**Міністерство освіти і науки України**

**Національний університет “Львівська політехніка”**

Кафедра ЕОМ



**Курсова робота**

з предмету: «Програмування, частина 2 (Об’єктно-орієнтованепрограмування)»

на тему:

**«Базові принципи об’єктно-орієнтованогопрограмування»**

Індивідуальне завдання:«Фармацевтична компанія»

*Виконано згідно методичних матеріалів*

**Виконав:**

ст. гр. КІ-15

Рикмас О. М.

**Прийняла:**

асистент

Козак Н.Б.

Львів-2020

**Анотація**

Для розв’язання багатьох практичних задач з різних галузей, необхідно використовувати базові принципи об’єктно-орієнтованого програмування, оскільки вони можуть адаптувати вирішення поставлених задач до їх швидкої зміни в реальному часі. Одним з шляхів забезпечення таких вимог є організація об’єктно-орієнтованих процесів і відповідних технічних засобів їх реалізації.

Причому, ефективність застосуванняоб’єктно-орієнтованого програмування у вирішенні поставленого завдання залежить, як від продуктивності комп’ютерів, так і від розмірів і структури пам’яті, пропускної здатності каналів зв’язку,

використанихоб’єктно-орієнтованихмов програмування, компіляторів, операційних систем, чисельних методів та інших математичних досліджень. Такий широкий обсяг параметрів вимагає проведення досліджень на різних рівнях: на рівні опису алгоритмів, створення спеціальних мов програмування, компіляторів, багатопроцесорних систем, неоднорідних систем, кластерів.

Для скорочення термінів розробки об’єктно-орієнтованих програм та дослідження їх роботи використовується моделювання.

***Метою виконання курсової роботи*** є засвоєння основних методів таалгоритмів об’єктно-орієнтованихструктур, принципів побудови відповідних структур, набуття початкових практичних навиків проектування таких засобів.

ЗМІСТ

[Завдання на курсову роботу 4](#_Toc40339204)

[1. Огляд та обґрунтовування вибору технологій об'єктно-орієнтованого програмування. 5](#_Toc40339205)

[1. Аналіз та розробка алгоритму згідно індивідуального завдання. 6](#_Toc40339206)

[1.1. Опис алгоритму та оцінка його складності. 6](#_Toc40339207)

[1.2. Аналіз особливостей алгоритму. 6](#_Toc40339208)

[2. Програмна реалізація алгоритму згідно індивідуального завдання. 8](#_Toc40339209)

[2.1. Опис структур даних програми. 8](#_Toc40339210)

[2.2. Схема викликів функцій або діаграма класів. 8](#_Toc40339211)

[2.3. Опис реалізації основних функцій/методів. 9](#_Toc40339212)

[2.3.1. Функція зчитування файлу та вивід на консоль. 9](#_Toc40339213)

[2.3.2. Функція запису всієї інформації про вибраний товар в допоміжний файл. 10](#_Toc40339214)

[2.3.3. Функція зміни кількості товару та запис в допоміжний файл. 10](#_Toc40339215)

[2.3.4. Функція сортування товару за його номером. 11](#_Toc40339216)

[2.3.5. Функція очищення файлу з всією інформацією про певний товар. 12](#_Toc40339217)

[2.3.6. Функція перепису файлу зі всією інформацією про певний товар. 13](#_Toc40339218)

[3. Відлагодження та тестування реалізованої програми згідно індивідуального завдання. 14](#_Toc40339219)

[3.1. Відлагодження та тестування окремих частин програми. 14](#_Toc40339220)

[3.2. Відлагодження та тестування програми загалом. 14](#_Toc40339221)

[4. Аналіз продуктивності (часу роботи) та структурної складності реалізованої програмзгідно індивідуального завдання. 17](#_Toc40339222)

[4.1. Аналіз структурної складності виконання програми. 17](#_Toc40339223)

[4.2. Аналіз часу виконання програми в залежності від об'єму вхідних даних. 17](#_Toc40339224)

[Висновки 18](#_Toc40339225)

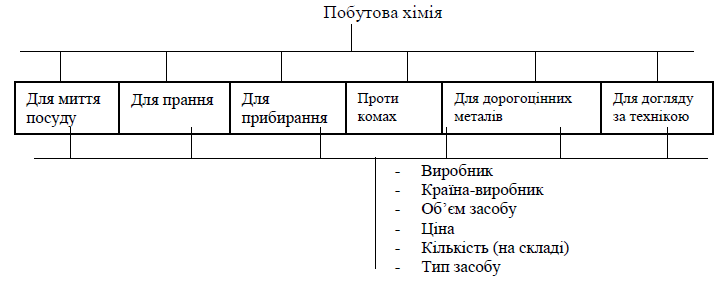
[Список літератури 19](#_Toc40339226)

[Додатки 20](#_Toc40339227)

[А. Лістинг модулів програми згідно індивідуального завдання 20](#_Toc40339228)

# Завдання на курсову роботу

Вхідні дані взяті з Інтернет - магазину «Розетка», а саме: назва товару, країна-виробник, ціна, об’єм, тип товару; так як я вибрав магазин побутової хімії, весь товар був розділений на 6 категорій: засоби для прання, засоби для миття посуду, засоби для прибирання, засоби проти шкідників, засоби по догляду за дорогоцінними металам та технікою.



*Рис. 1. Вигляд предметної галузі.*

# Огляд та обґрунтовування вибору технологій об'єктно-орієнтованого програмування.

Об'єктно-орієнтоване програмування (ООП) — одна з парадигм програмування, яка розглядає програму як множину «об'єктів», що взаємодіють між собою. Основу ООП складають три основні концепції: інкапсуляція, успадкування та поліморфізм. Одною з переваг ООП є краща модульність програмного забезпечення.

Успадкування (наслідування). Клас може мати «підкласи», спеціалізовані, розширені версії надкласу. Можуть навіть утворюватися цілі дерева успадкування. Підкласи успадковують атрибути та поведінку своїх батьківських класів, і можуть вводити свої власні. Успадкування може бути одиничне (один безпосередній батьківський клас) та множинне (кілька батьківських класів).

