МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА Комп’ютерних наук

Секція інформаційно-комунікаційних технологій

КУРСОВА РОБОТА

з дисципліни: «Програмування під платформу .NET»

Виконала: студентка Шелест А.В. групи № Ін-81/2

Перевірила: ст.викладач, к.т.н. Москаленко А.С.

СУМИ 2020

ЗМІСТ

[Вступ 3](#_Toc41238365)

[1 Теоретична частина 4](#_Toc41238366)

[2 Практична частина 6](#_Toc41238367)

[2.1 Діаграма класів 6](#_Toc41238368)

[2.2 Опис класу Shop 6](#_Toc41238369)

[2.3 Опис класу Product 8](#_Toc41238370)

[2.4 Опис класу Consumer 9](#_Toc41238371)

[2.5 Опис класу Program 10](#_Toc41238372)

[2.6 Файли проекту 11](#_Toc41238373)

[2.7 Результат роботи програми 11](#_Toc41238374)

[2.8 Виключення, оброблені в програмі 13](#_Toc41238375)

[2.9 Можливі помилки в роботі програми 14](#_Toc41238376)

[2.10 Приклад роботи програми 15](#_Toc41238377)

[2.11 Код програми 19](#_Toc41238378)

[Висновки 20](#_Toc41238379)

[Список використаної літератури 21](#_Toc41238380)

[Додаток А 22](#_Toc41238381)

[Додаток Б 24](#_Toc41238382)

[Додаток В 34](#_Toc41238383)

[Додаток Г 37](#_Toc41238384)

Вступ

У наш час інтернет-магазини є дуже розповсюдженими та зручними у використанні. Так як це актуально, метою цієї курсової роботи є створення моделі магазину мовою C# та опрацювання теорії про системи контролю версій, а саме – Git.

Для досягнення мети потрібно створити зручну для користувача програму та викласти її, використовуючи систему контролю версій Git. У програмі повинна бути можливість додати в кошик товар (інформація про корзину буде виводитись у форматі: зроблено замовлення у кількості N штук на суму Z грн.), видалити товар зробити замовлення/підтвердити, переглянути вміст кошика. Також повинна виводитися інформація про доставку в різні міста або країни та її ціну. Розраховувати знижку в залежності від суми замовлення або з нагоди дня народження клієнта. Клієнт повинен ознайомитися з контактною інформацією магазину та мати можливість обирати операції з замовленням. Також, потрібно зробити функцію пошуку товару за ключовим словом і сортування за ціною списку товарів.

# Теоретична частина

Система керування версіями використовується для керування версіями видів інформації. Системи керування версіями (СКВ) застосовують під час розробки ПЗ для документування та контролю над коректуваннями в електронних документах, які корегуються одночасно декількома людьми. Кожна версія позначена певною цифрою чи літерою, зміни записуються. Також зберігаються дані про того, хто виконав зміну та коли це відбулося [1].

СКВ використовуються також, якщо потрібно зберегти кожну копію зміни. Перевагами є :

Можливість повернути певні файли до попереднього стану.

Можливість повернути весь проект до попереднього стану.

Можливість побачити зміни.

Використання СКВ гарантує повернення того, що видалили [1].

Є 2 основні типи систем керування версіями: з централізованим та розподіленим сховищем.

Централізовані системи контролю версій (ЦСКВ) мають єдиний сервер з усіма версіями файлів, та певну кількість клієнтів, які є отримувачами файлів з центрального місця. Протягом довгого часу, ЦСКВ були стандартом для систем контролю версій [1].

Ці системи мають багато переваг. Наприклад, учасник проекту знає, в деякій мірі, чим займаються інші. Адміністратори контролюють хто і що може виконувати. Набагато легше керувати ЦСКВ, ніж мати справу з локальними базами даних для кожного клієнта [1].

Децентралізовані системи контролю версій - клієнти не просто отримують останній знімок файлів репозиторія: натомість вони є повною копією сховища разом з усією його історією. Таким чином, якщо сервер перестає працювати, через який працюють розробники, будь-який з клієнтських репозиторіїв може бути скопійований у сервер, щоб відновити його. Кожна копія є повною резервною копією всіх даних [1].

Основною відмінністю від інших систем є те, як Git зберігає дані. Багато СКВ зберігають інформацію як низку змін. Інші системи розглядають інформацію як список файлів та редагувань кожного з них за певний проміжок часу. Git зберігає свої дані як список знімків міні-файлової системи. Щоразу при зберіганні стану проекту, Git запам’ятовує як виглядають всі файли в даний час і зберігає посилання на цей знімок. Також, при зберіганні файлів, у які не вносилися змінні, Git просто робить посилання на попередній ідентичний цьому файл, який уже є в системі [1].

Особливості Git:

Майже всі операції є локальними. Більшість операцій у Git не потребують інформацію з інших комп’ютерів вашої мережі.

Git ­ цілісний. Неможливо змінити файл чи директорію таким чином, щоб Git про це не дізнався.

Git тільки додає дані. При певній діяльності в Git майже завжди відбувається тільки додавання інформації до бази даних Git [1].

