МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «Гомельский государственный технический

университет имени П.О. Сухого»

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

Отчёт по лабораторной работе №7

По дисциплине «Объектно-ориентированное программирование»

**«Наследование классов»**

Выполнил: студент

группы ИТИ-21

Говядкова П. Ю.

Принял: преподаватель

Карабчикова Е. А.

Гомель 2020

**Цель работы:** изучить основы синтаксиса объектно-ориентированного языка программирования, реализацию свойств, методов класса, наследования.

**Задание:**

1. Необходимо решить задачу, согласно варианту (Рисунок 1).
2. При создании классов руководствоваться Code Convention.
3. Весь код должен содержать элементы документирования.
4. При реализации иерархии классов использовать механизм наследования.
5. Разработать модульные тесты для верификации созданного класса.
6. Класс должен быть размещён в библиотеке классов.
7. Модульные тесты – в отдельном проекте.
8. В отдельном проекте реализовать интерфейс WPF.
9. WPF – приложение должно обеспечить ввод, редактирование и просмотр данных.

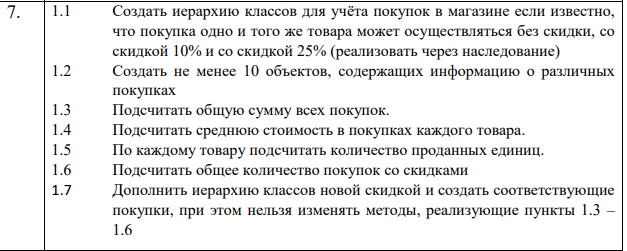


Рисунок 1 – Вариант задания

На рисунке 2 изображена структура решения.

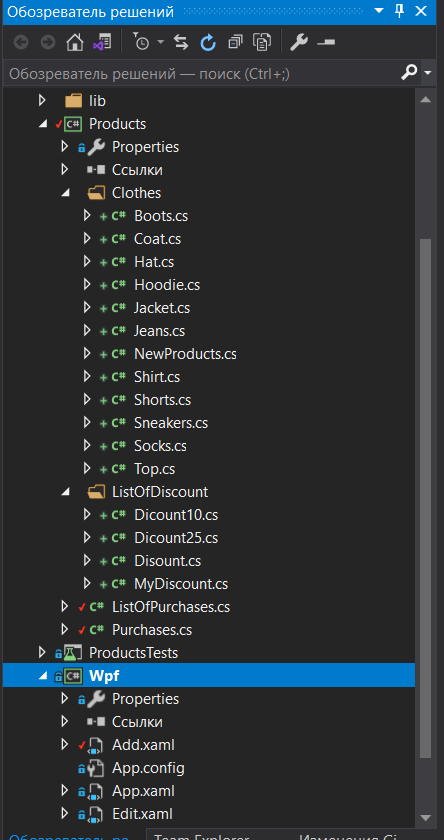


Рисунок 2 – Структура решения

На Рисунке 3 изображён вывод MainWindow.

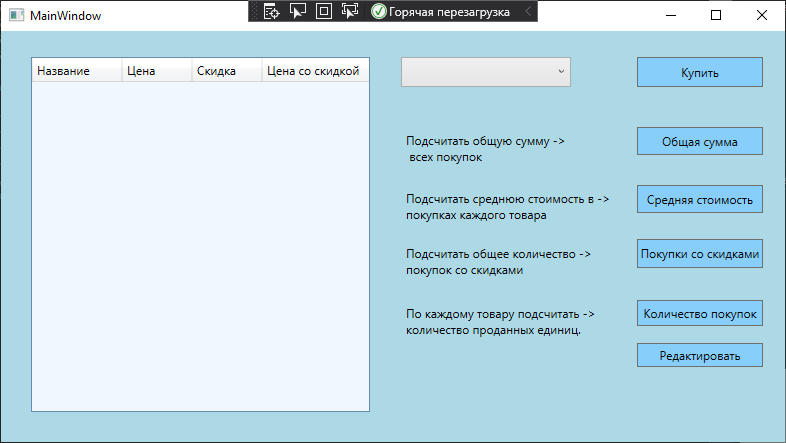


Рисунок 3 – Начало выполнения программы, вывод основной формы

На Рисунке 4 изображено выполнение пункта “Купить”, где можно выбрать товар, его цену, скидку.

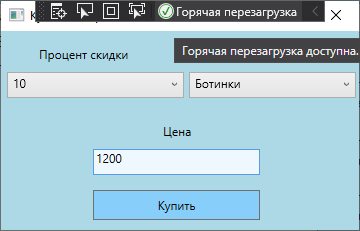


Рисунок 4 – Выполнение пункта “Купить”

На Рисунках 5–6 изображено выполнение пункта “Редактировать”, редактировать покупку.

