTemp.DisplayName.ToString();
                    j++;
                }
            }
        }

            void GetInfo(){ // управление получением информации
                CreateNodes();
                GetProcessorInfo();
                GetMotherBoardInfo();
                GetRAMInfo();
                GetVideoInfo();
                GetHDDInfo();
            }

            void GetProcessorInfo(){ // инфа про процессор
                ManagementObjectSearcher searcher = new
ManagementObjectSearcher("SELECT * FROM Win32_Processor");
                    foreach (ManagementObject wmi in searcher.Get())
                    {
                     try
                        {
                            treeView1.Nodes["CPU"].Nodes.Add("Name",
"Процесор: " + wmi.GetPropertyValue("Name").ToString());
```

```
                                        treeView1.Nodes["CPU"].Nodes.Add("Manufacturer",
"Виробник: " + wmi.GetPropertyValue("Manufacturer").ToString());
                                        treeView1.Nodes["CPU"].Nodes.Add("Count",
"Кількість ядер: " + Environment.ProcessorCount);
                                        treeView1.Nodes["CPU"].Nodes.Add("Description",
"Опис: " + wmi.GetPropertyValue("Description").ToString());
                                    }
                                catch { }
                            }
            }

            void GetMotherBoardInfo(){ // инфа о маме
                    ManagementObjectSearcher searcher = new
ManagementObjectSearcher("SELECT Manufacturer,Product, SerialNumber,Version FROM
Win32_BaseBoard");
                    ManagementObjectCollection information = searcher.Get();
                    foreach (ManagementObject obj in information)
                    {

      treeView1.Nodes["MainBoard"].Nodes.Add("Manufacturer","Виробник: "+
obj.GetPropertyValue("Manufacturer").ToString());

      treeView1.Nodes["MainBoard"].Nodes.Add("Product","Модель: "+
obj.GetPropertyValue("Product").ToString());

      treeView1.Nodes["MainBoard"].Nodes.Add("SerialNumber","Серійний номер: "+
obj.GetPropertyValue("SerialNumber").ToString());

      treeView1.Nodes["MainBoard"].Nodes.Add("Version","Версія: "+
obj.GetPropertyValue("Version").ToString());
                    }
            }

            void GetRAMInfo(){ // инфа о озу
                    ManagementObjectSearcher searcher = new
ManagementObjectSearcher("SELECT * FROM Win32_PhysicalMemory");
                    ManagementObjectCollection ram = searcher.Get();
                    string[]
strFormFactor={"Unknown","Other","SIP","DIP","ZIP","SOJ","Proprietary","SIMM","DIMM","TSO
P","PGA","RIMM","SODIMM","SRIMM","SMD","SSMP","QFP","TQFP","SOIC","LCC","PLCC","BGA","FPB
GA","LGA"};
                    string[] strMemoryType={"Unknown","Other","DRAM","Synchronous
DRAM","Cache
```

```
DRAM","EDO","EDRAM","VRAM","SRAM","RAM","ROM","Flash","EEPROM","FEPROM","EPROM","CDRAM","
3DRAM","SDRAM","SGRAM","RDRAM","DDR","DDR2","DDR2 FB-DIMM","","DDR3","FBD2"};
                    int indexRam=0;
                    foreach (ManagementObject qwe in ram)
                    {

                          try{
                              treeView1.Nodes["RAM"].Nodes.Add("DeviceLocator","Слот: "+
qwe.GetPropertyValue("DeviceLocator").ToString());

treeView1.Nodes["RAM"].Nodes[indexRam].Nodes.Add("Manufacturer","Виробник: "+
qwe.GetPropertyValue("Manufacturer").ToString());

      treeView1.Nodes["RAM"].Nodes[indexRam].Nodes.Add("MemoryType","Тип пам'яті: "+
strMemoryType[Convert.ToInt32(qwe.GetPropertyValue("MemoryType"))]);

      treeView1.Nodes["RAM"].Nodes[indexRam].Nodes.Add("Formfactor","Форм фактор: "+
strFormFactor[Convert.ToInt32(qwe.GetPropertyValue("FormFactor"))]);

      treeView1.Nodes["RAM"].Nodes[indexRam].Nodes.Add("Capasity","Об'єм: "+
Convert.ToDouble(qwe.GetPropertyValue("Capacity"))/1024/1024/1024 + " Гб");

      treeView1.Nodes["RAM"].Nodes[indexRam].Nodes.Add("Speed","Швидкість: "+
qwe.GetPropertyValue("Speed").ToString());

      treeView1.Nodes["RAM"].Nodes[indexRam].Nodes.Add("SerialNumber","Серійний номер:
"+ qwe.GetPropertyValue("SerialNumber").ToString());
                                indexRam++;
                          }
                          catch{};
                    }
              }

              void GetHDDInfo(){
                    //                                Локальные диски
                          int index=0;
                          foreach (DriveInfo dI in DriveInfo.GetDrives())
                          {

