Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра Програмної інженерії

КУРСОВА РОБОТА

ПОЯСНЮВАЛЬНА ЗАПИСКА

з дисципліни “Об’єктно-орієнтоване програмування”

СКЛАД

Керівник, ас. каф. ПІ Ляпота В.М.

Студент гр. ПІ-15-2 Яровий М.В.

Комісія: проф.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дудар З.В.

проф. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бондарєв М.В

Харків 2016

**ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ**

Кафедра: ***Програмної інженерії***

Дисципліна: Об’єктно-орієнтоване програмування

Спеціальність: ***Програмна інженерія***

Курс 1 .Група *ПІ-15-2* . Семестр 2 .

***ЗАВДАННЯ***

***на курсовий проект студента***

***Ярового Микити Володимировича***

1. Тема проекту: «Склад»
2. Термін здачі студентом закінченого проекту: ***“28” - травня - 2016 р***.
3. Вихідні дані до проекту:

Специфікація програми, методичні вказівки до виконання курсової роботи, інтернет посилання, книги та наукові публікації стосовно теми роботи, середовище Microsoft Visual Studio 2015 для розробки програми.

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |
| --- | --- | --- |
| *№* | *Назва етапу* | *Термін виконання* |
| 1 | Видача теми, узгодження і затвердження теми | 1-03-2016 р. |
| 2 | Аналіз предметної області | 23-04-2016 – 5-05-2016 р. |
| 3 | Розробка постановки задачі | 5-05-2016 – 6-05-2016 р. |
| 4 | Розробка об’єктної моделі | 6-05-2016 – 12-05-2016 р. |
| 5 | Кодування програмної системи | 12-05-2016 – 27-05-2016 р. |
| 6 | Тестування і доопрацювання розробленої програмної системи. | 12-05-2016 – 27-05-2016 р. |
| 7 | Оформлення пояснювальної записки | 27-05-2016 – 28-05-2016 р. |
| 8 | Публічний захист проекту перед комісією | 28-05-2016 р. |

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ляпота В.М.

«1» березня 2016р.

РЕФЕРАТ

Пояснювальна записка до курсової роботи: 33с., 15рис., 1 додаток, 5 джерел.

Метою роботи є розробка програмної системи під назвою “Склад” на засадах об'єктно-орієнтованого програмування.

Методи розробки базуються на використанні середи розробки Microsoft Visual Studio 2015, Windows Forms, платформи .NET Framework 4.5, мови програмування C#.

В результаті отримана програма під назвою “Склад”, яка дозволяє працювати з інформаційною базою даних, чию роль виконують txt файли. Розроблена програма дає змогу переглядати дані про товари, які знаходяться на складі. Також є можливості: додавання нових товарів на склад, відображення положення товарів на схемі складу, також існує можливість змінювати характеристики приміщення складу, наприклад при розширенні складу.

ПРОГРАМА, ОБ’ЕКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ, КЛАС, МОВА ПРОГРАМУВАННЯ C#, ПЛАТФОРМА .NET, СКЛАД, НАКЛАДНА, ВІДОМІСТЬ

ЗМІСТ

[ВСТУП 1](#_Toc452314104)

[1 СПЕЦИФІКАЦІЯ ПРОГРАМИ 3](#_Toc452314105)

[2 ПРОЕКТНА СПЕЦІФІКАЦІЯ 7](#_Toc452314106)

[2 ІНСТРУКЦІЯ КОРИСТУВАЧА 10](#_Toc452314107)

[ВИСНОВКИ 16](#_Toc452314108)

[ПЕРЕЛІК ПОСИЛАНЬ 17](#_Toc452314109)

[ДОДАТОК А 18](#_Toc452314110)

# ВСТУП

Склад — це споруда, різноманітне обладнання та внутрішня транспортна система, яка застосовується для прийому, розміщення та зберігання матеріальних цінностей, підготовки їх до споживання та видачі споживачу.

У логістиці склад виконує функцію акумуляції резервів матеріальних ресурсів, необхідних для демпфування коливань обсягів постачань і попиту, а також синхронізації швидкостей потоків товарів в системах просування від виробників до споживачів або потоків матеріалів в технологічних виробничих системах.

Склади є ключовим елементом системи логістики.

В даній курсовій роботі необхідно розробити програму «Склад», яка буде допомагати працівникові складу. Метою роботи є розробка програми, за допомогою якої можна буде додавати пакунки на склад, видаляти пакунки, друкувати прибуткову накладну та видаткову накладну, а також друкувати інвентарну відомість. Загалом програма повинна також мати можливість пошуку товарів, та відображення їх позиції на складі. За основу обрано об’єктно-орієнтований підхід.

Завдання виконання роботи:

1. вивчення предметної галузі з метою визначення головних принципів даної сфери;
2. проектування ієрархії класів, інтерфейсів, взаємодії компонентів на основі виділених принципів;
3. використання вбудованих елементів середовища розробки задля структуризації класів і оптимізації коду;
4. застосування принципу інкапсуляції до класів;
5. реалізація програмної взаємодії з базою даних;

Об’єктно-орієнтований підхід потребує повного розуміння основних концепцій, на яких він базується. В об’єктно-орієнтованому підході основними концепціями є об’єкти і класи.

На сьогоднішній день об’єктно-орієнтоване програмування є лідируючим в галузі прикладного програмування. Використання цього підходу надає розробнику всі необхідні інструменти для забезпечення функціональності і для супроводження проекту.

Розробка базується на використанні середовища розробки Microsoft Visual Studio 2015. В якості мови розробки була обрана мова С#.

# СПЕЦИФІКАЦІЯ ПРОГРАМИ

1. Вкладка «Change Storage»

Робота з програмою починається з головного вікна (рис 1.1). Тут користувач може побачити вікно в якому потрібно задати параметри складу, а саме ширину, довжину та висоту приміщення складу, кнопку при натисненні якої будуть змінюватися параметри складу. Зверху зліва можна побачити вкладки, які будуть використані для додавання\видалення продуктів зі складу, пошуку продуктів та відображення товарів на складі у вигляді схеми. Зліва від текстових полів буде відображено нову схему складу, яка формується автоматично і залежить від введених параметрів.

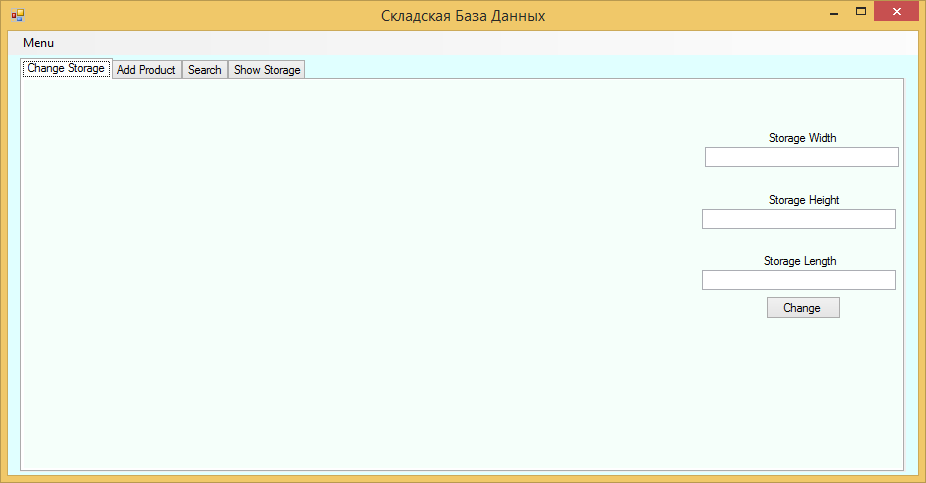


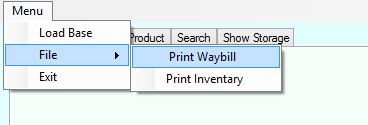
Рис. 1.1 – Головне вікно

Також можна побачити меню (рис.1.2). Як можна побачити в меню існує 3 кнопки: Load Base; File; Exit. При виборі «Load Base» буде загружено програмою дані про всі товари на складі та відображено у вкладках «Add Product», «Search».