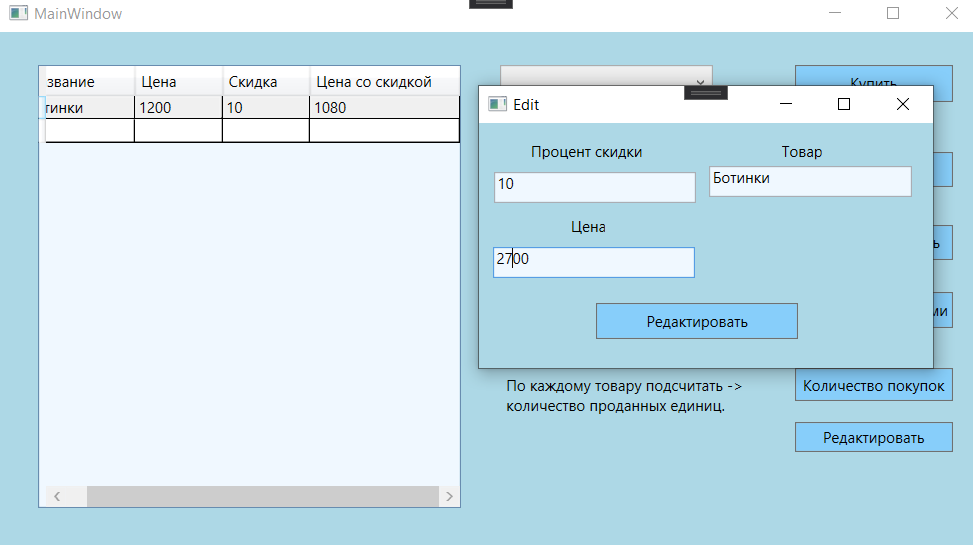


Рисунок 5 – Выполнение пункта “ Редактировать ”

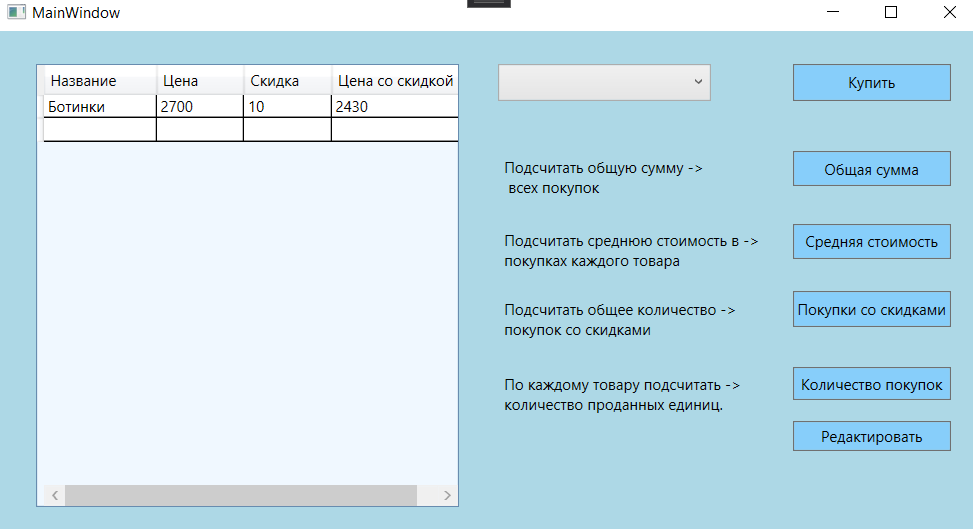


Рисунок 6 – Выполнение пункта “ Редактировать ”

На Рисунке 7 изображено выполнение пункта “Количество определённых покупок”.

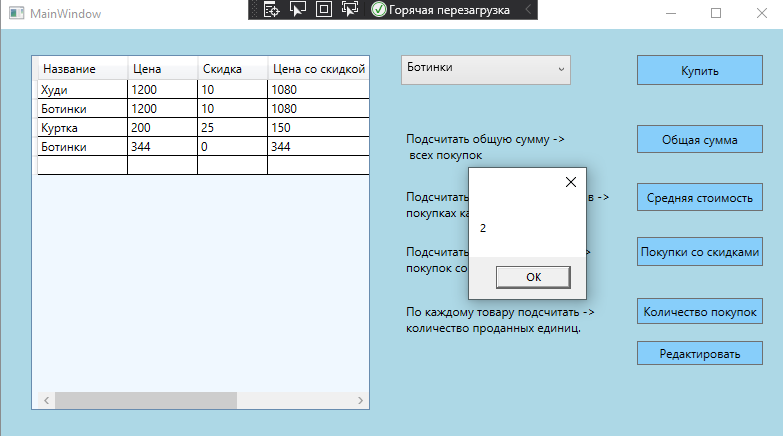


Рисунок 7 – Выполнение пункта “ Количество определённых покупок”

На Рисунке 8 изображено Выполнение пункта “Общая сумма покупок”, где можно узнать общую сумму всех покупок со скидкой и без.