Git має три основні стани файлів: збережений у коміті (commited), змінений (modified) та індексований (staged):

Збережений у коміті означає, що дані знаходяться в локальній базі даних.

Змінений означає, що файл було змінено, але зміни ще не збережено.

Індексований означає, що при позначенні зміненого файлу у поточній версії, ці зміни будуть в наступному знімку [1].

# Практична частина

* 1. Діаграма класів

Для того, щоб показати структуру ієрархії класів програми, використаємо діаграми класів UML [2]. Діаграма (рис. 1), створена у Visual Studio Community 2019 за допомогою Class Designer.

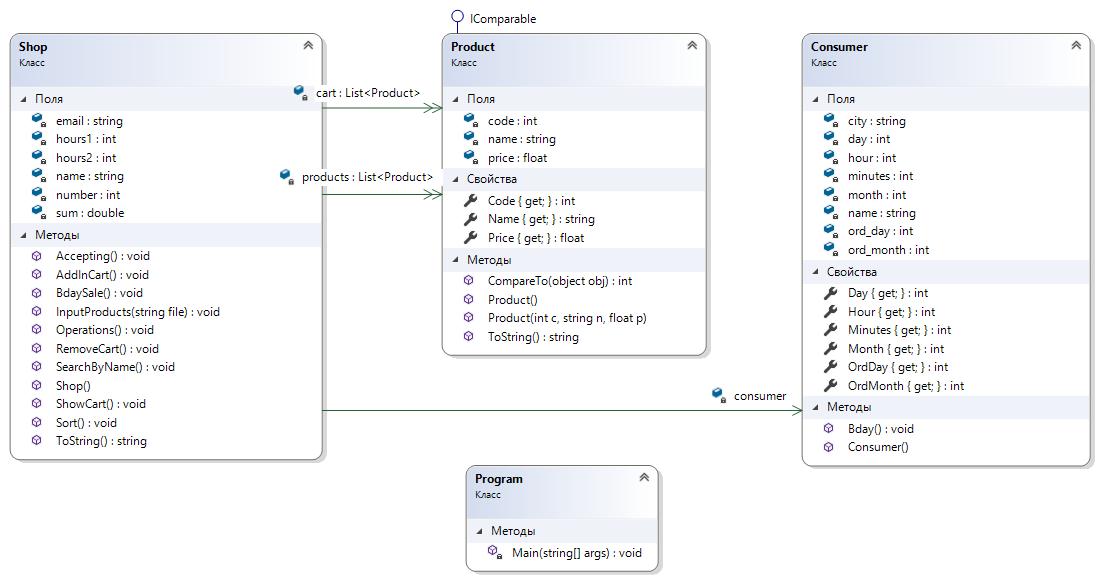


Рисунок 1 – Діаграма класів UML

Із діаграми видно усі поля, властивості та методи (з типами параметрів та їх типи даних, що є у кожному класі, а також, що клас Shop має асоціацію з такими класами як Product та Consumer.

* 1. Опис класу Shop

Клас Shop створений для виконання основних операцій клієнта і має поля (рис. 2):

* 1. Контактна інформація магазину (name, number, email. hours1, hours2).
  2. Загальна сума замовлення (sum).
  3. Інформація про клієнта (consumer).
  4. Список товарів (products).
  5. Кошик (cart).

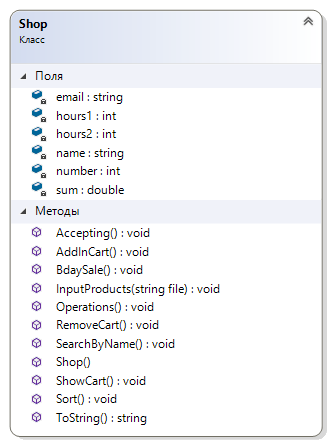


Рисунок 2 – Поля та методи класу Shop

Клас Shop має такі методи:

1. Для підтвердження замовлення та розрахунку доставки – Accepting().
2. Для додавання товару в кошик – AddInCart().
3. Для розрахунку знижки з нагоди дня народження клієнта–BdaySale().
4. Для зчитування списку товарів із файлу ­ InputProducts().
5. Для вибору операцій ­ Operations().
6. Для видалення товару з кошика ­ RemoveCart().
7. Для пошуку за найменуванням – SearchByName.
8. Для сортування за ціною ­ Sort().
9. Для виведення контактної інформації (назва, номер, пошта, часи роботи) про об’єкт класу типу string ­ ToString().
10. Для створення об’єкту класу та присвоєння полям об’єкту значень (конструктор за замовчуванням) ­ Shop().
11. Для показу вмісту кошика ­ ShowCart().
    1. Опис класу Product

Клас створений для надання товару коду, найменування, ціни – це і є поля, які зображено на діаграмі (рис. 3).

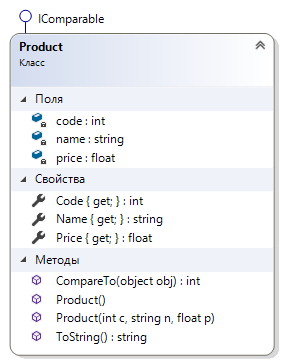


Рисунок 3 – Поля, властивості, методи класу Product

Також, є властивості для доступу (тільки читання) до коду (Code), найменування (Name) та ціни (Price).