Рисунок 1.2 – Меню

При виборі «Файл» користувачу стане доступним перелік друку ( див. рис. 1.4) в якому показані дві можливості: «друк накладної» (print Waybill) та «друк інвентарю» (print Waybill).

Рисунок 1.3-Меню друку накладних та інвентарної відомості

При виборі «друку накладної» буде роздруковано інформацію про останній завезений продукт. При виборі «друку інвентарю» буде роздуковано всю інформацію про товари на складі.

2. Вкладка «Add product»

За допомогою цієї вкладки користувач може додавати товари на склад та видаляти їх при вірному заповненні полів, які не включають «Код». У випадку правильного заповнення полів буде додано товар, а при заповненні поля «Код» товар буде видалено, звісно якщо він існує і знаходиться на складі.

Знизу буде відображено всі товари (див. рис. 1.5).

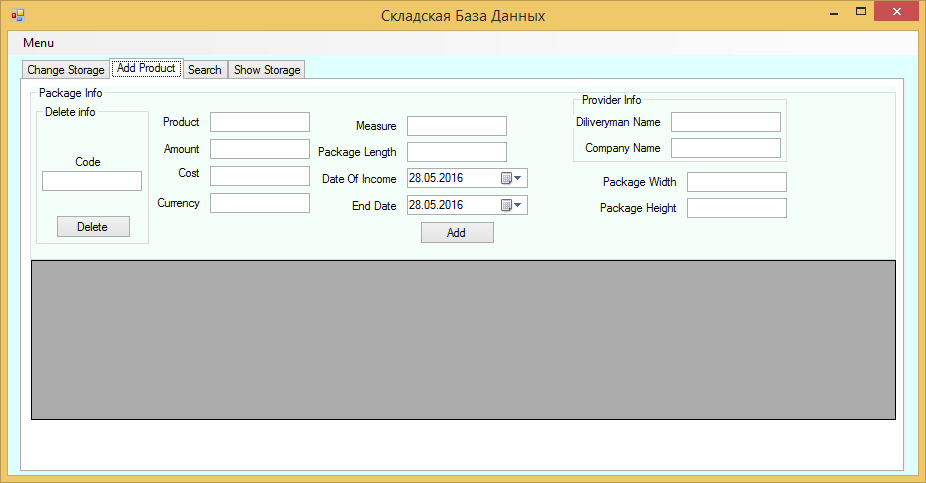


Рисунок 1.5 – Вкладка «Add Product»

3. Вкладка «Search»

Вкладка має вікно для відображення інформації у вигляді таблиці знизу, а зверху існує можливість пошуку по назві товару, при заповненні поля «Search By Product»; пошук по компанії постачальника, при правильному заповненні поля «Search By Company». Аналогічно при заповненні полів «Search By Deliveryman Name» та «Search By Date» відбувається пошук по імені постачальника та по даті. (див. рис. 1.6)

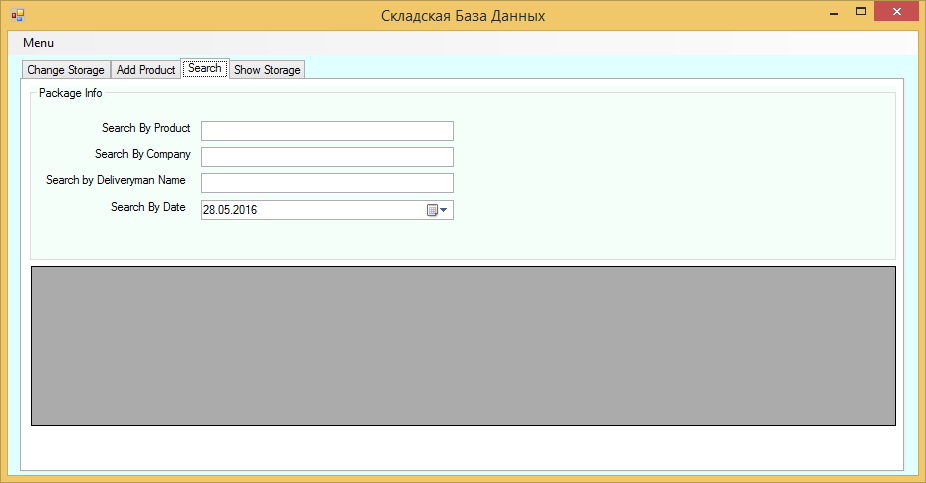


Рисунок 1.6 – Вкладка пошук

4. Вкладка «Show Storage»

В цьому вікні існує можливість показати на схемі план складу та розміщення на ньому товарів, а також масштабування зображення для зручного перегляду користувачем. (див. рис. 1.7).

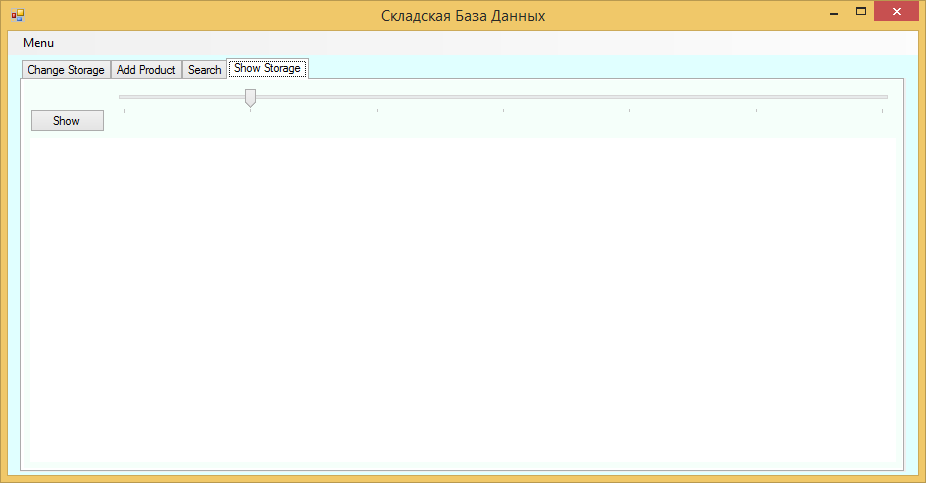


Рисунок 1.7 – Вкладка «Show Storage»

# 2 ПРОЕКТНА СПЕЦІФІКАЦІЯ

«Основным конструктивным элементом в языках ООП является модуль (module), представляющий собой логически связанную совокупность классов и объектов, а не подпрограмм, как в более ранних языках» [2].

В графічному інтерфейсі даної програми існує всього одна форма, та 4 вкладки, на кожній з яких існують методи для обробки даних, а саме функції: зчитування даних, заповнення колекції, збереження даних в файлах.

Всі файли, які зберігають дані мають розширення \*.txt.

На рисунку 2.1 зображено складові головної форми, а також класи помилок, які можливі при не правильному вводі даних в полях.

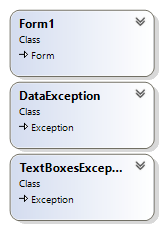


Рисунок 2.1 – Складові класу Form1

Назви класів помилок говорять самі за себе, помилки пов’язані з роботою текстових полів на формі, всі мають клас TextBoxesException, вони мають конструктор з текстом помилки, потім в формі ловляться всі помилки та формується діалогове вікно

На рисунку 2.2 показані всі методи та події форми-1, які необхідні для відображення, пошуку, внесення в базу, видалення з бази товарів. Події викликають певні методи форми, або у одиничних випадках виконують декілька дій, які не повторюються, через це не має необхідності в написанні цілого методу для їх виконання.