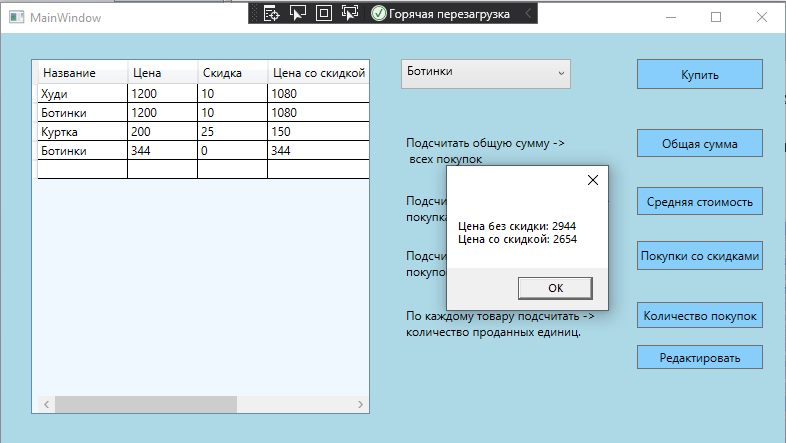


Рисунок 8 – Выполнение пункта “ Общая сумма покупок ”

На Рисунке 9 изображено выполнение пункта “Средняя стоимость в покупках каждого товара”.

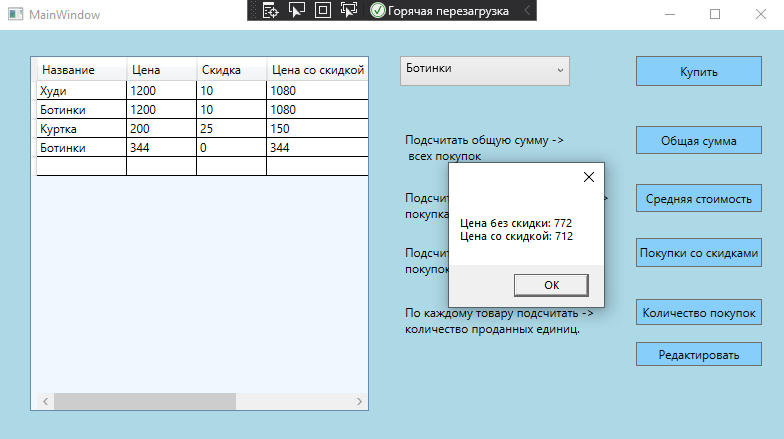


Рисунок 9 ­­­– Выполнение пункта “ Средняя стоимость в покупках каждого товара”

На Рисунке 10 изображено выполнение пункта “Покупки со скидками”, где можно узнать количество покупок со скидками.

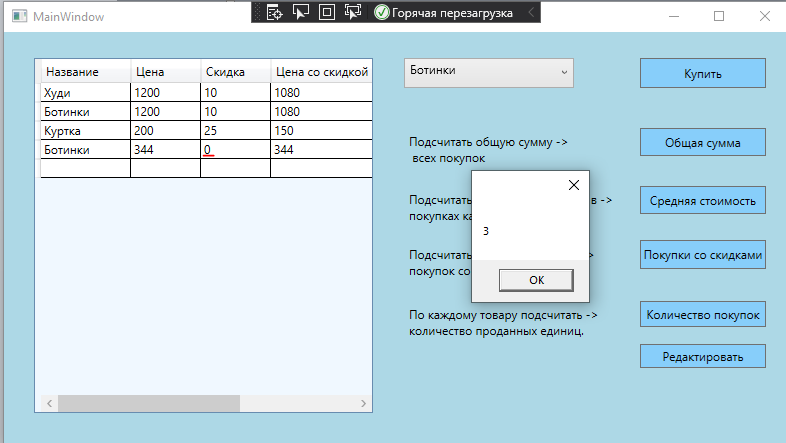


Рисунок 10 ­­­– Выполнение пункта “ Покупки со скидками ”

На Рисунке 11 изображено выполнение пункта “Количество покупок”, где можно узнать количество проданных единиц по каждому товару.

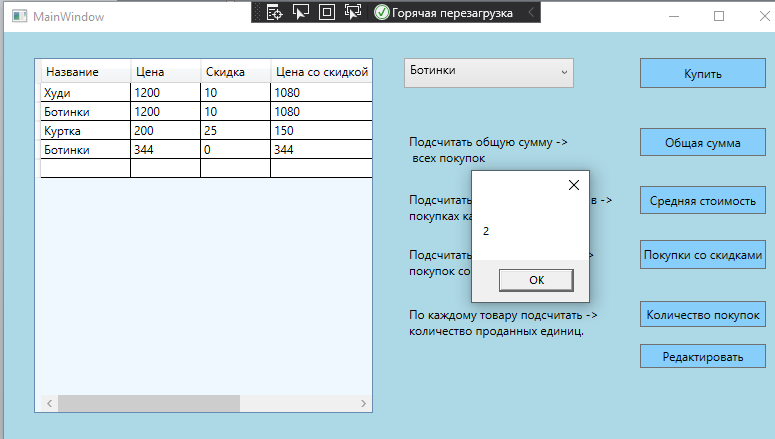


Рисунок 11 – Выполнение пункта “Количество покупок”

На Рисунке 12 изображено выполнение пункта, где требуется добавить новую скидку и покупки.