Клас Product має методи за допомогою яких можна:

1. Відсортувати товар за ціною ­ CompareTo (object obj).
2. Вивести інформацію про товар ­ Output().
3. Створити екземпляр класу ­ конструктор за замовчуванням Product().
4. Створити екземпляр класу ­ конструктор з параметрами Product(int c,string n,float p).
5. Вивести інформацію про об’єкт класу ­ toString().
   1. Опис класу Consumer

Клас для надання та читання інформації клієнта. З частини діаграми (рис. 4) видно, що поля ­ це інформація про клієнта:

1. Місто проживання – city.
2. День народження – day.
3. Година замовлення – hour.
4. Хвилина замовлення – minutes.
5. Місяць народження – month.
6. Ім’я ­ name.
7. День замовлення ­ ord\_day.
8. Місяць замовлення ­ ord\_month.

Також є властивості, призначені для доступу (читання) до полів:

1. Day.
2. Hour.
3. Minutes.
4. Month.
5. OrdDay.
6. OrdMonth.

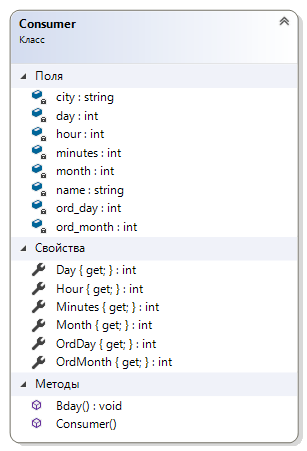


Рисунок 4 – Поля, властивості та методи класу Consumer

* 1. Опис класу Program

Клас, який має метод Main (рис. 5) – точку входу програми.

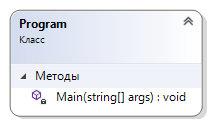


Рисунок 5 ­ Метод класу Program

У цьому методі створено екземпляр класу Shop, викликані усі методи, що створені для належної роботи програми та оброблено виключення.

* 1. Файли проекту

Файл list1.txt (рис. 6) розміщується в папці Debug. Він потрібен для зчитування асортименту товару у методі класу Shop (у форматі Код – Наіменування – Ціна):

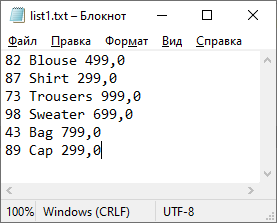


Рисунок 6 – Файл з даними

* 1. Результат роботи програми

Виведення інформації про доставку у різні країни, контактна інформація магазину , список товару, сортування товару зображено на рисунках 7-8.