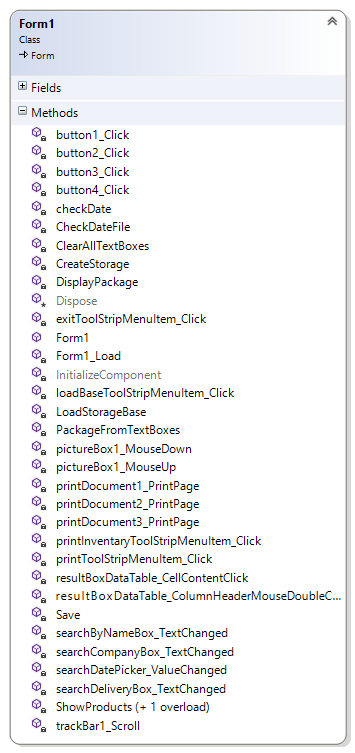


Рисунок 2.2 – Методи головної форми

Основні методи головної форми:

1. CreateStorage() – створює склад з параметрами вказаними в вкладці Change Storage;
2. Save() – зберігає всі товари в базі;
3. ShowProducts() – відображує товари в елементі DataGridView;
4. trackBar1\_Scroll() – змінює масштаб відображення складу на вкладці Show Storage
5. ClearAllTextBoxes – очищує всі текстові поля на формі, необхідно викликати цей метод після зчитування даних.

Взаємозв’язки класів показано на рис.2.3

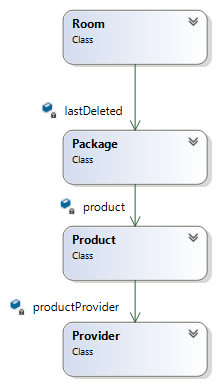


Рис. 2.3 – Взаємозв’язки класі

# ІНСТРУКЦІЯ КОРИСТУВАЧА

1. Зміна параметрів складу. Вкладка «Change Storage»

Для того щоб змінити параметри складу, такі як: довжина, висота, ширина необхідно перейти до вкладки «Change Storage» та заповнити текстові поля. Після заповнення текстових полів «Storage Width», «Storage Height», «Storage Length» числовими значеннями, які означають ширину, висоту та довжину складу в метрах, цілими числами, з’явиться схема складу з заданими параметрами.

На рис.3.1 показано який результат буде відображено при правильному заповненні всіх текстових полів та при натисненні кнопки “Change”. У випадку зображеному на рисунку створено схему складу 300м шириною, 10 метрів висотою, 100 метрів довжиною. Висота складу не відображується на рисунку.

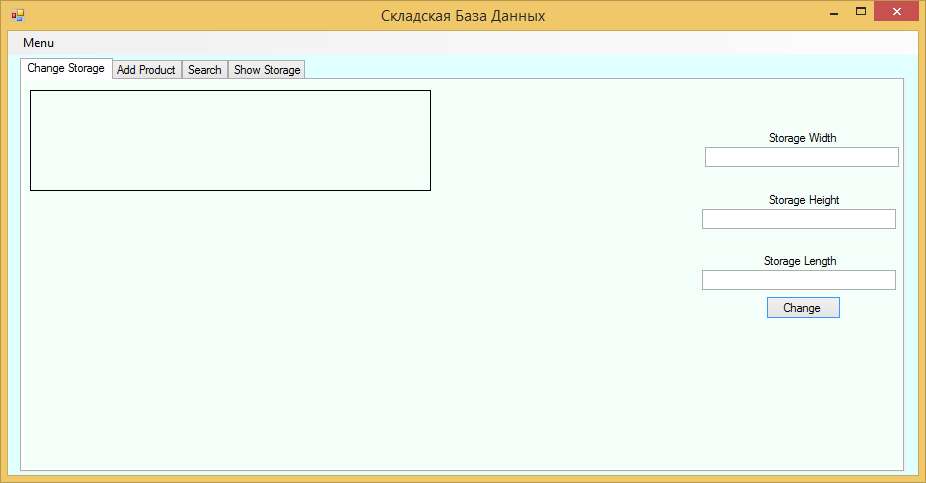


Рисунок 3.1- Зміна параметрів складу.

У випадку не правильного заповнення текстових полів буде отримано вікно з описом помилки, яке допоможе користувачу зрозуміти в якому текстовому полі він ввів не правильні данні. Приклад такого вікна наведено на рисунку 3.2.

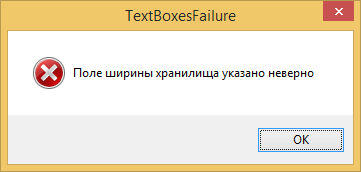


Рис. 3.2 – Вікно помилки

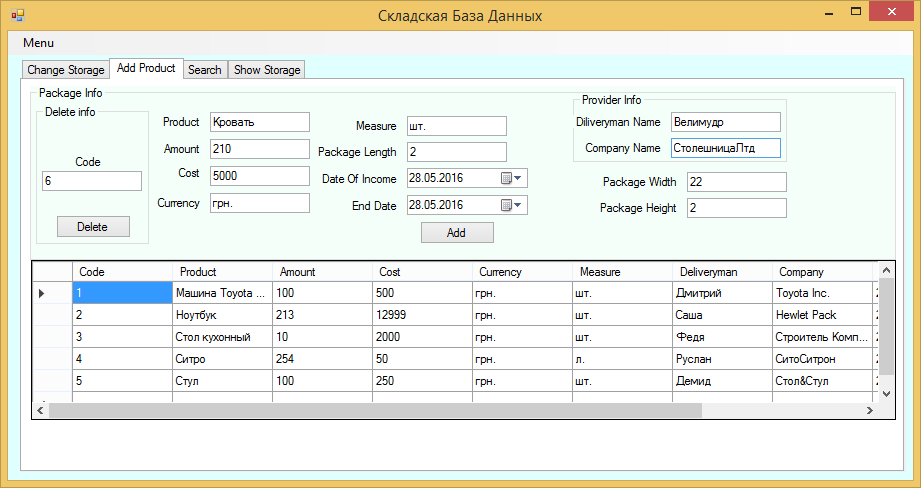
1. Внесення товару на склад.

Опис текстових полів:

1. “Product” зберігає назву продукту;
2. “Amount” зберігає кількість товарних одиниць;
3. “Cost” зберігає числову частину вартості товару;
4. “Currency” зберігає валютну частину вартості продукту;
5. “Measure” зберігає одиниці виміру, наприклад для водних товарів це метри;
6. “Package Length” зберігає довжину всієї групи даного товару;
7. “Date of Income” зберігає дату завозу продукту;
8. “End Date” зберігає дату коли товар заберуть зі складу;
9. “Package Width” зберігає ширину всієї групи даного товару;
10. “Package Heigth” зберігає висоту всієї групи даного товару;
11. “Deliveryman Name” зберігає ім’я постачальника;
12. “Company Name” зберігає назву компанії постачальника.

Код товару формується автоматично.

Заповнення полів для внесення товару на склад (див. рис. 3.3).