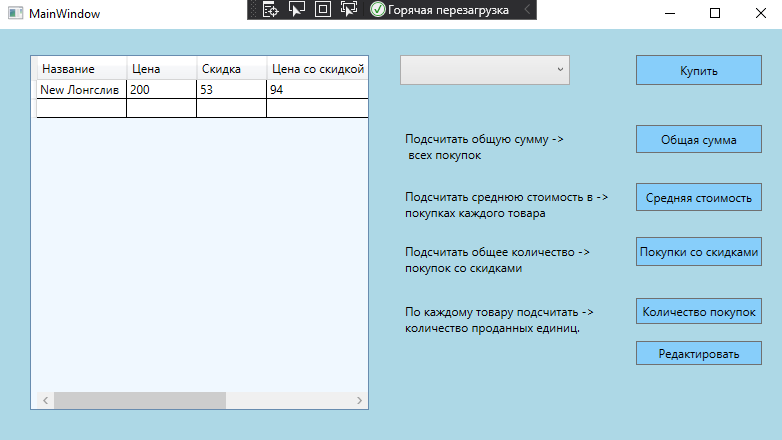


Рисунок 12 – Выполнение пункта “Количество покупок”

На Рисунках 13 – 16 изображены ситуации, когда список покупок пуст.

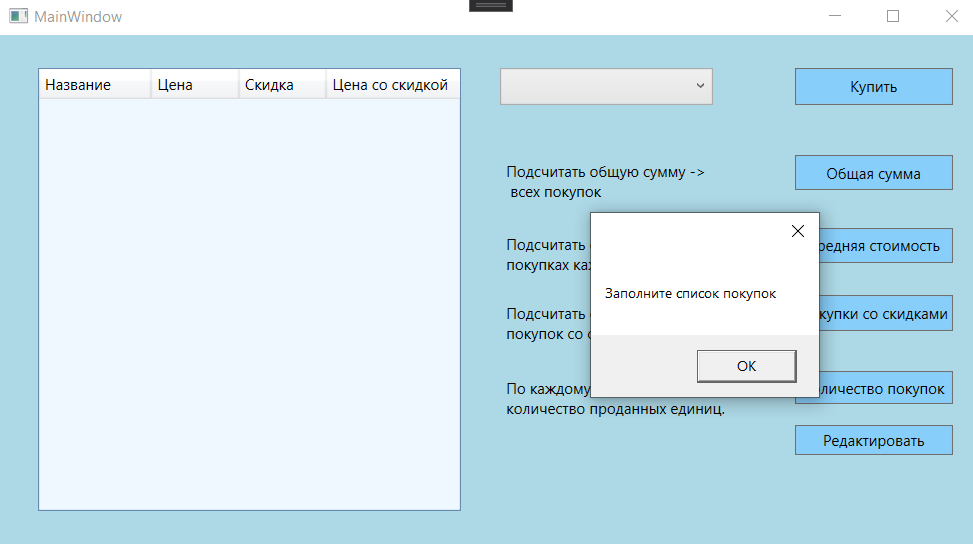


Рисунок 13 – Выполнение пункта “Общая сумма покупок”, когда нет покупок

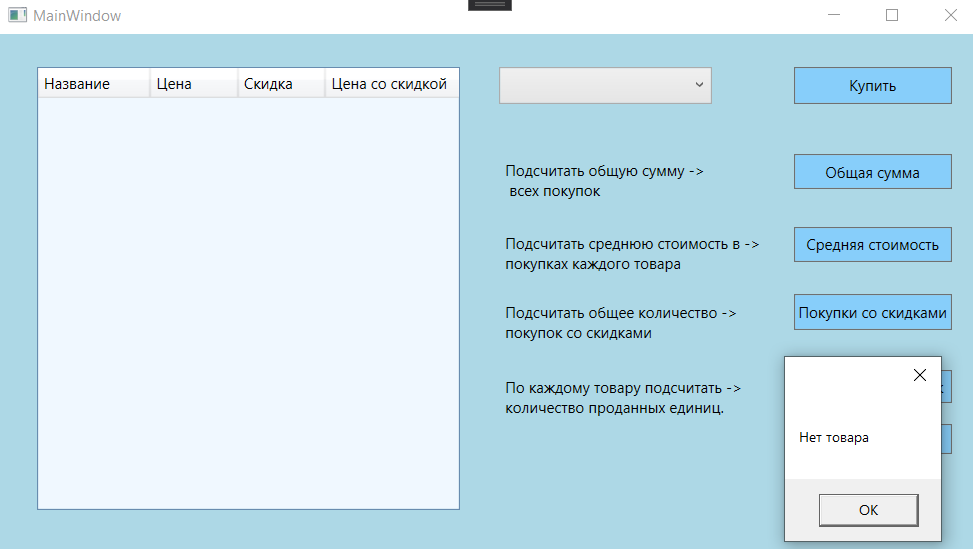


Рисунок 14 – Выполнение пункта “Стоимость товара в покупках”, когда нет покупок

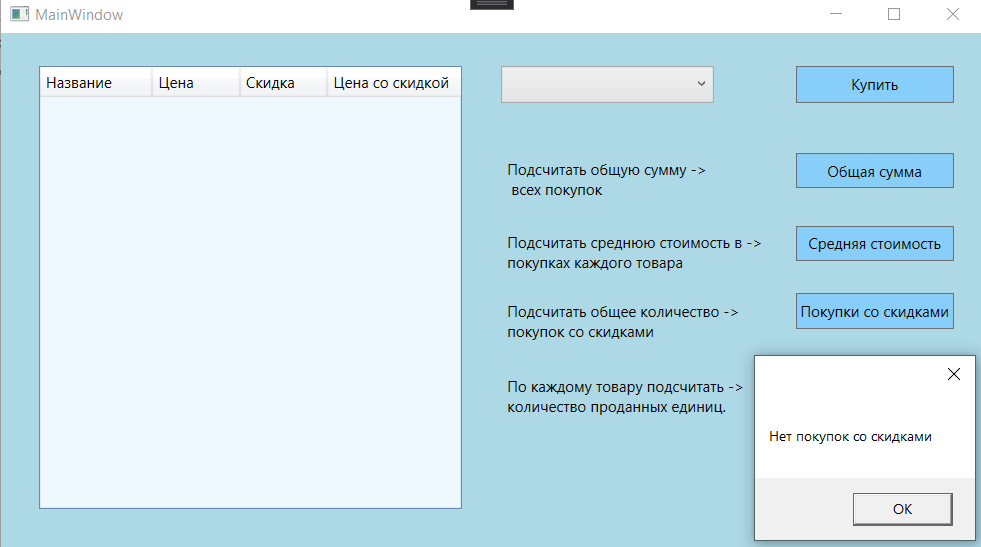


Рисунок 15 – Выполнение пункта “Общее количество покупок со скидками”, когда нет покупок