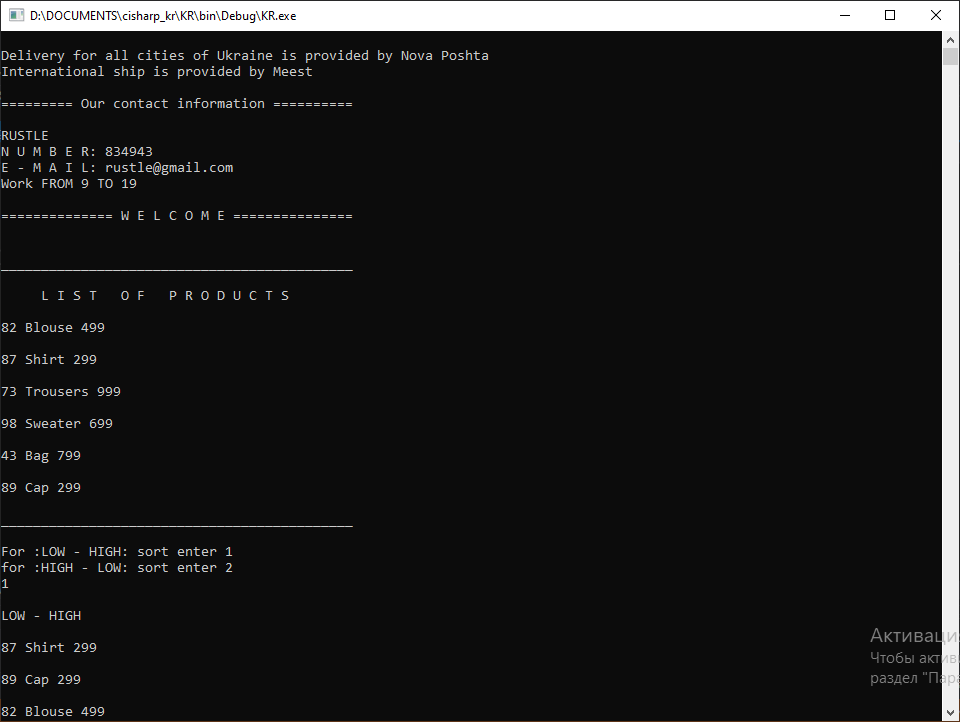


Рисунок 7 – Результат виконання програми

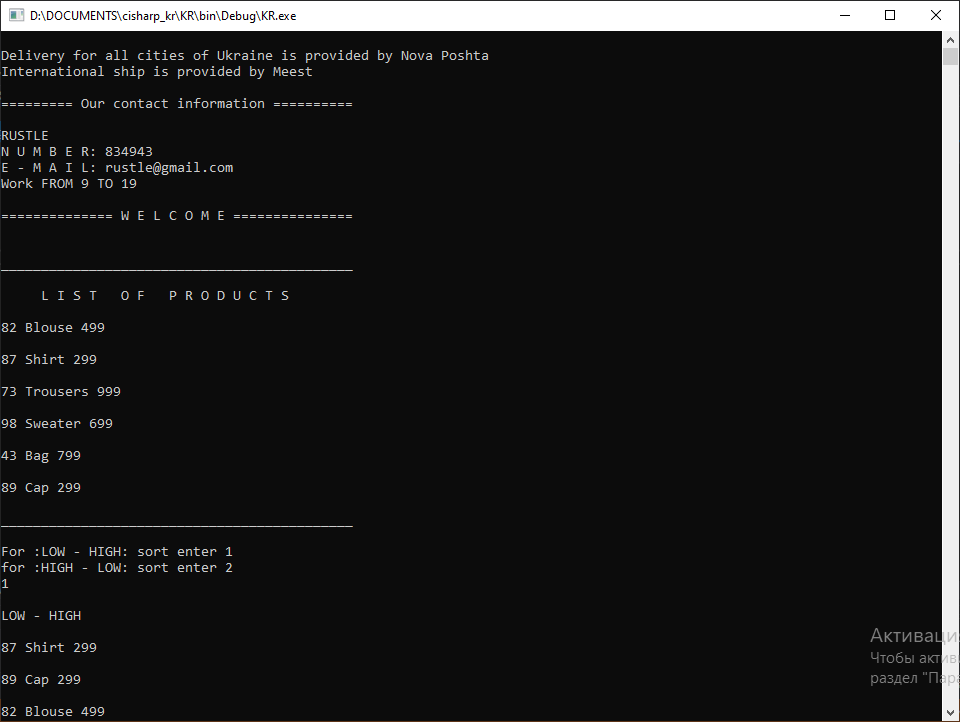


Рисунок 8 – Результат виконання програми

Операція пошуку, вибір операції додавання товару (рис. 9).



Рисунок 9 – Результат виконання програми

Підтвердження замовлення, введення дати народження, отримання кінцевої суми замовлення враховуючи знижки та доставку (рис. 10).

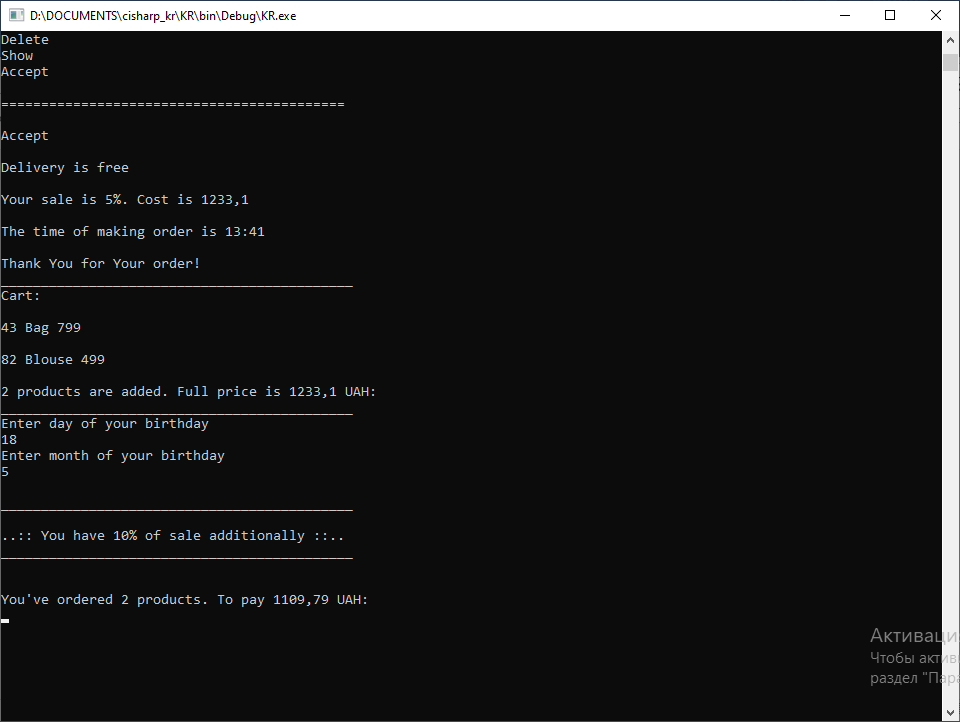


Рисунок 10 – Результат виконання програми

* 1. Виключення, оброблені в програмі

У програмі є обробка таких виключень:

1. FormatException.
2. OverflowException.
3. Overflow.
4. OutOfMemoryException.
5. FileNotFoundException.
6. IndexOutOfRangeException.

Якщо у блоці try відбувається виключення типу FormatException, виводиться повідомлення «Invalid format of input».