 Рисунок 3.3 – Правильне заповнення полів

При натисканні кнопки «Add» відбувається внесення товару в базу даних, звісно у випадку, якщо товар не може поміститися на складі буде викликано вікно яке попередить про це користувача, після натиснення кнопки знизу з’явиться новий товар (Див. рис. 3.4)

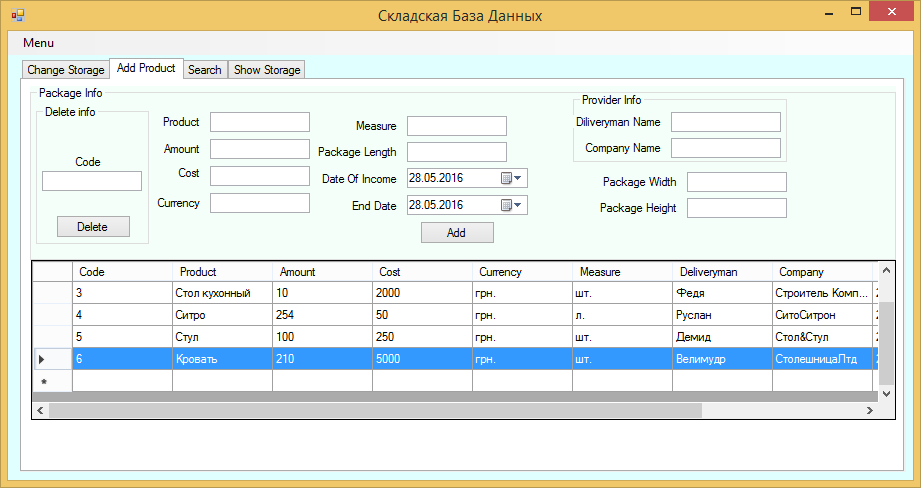


Рисунок. 3.4-Додавання товару

При введені коду товару буде видалено товар з таким кодом(рис 3.4). На рисунку 3.4 видалено товар з кодом 6. Тобто товар який, був щойно доданий може бути видалений по коду. В нашому випадку це код 6.

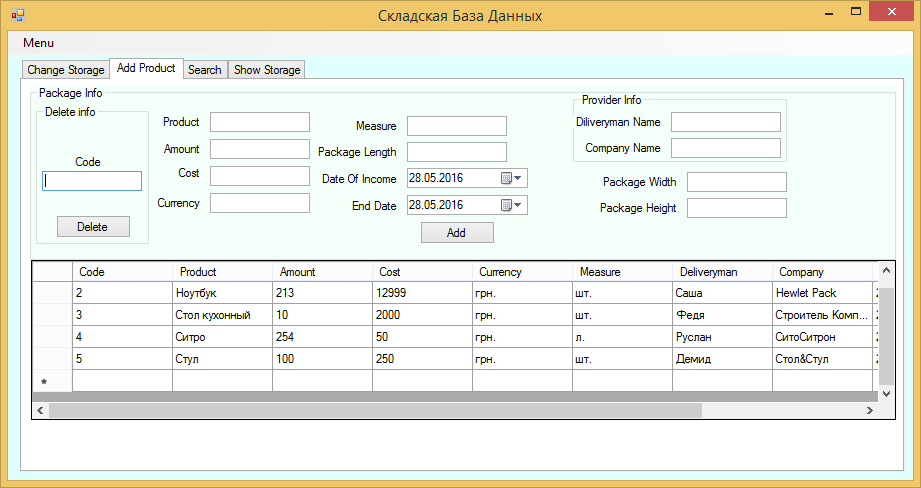


Рисунок 3.4- Видалення товару

При неправильному внесенні даних або у випадку якщо було натиснуто кнопку “Add” буде видано вікно помилки. Приклад вікна помилки у випадку відсутності даних в текстовому полі зображено на рисунку 3.5.

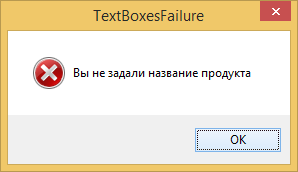


Рис 3.5 – Вікно помилки-повідомлення

3. Пошук товарів.

Вкладка «Search» (рис 3.5) має вікно для відображення інформації у вигляді таблиці знизу, а зверху існує можливість пошуку по назві товару, при заповненні поля «Search By Product»; пошук по компанії постачальника, при правильному заповненні поля «Search By Company».

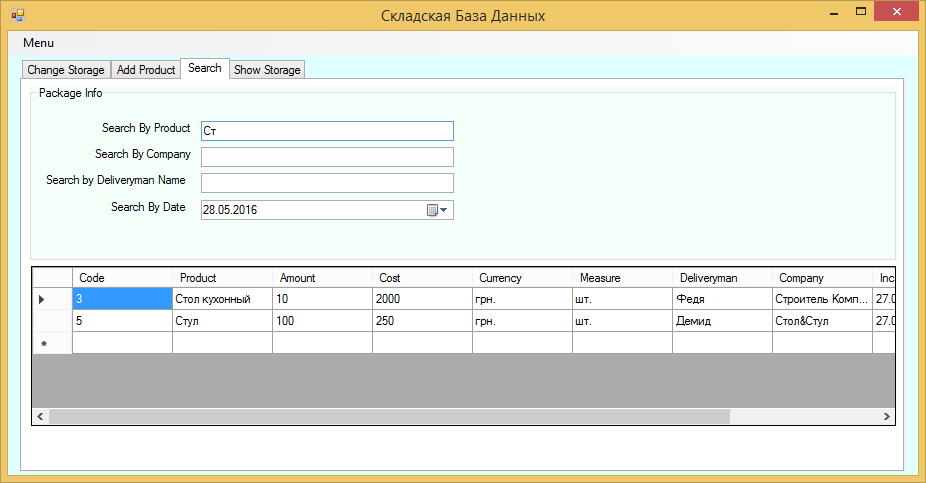


Рисунок 3.5 Приклад пошуку за назвою

Аналогічно при заповненні полів «Search By Deliveryman Name» та «Search By Date» відбувається пошук по імені постачальника та по даті відповідно. Приклад пошуку за назвою продукту наведено на рисунку 3.5. При одночасному пошуку по декільком полям, буде виведено результат пошуку лише по останньому з них.

4. Схема складу та масштабування

Вкладка «Show Storage». При натисненні кнопки «Show» з’явиться відображення складу (див. рис. 3.7), ця кнопка відображує склад тільки при масштабі 1. Для збільшення масштабу потрібно пересунути бігунок вправо, а для зменшення вліво. При наведені на товар інформація про нього буде показана в місці де знаходиться курсор.

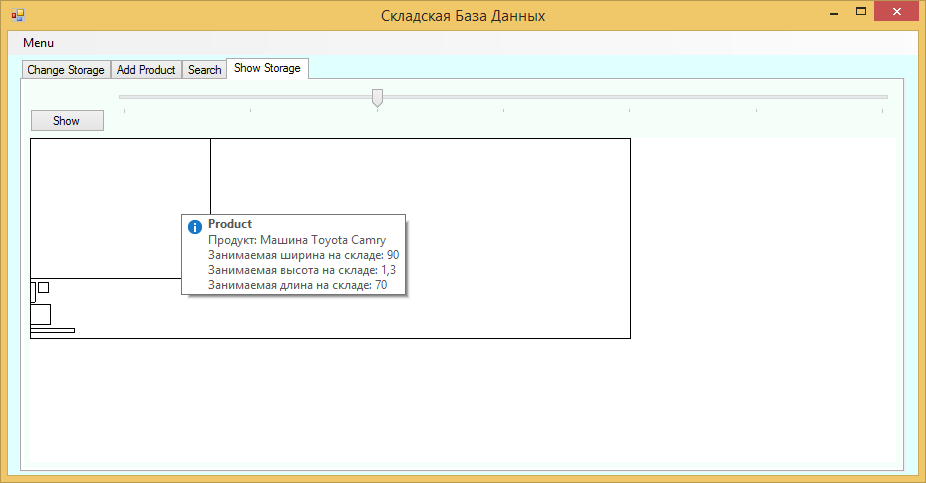


Рисунок 3.6 Масштабування складської схеми

# ВИСНОВКИ

В результаті курсової роботи була отримана готова програма – програма «Склад». За допомогою цієї програми можна переглядати кількість товарів на складі, отримувати перелік всіх товарів та відшукувати товари за їх характеристиками.

Інтерфейс програми простий, зрозумілий та зручний в використанні. Цією програмою без зусиль зможе користуватися навіть недосвідчений користувач.

Ця програма створена перш за все для людей, які слідкують за складськими приміщеннями, з метою покращення та полегшення праці цих людей.