                                if(dI.IsReady == true){
                                      treeView1.Nodes["HDD"].Nodes.Add("Name", "Диск:
" + dI.Name);
```

```
      //treeView1.Nodes["HDD"].Nodes[index].Nodes.Add("Format", "Формат диску: " +
dI.DriveFormat);
                                          double
Size=Convert.ToDouble(dI.TotalSize)/1024/1024/1024;

      //treeView1.Nodes["HDD"].Nodes[index].Nodes.Add("Size", "Розмір диску: " +
Size.ToString("F"+2) + " Гб");
                                          double
FreeSize=Convert.ToDouble(dI.AvailableFreeSpace)/1024/1024/1024;

      //treeView1.Nodes["HDD"].Nodes[index].Nodes.Add("FreeSize", "Вільне місце на
диску: " + FreeSize.ToString("F" +2) +" Гб");
                                    }
                                    index++;
                              }
            }

            void GetVideoInfo(){ // инфа видео адаптер
                  ManagementObjectSearcher searcher = new
ManagementObjectSearcher("SELECT * FROM Win32_VideoController");
                        int indexVideo=0;
                        foreach (ManagementObject wmi in searcher.Get())
                        {
                         try
                               {
                                     treeView1.Nodes["GPU"].Nodes.Add("Name", "Назва:
" + wmi.GetPropertyValue("Name").ToString());

      treeView1.Nodes["GPU"].Nodes[indexVideo].Nodes.Add("VideoProcessor", "Відео
процесор: " + wmi.GetPropertyValue("VideoProcessor").ToString());

      treeView1.Nodes["GPU"].Nodes[indexVideo].Nodes.Add("DriverVersion", "Версія
драйверу: " + wmi.GetPropertyValue("DriverVersion").ToString());
                                          double
AdapterRAM=Convert.ToDouble(wmi.GetPropertyValue("AdapterRAM"))/1024/1024/1024;

      treeView1.Nodes["GPU"].Nodes[indexVideo].Nodes.Add("AdapterRAM", "Об'єм пам'яті: "
+ AdapterRAM.ToString("F"+ 2) + " Гб");
                                          indexVideo++;
                               }
                          catch { }
                        }
```

```
            }

            void Clear(){
                   comboBox3.Items.Clear();
            }

            void GetSMART(){ // получение смарта

                                ManagementObjectSearcher WMISearch =new
ManagementObjectSearcher("Select * from Win32_DiskDrive");
                                       ManagementObjectCollection Drives = WMISearch.Get();
                                       int ind=comboBox3.SelectedIndex;
                        //Clear();
                        //foreach ( ManagementObject Drive in Drives )
                        //{
                        //
comboBox3.Items.Add(Drive.Properties["Model"].Value.ToString());
                        //}
                                       SMART smartdata;
                                       int IsTemperature=0;
                                       dataGridView1.Rows.Clear();
                                       WMISearch.Scope=new ManagementScope(@"\root\wmi" );
                                       WMISearch.Query=new ObjectQuery("Select * from
MSStorageDriver_FailurePredictData");
                                       ManagementObjectCollection FailDataSet =
WMISearch.Get();
                                       ManagementObject mo =
FailDataSet.OfType<ManagementObject>().ElementAtOrDefault(ind);
                                       try{
                                              Byte[] data2 =
(Byte[])mo.Properties["VendorSpecific"].Value;
                                              for (int i = 0; i < data2[0] – 1; i++)
                                              {
                                                     int start=i*12;
                                                     switch(data2[start+2])
                                                     {
                                                            case 1:

      smartdata.RawReadErrorRate.value = data2[start + 5];

      smartdata.RawReadErrorRate.worstvvalue=data2[start+6];

      smartdata.RawReadErrorRate.rawvalue=data2[start+7];
```

```
        this.dataGridView1.Rows.Add();

        this.dataGridView1.Rows[0].Cells[0].Value ="1";

        this.dataGridView1.Rows[0].Cells[1].Value ="Raw Read Error Rate";

        this.dataGridView1.Rows[0].Cells[2].Value
=smartdata.RawReadErrorRate.value.ToString();

        this.dataGridView1.Rows[0].Cells[3].Value
=smartdata.RawReadErrorRate.worstvvalue.ToString();