Інкапсуляція. Приховування деталей про роботу класів від об'єктів, що їх використовують або надсилають їм повідомлення. Інкапсуляція досягається шляхом вказування, які класи можуть звертатися до членів об'єкта. Як наслідок, кожен об'єкт представляє кожному іншому класу певний інтерфейс — члени, доступні іншим класам. Інкапсуляція потрібна для того, аби запобігти використанню користувачами інтерфейсу тих частин реалізації, які, швидше за все, будуть змінюватись. Це дозволить полегшити внесення змін, тобто, без потреби змінювати і користувачів інтерфейсу. Часто, члени класу позначаються як публічні (англ. public), захищені (англ. protected) та приватні (англ.private), визначаючи, чи доступні вони всім класам, підкласам, або лише до класу в якому їх визначено.

Поліморфізм. Поліморфізм означає залежність поведінки від класу, в якому ця поведінка викликається, тобто, два або більше класів можуть реагувати по-різному на однакові повідомлення. На практиці - це реалізовується шляхом реалізації ряду підпрограм (функцій, процедур, методів тощо) з однаковими іменами, але з різними параметрами. В залежності від того, що передається і вибирається відповідна підпрограма.

# Аналіз та розробка алгоритму згідно індивідуального завдання.

# Опис алгоритму та оцінка його складності.

В алгоритмі наявні 3 конструкції switch, що утворюють меню консольної програми. Перше розгалуження відповідає за головне меню програми, що включає такі можливості:

а) перейти до розділів товару;

б) перейти до корзини;

в) завершити роботу.

При виконанні переходу до розділів товару перед користувачем з’являється допоміжне меню вибору типу товару (всього їх 6). Після чого користувач отримує інформацію про товар, і може здійснювати купівлю товару (а саме збереження товару до корзини покупця).

При виконанні переходу до корзини перед користувачем з’являється інформація про щойно збережений товар (якщо він є), і тоді він може купити його, після чого підрахується ціна купленого товару і збережеться профіт магазину.

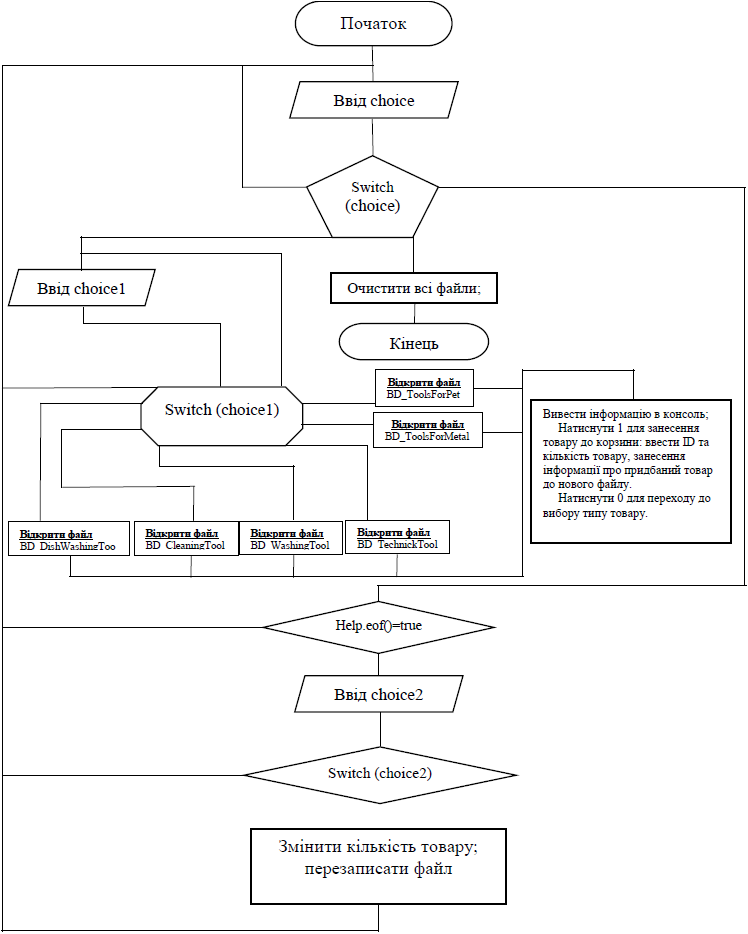
# Аналіз особливостей алгоритму.

В кожній конструкції switch викликаються методи класу.

а) в першій конструкції switch при виборі завершення роботи викликаються функції очищення всіх допоміжних файлів, та їх закриття.

б) в другій конструкції switch при виборі певного товару викликається функція одного з шести дочірних класів і зчитується певний файл, після чого користувач одержує всю інформацію про певний тип товару.

в) в третій конструкції switch викликаються функції зчитування купленого товару, його сортування, підрахунок ціни купленого товару та зміни кількості товару (в залежності від того скільки товару куплено) в головних файлах, де міститься вся інформація про товари.



*Рис. 2. Схема алгоритму роботи програми.*

# Програмна реалізація алгоритму згідно індивідуального завдання.

# Опис структур даних програми.

В програмі наявні змінні:

• цілого типу для виконання умови switch;

• цілого типу для зчитування з файлу, виконання обчислень та виведення в консоль/файл;

• масиви типу char для зчитування з файлу, виведення в консоль/файл;

• об’єкти класів для виконання методів.

• Порядок викликів функцій або діаграма класів.