В курсовій роботі реалізований об’єктно-орієнтований підхід в створенні програмного забезпечення інформаційного характеру, який дає можливість на перших етапах розробки визначити необхідний набір функцій майбутньої програми, її склад і структуру баз даних необхідну для реалізації визначених функцій. Все це захищає від майбутньої необхідності в виправленні вже створених компонентів програми.

Шляхи вдосконалення та розвитку програми: використання більш розширеної бази даних; знаходження кращого алгоритму для розміщення товарів на складі, створення більш приємного інтерфейсу, доступного навіть компаніям, що зберігають товари на складі, також шляхом вдосконалення буде можливість змінювати позицію товару в складському приміщенні.

# ПЕРЕЛІК ПОСИЛАНЬ

1. Бондарев, В.М. Объектно-ориентированное программирование на С# : учеб. пособ. /В.М. Бондарев. – Х.: Компания СМИТ, 2009 – 224 с.

2. Буч, Г. Объектно-ориентированный анализ и проектирование : пер. с англ – М.: 000 "И.Д. Вильяме", 2008 - 720 с.

3. Мандер, Т. Разработка пользовательского интерфейса : пер. с англ. – М.: ДМК Пресс, 2008 – 412 с.

4. Троелсен, Э. Язык программирования C# 5.0 и платформа .NET 4.5, 6-е изд. : пер. с англ. – М.: Вильямс, 2013 – 1312 с.

5. Шилдт, Г. C# 4.0.: Полное руководство : пер. с англ. – М.: Вильямс, 2011 – 1056 с.

# ДОДАТОК А

Код програми

Клас Room

class Room

{

private List<Package> packageList = new List<Package>(); //Список товаров на складе

private bool[][] positions;//матрица свободных\занятых позиций

private int roomWidth;

private int roomLength;

private int roomHeight;

private int lastNumber;

private Package lastDeleted;

public Room()

{

roomWidth = 100;

roomHeight = 100;

roomLength = 100;

lastNumber = 1;

positions = new bool[100][];

for (int i = 0; i < 100; i++)

{

positions[i] = new bool[100];

for (int j = 0; j < 100; j++)

{

positions[i][j] = false;

}

}

}

public Room(int w, int l, int h)

{

roomWidth = w;

roomLength = l;

roomHeight = h;

lastNumber = 1;

positions = new bool[w][];

for (int i=0; i<w; i++)

{

positions[i] = new bool[l];

for (int j=0; j<l; j++)

{

positions[i][j] = false;

}

}

}

public int RoomWidth//Свойство ширины комнаты

{

get { return roomWidth; }

set { roomWidth = value; }

}

public int RoomLength//Свойство длины комнаты

{

get { return roomLength; }

set { roomLength = value; }

}

public int RoomHeight//Свойство высоты комнаты

{

get { return roomHeight; }

set { roomHeight = value; }

}

public Package LastDeleted

{

get { return lastDeleted; }

}

public void AddPackage(Package p)//Добавление товара в список товаров и занесение в матрицу занятых позиций

{

for (int i=0; i<roomWidth-p.Width; i++)

{

for (int j=0; j<roomLength-p.Length; j++)

{

//if (i + p.Width > roomWidth) break;

//if (j + p.Length > roomLength) break;

if (i+p.Width > roomWidth && j + p.Length > roomLength)

throw new RoomException("Комната заполнена, попробуйте перейти в другую комнату");

if (isFree(i, j, (int)p.Width + 1, (int)p.Length + 1)) {

p.NewPosition(i, j);

p.Code = lastNumber;

packageList.Add(p);

lastNumber++;

ocupatePosition(i, j,(int) p.Width+1,(int)p.Length+1);

return;

}

}

}

}

private void ocupatePosition(int x, int y, int w, int l)//Метод для отметки занятых позиций начиная с (х,у) шириной w и длиной l

{

if (x + w > roomWidth || y + l > roomLength) throw new RoomException("В комнате не может быть размещен данный продукт");

for (int i = x; i <=x+ w; i++)

{

for (int j = y; j <=y+l; j++)

{

positions[i][j] = true;

}

}

}

private bool isFree(int x, int y, int w, int l)//Метод проверки свободности позиций

{

if (x + w > roomWidth || y + l > roomLength) throw new RoomException("В комнате не может быть размещен данный продукт");

for (int i=x; i<=x+w; i++)

{

for (int j=y; j<=y+l; j++)

{

if (positions[i][j] != false)

return false;

}

}

return true;

}

public Package[] AllProducts()//Метод возращающий массив всех товаров на складе

{

return packageList.ToArray();

}

public IEnumerable<Package> SearchName(string s)//Метод поиска по имени, возвращает элементы содержащие часть строки в имени

{

for (int i = 0; i < packageList.Count; i++)

{

if (packageList[i].Product.Name.Contains(s))

yield return packageList[i];

}

}

public IEnumerable<Package> SearchDeliveryman(string s)//Поиск по Доставщику

{

for (int i = 0; i < packageList.Count; i++)

{

if (packageList[i].Product.ProductProvider.Deliveryman.Contains(s))

yield return packageList[i];

}

}

public IEnumerable<Package> SearchCompany(string s)//Поиск по компании поставщику

{

for (int i = 0; i < packageList.Count; i++)

{

if (packageList[i].Product.ProductProvider.CompanyName.Contains(s))

yield return packageList[i];

}

}

public IEnumerable<Package> SearchDate(string s)//Поиск по дате

{

for (int i = 0; i < packageList.Count; i++)

{

if (packageList[i].Product.CloseDate(s))

yield return packageList[i];

}

}

public void View(Graphics g)//Графическое отображение хранилища без масштабирования

{

g.DrawRectangle(new Pen(Color.Black), 0, 0, roomWidth, roomLength);

foreach(Package p in packageList)

{

p.Draw(g);

}

}

public void View(Graphics g, int n)//Графическое отображение хранилища с масштабированием

{

g.Clear(Color.White);

g.DrawRectangle(new Pen(Color.Black), 0, 0, roomWidth\*n, roomLength\*n);

foreach (Package p in packageList)

{

p.Draw(g,n);

}

}

private void ClearPositions(int x,int y,int w,int l)

{

if (x + w > roomWidth || y + l > roomLength) throw new RoomException("В комнате не может быть размещен данный продукт");

for (int i = x; i <= x + w; i++)

{

for (int j = y; j <= y + l; j++)

{

positions[i][j] = false;

}

}

}

public void Delepe(int c)//Удаление Товара c

{

if (c < 0) throw new RoomException("Невозможно удалить несуществующее");

if (c < lastNumber)

{

foreach (Package p in packageList)

if (p.Code == c)

{

lastDeleted = p;

ClearPositions(p.Position.X, p.Position.Y, (int) p.Width+1, (int)p.Length+1);

packageList.Remove(p);

return;

}

}

else

throw new RoomException("Такого элемента нету на складе");

}

public override string ToString()//Метод для получения строки, которую будет записано в файл-базу

{

string rez = "";

foreach(Package p in packageList)

{

rez += p.ToString() + "\r\n";

}

return rez;

}

public Package MouseOverTheElement(float mx, float my, int n)//Проверка позиции мыши над элементом pictureBox

{

foreach(Package p in packageList)

{

if (p.CheckPosition(mx, my, n))

return p;

}

return null;

}

public Package LastAdded//Свойство возвращающее последний добавленный элемент

{

get {

if (packageList.Count!=0) return packageList.Last();

else throw new RoomException("База данных пуста, попробуйте добавить что-то");

}

}

public string StringToPrintInverntary()//Получение строки для печати приездной накладной

{

string rez="";

foreach (Package p in packageList)

{

rez += p.StringToPrint();

}

return rez;

}

}

internal class RoomException : Exception

{

public RoomException()

{

}

public RoomException(string message) : base(message)

{

}

public RoomException(string message, Exception innerException) : base(message, innerException)