Якщо у блоці try відбувається виключення типу OverflowException, виводиться повідомлення «Overflow».

Якщо у блоці try відбувається виключення типу OutOfMemoryException, виводиться повідомлення «Out of memory to create new object».

Якщо у блоці try відбувається виключення типу FileNotFoundException, виводиться повідомлення «File doesn't exist».

Якщо у блоці try відбувається виключення типу IndexOutOfRangeException, виводиться повідомлення «Index out of range».

* 1. Можливі помилки в роботі програми

Якщо файл порожній – виводиться повідомлення «List of products is empty. Enter any key to exit... « і програма очікує натиснення на будь-яку клавішу, щоб закінчити роботу.

Якщо при додаванні або видаленні товару з кошика введено невірний код – виводиться повідомлення «Incorrect code».

Якщо при відповіді на питання «Continue? Enter 1 to continue or 0 to do not"» вводится щось, окрім 1 та 0 – виводиться повідомлення «Incorrect symbol».

Якщо при відповіді на питання «Do you want to delete item? Enter 1 to do or 0 to do not» вводится щось, окрім 1 та 0 – виводиться повідомлення «!!! Incorrect answer !!!».

Якщо кошик порожній - виводиться повідомлення Your cart is empty. Add something in your Shopping cart».

Якщо при пошуку товару введено невірне найменування – виводиться повідомлення «Incorrect name».

Якщо при введенні дати народження клієнта введено невірний день або місяць (день та місяць не можуть бути від’ємними, днів максимум 31, а місяців 12) – виводиться повідомлення «Incorrect date».

* 1. Приклад роботи програми

Одразу після запуску програми виводиться повідомлення про доставку, контактну інформацію та список товарів (рис. 11). Програма пропонує відсортувати список товарів за ціною.

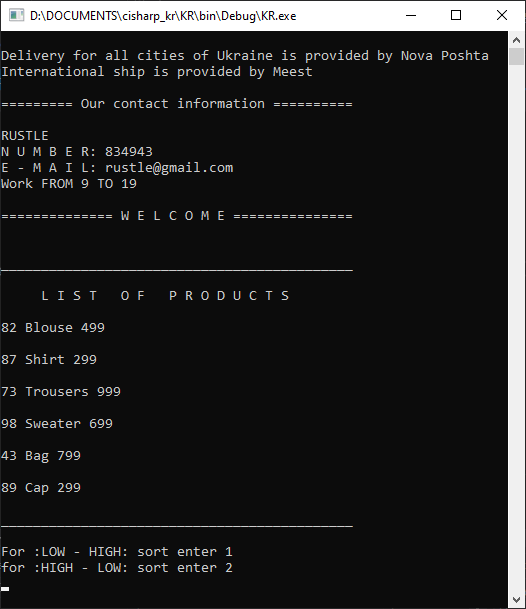


Рисунок 11 – Приклад роботи програми

Натискаємо 1 (сортування за зростанням). Виводиться відсортований список (рис. 12).

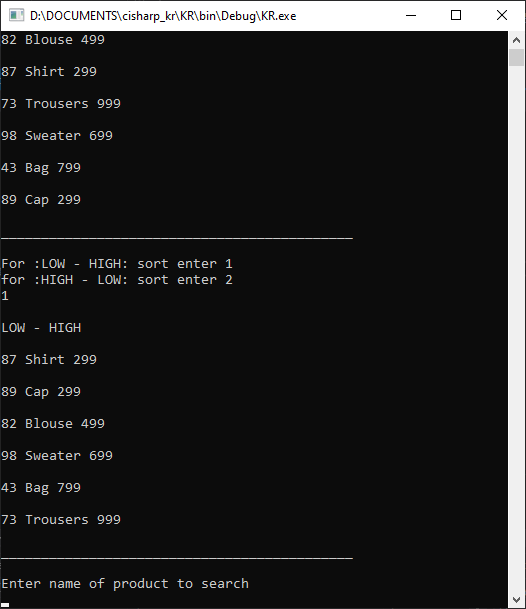


Рисунок 12 ­ Приклад роботи програми

Програма пропонує ввести ім’я для пошуку товару (рис. 13).

Вводимо Bag. Виводиться інформацію про товар.

Програма пропонує обрати операцію. Обираємо додавання товару в кошик.

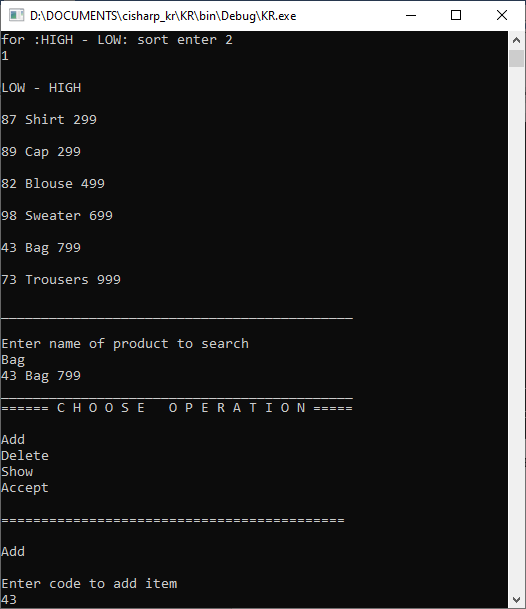


Рисунок 13 – Приклад роботи програми

Вводимо код товару. Виводиться повідомлення про продовження операції. Натискаємо 0(ні). Обираємо підтвердження замовлення. Виводиться повідомлення про доставку, повну суму та час замовлення. Виводиться вміст кошика та сума (рис. 14).