        this.dataGridView1.Rows[0].Cells[4].Value
=smartdata.RawReadErrorRate.rawvalue.ToString();

        if(smartdata.RawReadErrorRate.rawvalue>10)
dataGridView1.Rows[0].Cells[0].Style.BackColor=Color.Yellow;

        if(smartdata.RawReadErrorRate.rawvalue>10)
dataGridView1.Rows[0].Cells[1].Style.BackColor=Color.Yellow;

        if(smartdata.RawReadErrorRate.rawvalue>10)
dataGridView1.Rows[0].Cells[2].Style.BackColor=Color.Yellow;

        if(smartdata.RawReadErrorRate.rawvalue>10)
dataGridView1.Rows[0].Cells[3].Style.BackColor=Color.Yellow;

        if(smartdata.RawReadErrorRate.rawvalue>10)
dataGridView1.Rows[0].Cells[4].Style.BackColor=Color.Yellow;
                                                        break;
                                                        case 3:
                                                               smartdata.SpinUpTime.value =
data2[start + 5];

        smartdata.SpinUpTime.worstvvalue=data2[start+6];

        smartdata.SpinUpTime.rawvalue=data2[start+7];

        this.dataGridView1.Rows.Add();

        this.dataGridView1.Rows[1].Cells[0].Value ="3";
```

```
        this.dataGridView1.Rows[1].Cells[1].Value ="Spin Up Time";

        this.dataGridView1.Rows[1].Cells[2].Value =smartdata.SpinUpTime.value.ToString();

        this.dataGridView1.Rows[1].Cells[3].Value
=smartdata.SpinUpTime.worstvvalue.ToString();

        this.dataGridView1.Rows[1].Cells[4].Value
=smartdata.SpinUpTime.rawvalue.ToString();
                                                        break;
                                                        case 4:

        smartdata.StartStopCount.value = data2[start + 5];

        smartdata.StartStopCount.worstvvalue=data2[start+6];

        smartdata.StartStopCount.rawvalue=data2[start+7];

        this.dataGridView1.Rows.Add();

        this.dataGridView1.Rows[2].Cells[0].Value ="4";

        this.dataGridView1.Rows[2].Cells[1].Value ="Start/Stop Count";

        this.dataGridView1.Rows[2].Cells[2].Value
=smartdata.StartStopCount.value.ToString();

        this.dataGridView1.Rows[2].Cells[3].Value
=smartdata.StartStopCount.worstvvalue.ToString();

        this.dataGridView1.Rows[2].Cells[4].Value
=smartdata.StartStopCount.rawvalue.ToString();
                                                        break;
                                                        case 5:

        smartdata.ReallocatedSectorCount.value = data2[start + 5];

        smartdata.ReallocatedSectorCount.worstvvalue=data2[start+6];

        smartdata.ReallocatedSectorCount.rawvalue=data2[start+7];

        this.dataGridView1.Rows.Add();
```

```
        this.dataGridView1.Rows[3].Cells[0].Value ="5";

        this.dataGridView1.Rows[3].Cells[1].Value ="RealLocated Sector Count";

        this.dataGridView1.Rows[3].Cells[2].Value
=smartdata.ReallocatedSectorCount.value.ToString();

        this.dataGridView1.Rows[3].Cells[3].Value
=smartdata.ReallocatedSectorCount.worstvvalue.ToString();

        this.dataGridView1.Rows[3].Cells[4].Value
=smartdata.ReallocatedSectorCount.rawvalue.ToString();

        if(smartdata.ReallocatedSectorCount.rawvalue>5)
dataGridView1.Rows[3].Cells[0].Style.BackColor=Color.Red;

        if(smartdata.ReallocatedSectorCount.rawvalue>5)
dataGridView1.Rows[3].Cells[1].Style.BackColor=Color.Red;

        if(smartdata.ReallocatedSectorCount.rawvalue>5)
dataGridView1.Rows[3].Cells[2].Style.BackColor=Color.Red;

        if(smartdata.ReallocatedSectorCount.rawvalue>5)
dataGridView1.Rows[3].Cells[3].Style.BackColor=Color.Red;

        if(smartdata.ReallocatedSectorCount.rawvalue>5)
dataGridView1.Rows[3].Cells[4].Style.BackColor=Color.Red;
                                                        break;
                                                        case 7:

        smartdata.SeekErrorRate.value = data2[start + 5];

        smartdata.SeekErrorRate.worstvvalue=data2[start+6];

        smartdata.SeekErrorRate.rawvalue=data2[start+7];

        this.dataGridView1.Rows.Add();

        this.dataGridView1.Rows[4].Cells[0].Value ="7";