# Схема викликів функцій або діаграма класів.

Порядок викликів функцій:

• одержання заголовку таблиці;

• зчитування файлу та вивід його вмісту на консоль;

• отримання номеру вибраного товару;

• отримання кількості вибраного товару;

• запис всієї інформації про вибраний товар в допоміжний файл;

• отримання заголовку для таблиці корзини товарів;

• вивід товару в корзині;

• здійснення купівлі товару: підрахунок профіту та його збереження;

• очищення змінної профіту;

• сортування купленого товару за типом;

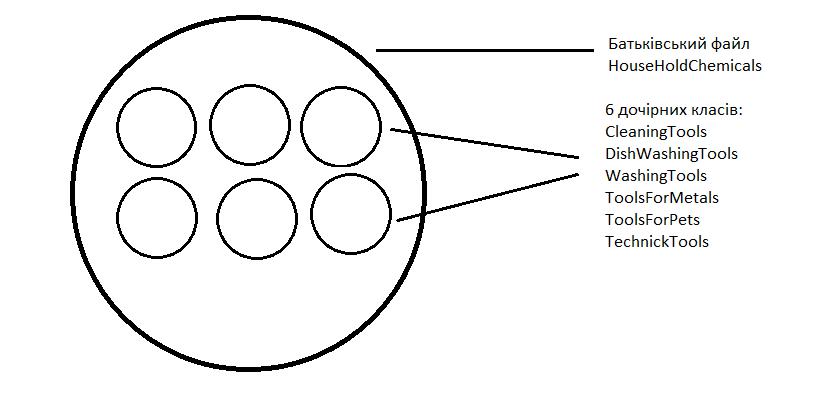
• сортування купленого товару за збільшенням його номеру, зміна його кількості;

• очищення файлу з всією інформацією про певний товар;

• перепис файлу зі зміненою кількістю певного товару;

• очищення всіх допоміжних файлів.

Діаграма класів наведена нижче.



*Рис. 3.1. Спрощений вигляд діаграми класів програмного проекту.*

# Опис реалізації основних функцій/методів.

# Функція зчитування файлу та вивід на консоль.

В лістингу 1 наведено функцію зчитування файлу та вивід на консоль.

*Лістинг 1*

|  |
| --- |
| void CleaningTools::ReadFile()  {char Name[15], CountryProducer[15], Type[15]; int Size, Cost, Amount, Number=0, ID;  ifstream BD\_CT("BD\_CleaningTools.txt");  while (!BD\_CT.eof()) // читаю весь файл построчно  {BD\_CT >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; Number++;  cout << "|" << setw(2) << left << Number << "|" << setw(12) << Name << "|" << setw(10) << CountryProducer << "|" << setw(4) << Size << "|" << setw(4) << Cost << "|" << setw(9) << Amount << "|" << setw(11) << Type << "|" << setw(3) << ID << "|" << endl; }  cout << "------------------------------------------------------------------------------------" << endl;  BD\_CT.close();  } |

# Функція запису всієї інформації про вибраний товар в допоміжний файл.

В лістингу 2 наведено функцію запису всієї інформації про вибраний товар в допоміжний файл.

*Лістинг 2*

|  |
| --- |
| void CleaningTools::SendProduct(int ID, int Amount)   * запис товарів в файл (корзина)   {  char Name[15], CountryProducer[15], Type[15];  int Size, Cost, numerosity, Number = 0;  ifstream BD\_CT("BD\_CleaningTools.txt");  ofstream Korzina("Korzina.txt", ios\_base::app);  int fID; m\_ID = ID; m\_Amount = Amount;  while (!BD\_CT.eof())  {  BD\_CT >> fID >> Name >> CountryProducer >> Size >> Cost >> numerosity >> Type; Number++;  if (fID == m\_ID)  {  Korzina << "\n" << fID << " " << Name << " " << CountryProducer << " " << Size << " " << Cost << " " << m\_Amount << " " << Type;  }  }  Korzina.close();  BD\_CT.close();  } |

# Функція зміни кількості товару та запис в допоміжний файл.

В лістингу 3 наведено функцію зміни кількості товару та запис в допоміжний файл.

*Лістинг 3*

|  |
| --- |
| void CleaningTools::WriteHelp()  {  ifstream CT\_Korzina("CT\_Korzina.txt");  ifstream BD\_CT("BD\_CleaningTools.txt");  ofstream Help("CT\_Help.txt", ios\_base::app);  char K\_Name[15], CountryProducer[15], Type[15];  char Name[15];  int Size, Cost, ID, K\_Amount, K\_ID, G\_Amount, N\_Amount; while (!BD\_CT.eof())  {BD\_CT >> K\_ID >> K\_Name >> CountryProducer >> Size >> Cost >> K\_Amount >> Type; CT\_Korzina >> ID >> Name >> CountryProducer >> Size >> Cost >> N\_Amount >> Type;    if (K\_ID == ID) { G\_Amount = K\_Amount - N\_Amount; }  Help << "\n" << K\_ID << " " << K\_Name << " " << CountryProducer << " " << Size << " " <<  Cost << " " << G\_Amount << " " << Type;  }  BD\_CT.close();  Help.close();  CT\_Korzina.close();  } |

# Функція сортування товару за його номером.

В лістингу 4 наведено функцію сортування товару за його номером.

*Лістинг 4*

|  |
| --- |
| void CleaningTools::ArrangeKorzina()  {  char Name[15], CountryProducer[15], Type[15];  int Size, Cost, Amount, ID, mID = 200;  check:  ifstream Korzina("Korzina.txt");  ofstream CT\_Korzina("CT\_Korzina.txt", ios\_base::app); while (!Korzina.eof())  {Korzina >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; if (ID == mID)  {CT\_Korzina << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size   * " " << Cost << " " << Amount << " " << Type; Korzina.close();   CT\_Korzina.close(); mID++;  if (mID < 202) { goto check; }  }  }  Korzina.close();  ifstream BD\_CT("BD\_CleaningTools.txt");  while (!BD\_CT.eof())  {BD\_CT >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; Amount = 0;  if (ID == mID)  {CT\_Korzina << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size   * " " << Cost << " " << Amount << " " << Type;   }  }  CT\_Korzina.close();  BD\_CT.close();  mID++;  if (mID < 202) { goto check; }  } |

# Функція очищення файлу з всією інформацією про певний товар.

В лістингу 5 наведено функцію очищення файлу з всією інформацією про певний товар.

*Лістинг 5*

|  |
| --- |
| void CleaningTools::fClean()  {  ofstream BD\_CT("BD\_CleaningTools.txt", ios\_base::trunc);  BD\_CT.close();  } |

# Функція перепису файлу зі всією інформацією про певний товар.

В лістингу 6 наведено функцію перепису файлу зі всією інформацією про певний товар.