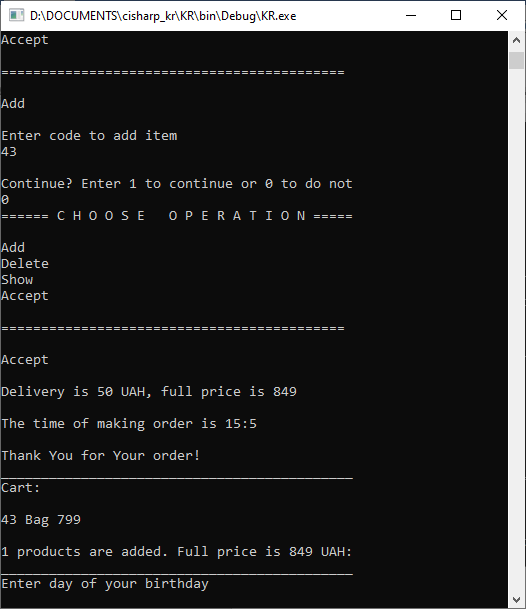


Рисунок 14 – Приклад роботи програми

Вводимо день народження. Потім місяць. У цифровому форматі.

Виводиться повідомлення про знижку та кінцеву суму замовлення (рис. 15).

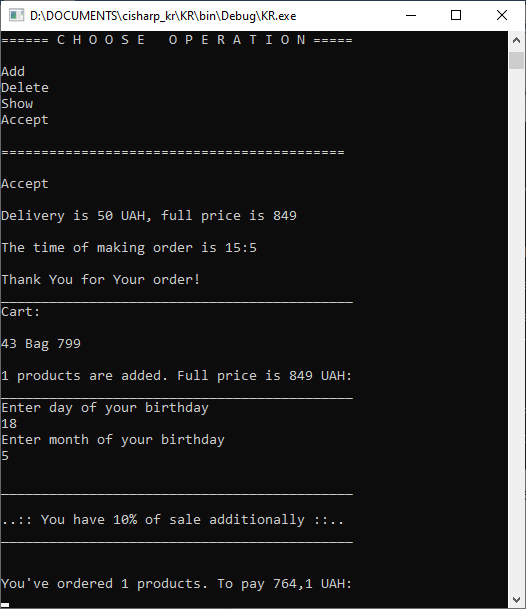


Рисунок 15 – Приклад роботи програми

* 1. Код програми

Код можна знайти в додатках:

1. Клас Program в додатку А.
2. Клас Shop в додатку Б.
3. Клас Consumer в додатку В.
4. Клас Product в додатку Г.

Для зручного перегляду коду та доступу до нього курсову роботу, також, можна знайти за посиланням <https://github.com/alineriia/KR>

Висновки

У ході виконання цієї курсової роботи було створено модель магазину мовою C# за допомогою інтегрованого середовища розробки Visual Studio Community 2019 і засвоєно теорію щодо систем керування версіями, а саме ­ Git. Програма надає можливість додавати товар у кошик, розраховувати доставку, видаляти товар з кошика, розраховувати знижки, підтверджувати замовлення, сортувати список товарів за спаданням і за зростанням, пошук товару за найменуванням.

Для виконання цієї курсової роботи були використані класи, інтерфейси. За допомогою інтегрованого середовища розробки Visual Studio Community 2019 створено діаграму класів UML та опрацьовано матеріал про UML. Закріплено навички роботи з методами, полями, властивостями та файлами. Також, було оброблено декілька типів виключень. Для доступу викладено курсову роботу на GitHub за допомогою інтегрованого середовища розробки Visual Studio Community 2019. Також, було закріплено знання про об’єктно-орієнтоване програмування та мову С#. Загалом, засвоєно матеріал попередніх лабораторних робіт.

Труднощі виникали в процесі зв’язки класів між собою

Створену програму можна вдосконалити базою даних та використанням Windows форм. Також, можна змінити умови доставки та розрахунку знижок.

Список використаної літератури

1. Chacon, S., & Straub, B. (2014). Pro Git (2nd ed.). Berkeley, CA: Apress.
2. Gervasi, O., & Gavrilova, M. The 2007 International Conference on Computational Science and its Applications : proceedings : ICCSA 2007 : Kuala Lumpur, Malaysia, August 26-29, 2007 : selected papers.h.