        this.dataGridView1.Rows[4].Cells[1].Value ="Seek Error Rate";
```

```
       this.dataGridView1.Rows[4].Cells[2].Value
=smartdata.SeekErrorRate.value.ToString();

       this.dataGridView1.Rows[4].Cells[3].Value
=smartdata.SeekErrorRate.worstvvalue.ToString();

       this.dataGridView1.Rows[4].Cells[4].Value
=smartdata.SeekErrorRate.rawvalue.ToString();
                                                    break;
                                                    case 10:

       smartdata.SpinRetryCount.value = data2[start + 5];

       smartdata.SpinRetryCount.worstvvalue=data2[start+6];

       smartdata.SpinRetryCount.rawvalue=data2[start+7];

       this.dataGridView1.Rows.Add();

       this.dataGridView1.Rows[5].Cells[0].Value ="10";

       this.dataGridView1.Rows[5].Cells[1].Value ="Spin Retry Count";

       this.dataGridView1.Rows[5].Cells[2].Value
=smartdata.SpinRetryCount.value.ToString();

       this.dataGridView1.Rows[5].Cells[3].Value
=smartdata.SpinRetryCount.worstvvalue.ToString();

       this.dataGridView1.Rows[5].Cells[4].Value
=smartdata.SpinRetryCount.rawvalue.ToString();
                                                    break;
                                                    case 12:

       smartdata.PowerCycleCount.value = data2[start + 5];

       smartdata.PowerCycleCount.worstvvalue=data2[start+6];

       smartdata.PowerCycleCount.rawvalue=data2[start+7];

       this.dataGridView1.Rows.Add();
```

```
        this.dataGridView1.Rows[6].Cells[0].Value ="12";

        this.dataGridView1.Rows[6].Cells[1].Value ="Power Cycle Count";

        this.dataGridView1.Rows[6].Cells[2].Value
=smartdata.PowerCycleCount.value.ToString();

        this.dataGridView1.Rows[6].Cells[3].Value
=smartdata.PowerCycleCount.worstvvalue.ToString();

        this.dataGridView1.Rows[6].Cells[4].Value
=smartdata.PowerCycleCount.rawvalue.ToString();
                                                        break;
                                                        // получаем температуру если диск
имеет станрадтный смарт
                                                        case 194:
                                                               smartdata.Temperature.value
= data2[start+5];

        smartdata.Temperature.worstvvalue=data2[start+6];

        smartdata.Temperature.rawvalue=data2[start+7];

        this.dataGridView1.Rows.Add();

        this.dataGridView1.Rows[7].Cells[0].Value ="194";

        this.dataGridView1.Rows[7].Cells[1].Value ="Temperature";

        this.dataGridView1.Rows[7].Cells[2].Value =smartdata.Temperature.value.ToString();

        this.dataGridView1.Rows[7].Cells[3].Value
=smartdata.Temperature.worstvvalue.ToString();

        this.dataGridView1.Rows[7].Cells[4].Value
=smartdata.Temperature.rawvalue.ToString();
                                                               IsTemperature=1;
                                                        break;


                                                }
```

```
                                                // получаем температуру из записей если
диск имеет не стандартный смарт
                                                // и не выводим значение пока не будет 5
строки предыдущие выведено
                                                if
((IsTemperature==0)&&(dataGridView1.Rows.Count==7)){
                                                       smartdata.Temperature.value =
data2[173];

       smartdata.Temperature.worstvvalue=data2[174];

       smartdata.Temperature.rawvalue=data2[175];
                                                       dataGridView1.Rows.Add();

       this.dataGridView1.Rows[7].Cells[0].Value ="194";

       this.dataGridView1.Rows[7].Cells[1].Value ="Temperature";

       this.dataGridView1.Rows[7].Cells[2].Value =smartdata.Temperature.value.ToString();

       this.dataGridView1.Rows[7].Cells[3].Value
=smartdata.Temperature.worstvvalue.ToString();

       this.dataGridView1.Rows[7].Cells[4].Value
=smartdata.Temperature.rawvalue.ToString();
                                                       }

                                               }
                                               }
                                               catch{
                                                       MessageBox.Show("Не можливо отримати
значення");
                                               }

             }

             void GetProcess(){ // получение списка процесов
                    int j=0;
                           dataGridView2.Rows.Clear();
                           foreach(Process process in Process.GetProcesses())
                           {
                                  dataGridView2.Rows.Add();
                           // выводим id и имя процесса
```

```
                        dataGridView2.Rows[j].Cells[0].Value = process.Id.ToString();
                        dataGridView2.Rows[j].Cells[1].Value =
process.ProcessName.ToString();
                        j++;
                        }