*Лістинг 6*

|  |
| --- |
| void CleaningTools::Refresh()  {  char Name[15], CountryProducer[15], Type[15];    int Size, Cost, Amount, ID, mID = 100;  ifstream Help("CT\_Help.txt");  ofstream BD\_CT("BD\_CleaningTools.txt", ios\_base::app); while (!Help.eof())  {  Help >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type;  BD\_CT << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size << " " << Cost << " " << Amount << " " << Type;  }  BD\_CT.close();  Help.close();  } |

# Відлагодження та тестування реалізованої програми згідно індивідуального завдання.

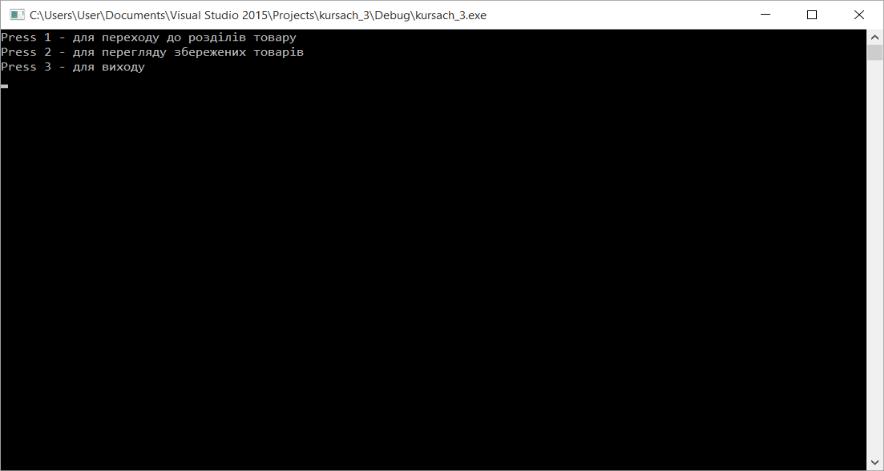
# Відлагодження та тестування окремих частин програми.

Під час тестування окремих частин програм з’являлись певні недоліки та помилки. Щоб все правильно працювало потрібно було не дати можливості користувачу змушувати програму некоректно працювати. Тож всі можливі неполадки були ізольовані і опрацьовані так, щоб з’являлось певне повідомлення з певною інформацією, звідки користувач дізнавався як слід користуватися програмою, щоб уникати таких невизначеностей.

# Відлагодження та тестування програми загалом.

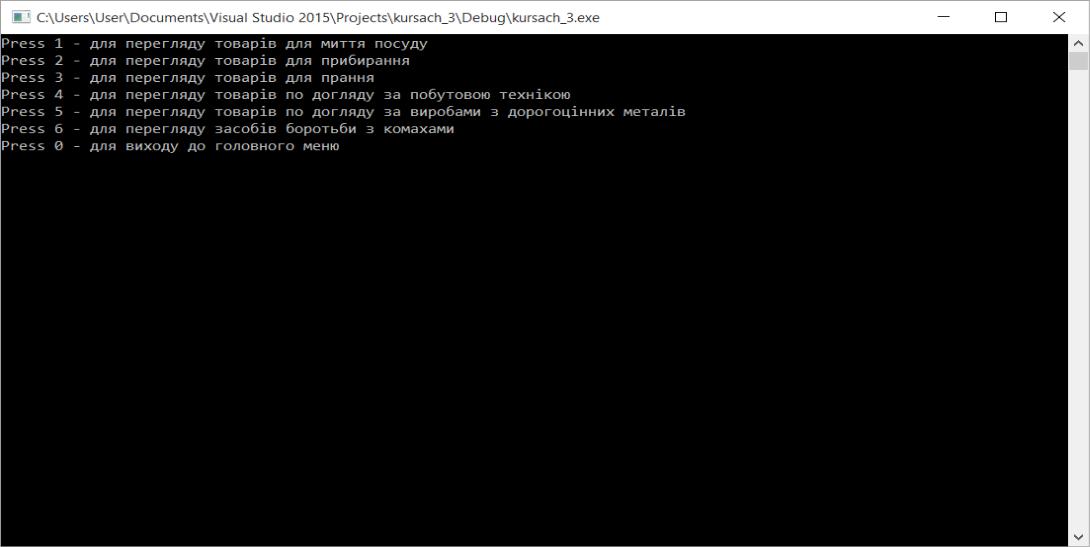
Подібні кроки були виконані і з перевіркою всієї програми. Нижче наведені скріншоти, які показують коректність роботи програми:

На рисунку 4.1. показано головне меню програми.



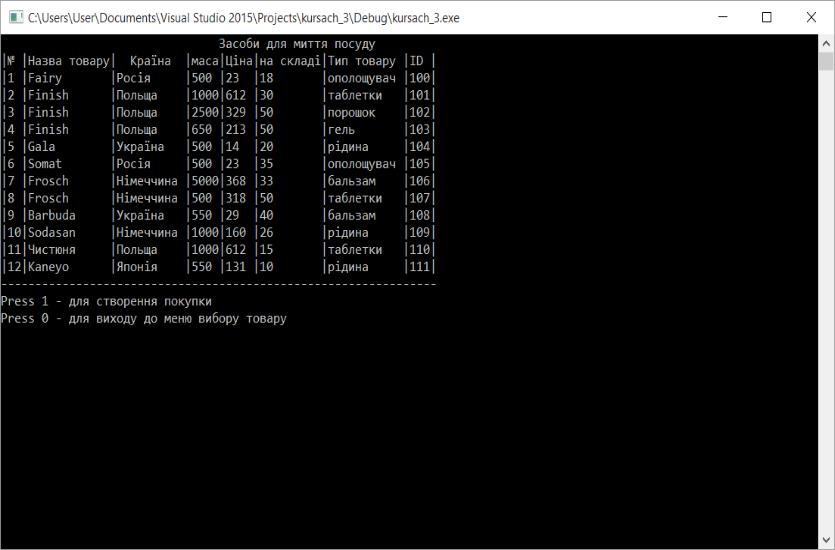
*Рис. 4.1. Головне меню програми.*

На рисунку 4.2. показано меню вибору категорії товарів.