Додаток А

using System;

using System.Collections.Generic;

using System.Collections;

using System.Linq;

using System.Text;

using System.IO;

using System.Threading.Tasks;

namespace KR

{

class Program

{

static void Main(string[] args)

{

const string q = "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_";

try

{

Shop store = new Shop();

Console.WriteLine(store);

Console.WriteLine(q);

Console.WriteLine("\n L I S T O F P R O D U C T S\n");

store.InputProducts("list1.txt");

Console.WriteLine(q);

store.Sort();

Console.WriteLine(q);

store.SearchByName();

Console.WriteLine(q);

store.Operations();

Console.WriteLine(q);

Console.WriteLine("Cart:");

store.ShowCart();

Console.WriteLine(q);

store.BdaySale();

}

catch (FormatException)

{

Console.WriteLine("Invalid format of input");

}

catch (OverflowException)

{

Console.WriteLine("Overflow");

}

catch (OutOfMemoryException)

{

Console.WriteLine("Out of memory to create new object");

}

catch (IndexOutOfRangeException)

{

Console.WriteLine("Index is out of range");

}

catch (FileNotFoundException)

{

Console.WriteLine("File doesn't exist");

}

Console.ReadKey();

}

}

}

Додаток Б

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.IO;

using System.Text;

using System.Threading.Tasks;

namespace KR

{

class Shop

{

string name;

int number;

string email;

int hours1;

int hours2;

double sum = 0;

List <Product> products = new List<Product>();

Consumer consumer = new Consumer();

public void BdaySale() // Метод для знижки у випадку дня народження клієнта

{

consumer.Bday();

if ((consumer.Day >= consumer.OrdDay || consumer.Day <= consumer.OrdDay + 3) && consumer.Month == consumer.OrdMonth)

{

sum = sum \* 0.9;

Console.WriteLine("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n..:: You have 10% of sale additionally ::..\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

Console.WriteLine("\nYou've ordered {0} products. To pay {1} UAH:", cart.Count, sum);

}

else

{

Console.WriteLine("\n..:: Unfortunately, you don't have any extra sales ::..");

}

}

public void InputProducts(string file) // Метод зчитування списку товарів з файлу

{

string line;

using (StreamReader MyFile = new StreamReader(file))

{

while ((line = MyFile.ReadLine()) != null)

{

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

if (data.Length == 3)

{

Product NewProduct = new Product(int.Parse(data[0]), data[1], float.Parse(data[2]));

products.Add(NewProduct);

}

}

}

if (products.Count != 0)

{

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

{

Console.WriteLine(products[i] + "\n");

}

}

else

{

Console.WriteLine("List of products is empty. Enter any key to exit... "); Console.ReadKey();

}

}

List<Product> cart = new List<Product>();

public void AddInCart() // Метод додавання товару в кошик

{

while (true)

{

Console.WriteLine("\nEnter code to add item");

int cheker = 0;

int code1 = int.Parse(Console.ReadLine());

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

{

if (products[i].Code == code1)

{

cart.Add(products[i]);

sum += products[i].Price;

cheker++;

}

}

if (cheker == 0)

{

Console.WriteLine("Incorrect code");

}

Console.WriteLine("\nContinue? Enter 1 to continue or 0 to do not");

int a = int.Parse(Console.ReadLine());

if (a == 0) { break; }

else

{

if (a == 1) { continue; }

else

Console.WriteLine("Incorrect symbol");

}

}

}

public void RemoveCart() //Метод для видалення товару з кошику

{

if (cart.Count != 0)

{

Console.WriteLine("\nDo you want to delete item? Enter 1 to do or 0 to do not");

int answ1 = int.Parse(Console.ReadLine()); int cheker = 0;

if (answ1 == 1)

{

while (true)

{

Console.WriteLine("Enter code to delete item");

int code1 = int.Parse(Console.ReadLine());

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

{

if (cart[i].Code == code1)

{

sum -= cart[i].Price;

cart.Remove(cart[i]);

cheker++;

}

if (cheker == 0)

{

Console.WriteLine("!!! Incorrect code !!!");

}

}

Console.WriteLine("Continue? Enter 1 to continue or 0 to do not");

int answ2 = int.Parse(Console.ReadLine());

if (answ2 == 0) { break; }

else

{

if (answ2 == 1) { continue; }

else

Console.WriteLine("Incorrect symbol");

}

}

}

else

{

if (answ1 != 0)

{

Console.WriteLine("\n!!! Incorrect answer !!!");

RemoveCart();

}

}

}

else

{

Console.WriteLine("\nYour cart is empty. Add something in your Shopping cart");

}

}

public void Sort() // Метод для сортування списку товарів за ціною

{

Console.WriteLine("\nFor :LOW - HIGH: sort enter 1\nfor :HIGH - LOW: sort enter 2");

string ans = Console.ReadLine();

switch (ans)

{

case "1":

Console.WriteLine("\nLOW - HIGH\n");

products.Sort();

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

{

Console.WriteLine(products[i] + "\n");

}

break;

case "2":

Console.WriteLine("\nHIGH - LOW\n");

products.Sort();

for (int i = products.Count - 1; i > -1; i--)

{

Console.WriteLine(products[i] + "\n");

}

break;

default:

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

{

Console.WriteLine(products[i] + "\n");

}

break;