            }

            void CreateNodes(){// создание записей в информацию
                  treeView1.Nodes.Clear();
                  treeView1.Nodes.Add("System", "Операційна система");
                  treeView1.Nodes.Add("CPU","Процесор");
                  treeView1.Nodes.Add("MainBoard","Материнська плата");
                  treeView1.Nodes.Add("RAM","Оперативна  пам'ять");
                  treeView1.Nodes.Add("GPU","Відеокарта");
                  treeView1.Nodes.Add("HDD","Жорсткий диск");
                  treeView1.Nodes["System"].Nodes.Add("OSVersion", "Операционная
система: " + Environment.OSVersion);
            treeView1.Nodes["System"].Nodes.Add("SystemDirectory", "Системна
директорія: " + Environment.SystemDirectory);
            treeView1.Nodes["System"].Nodes.Add("UserName", "Ім'я користувача: "+
Environment.UserName);
            treeView1.Nodes["System"].Nodes.Add("HostName","Мережеве ім'я:
"+System.Net.Dns.GetHostName());
            }

            void Form1FormClosed(object sender, FormClosedEventArgs e)// закрытие
приложения
            {
                  Application.Exit();
            }

            void Button1Click(object sender, EventArgs e) // сохранение смарта
            {
                  if (saveFileDialog1.ShowDialog() == DialogResult.Cancel)
          return;
                   // получаем выбранный файл
                  string filename = saveFileDialog1.FileName;
                  // создаем файл и тут же записываем корневые элементы из файла
образца
                  File.Copy("examples\\smart.xml", filename);
                  // активируем корневой узел XML
                  XDocument xdoc = XDocument.Load(filename);
```

```
                    XElement root = xdoc.Element("SMARTS");
                    // производим запись показателей SMART
                    for(int i=0;i<8;i++){
                    root.Add(
                                    new XElement("smart",
                                    new XElement("id",
dataGridView1.Rows[i].Cells[0].Value),
                                    new XElement("current",
dataGridView1.Rows[i].Cells[2].Value),
                                    new XElement("worst",
dataGridView1.Rows[i].Cells[3].Value),
                                    new XElement("raw",
dataGridView1.Rows[i].Cells[4].Value),
                                    new XElement("hostname",System.Net.Dns.GetHostName())
                            )
                    );
                    }
                    // сохраняем файл
                    xdoc.Save(filename);

            }

            void Button2Click(object sender, EventArgs e) // открытие файла смарт
            {
                    panel4.Show();
                    int j=0;
                    try{
                    if (openFileDialog1.ShowDialog() == DialogResult.Cancel) return;
                    XDocument xdoc = XDocument.Load(openFileDialog1.FileName);
                    XElement root = xdoc.Element("SMARTS");
                    foreach (XElement BDelement in root.Elements("smart"))
                    {
                            dataGridView1.Rows[j].Cells[0].Value =
BDelement.Element("id").Value;
                    dataGridView1.Rows[j].Cells[2].Value =
BDelement.Element("current").Value;
                    dataGridView1.Rows[j].Cells[3].Value =
BDelement.Element("worst").Value;
                    dataGridView1.Rows[j].Cells[4].Value =
BDelement.Element("raw").Value;
                    label8.Text="Режим перегляду файлу S.M.A.R.T.
"+BDelement.Element("hostname").Value;
                    j++;
```

```
                        }
                }
                catch{label8.Text="Невдалось відкрити файл";}
        }

        void Form1Load(object sender, EventArgs e) // загрузка формы
        {
                panel5.Width=23;
                panel4.Hide();
        ManagementObjectSearcher WMISearch = new ManagementObjectSearcher("Select *
from Win32_DiskDrive");
        ManagementObjectCollection Drives = WMISearch.Get();
        Clear();
        foreach (ManagementObject Drive in Drives)
        {
            comboBox3.Items.Add(Drive.Properties["Model"].Value.ToString());
        }
        comboBox3.SelectedIndex = 0;
                comboBox1.Width=542;
                comboBox2.Width=542;
                label6.Width=542;
                label7.Width=542;
                button5.Left=213;
                textBox1.Width=542;
                label9.Width=542;
                GetInfo();

                label10.Text=DateTime.Today.ToString();
        }

        void Label4Click(object sender, EventArgs e) // управление боковым меню
вкладки ТО
        {
                if(label4.Text==">"){
                        panel5.Width=23;
                        label4.Text="<";
                        comboBox1.Width=542;
                        comboBox2.Width=542;
                        label6.Width=542;
                        label7.Width=542;
                        button5.Left=213;
                        textBox1.Width=542;
                        label9.Width=542;
```