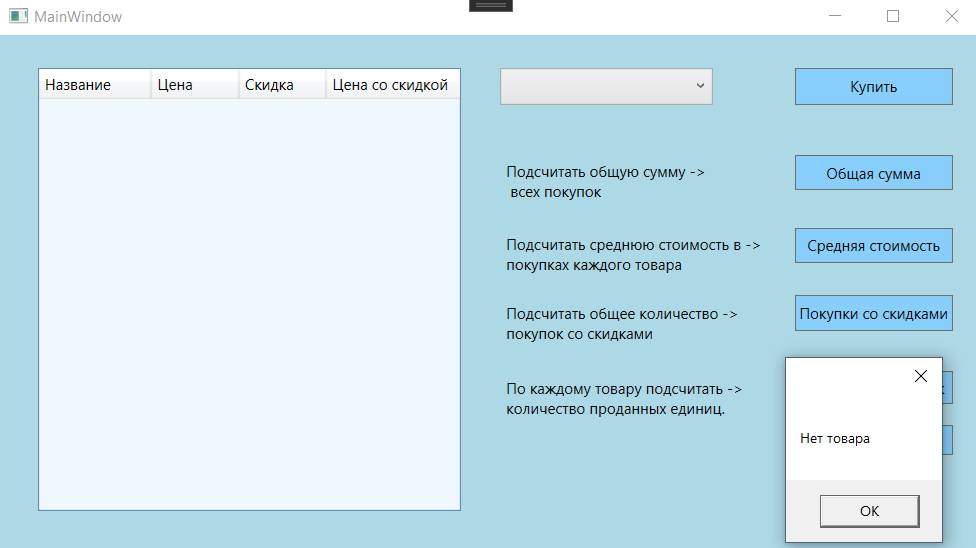


Рисунок 16 – Выполнение пункта “Количество проданных единиц”, когда нет покупок

На Рисунке 17 изображено выполнение тестов

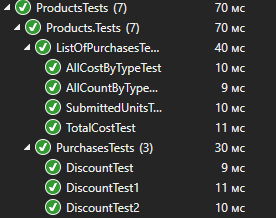


Рисунок 17 – Прохождение модульных тестов

На Рисунке 18 изображены созданные xml файлы

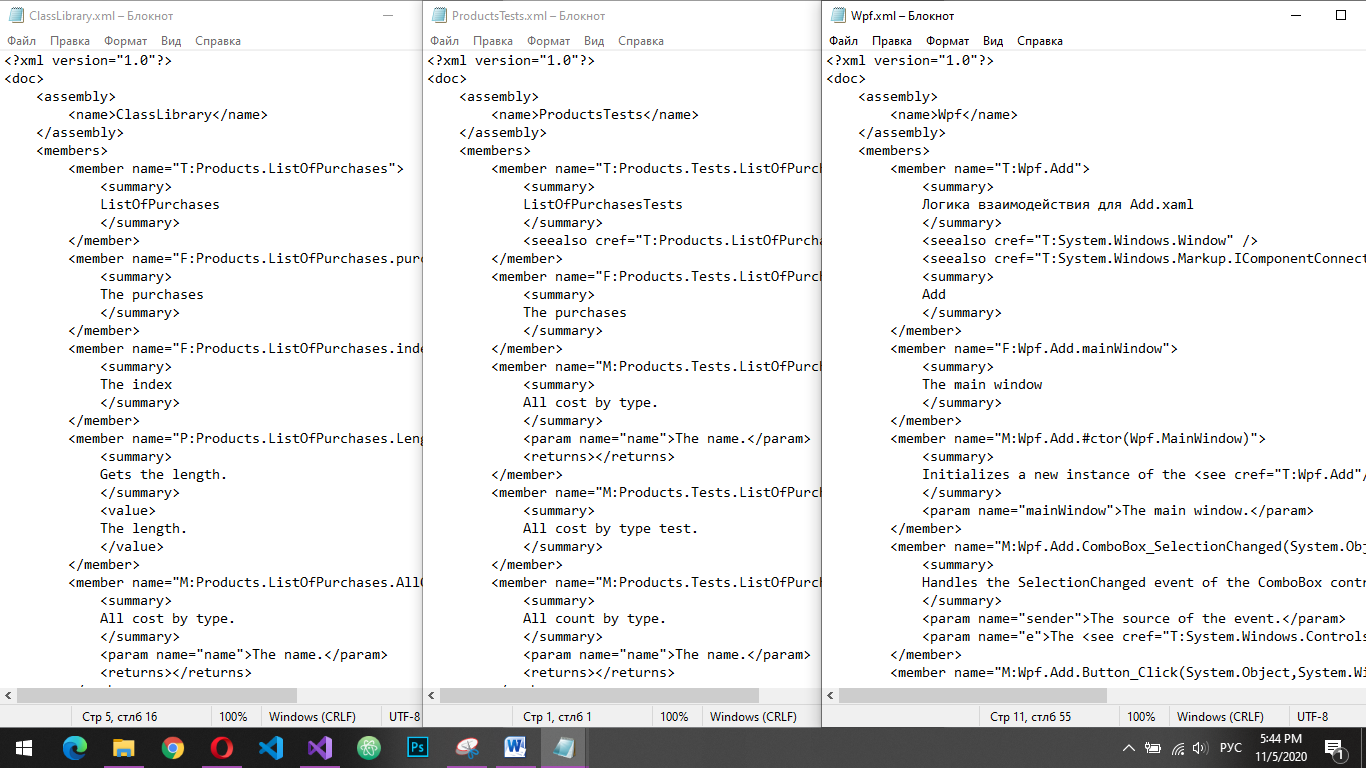