{

}

protected RoomException(SerializationInfo info, StreamingContext context) : base(info, context)

{

}

}

Клас Package

class Package

{

private Product product;

private int code;

private Point position;

private float width;

private float height;

private float length;

public Package(Product p, int w, int h, int l)

{

product = p;

position.X = 0;

position.Y = 0;

width = w;

height = h;

length = l;

}

public Package()

{

position.X = 0;

position.Y = 0;

width = 1;

height = 1;

length = 1;

product = new Product();

}

public Package(string[] fileString)

{

int temp = 0;

product = new Product(fileString);

if (int.TryParse(fileString[9], out temp))

position.X = temp;

else

throw new PackageException("Ошибка в файле. Невозможно получить позицию пакета");

if (int.TryParse(fileString[10], out temp))

position.Y = temp;

else

throw new PackageException("Ошибка в файле. Невозможно получить позицию пакета");

if (!float.TryParse(fileString[11], out width))

throw new PackageException("Ширина пакета не указана");

if (!float.TryParse(fileString[12], out height))

throw new PackageException("Высота пакета не указана");

if (!float.TryParse(fileString[13], out length))

throw new PackageException("Длина пакета не указана");

}

public int Code { get { return code; } set { code = value; } }

public Point Position { get { return position; } set { position = value; } }

public float Width { get { return width; } set { width = value; } }

public float Height { get { return height; } set { height = value; } }

public float Length { get { return length; } set { length = value; } }

public bool CheckPosition(float mx, float my, int mas)

{

mx /= mas;

my /= mas;

return position.X <= mx && position.X + width >= mx && position.Y <= my && position.Y + length >= my;

}

public string ToolString()

{

return "Продукт: "+product.Name+Environment.NewLine + "Занимаемая ширина на складе: " + width + Environment.NewLine + "Занимаемая высота на складе: " + height + Environment.NewLine + "Занимаемая длина на складе: " + length + Environment.NewLine;

}

public void NewPosition(int x, int y)

{

position.X = x;

position.Y = y;

}

public Product Product { get { return product; } set { product = value; } }

public void Draw(Graphics g)

{

g.DrawRectangle(new Pen(Color.Black), position.X, position.Y, width, length);

}

public void Draw(Graphics g, int n)

{

g.DrawRectangle(new Pen(Color.Black), position.X \* n, position.Y \* n, width \* n, length \* n);

}

public void DrawFill(Graphics g,int n)

{

g.FillRectangle(new SolidBrush(Color.LawnGreen),position.X\*n,position.Y,width\*n,length\*n);

}

public void DrawFill(Graphics g, int n, Color c)

{

g.FillRectangle(new SolidBrush(c), position.X \* n, position.Y, width \* n, length \* n);

}

public override string ToString()

{

return product.ToString() + position.X + ";" + position.Y + ";" + width + ";" + height + ";" + length + ";";

}

public string StringToPrint()

{

return product.StringToPrint()+"Занимаемая ширина на складе: \t"+ width + Environment.NewLine + "Занимаемая высота на складе: \t" + height + Environment.NewLine+ "Занимаемая длина на складе: \t" + length + "\r\n\r\n\r\n"+ DateTime.Today.ToString("d")+"\r\n"+ "-----------------------------------------------"+Environment.NewLine;

}

}

class PackageException : Exception

{

public PackageException(string message) : base(message)

{

}

}

Клас Product

class Product

{

private string name;

private int amount;

private float cost;

private string currency;

private string measure;

private Provider productProvider;

private string dateOfIncome;

private string endDate;

public Product()

{

productProvider = new Provider();

}

public Product(string n, int a, int c, string cur, string m)

{

name = n;

amount = a;

cost = c;

currency = cur;

measure = m;

}

public Product(string[] fileLine)

{

productProvider = new Provider();

if (fileLine[0] != "")

name = fileLine[0];

else

throw new ProductException("В файле отсутствует поле названия продукта");

if (!int.TryParse(fileLine[1], out amount))

throw new ProductException("Ошибка в хранении поля количества");

if (!float.TryParse(fileLine[2], out cost))

throw new ProductException("Ошибка в хранении поля числовой части цены");

if (fileLine[3] == "")

throw new ProductException("В файле отсутствует поле валюты");

else

currency = fileLine[3];

if (fileLine[4] == "")

throw new ProductException("В файле отсутствует поле меры измерения");

else

measure = fileLine[4];

if (fileLine[5] == "")

throw new ProductException("В файле отсутствует имя поставщика");

else

productProvider.Deliveryman = fileLine[5];

if (fileLine[6] == "")

throw new ProductException("В файле отсутствует фирма поставщик");

else

productProvider.CompanyName = fileLine[6];

if (fileLine[7] == "")

throw new ProductException("В файле отсутствует дата завоза продукта "+name);

else

dateOfIncome = fileLine[7];

if (fileLine[8] == "")

throw new ProductException("В файле отсутствует дата вывода продукта "+name);

else

endDate= fileLine[8];

}

public string Name { get { return name; } set { name = value; } }

public int Amount { get { return amount; } set { amount = value; } }

public float Cost { get { return cost; } set { cost = value; } }

public string Currency { get { return currency; } set { currency = value; } }

public string Measure { get { return measure; } set { measure = value; } }

public Provider ProductProvider { get { return productProvider; } set { productProvider = value; } }

public string DateOfIncome { get { return dateOfIncome; } set { dateOfIncome = value; } }

public string EndDate { get { return endDate; } set { endDate = value; } }

private int[] getIntDate(string s)

{

string[] st = s.Split('.');

int[] rez= { int.Parse(st[0]), int.Parse(st[1]), int.Parse(st[2]) };

return rez;

}

private bool dateIsLess(int[] checkingDate, int[] thisClassDate)

{

return checkingDate[2] < thisClassDate[2] || (checkingDate[2] == thisClassDate[2] && checkingDate[1] < thisClassDate[1]) || (checkingDate[2] == thisClassDate[2] && checkingDate[1] == thisClassDate[1] && checkingDate[0] <= thisClassDate[0]);

}

public string StringToPrint()

{

return "Название продукта:\t\t"+name + "\r\n Количество: \t\t\t" + amount + "\r\n Цена:\t\t\t\t" + cost + "\r\n Валюта:\t\t\t\t" + currency + "\r\n Единица измерения: \t\t" + measure + "\r\n" + productProvider.StringToPrint() + "\r\n Дата завоза товара:\t\t" + dateOfIncome + "\r\n Дата вывоза:\t\t\t" + endDate + "\r\n";

}

public bool CloseDate(string d)

{

int[] temporary = getIntDate(d);

return dateIsLess(temporary, getIntDate(endDate)) && dateIsLess(getIntDate(dateOfIncome), temporary);

}

public override string ToString()

{

return name + ";" + amount + ";" + cost + ";" + currency + ";" + measure + ";" + productProvider.ToString() + dateOfIncome + ";" + endDate + ";";

}

}

class ProductException : Exception

{

public ProductException(string message) : base(message)

{

}

}

Клас Provider

class Provider

{

private string deliverymanName;

private string companyName;

public Provider()

{

}

public Provider(string d,string cName)

{

deliverymanName = d;

companyName = cName;

}

public string CompanyName//Свойство названия компании

{

get { return companyName; }

set { companyName = value; }

}

public string Deliveryman//Свойство имени поставщика

{

get { return deliverymanName; }

set { deliverymanName = value; }

}

public override string ToString()

{

return deliverymanName + ";" + companyName + ";";

}

public string StringToPrint()

{

return "Поставщик:\t\t\t" + deliverymanName + "\r\nКомпания поставщик:\t" + companyName +"\r\n";

}

}

Класс Form1 Основні методи

public partial class Form1 : Form

{

DataTable table =new DataTable();

private Room current;

public Form1()