```
                }
                else{
                        panel5.Width=242;
                        label4.Text=">";
                        comboBox1.Width=312;
                        comboBox2.Width=312;
                        label6.Width=312;
                        label7.Width=312;
                        button5.Left=103;
                        textBox1.Width=312;
                        label9.Width=312;
                }
        }

        void Button5Click(object sender, EventArgs e) // заполнение ТО
        {
                if((comboBox1.Text!="")&&(comboBox2.Text!="")){
                int count=0;

                XDocument xdoc = XDocument.Load("Maintenance.xml");
                foreach (XElement Element in
xdoc.Element("Maintenances").Elements("maintenance"))
                {
                        count++;
                }
                XElement root = xdoc.Element("Maintenances");
                root.Add(
                                new XElement("maintenance",
                                new XAttribute("id",count+1),
                                new XElement("inventory", textBox1.Text),
                                new XElement("Part", comboBox1.Text),
                                new XElement("Crash", comboBox2.Text),
                                new XElement("date", DateTime.Today)
                                )

                );
                xdoc.Save("Maintenance.xml");
                }
                else{
                        MessageBox.Show("Заповніть всі неодхіні поля");
                }
                MaintenanceLoad();
```

```
            }

            void MaintenanceLoad(){ // загрузка програмного файла ТО в таблицу
                  int j=0;
                  dataGridView3.Rows.Clear();
                  XDocument xdoc = XDocument.Load("Maintenance.xml");
                  foreach (XElement BDelement in
xdoc.Element("Maintenances").Elements("maintenance"))
                  {
                        dataGridView3.Rows.Add();
                        dataGridView3.Rows[j].Cells[0].Value =
BDelement.Attribute("id").Value;
                  dataGridView3.Rows[j].Cells[1].Value =
BDelement.Element("inventory").Value;
                  dataGridView3.Rows[j].Cells[2].Value =
BDelement.Element("Part").Value;
                  dataGridView3.Rows[j].Cells[3].Value =
BDelement.Element("Crash").Value;
                  dataGridView3.Rows[j].Cells[4].Value =
BDelement.Element("date").Value;
                  j++;
                  }
            }

            void ПопередняВерсіяФайлууЗвітуЗЦьогоПКToolStripMenuItemClick(object
sender, EventArgs e) // загрузка файла выбраного пользователем и слияние с программным
файлом ТО
            {
                  int j=0;
                  if (openFileDialog1.ShowDialog() == DialogResult.Cancel)
           return;
                  XDocument xdoc = XDocument.Load(openFileDialog1.FileName);
                  foreach (XElement BDelement in
xdoc.Element("Maintenances").Elements("maintenance"))
                  {
                        dataGridView3.Rows.Add();
                        dataGridView3.Rows[j].Cells[0].Value =
BDelement.Attribute("id").Value;
                  dataGridView3.Rows[j].Cells[1].Value =
BDelement.Element("inventory").Value;
                  dataGridView3.Rows[j].Cells[2].Value =
BDelement.Element("Part").Value;
```

```
                    dataGridView3.Rows[j].Cells[3].Value =
BDelement.Element("Crash").Value;
                    dataGridView3.Rows[j].Cells[4].Value =
BDelement.Element("date").Value;
                    j++;
                    }
                    xdoc = XDocument.Load("Maintenance.xml");
                    foreach (XElement BDelement in
xdoc.Element("Maintenances").Elements("maintenance"))
                    {
                           dataGridView3.Rows.Add();
                           dataGridView3.Rows[j].Cells[0].Value =
BDelement.Attribute("id").Value;
                    dataGridView3.Rows[j].Cells[1].Value =
BDelement.Element("inventory").Value;
                    dataGridView3.Rows[j].Cells[2].Value =
BDelement.Element("Part").Value;
                    dataGridView3.Rows[j].Cells[3].Value =
BDelement.Element("Crash").Value;
                    dataGridView3.Rows[j].Cells[4].Value =
BDelement.Element("date").Value;
                    j++;
                    }
                    xdoc.Element("Maintenances").RemoveAll();
                    for(int i=0;i<dataGridView3.Rows.Count-1;i++){
                           try{
                           XElement root = xdoc.Element("Maintenances");
                           root.Add(
                                  new XElement("maintenance",
                                  new XAttribute("id",i),
                                  new XElement("inventory", textBox1.Text),
                                  new XElement("Part", comboBox1.Text),
                                  new XElement("Crash", comboBox2.Text),
                                  new XElement("date", DateTime.Today)
                            )

                           );
                           }
                           catch{ };
                    xdoc.Save("Maintenance.xml");