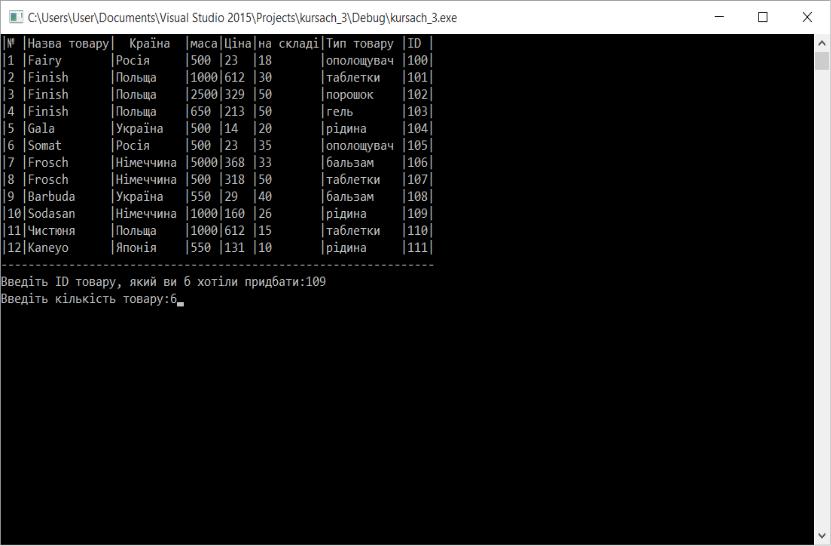
*Рис. 4.2. Меню вибору категорії товарів.*

На рисунку 4.3. показано навігація по списку товарів.



*Рис. 4.3. Навігація по списку товарів.*

На рисунку 4.4. показаний вибір товарів для придбання товарів.



*Рис. 4.4. Вибір товарів для придбання.*

На рисунку 4.5. показано підтвердження покупки обраних товарів.



*Рис. 4.5. Підтвердження покупки обраних товарів.*

# Аналіз продуктивності (часу роботи) та структурної складності реалізованої програмзгідно індивідуального завдання.

# Аналіз структурної складності виконання програми.

Оцінюючи структуру програми, потрібно наголосити, що наявні 3 конструкції switch, які полегшують не тільки користування програмою в ролі користувача, але як на мене, полегшують власне кодування. Ці конструкції начебто уявно відділяють блок коду від усієї програми, що покращує сприйняття й розуміння виконаних та запланованих цілей (функцій, операцій). Як висновок, можу сказати що, виконання програми на структурному рівні не є складним.

# Аналіз часу виконання програми в залежності від об'єму вхідних даних.

Всі операції всередині програми виконуються миттєво, немає затримки між переходами по меню та виведенням інформації на консоль. Читання даних з файлу та виконання операцій з числами виконується бездоганно.

# Висновки

Врезультаті виконання курсової роботи, я вивчив базові принципи об’єктно-орієнтованого програмування та закріпив знання на практиці. Створив програму, яка характеризує біжучий стан галузі магазину побутової хімії. Використані основні принципи об’єктно-орієнтованого програмування, а саме наслідування та інкапсуляція. Розроблене консольне меню.

# Список літератури

1. *Прата С. Язык программирования C++. Лекции и упражнения. 6-е издание / Стивен Прата; пер. с англ. – М.: ООО «И.Д. Вильямс», 2012. – 1248 с.: ил. – Парал. тит. англ.*
2. *Полный справочник по C++, 4-е издание / ГербертШилдт — М.: «Вильямс», 2011. - 800 с.*
3. *Chris Giametta «Pro Flex on Spring», 2009.- p.445.*
4. *Роберт Дж. Оберг «ТехнологияCOM + Основы и программирование = Understanding and ProgrammingCOM+: A Practical Guideto Windows 2000 First Edition». — М.:* [*«Вильямс»,*](http://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%BB%D1%8C%D1%8F%D0%BC%D1%81_%28%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE%29) *2000.- С. 480.*
5. *Тамре Л. «Введение в тестирование програмного обеспечения ».- М.: Издательский дом*

*«Вильямс», 2003.- С. 368.*

1. *Липаев В.В. «Обеспечение качества программных средств. Методы и стандарты». - М.:*

*Синтег, 2001.- С. 246.*

1. *Макгрегор Дж., Сайкс Д. «Тестирование объектно-ориентированного програмного обеспечения».- К: Диасофт, 2002.- С. 432.*
2. *Эммерих В. Конструирование распределенных объектов. Методы и средства*

*программирования интероперабельных объектов в архитектурах OMG/CORBA, Microsoft/COM и Java/RMI. Пер. с англ. — М.: Мир, 2002. — С. 510.*

*9. Макгрегор Дж., Сайкс Д. Тестирование объектно-ориентированного програмного обеспечения.- К: Диасофт, 2002.- С. 432.*

1. *Тамре Л. Введение в тестирование програмного обеспечения.- М.: Издательский дом―Вильямс‖, 2003.- С. 368.*
2. *Э. Гамм, Р. Хелм, Р. Джонсон, Д. Влиссидес «Приѐмы объектно-ориентированного проектирования. Паттерны проектирования», Питер, Москва, 2007.- С. 366.*
3. *Джон Влиссидес «Применение шаблонов проектирования», Москва, 2003.- С. 130.*

# Додатки

## А. Лістинг модулів програми згідно індивідуального завдання

void CleaningTools::ReadFile()

{char Name[15], CountryProducer[15], Type[15]; int Size, Cost, Amount, Number=0, ID;

ifstream BD\_CT("BD\_CleaningTools.txt");

while (!BD\_CT.eof()) // читаю весь файл построчно

{BD\_CT >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; Number++;

cout << "|" << setw(2) << left << Number << "|" << setw(12) << Name << "|" << setw(10) << CountryProducer << "|" << setw(4) << Size << "|" << setw(4) << Cost << "|" << setw(9) << Amount << "|" << setw(11) << Type << "|" << setw(3) << ID << "|" << endl; }

cout << "------------------------------------------------------------------------------------" << endl;

BD\_CT.close();

}

void CleaningTools::SendProduct(int ID, int Amount)

* запис товарів в файл (корзина)

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, numerosity, Number = 0;

ifstream BD\_CT("BD\_CleaningTools.txt");

ofstream Korzina("Korzina.txt", ios\_base::app);

int fID; m\_ID = ID; m\_Amount = Amount;

while (!BD\_CT.eof())