Рисунок 18 – Хml файлы

На Рисунке 19 изображены иерархии классов Покупки и Скидка.

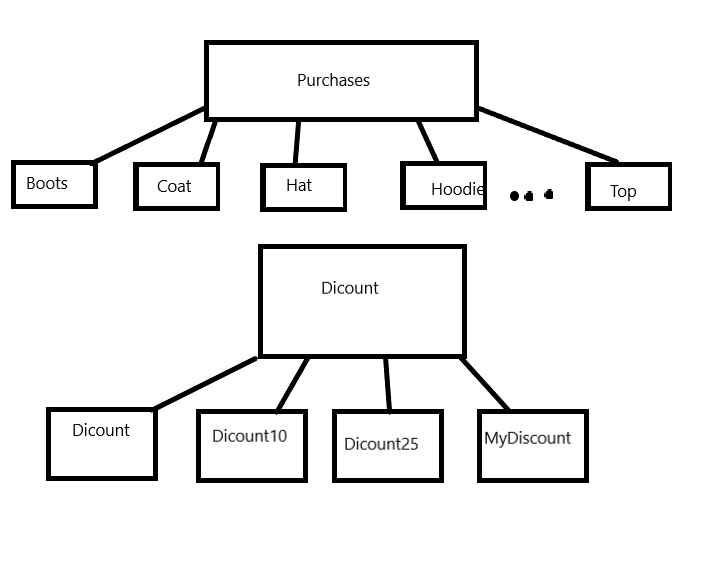


Рисунок 19 – Иерархии классов

**Вывод:** в ходе лабораторной работы были изучены основы синтаксиса объектно-ориентированного языка программирования. При реализации иерархии классов использован механизм наследования.