}

}

public void Operations() // Метод для вибору операції

{

Console.WriteLine("====== C H O O S E O P E R A T I O N =====\n\nAdd\nDelete\nShow\nAccept\n\n===========================================\n");

string sw = Console.ReadLine();

switch (sw)

{

case ("Add"):

AddInCart(); Operations();

break;

case ("Delete"):

RemoveCart(); Operations();

break;

case ("Show"):

ShowCart(); Operations();

break;

case ("Accept"):

Accepting();

Console.WriteLine("Thank You for Your order!");

break;

default:

Console.WriteLine("Incorrect operation");

Operations();

break;

}

}

public void Accepting() // Метод для підтвердження замовлення

{

if (cart.Count != 0)

{

int delivery = 50;

float V = 1000; float V1 = 1500;

if (sum >= V && sum < V1)

{

sum = 0.95 \* sum; Console.WriteLine("\nDelivery is free\n\nYour sale is 5%. Cost is {0}", sum);

}

else

{

if (sum >= V1)

{

sum = 0.93 \* sum;

Console.WriteLine("\nDelivery is free\n\nYour sale is 7%. Cost is {0}", sum);

}

else

{

sum += delivery;

Console.WriteLine("\nDelivery is {0} UAH, full price is {1}", delivery, sum);

}

}

Console.WriteLine("\nThe time of making order is {0}:{1}\n", consumer.Hour, consumer.Minutes);

}

else

{

Console.WriteLine("\nYour cart is empty. Add something in your Shopping cart");

Operations();

}

}

public void ShowCart() // Метод для показу переліку товарів у кошику

{

if (cart.Count != 0)

{

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

{

Console.WriteLine("\n" + cart[i]);

}

Console.WriteLine("\n{0} products are added. Full price is {1} UAH:", cart.Count, sum);

}

else

{

Console.WriteLine("\nYour cart is empty. Add something in your Shopping cart");

Operations();

}

}

public Shop() // Конструктор за замовчуванням

{

name = "RUSTLE";

number = 834943;

email = "rustle@gmail.com";

hours1 = 9;

hours2 = 19;

sum = 0;

if (consumer.Hour < 9 || consumer.Hour > 19)

{

Console.WriteLine("Proccesing will be at {0} to {1}", hours1, hours2);

}

Console.WriteLine("\nDelivery for all cities of Ukraine is provided by Nova Poshta\nInternational ship is provided by Meest");

}

public override string ToString() // Метод для виведення інформації про магазин

{

string line = String.Empty;

line += ("\n========= Our contact information ==========\n\n"+name +" " );

line += ("\nN U M B E R: "+number + " " );

line += ("\nE - M A I L: "+email + " ");

line += ("\nWork FROM " + hours1+" TO "+hours2+ "\n\n============== W E L C O M E ===============\n\n");

return line;

}

public void SearchByName() //Метод пошуку товару за наіменуванням

{

int cheker = 0;

Console.WriteLine("\nEnter name of product to search");

string name1 = Console.ReadLine();

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

{

if (products[i].Name.Equals(name1)) {

Console.WriteLine(products[i]); cheker++;

}

}

if (cheker == 0)

{

Console.WriteLine("Incorrect name");

SearchByName();

}

}

}

}

Додаток В

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace KR

{

class Consumer

{

string name;

int day;

int month;

string city;

int ord\_day;

int ord\_month;

int hour;

int minutes;

public int Day

{

get{return day;}

}

public int Month

{

get { return month; }

}

public Consumer() // Конструктор за замовчуванням

{

name = "Alicia Patrick";

city = "Sumy";

ord\_day = DateTime.Now.Day;

ord\_month = DateTime.Now.Month;

hour = DateTime.Now.Hour;

minutes = DateTime.Now.Minute;

}

public void Bday() // Метод для введення дня народження клієнта

{

Console.WriteLine("Enter day of your birthday");

day = int.Parse(Console.ReadLine());

if (day > 31 || day < 1)

{

Console.WriteLine("Incorrect date");

Bday();

}

else

{

Console.WriteLine("Enter month of your birthday");

month = int.Parse(Console.ReadLine());

if (month > 12 || month < 1)

{

Console.WriteLine("Incorrect date");

Bday();

}

}

}

public int OrdDay

{

get { return ord\_day; }

}

public int Hour

{

get { return hour; }

}

public int Minutes

{

get { return minutes; }

}

public int OrdMonth

{

get { return ord\_month; }

}

}

}

Додаток Г

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace KR

{

class Product: IComparable

{

float price;

string name;

int code;

public Product() { } // Порожній конструктор

public Product(int c, string n, float p) // Конструктор з параметрами

{

code = c;

name = n;

price = p;

}

public override string ToString() // Метод для рядкового представлення об'єкту

{

string line = String.Empty;

line += (code+ " ");

line += (name + " ");

line += (price);

return line;

}

public float Price

{

get

{

return price;

}

}

public int Code

{

get

{

return code;

}

}

public string Name

{

get

{

return name;

}

}

public int CompareTo(object obj) // Метод для сортування об'єктів за ціною

{

Product b = (Product)obj;

if (Price == b.Price) return 0;

else if (Price > b.Price) return 1;

else return -1;

}

}

}