{

InitializeComponent();

startDatePicker.Format = DateTimePickerFormat.Custom;

startDatePicker.CustomFormat = "dd.MM.yyyy";

endDatePicker.Format = DateTimePickerFormat.Custom;

endDatePicker.CustomFormat = "dd.MM.yyyy";

searchDatePicker.Format = DateTimePickerFormat.Custom;

searchDatePicker.CustomFormat = "dd.MM.yyyy";

trackBar1.Maximum = 6;

trackBar1.TickFrequency = 1;

trackBar1.LargeChange = 3;

trackBar1.SmallChange = 1;

trackBar1.Value = 1;

pictureBox1.BackColor = Color.White;

current = new Room();

}

private Package PackageFromTextBoxes()//Получение информации о продукте из текстовых полей формы

{

float valo;

int val;

Package pr = new Package();

//packageName

if (packageNameBox.Text != null && packageNameBox.Text != "")

{

pr.Product.Name = packageNameBox.Text;

}

else

throw new TextBoxesException("Вы не задали название продукта");

//amount

if (amountBox.Text != null && amountBox.Text != "")

if (int.TryParse(amountBox.Text, out val))

pr.Product.Amount = val;

else

throw new TextBoxesException("В поле количества введены не коректные данные.");

else

throw new TextBoxesException("Содержимое поля количества продуктов пусто. Введите количество продуктов.");

//cost

if (costBox.Text != null && costBox.Text != "")

{

if (float.TryParse(costBox.Text, out valo))

pr.Product.Cost = valo;

else

throw new TextBoxesException("Введенные данные цены не являются числовыми");

}

else

throw new TextBoxesException("Поле цена пустое. Введите цену продукта");

//currency

if (currencyBox.Text != null && currencyBox.Text != "")

pr.Product.Currency = currencyBox.Text;

else

throw new TextBoxesException("Валюта не указана");

//measure

if (measureBox.Text != null && measureBox.Text != "")

pr.Product.Measure = measureBox.Text;

else

throw new TextBoxesException("Единица измерения не указана");

//provider`s info

//deliveryman Name

if (deliverymanNameBox.Text != null && deliverymanNameBox.Text != "")

pr.Product.ProductProvider.Deliveryman = deliverymanNameBox.Text;

else

throw new TextBoxesException("Доставщик не указан");

//companyName

if (companyNameBox.Text != null && companyNameBox.Text != "")

pr.Product.ProductProvider.CompanyName = companyNameBox.Text;

else

throw new TextBoxesException("Компания не указана");

//date

checkDate(); //Проверка даты в полях

pr.Product.DateOfIncome = startDatePicker.Text;

pr.Product.EndDate = endDatePicker.Text;

//Width

if (float.TryParse(packageWidthBox.Text, out valo))

pr.Width = valo;

else

throw new TextBoxesException("Не числовое значение в ширине пакета");

//Height

if (float.TryParse(packageHeightBox.Text, out valo))

pr.Height = valo;

else

throw new TextBoxesException("Не числовое значение в высоты пакета");

//Length

if (float.TryParse(packageLengthBox.Text, out valo))

pr.Length = valo;

else

throw new TextBoxesException("Не числовое значение в длине пакета");

return pr;

}

private void checkDate()//Проверка даты на форме

{

int val;

string[] parser = startDatePicker.Text.Split('.');

int[] date1 = new int[3];

for (int i = 0; i < parser.Length; i++)

if (Int32.TryParse(parser[i], out val))

date1[i] = val;

else

throw new TextBoxesException("Не правильный формат даты завоза продукта");

//EndDate

parser = endDatePicker.Text.Split('.');

int[] date2 = new int[3];

for (int i = 0; i < parser.Length; i++)

if (Int32.TryParse(parser[i], out val))

date2[i] = val;

else

throw new TextBoxesException("Дата вывоза не правильного формата");

if (date2[2] < date1[2] || (date1[2] == date2[2] && date1[1] > date2[1]) || (date1[2] == date2[2] && date1[1] == date2[1] && date1[0] > date2[0]))

throw new TextBoxesException("Дата вывоза должна быть больше чем дата завоза");

}

private bool CheckDateFile(string s)//Метод выберает по дате какие данные загружать из файла, а какие нет

{

string[] currdate = DateTime.Today.ToString("d").Split('.');

string[] checkingDate = s.Split('.');

return (Convert.ToInt32(currdate[2]) < Convert.ToInt32(checkingDate[2])) || (Convert.ToInt32(currdate[2]) == Convert.ToInt32(checkingDate[2]) && Convert.ToInt32(currdate[1]) < Convert.ToInt32(checkingDate[1])) || (Convert.ToInt32(currdate[2]) == Convert.ToInt32(checkingDate[2]) && Convert.ToInt32(currdate[1]) == Convert.ToInt32(checkingDate[1]) && Convert.ToInt32(checkingDate[0]) >= Convert.ToInt32(currdate[0]));

}

private Room LoadStorageBase()//Загрузка данных из файла

{

Room room = new Room(current.RoomWidth,current.RoomLength,current.RoomHeight);

using (TextReader reader = new StreamReader("DataBase.txt"))

{

string line;

while ((line = reader.ReadLine()) != null&&line!="")

{

string[] sparse = line.Split(';');

if (CheckDateFile(sparse[8]))

{

room.AddPackage(new Package(sparse));

}

}

current = room;

return room;

}

}

private void DisplayPackage(Package p)//Добавление продукта в таблицу для отображения

{

DataRow r = table.NewRow();

r["Code"] = p.Code;

r["Product"] = p.Product.Name;

r["Amount"] = p.Product.Amount;

r["Cost"] = p.Product.Cost;

r["Currency"] = p.Product.Currency;

r["Measure"] = p.Product.Measure;

r["Deliveryman"] = p.Product.ProductProvider.Deliveryman;

r["Company"] = p.Product.ProductProvider.CompanyName;

r["Income Date"] = p.Product.DateOfIncome;

r["End Date"] = p.Product.EndDate;

table.Rows.Add(r);

}

private void ShowProducts(Room Storage1)//Отображение продуктов получаемых массивом

{

table.Rows.Clear();

Package[] ps = Storage1.AllProducts();

if (table.Columns.Count == 0)

{

table.Columns.Add("Code");

table.Columns.Add("Product");

table.Columns.Add("Amount");

table.Columns.Add("Cost");

table.Columns.Add("Currency");

table.Columns.Add("Measure");

table.Columns.Add("Deliveryman");

table.Columns.Add("Company");

table.Columns.Add("Income Date");

table.Columns.Add("End Date");

}

for (int i = 0; i < ps.Length; i++)

{

DisplayPackage(ps[i]);

}

resultBoxDataTable.DataSource = table;

}

private void Save(Room s1)//Сохранить в файл данные

{

using (TextWriter tw = new StreamWriter("DataBase.txt"))

{

tw.Write(s1.ToString());

}

}

private void Form1\_Load(object sender, EventArgs e)

{

}

private void ClearAllTextBoxes()//Очистка всех текстовых полей, например после получения всей информации из них

{

packageNameBox.Clear();

amountBox.Clear();

costBox.Clear();

currencyBox.Clear();

measureBox.Clear();

deliverymanNameBox.Clear();

companyNameBox.Clear();

endDatePicker.ResetText();

startDatePicker.ResetText();

packageHeightBox.Clear();

packageLengthBox.Clear();

packageWidthBox.Clear();

}

private void button1\_Click(object sender, EventArgs e)//Добавление продукта в базу данных

{

try

{

Package pr = PackageFromTextBoxes();

current.AddPackage(pr);

Save(current);

ShowProducts(current);

ClearAllTextBoxes();

}

catch (TextBoxesException ex)

{

MessageBox.Show(ex.Message, "TextBoxesFailure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

catch (DataException)

{

MessageBox.Show("Попробуйте изменить файл, или проверьте вручную данные в файле на соответсвие типов", "Данные в файле ошибочны", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

catch (PackageException ex)