**ПРИЛОЖЕНИЕ А**

**Листинг программы**

Jeans

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Products

{

/// <summary>

/// Jeans

/// </summary>

/// <seealso cref="Products.Purchases" />

class Jeans : Purchases

{

/// <summary>

/// Initializes a new instance of the <see cref="Jeans"/> class.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="percent">The percent.</param>

public Jeans(string name, double cost, int percent) : base("Джинсы", cost, percent) { }

}

}

Coat

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Products

{

/// <summary>

/// Coat

/// </summary>

/// <seealso cref="Products.Purchases" />

class Coat : Purchases

{

/// <summary>

/// Initializes a new instance of the <see cref="Coat"/> class.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="percent">The percent.</param>

public Coat(string name, double cost, int percent) : base("Пальто", cost, percent) { }

}

}

Hat

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Products

{

/// <summary>

/// Hat

/// </summary>

/// <seealso cref="Products.Purchases" />

class Hat : Purchases

{

/// <summary>

/// Initializes a new instance of the <see cref="Hat"/> class.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="percent">The percent.</param>

public Hat(string name, double cost, int percent) : base("Шапка", cost, percent) { }

}

}

Hoodie

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Products

{

/// <summary>

/// Hoodie

/// </summary>

/// <seealso cref="Products.Purchases" />

class Hoodie : Purchases

{

/// <summary>

/// Initializes a new instance of the <see cref="Hoodie"/> class.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="percent">The percent.</param>

public Hoodie(string name, double cost, int percent) : base("Худи", cost, percent) { }

}

}

Jacket

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Products

{

/// <summary>

/// Jacket

/// </summary>

/// <seealso cref="Products.Purchases" />

class Jacket : Purchases

{

/// <summary>

/// Initializes a new instance of the <see cref="Jacket"/> class.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="percent">The percent.</param>

public Jacket(string name, double cost, int percent) : base("Куртка", cost, percent) { }

}

}

Boots

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Products

{

/// <summary>

/// Boots

/// </summary>

/// <seealso cref="Products.Purchases" />

class Boots : Purchases

{

/// <summary>

/// Initializes a new instance of the <see cref="Boots"/> class.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="percent">The percent.</param>

public Boots(string name, double cost, int percent) : base("Ботинки", cost, percent) { }

}

}

NewProducts

namespace Products

{

/// <summary>

/// NewProducts

/// </summary>

/// <seealso cref="Products.Purchases" />

internal class NewProducts : Purchases

{

/// <summary>

/// Initializes a new instance of the <see cref="NewProducts"/> class.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="percent">The percent.</param>

public NewProducts(string name, double cost, int percent) : base(name, cost, percent)

{

}

}

}

Shirt

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Products

{

/// <summary>

/// Shirt

/// </summary>

/// <seealso cref="Products.Purchases" />

class Shirt : Purchases

{

/// <summary>

/// Initializes a new instance of the <see cref="Shirt"/> class.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="percent">The percent.</param>

public Shirt(string name, double cost, int percent) : base("Рубашка", cost, percent) { }

}

}

Shorts

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Products

{

/// <summary>

/// Shorts

/// </summary>

/// <seealso cref="Products.Purchases" />

class Shorts : Purchases

{

/// <summary>

/// Initializes a new instance of the <see cref="Shorts"/> class.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="percent">The percent.</param>

public Shorts(string name, double cost, int percent) : base("Шорты", cost, percent) { }

}

}

Sneakers

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Products

{

/// <summary>

/// Sneakers

/// </summary>

/// <seealso cref="Products.Purchases" />

class Sneakers : Purchases

{

/// <summary>

/// Initializes a new instance of the <see cref="Sneakers"/> class.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="percent">The percent.</param>

public Sneakers(string name, double cost, int percent) : base("Кроссовки", cost, percent) { }

}

}

Socks

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Products

{

/// <summary>

/// Socks

/// </summary>

/// <seealso cref="Products.Purchases" />

class Socks : Purchases

{

/// <summary>

/// Initializes a new instance of the <see cref="Socks"/> class.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="percent">The percent.</param>

public Socks(string name, double cost, int percent) : base("Носки", cost, percent) { }

}

}

Top

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Products

{

/// <summary>

/// Top

/// </summary>

/// <seealso cref="Products.Purchases" />

class Top : Purchases

{

/// <summary>

/// Initializes a new instance of the <see cref="Top"/> class.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="percent">The percent.</param>

public Top(string name, double cost, int percent) : base("Топ", cost, percent) { }

}

}

Purchases

namespace Products

{

/// <summary>

/// Purchases

/// </summary>

public class Purchases

{

/// <summary>

/// The name

/// </summary>

private string \_name;

/// <summary>

/// The cost

/// </summary>

private double \_cost;

/// <summary>

/// The percent

/// </summary>

private int \_percent;

/// <summary>

/// Initializes a new instance of the <see cref="Purchases"/> class.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="percent">The percent.</param>

public Purchases(string name, double cost, int percent)

{

this.\_name = name;

this.\_cost = cost;

this.\_percent = percent;

}

/// <summary>

/// Initializes a new instance of the <see cref="Purchases"/> class.