{

BD\_CT >> fID >> Name >> CountryProducer >> Size >> Cost >> numerosity >> Type; Number++;

if (fID == m\_ID)

{

Korzina << "\n" << fID << " " << Name << " " << CountryProducer << " " << Size << " " << Cost << " " << m\_Amount << " " << Type;

}

}

Korzina.close();

BD\_CT.close();

}

void CleaningTools::WriteHelp()

{

ifstream CT\_Korzina("CT\_Korzina.txt");

ifstream BD\_CT("BD\_CleaningTools.txt");

ofstream Help("CT\_Help.txt", ios\_base::app);

char K\_Name[15], CountryProducer[15], Type[15];

char Name[15];

int Size, Cost, ID, K\_Amount, K\_ID, G\_Amount, N\_Amount; while (!BD\_CT.eof())

{BD\_CT >> K\_ID >> K\_Name >> CountryProducer >> Size >> Cost >> K\_Amount >> Type; CT\_Korzina >> ID >> Name >> CountryProducer >> Size >> Cost >> N\_Amount >> Type;

if (K\_ID == ID) { G\_Amount = K\_Amount - N\_Amount; }

Help << "\n" << K\_ID << " " << K\_Name << " " << CountryProducer << " " << Size << " " <<

Cost << " " << G\_Amount << " " << Type;

}

BD\_CT.close();

Help.close();

CT\_Korzina.close();

}

void CleaningTools::ArrangeKorzina()

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, Amount, ID, mID = 200;

check:

ifstream Korzina("Korzina.txt");

ofstream CT\_Korzina("CT\_Korzina.txt", ios\_base::app); while (!Korzina.eof())

{Korzina >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; if (ID == mID)

{CT\_Korzina << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size

* " " << Cost << " " << Amount << " " << Type; Korzina.close();

CT\_Korzina.close(); mID++;

if (mID < 202) { goto check; }

}

}

Korzina.close();

ifstream BD\_CT("BD\_CleaningTools.txt");

while (!BD\_CT.eof())

{BD\_CT >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; Amount = 0;

if (ID == mID)

{CT\_Korzina << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size

* " " << Cost << " " << Amount << " " << Type;

}

}

CT\_Korzina.close();

BD\_CT.close();

mID++;

if (mID < 202) { goto check; }

}

void CleaningTools::fClean()

{

ofstream BD\_CT("BD\_CleaningTools.txt", ios\_base::trunc);

BD\_CT.close();

}

void CleaningTools::Refresh()

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, Amount, ID, mID = 100;

ifstream Help("CT\_Help.txt");

ofstream BD\_CT("BD\_CleaningTools.txt", ios\_base::app); while (!Help.eof())

{

Help >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type;

BD\_CT << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size << " " << Cost << " " << Amount << " " << Type;

}

BD\_CT.close();

Help.close();

}

void CleaningTools::ReadFile()

{char Name[15], CountryProducer[15], Type[15]; int Size, Cost, Amount, Number=0, ID;

ifstream BD\_CT("BD\_CleaningTools.txt");

while (!BD\_CT.eof()) // читаю весь файл построчно

{BD\_CT >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; Number++;

cout << "|" << setw(2) << left << Number << "|" << setw(12) << Name << "|" << setw(10) << CountryProducer << "|" << setw(4) << Size << "|" << setw(4) << Cost << "|" << setw(9) << Amount << "|" << setw(11) << Type << "|" << setw(3) << ID << "|" << endl; }

cout << "------------------------------------------------------------------------------------" << endl;

BD\_CT.close();

}

void CleaningTools::SendProduct(int ID, int Amount)

* запис товарів в файл (корзина)

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, numerosity, Number = 0;

ifstream BD\_CT("BD\_CleaningTools.txt");

ofstream Korzina("Korzina.txt", ios\_base::app);

int fID; m\_ID = ID; m\_Amount = Amount;

while (!BD\_CT.eof())

{

BD\_CT >> fID >> Name >> CountryProducer >> Size >> Cost >> numerosity >> Type; Number++;

if (fID == m\_ID)

{

Korzina << "\n" << fID << " " << Name << " " << CountryProducer << " " << Size << " " << Cost << " " << m\_Amount << " " << Type;

}

}

Korzina.close();

BD\_CT.close();

}

void CleaningTools::WriteHelp()

{

ifstream CT\_Korzina("CT\_Korzina.txt");

ifstream BD\_CT("BD\_CleaningTools.txt");

ofstream Help("CT\_Help.txt", ios\_base::app);

char K\_Name[15], CountryProducer[15], Type[15];

char Name[15];

int Size, Cost, ID, K\_Amount, K\_ID, G\_Amount, N\_Amount; while (!BD\_CT.eof())

{BD\_CT >> K\_ID >> K\_Name >> CountryProducer >> Size >> Cost >> K\_Amount >> Type; CT\_Korzina >> ID >> Name >> CountryProducer >> Size >> Cost >> N\_Amount >> Type;

if (K\_ID == ID) { G\_Amount = K\_Amount - N\_Amount; }

Help << "\n" << K\_ID << " " << K\_Name << " " << CountryProducer << " " << Size << " " <<

Cost << " " << G\_Amount << " " << Type;

}

BD\_CT.close();

Help.close();

CT\_Korzina.close();

}

void CleaningTools::ArrangeKorzina()

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, Amount, ID, mID = 200;

check:

ifstream Korzina("Korzina.txt");

ofstream CT\_Korzina("CT\_Korzina.txt", ios\_base::app); while (!Korzina.eof())

{Korzina >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; if (ID == mID)

{CT\_Korzina << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size

* " " << Cost << " " << Amount << " " << Type; Korzina.close();

CT\_Korzina.close(); mID++;

if (mID < 202) { goto check; }

}

}

Korzina.close();

ifstream BD\_CT("BD\_CleaningTools.txt");

while (!BD\_CT.eof())

{BD\_CT >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; Amount = 0;

if (ID == mID)

{CT\_Korzina << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size

* " " << Cost << " " << Amount << " " << Type;

}

}

CT\_Korzina.close();

BD\_CT.close();

mID++;

if (mID < 202) { goto check; }

}

void CleaningTools::fClean()