{

MessageBox.Show(ex.Message, "PackageFailure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

catch (ProductException ex)

{

DialogResult result = MessageBox.Show(ex.Message, "ProductFailure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void resultBoxDataTable\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void ShowProducts(IEnumerable<Package> p)//метод отображения товаров в Табличном виде в вкладке Search

{

table.Rows.Clear();

if (table.Columns.Count == 0)

{

table.Columns.Add("Code");

table.Columns.Add("Product");

table.Columns.Add("Amount");

table.Columns.Add("Cost");

table.Columns.Add("Currency");

table.Columns.Add("Measure");

table.Columns.Add("Deliveryman");

table.Columns.Add("Company");

table.Columns.Add("Income Date");

table.Columns.Add("End Date");

}

foreach (var i in p)

{

DisplayPackage(i);

}

searchDataGrid.DataSource = table;

}

private void searchByNameBox\_TextChanged(object sender, EventArgs e)//метод поиска товара по названию продукта

{

ShowProducts(current.SearchName(searchByNameBox.Text));

}

private void searchDatePicker\_ValueChanged(object sender, EventArgs e)//метод поиска по дате

{

ShowProducts(current.SearchDate(searchDatePicker.Text));

}

private void searchDeliveryBox\_TextChanged(object sender, EventArgs e)//метод поиска по имени поставщика

{

ShowProducts(current.SearchDeliveryman(searchDeliveryBox.Text));

}

private void searchCompanyBox\_TextChanged(object sender, EventArgs e)//метод поиска по названию компании-поставщика

{

ShowProducts(current.SearchCompany(searchCompanyBox.Text));

}

private void loadBaseToolStripMenuItem\_Click(object sender, EventArgs e)

{

current=LoadStorageBase();

ShowProducts(current);

}

private void button2\_Click(object sender, EventArgs e)

{

if (trackBar1.Value == 1)

{

Graphics g = pictureBox1.CreateGraphics();

current.View(g);

}

}

private void resultBoxDataTable\_ColumnHeaderMouseDoubleClick(object sender, DataGridViewCellMouseEventArgs e)

{

}

private void trackBar1\_Scroll(object sender, EventArgs e)

{

Graphics g = pictureBox1.CreateGraphics();

current.View(g, trackBar1.Value);

}

private void printToolStripMenuItem\_Click(object sender, EventArgs e)

{

try

{

using (TextWriter tw = new StreamWriter("PrintLastIn.txt"))

{

tw.Write(current.LastAdded.StringToPrint());

}

printDialog1.AllowSomePages = true;

printDialog1.ShowHelp = true;

printDialog1.Document = printDocument1;

DialogResult result = printDialog1.ShowDialog();

if (result == DialogResult.OK)

{

printDocument1.Print();

}

}

catch (RoomException ex)

{

MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void printInventaryToolStripMenuItem\_Click(object sender, EventArgs e)

{

using (TextWriter tw = new StreamWriter("Inventary.txt"))

{

tw.Write(current.StringToPrintInverntary());

}

printDialog1.AllowSomePages = true;

printDialog1.ShowHelp = true;

printDialog1.Document = printDocument2;

DialogResult result = printDialog1.ShowDialog();

if (result == DialogResult.OK)

{

printDocument2.Print();

}

}

private void pictureBox1\_MouseDown(object sender, MouseEventArgs e)

{

Package p = current.MouseOverTheElement(e.X, e.Y, trackBar1.Value);

if (p!= null)

{

toolTip1.Show("", this);

toolTip1.ToolTipTitle = "Product";

toolTip1.ToolTipIcon = ToolTipIcon.Info;

toolTip1.SetToolTip(pictureBox1, p.ToolString());

}

else

{

toolTip1.SetToolTip(pictureBox1, "");

}

}

private void exitToolStripMenuItem\_Click(object sender, EventArgs e)

{

Close();

}

private void printDocument2\_PrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e)

{

string text = current.StringToPrintInverntary();

System.Drawing.Font printFont = new System.Drawing.Font("Times New Roman", 14, System.Drawing.FontStyle.Regular);

// Draw the content.

e.Graphics.DrawString(text, printFont, System.Drawing.Brushes.Black, 10, 10);

}

private void pictureBox1\_MouseUp(object sender, MouseEventArgs e)

{

toolTip1.Hide(this);

}

private Room CreateStorage()// метод создания хранилища с параметрами из текстбоксов

{

int w=0, h=0, l=0;

if (roomWidthBox.Text != null && roomWidthBox.Text != "")

{

if (!int.TryParse(roomWidthBox.Text, out w))

throw new TextBoxesException("Поле ширины хранилища указано неверно");

}

else

throw new TextBoxesException("Содержимое поля ширины хранилища отсутствует.");

if (roomLengthBox.Text != null && roomLengthBox.Text != "")

{

if (!int.TryParse(roomLengthBox.Text, out l))

throw new TextBoxesException("Поле длины хранилища указано неверно");

}

else

throw new TextBoxesException("Содержимое поля длины хранилища отсутствует.");

if (roomHeightBox.Text != null && roomHeightBox.Text != "")

{

if (!int.TryParse(roomHeightBox.Text, out h))

throw new TextBoxesException("Поле высоты хранилища указано неверно");

}

else

throw new TextBoxesException("Содержимое поля высоты хранилища отсутствует.");

roomHeightBox.Clear();

roomLengthBox.Clear();

roomWidthBox.Clear();

return new Room(w, l, h);

}

private void button3\_Click(object sender, EventArgs e)

{

try

{

current = CreateStorage();

Graphics g = pictureBox2.CreateGraphics();

current.View(g);

}

catch (TextBoxesException ex)

{

MessageBox.Show(ex.Message, "TextBoxesFailure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

catch (PackageException ex)

{

MessageBox.Show(ex.Message, "PackageFailure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

catch (ProductException ex)

{

DialogResult result = MessageBox.Show(ex.Message, "ProductFailure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void printDocument1\_PrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e)

{

string text = current.LastAdded.StringToPrint();

System.Drawing.Font printFont = new System.Drawing.Font("Times New Roman", 14, System.Drawing.FontStyle.Regular);

// Draw the content.

e.Graphics.DrawString(text, printFont, System.Drawing.Brushes.Black, 10, 10);

}

private void button4\_Click(object sender, EventArgs e)

{

try

{

int code;

if (int.TryParse(codeBox.Text, out code))

current.Delepe(code);

else

throw new TextBoxesException("В поле Code должен быть указан только целочисленный код товара");

Save(current);

ShowProducts(current);

using (TextWriter tw = new StreamWriter("PrintLastDeleted.txt"))

{

tw.Write(current.LastDeleted.StringToPrint());

}

printDialog1.AllowSomePages = true;

printDialog1.ShowHelp = true;

printDialog1.Document = printDocument3;

DialogResult result = printDialog1.ShowDialog();

if (result == DialogResult.OK)

{

printDocument3.Print();

}

}

catch (TextBoxesException ex)

{

MessageBox.Show(ex.Message, "TextBoxFailure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

catch (RoomException ex)

{

MessageBox.Show(ex.Message, "RoomFailure", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void printDocument3\_PrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e)

{

string text = current.LastDeleted.StringToPrint();

System.Drawing.Font printFont = new System.Drawing.Font("Times New Roman", 14, System.Drawing.FontStyle.Regular);

// Draw the content.

e.Graphics.DrawString(text, printFont, System.Drawing.Brushes.Black, 10, 10);

}

}

class DataException : Exception

{

}

class TextBoxesException : Exception

{

public TextBoxesException()

{

}

public TextBoxesException(string message) : base(message)

{

}

public TextBoxesException(string message, Exception innerException) : base(message, innerException)

{

}

protected TextBoxesException(SerializationInfo info, StreamingContext context) : base(info, context)

{

}

}