                    }
             }
```

```
            void Button3Click(object sender, EventArgs e) // сохранение файла ТО с
именем указамым пользователем и по пути указаным пользователем
            {
                  if (saveFileDialog1.ShowDialog() == DialogResult.Cancel)
           return;
                   // получаем выбранный файл
                  string filename = saveFileDialog1.FileName;
                  // создаем файл и тут же записываем корневые элементы из файла
образца
                  File.Copy("examples\\maintenance.xml", filename);
                  // активируем корневой узел XML
                  XDocument xdoc = XDocument.Load(filename);
                  XElement root = xdoc.Element("Maintenances");
                  // производим запись показателей SMART
                  for(int i=0;i<dataGridView3.Rows.Count;i++){
                  root.Add(
                              new XElement("maintenance",
                              new XAttribute("id",i),
                              new XElement("inventory",
dataGridView3.Rows[i].Cells[1].Value),
                              new XElement("Part",
dataGridView3.Rows[i].Cells[2].Value),
                              new XElement("Crash",
dataGridView3.Rows[i].Cells[3].Value),
                              new XElement("date",
dataGridView3.Rows[i].Cells[4].Value)
                               )

                             );
                  }
                  // сохраняем файл
                  xdoc.Save(filename);
            }

            void Button4Click(object sender, EventArgs e) // расположение выпадающего
списка открытия файла ТО под кнопкой открыть файл
            {
                  Point screenPoint = button4.PointToScreen(new Point(button4.Left,
button4.Bottom));
                  if (screenPoint.Y + contextMenuStrip1.Size.Height >
Screen.PrimaryScreen.WorkingArea.Height) {
```

```
                    contextMenuStrip1.Show(button4, new Point(0, –
contextMenuStrip1.Size.Height));
                    }
                    else {
                    contextMenuStrip1.Show(button4, new Point(0, button4.Height));
                    }
             }

             void Button6Click(object sender, EventArgs e) // открытие справки смарт в
браузере по умолчанию
             {
                    Process.Start("smarthelp.html");
             }

             void ЗІншогоПКToolStripMenuItemClick(object sender, EventArgs e) //
открытие файла выбраного пользователем и загрузка его в таблицу
             {
                    int j = 0;
                    if (openFileDialog1.ShowDialog() == DialogResult.Cancel)
                           return;
                    XDocument xdoc = XDocument.Load(openFileDialog1.FileName);
                    foreach (XElement BDelement in
xdoc.Element("Maintenances").Elements("maintenance")) {
                           dataGridView3.Rows.Add();
                           dataGridView3.Rows[j].Cells[0].Value =
BDelement.Attribute("id").Value;
                           dataGridView3.Rows[j].Cells[1].Value =
BDelement.Element("inventory").Value;
                           dataGridView3.Rows[j].Cells[2].Value =
BDelement.Element("Part").Value;
                           dataGridView3.Rows[j].Cells[3].Value =
BDelement.Element("Crash").Value;
                           dataGridView3.Rows[j].Cells[4].Value =
BDelement.Element("date").Value;
                           j++;
                    }
             }
             void Panel1Paint(object sender, PaintEventArgs e)
             {

             }
             void ComboBox3SelectedIndexChanged(object sender, EventArgs e)
             {
```

```
                GetSMART();
            }


        private void button7_Click(object sender, EventArgs e)
        {
            richTextBox1.Hide();
            button7.Hide();
        }

        private void інформаціяToolStripMenuItem_Click(object sender, EventArgs e)
        {
            richTextBox1.Show();
            richTextBox1.Clear();
            button7.Show();
            Process[] proc = Process.GetProcesses();
            foreach (Process process in proc)
                if (process.ProcessName ==
dataGridView2.Rows[dataGridView2.CurrentCell.RowIndex].Cells[1].Value.ToString())
                {
                    richTextBox1.Text += "Процес" + "\n";
                    richTextBox1.Text += "Name: " + process.ProcessName.ToString() +
"\n";
                    richTextBox1.Text += "ID: " + process.Id.ToString() + "\n";
                    richTextBox1.Text += "MachineName: " + process.MachineName.ToString()
+ "\n";
                    richTextBox1.Text += "Modules: " + process.Modules.ToString() + "\n";
                    richTextBox1.Text += "StartTime: " + process.StartTime.ToString() +
"\n";
                    richTextBox1.Text += "VirtualMemorySize64: " + process.Id.ToString()
+ "\n";
                    richTextBox1.Text += "Модулі" + "\n";
                    ProcessModuleCollection modules = process.Modules;
                    foreach(ProcessModule module in modules)
                        {
                            richTextBox1.Text += "\n";
                            richTextBox1.Text += "ModuleName: " +
module.ModuleName.ToString() + "\n";
                            richTextBox1.Text += "BaseAddress: " +
module.BaseAddress.ToString() + "\n";
                            richTextBox1.Text += "FileName: " +
module.FileName.ToString() + "\n";
```