{

ofstream BD\_CT("BD\_CleaningTools.txt", ios\_base::trunc);

BD\_CT.close();

}

void CleaningTools::Refresh()

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, Amount, ID, mID = 100;

ifstream Help("CT\_Help.txt");

ofstream BD\_CT("BD\_CleaningTools.txt", ios\_base::app); while (!Help.eof())

{

Help >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type;

BD\_CT << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size << " " << Cost << " " << Amount << " " << Type;

}

BD\_CT.close();

Help.close();

}

void CleaningTools::ReadFile()

{char Name[15], CountryProducer[15], Type[15]; int Size, Cost, Amount, Number=0, ID;

ifstream BD\_CT("BD\_CleaningTools.txt");

while (!BD\_CT.eof()) // читаю весь файл построчно

{BD\_CT >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; Number++;

cout << "|" << setw(2) << left << Number << "|" << setw(12) << Name << "|" << setw(10) << CountryProducer << "|" << setw(4) << Size << "|" << setw(4) << Cost << "|" << setw(9) << Amount << "|" << setw(11) << Type << "|" << setw(3) << ID << "|" << endl; }

cout << "------------------------------------------------------------------------------------" << endl;

BD\_CT.close();

}

void CleaningTools::SendProduct(int ID, int Amount)

* запис товарів в файл (корзина)

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, numerosity, Number = 0;

ifstream BD\_CT("BD\_CleaningTools.txt");

ofstream Korzina("Korzina.txt", ios\_base::app);

int fID; m\_ID = ID; m\_Amount = Amount;

while (!BD\_CT.eof())

{

BD\_CT >> fID >> Name >> CountryProducer >> Size >> Cost >> numerosity >> Type; Number++;

if (fID == m\_ID)

{

Korzina << "\n" << fID << " " << Name << " " << CountryProducer << " " << Size << " " << Cost << " " << m\_Amount << " " << Type;

}

}

Korzina.close();

BD\_CT.close();

}

void CleaningTools::WriteHelp()

{

ifstream CT\_Korzina("CT\_Korzina.txt");

ifstream BD\_CT("BD\_CleaningTools.txt");

ofstream Help("CT\_Help.txt", ios\_base::app);

char K\_Name[15], CountryProducer[15], Type[15];

char Name[15];

int Size, Cost, ID, K\_Amount, K\_ID, G\_Amount, N\_Amount; while (!BD\_CT.eof())

{BD\_CT >> K\_ID >> K\_Name >> CountryProducer >> Size >> Cost >> K\_Amount >> Type; CT\_Korzina >> ID >> Name >> CountryProducer >> Size >> Cost >> N\_Amount >> Type;

if (K\_ID == ID) { G\_Amount = K\_Amount - N\_Amount; }

Help << "\n" << K\_ID << " " << K\_Name << " " << CountryProducer << " " << Size << " " <<

Cost << " " << G\_Amount << " " << Type;

}

BD\_CT.close();

Help.close();

CT\_Korzina.close();

}

void CleaningTools::ArrangeKorzina()

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, Amount, ID, mID = 200;

check:

ifstream Korzina("Korzina.txt");

ofstream CT\_Korzina("CT\_Korzina.txt", ios\_base::app); while (!Korzina.eof())

{Korzina >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; if (ID == mID)

{CT\_Korzina << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size

* " " << Cost << " " << Amount << " " << Type; Korzina.close();

CT\_Korzina.close(); mID++;

if (mID < 202) { goto check; }

}

}

Korzina.close();

ifstream BD\_CT("BD\_CleaningTools.txt");

while (!BD\_CT.eof())

{BD\_CT >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; Amount = 0;

if (ID == mID)

{CT\_Korzina << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size

* " " << Cost << " " << Amount << " " << Type;

}

}

CT\_Korzina.close();

BD\_CT.close();

mID++;

if (mID < 202) { goto check; }

}

void CleaningTools::fClean()

{

ofstream BD\_CT("BD\_CleaningTools.txt", ios\_base::trunc);

BD\_CT.close();

}

void CleaningTools::Refresh()

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, Amount, ID, mID = 100;

ifstream Help("CT\_Help.txt");

ofstream BD\_CT("BD\_CleaningTools.txt", ios\_base::app); while (!Help.eof())

{

Help >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type;

BD\_CT << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size << " " << Cost << " " << Amount << " " << Type;

}

BD\_CT.close();

Help.close();

}

void CleaningTools::ReadFile()

{char Name[15], CountryProducer[15], Type[15]; int Size, Cost, Amount, Number=0, ID;

ifstream BD\_CT("BD\_CleaningTools.txt");

while (!BD\_CT.eof()) // читаю весь файл построчно

{BD\_CT >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; Number++;

cout << "|" << setw(2) << left << Number << "|" << setw(12) << Name << "|" << setw(10) << CountryProducer << "|" << setw(4) << Size << "|" << setw(4) << Cost << "|" << setw(9) << Amount << "|" << setw(11) << Type << "|" << setw(3) << ID << "|" << endl; }

cout << "------------------------------------------------------------------------------------" << endl;

BD\_CT.close();

}

void CleaningTools::SendProduct(int ID, int Amount)

* запис товарів в файл (корзина)

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, numerosity, Number = 0;

ifstream BD\_CT("BD\_CleaningTools.txt");

ofstream Korzina("Korzina.txt", ios\_base::app);

int fID; m\_ID = ID; m\_Amount = Amount;

while (!BD\_CT.eof())

{

BD\_CT >> fID >> Name >> CountryProducer >> Size >> Cost >> numerosity >> Type; Number++;

if (fID == m\_ID)

{

Korzina << "\n" << fID << " " << Name << " " << CountryProducer << " " << Size << " " << Cost << " " << m\_Amount << " " << Type;

}

}

Korzina.close();

BD\_CT.close();

}

void CleaningTools::WriteHelp()

{

ifstream CT\_Korzina("CT\_Korzina.txt");

ifstream BD\_CT("BD\_CleaningTools.txt");

ofstream Help("CT\_Help.txt", ios\_base::app);

char K\_Name[15], CountryProducer[15], Type[15];

char Name[15];

int Size, Cost, ID, K\_Amount, K\_ID, G\_Amount, N\_Amount; while (!BD\_CT.eof())

{BD\_CT >> K\_ID >> K\_Name >> CountryProducer >> Size >> Cost >> K\_Amount >> Type; CT\_Korzina >> ID >> Name >> CountryProducer >> Size >> Cost >> N\_Amount >> Type;

if (K\_ID == ID) { G\_Amount = K\_Amount - N\_Amount; }

Help << "\n" << K\_ID << " " << K\_Name << " " << CountryProducer << " " << Size << " " <<

Cost << " " << G\_Amount << " " << Type;

}

BD\_CT.close();

Help.close();

CT\_Korzina.close();

}

void CleaningTools::ArrangeKorzina()

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, Amount, ID, mID = 200;

check:

ifstream Korzina("Korzina.txt");

ofstream CT\_Korzina("CT\_Korzina.txt", ios\_base::app); while (!Korzina.eof())

{Korzina >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; if (ID == mID)

{CT\_Korzina << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size

* " " << Cost << " " << Amount << " " << Type; Korzina.close();

CT\_Korzina.close(); mID++;

if (mID < 202) { goto check; }

}

}

Korzina.close();

ifstream BD\_CT("BD\_CleaningTools.txt");

while (!BD\_CT.eof())

{BD\_CT >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; Amount = 0;

if (ID == mID)

{CT\_Korzina << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size

* " " << Cost << " " << Amount << " " << Type;

}

}

CT\_Korzina.close();

BD\_CT.close();

mID++;

if (mID < 202) { goto check; }

}

void CleaningTools::fClean()

{

ofstream BD\_CT("BD\_CleaningTools.txt", ios\_base::trunc);

BD\_CT.close();

}

void CleaningTools::Refresh()

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, Amount, ID, mID = 100;

ifstream Help("CT\_Help.txt");

ofstream BD\_CT("BD\_CleaningTools.txt", ios\_base::app); while (!Help.eof())

{

Help >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type;

BD\_CT << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size << " " << Cost << " " << Amount << " " << Type;

}

BD\_CT.close();

Help.close();

}

void CleaningTools::ReadFile()

{char Name[15], CountryProducer[15], Type[15]; int Size, Cost, Amount, Number=0, ID;

ifstream BD\_CT("BD\_CleaningTools.txt");

while (!BD\_CT.eof()) // читаю весь файл построчно

{BD\_CT >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; Number++;

cout << "|" << setw(2) << left << Number << "|" << setw(12) << Name << "|" << setw(10) << CountryProducer << "|" << setw(4) << Size << "|" << setw(4) << Cost << "|" << setw(9) << Amount << "|" << setw(11) << Type << "|" << setw(3) << ID << "|" << endl; }

cout << "------------------------------------------------------------------------------------" << endl;

BD\_CT.close();

}

void CleaningTools::SendProduct(int ID, int Amount)

* запис товарів в файл (корзина)

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, numerosity, Number = 0;

ifstream BD\_CT("BD\_CleaningTools.txt");

ofstream Korzina("Korzina.txt", ios\_base::app);

int fID; m\_ID = ID; m\_Amount = Amount;

while (!BD\_CT.eof())

{

BD\_CT >> fID >> Name >> CountryProducer >> Size >> Cost >> numerosity >> Type; Number++;

if (fID == m\_ID)

{

Korzina << "\n" << fID << " " << Name << " " << CountryProducer << " " << Size << " " << Cost << " " << m\_Amount << " " << Type;

}

}

Korzina.close();

BD\_CT.close();

}

void CleaningTools::WriteHelp()

{

ifstream CT\_Korzina("CT\_Korzina.txt");

ifstream BD\_CT("BD\_CleaningTools.txt");

ofstream Help("CT\_Help.txt", ios\_base::app);

char K\_Name[15], CountryProducer[15], Type[15];

char Name[15];

int Size, Cost, ID, K\_Amount, K\_ID, G\_Amount, N\_Amount; while (!BD\_CT.eof())

{BD\_CT >> K\_ID >> K\_Name >> CountryProducer >> Size >> Cost >> K\_Amount >> Type; CT\_Korzina >> ID >> Name >> CountryProducer >> Size >> Cost >> N\_Amount >> Type;

if (K\_ID == ID) { G\_Amount = K\_Amount - N\_Amount; }

Help << "\n" << K\_ID << " " << K\_Name << " " << CountryProducer << " " << Size << " " <<

Cost << " " << G\_Amount << " " << Type;

}

BD\_CT.close();

Help.close();

CT\_Korzina.close();

}

void CleaningTools::ArrangeKorzina()

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, Amount, ID, mID = 200;

check:

ifstream Korzina("Korzina.txt");

ofstream CT\_Korzina("CT\_Korzina.txt", ios\_base::app); while (!Korzina.eof())

{Korzina >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; if (ID == mID)

{CT\_Korzina << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size

* " " << Cost << " " << Amount << " " << Type; Korzina.close();

CT\_Korzina.close(); mID++;

if (mID < 202) { goto check; }

}

}

Korzina.close();

ifstream BD\_CT("BD\_CleaningTools.txt");

while (!BD\_CT.eof())

{BD\_CT >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type; Amount = 0;

if (ID == mID)

{CT\_Korzina << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size

* " " << Cost << " " << Amount << " " << Type;

}

}

CT\_Korzina.close();

BD\_CT.close();

mID++;

if (mID < 202) { goto check; }

}

void CleaningTools::fClean()

{

ofstream BD\_CT("BD\_CleaningTools.txt", ios\_base::trunc);

BD\_CT.close();

}

void CleaningTools::Refresh()

{

char Name[15], CountryProducer[15], Type[15];

int Size, Cost, Amount, ID, mID = 100;

ifstream Help("CT\_Help.txt");

ofstream BD\_CT("BD\_CleaningTools.txt", ios\_base::app); while (!Help.eof())

{

Help >> ID >> Name >> CountryProducer >> Size >> Cost >> Amount >> Type;

BD\_CT << "\n" << ID << " " << Name << " " << CountryProducer << " " << Size << " " << Cost << " " << Amount << " " << Type;

}

BD\_CT.close();

Help.close();

}