```
                            richTextBox1.Text += "EntryPointAddress: " +
module.EntryPointAddress.ToString() + "\n";
                        richTextBox1.Text += "ModuleMemorySize:
"+module.ModuleMemorySize.ToString();
                        }
                    }
            richTextBox1.Text += "\n";
        }

        private void завершитПроцесToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Process[] proc = Process.GetProcesses();
            foreach (Process process in proc)
                if (process.ProcessName ==
dataGridView2.Rows[dataGridView2.CurrentCell.RowIndex].Cells[1].Value.ToString())
                {
                    process.Kill();
                }
        }

        private void dataGridView2_CellClick(object sender, DataGridViewCellEventArgs e)
        {
            dataGridView2.ContextMenuStrip.Show(Cursor.Position);
        }

        private void label5_MouseEnter(object sender, EventArgs e)
        {
            label5.BackColor = Color.Purple;
        }

        private void label5_MouseLeave(object sender, EventArgs e)
        {
            label5.BackColor = Color.DarkMagenta;
        }
        private void dataGridView4_CellClick(object sender, DataGridViewCellEventArgs e)
        {
            dataGridView4.ContextMenuStrip.Show(Cursor.Position);
        }

        private void інформаціяToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            try
            {
```

```
                ManagementObject wmiService;
                wmiService = new ManagementObject("Win32_Service.Name='" +
dataGridView4.Rows[dataGridView4.CurrentCell.RowIndex].Cells[0].Value.ToString() + "'");
                wmiService.Get();
                richTextBox2.Clear();
                richTextBox2.Show();
                richTextBox2.Text = wmiService["Description"].ToString();
                button8.Show();
            }
            catch { MessageBox.Show("Непередбачувана помилка"); }
        }

        private void button8_Click(object sender, EventArgs e)
        {
            richTextBox2.Hide();
            button8.Hide();
        }

        private void запуститиToolStripMenuItem_Click(object sender, EventArgs e)
        {
            ServiceController service = new
ServiceController(dataGridView4.Rows[dataGridView4.CurrentCell.RowIndex].Cells[0].Value.T
oString());
            // Проверяем не запущена ли служба
            if (service.Status != ServiceControllerStatus.Running)
            {
                // Запускаем службу
                service.Start();
                // В течении минуты ждём статус от службы
                service.WaitForStatus(ServiceControllerStatus.Running,
TimeSpan.FromMinutes(1));
                MessageBox.Show("Служба была успешно запущена!");
            }
            else
            {
                MessageBox.Show("Служба уже запущена!");
            }
        }
        private void зупинитиToolStripMenuItem_Click(object sender, EventArgs e)
        {
            ServiceController service = new
ServiceController(dataGridView4.Rows[dataGridView4.CurrentCell.RowIndex].Cells[0].Value.T
oString());
```

```
            // Если служба не остановлена
            if (service.Status != ServiceControllerStatus.Stopped)
            {
                // Останавливаем службу
                service.Stop();
                service.WaitForStatus(ServiceControllerStatus.Stopped,
TimeSpan.FromMinutes(1));
                MessageBox.Show("Служба была успешно остановлена!");
            }
            else
            {
                MessageBox.Show("Служба уже остановлена!");
            }
        }

        private void перезапуститиToolStripMenuItem_Click(object sender, EventArgs e)
        {
            ServiceController service = new
ServiceController(dataGridView4.Rows[dataGridView4.CurrentCell.RowIndex].Cells[0].Value.T
oString());
            TimeSpan timeout = TimeSpan.FromMinutes(1);
            if (service.Status != ServiceControllerStatus.Stopped)
            {
                MessageBox.Show("Перезапуск службы. Останавливаем службу...");
                // Останавливаем службу
                service.Stop();
                service.WaitForStatus(ServiceControllerStatus.Stopped, timeout);
                MessageBox.Show("Служба была успешно остановлена!");
            }
            if (service.Status != ServiceControllerStatus.Running)
            {
                MessageBox.Show("Перезапуск службы. Запускаем службу...");
                // Запускаем службу
                service.Start();
                service.WaitForStatus(ServiceControllerStatus.Running, timeout);
                MessageBox.Show("Служба была успешно запущена!");
            }
        }
            }
}
```