/// </summary>

public Purchases()

{

}

/// <summary>

/// Gets or sets the name.

/// </summary>

/// <value>

/// The name.

/// </value>

public string Name

{

get

{

return \_name;

}

set

{

\_name = value;

}

}

/// <summary>

/// Gets or sets the cost.

/// </summary>

/// <value>

/// The cost.

/// </value>

public double Cost

{

get

{

return \_cost;

}

set

{

\_cost = value;

}

}

/// <summary>

/// Gets or sets the percent.

/// </summary>

/// <value>

/// The percent.

/// </value>

public int Percent

{

get

{

return \_percent;

}

set

{

\_percent = value;

}

}

/// <summary>

/// Calculates a discount for a product.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="percent">The percent.</param>

/// <returns></returns>

public double Discount( double cost, int percent)

{

double decimal\_percentage;

double new\_percentage;

double result\_cost;

decimal\_percentage = 100 - percent;

new\_percentage = decimal\_percentage / 100;

result\_cost = cost \* new\_percentage;

return result\_cost;

}

/// <summary>

/// Creates new cost.

/// </summary>

/// <value>

/// The new cost.

/// </value>

public double NewCost

{

get

{

double \_newcost = Discount(\_cost, \_percent);

return \_newcost;

}

}

}

}

ListOfPurchases

using Products.Discount;

using System;

using System.IO;

namespace Products

{

/// <summary>

/// ListOfPurchases

/// </summary>

public class ListOfPurchases

{

/// <summary>

/// The purchases

/// </summary>

public Purchases[] purchases = new Purchases[] { };

/// <summary>

/// The index

/// </summary>

public int index = 0;

/// <summary>

/// Gets the length.

/// </summary>

/// <value>

/// The length.

/// </value>

public int Length

{

get

{

return purchases.Length;

}

}

//модификатор virtual для переопределения методов в тестах.

/// <summary>

/// All cost by type.

/// </summary>

/// <param name="name">The name.</param>

/// <returns></returns>

public virtual string AllCostByType(string name)

{

double all\_cost = 0;

double all\_cost\_with\_discount = 0;

int count = 0;

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

{

if (purchases[i].Name == name)

{

count++;

all\_cost += purchases[i].Cost;

all\_cost\_with\_discount += purchases[i].NewCost;

}

}

if (count != 0)

{

all\_cost /= count;

all\_cost\_with\_discount /= count;

}

else all\_cost = 0;

return "Цена без скидки: " + Convert.ToString(all\_cost) + "\nЦена со скидкой: " + Convert.ToString(all\_cost\_with\_discount);

}

/// <summary>

/// All count by type.

/// </summary>

/// <param name="name">The name.</param>

/// <returns></returns>

public virtual string AllCountByType(string name)

{

int count = 0;

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

{

if (purchases[i].Name == name)

{

count++;

}

}

return Convert.ToString(count);

}

/// <summary>

/// Total cost.

/// </summary>

/// <returns></returns>

public virtual string TotalCost()

{

double all\_cost\_with\_discount = 0;

double all\_cost = 0;

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

{

all\_cost\_with\_discount += purchases[i].NewCost;

all\_cost += purchases[i].Cost;

}

return "Цена без скидки: " + Convert.ToString(all\_cost) + "\nЦена со скидкой: " + Convert.ToString(all\_cost\_with\_discount);

}

/// <summary>

/// Submitted units.

/// </summary>

/// <returns></returns>

public virtual string SubmittedUnits()

{

int count = 0;

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

{

if (purchases[i].Percent != 0)

{

count++;

}

}

return Convert.ToString(count);

}

/// <summary>

/// Add the product.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="disc">The disc.</param>

public void Add(string name, double cost, Dicount disc)

{

index = purchases.Length + 1;

Array.Resize(ref purchases, index--);

switch (name)

{

case "Ботинки":

{

purchases[index] = new Boots(name, cost, disc);

break;

}

case "Пальто":

{

purchases[index] = new Coat(name, cost, disc);

break;

}

case "Шапка":

{

purchases[index] = new Hat(name, cost, disc);

break;

}

case "Худи":

{

purchases[index] = new Hoodie(name, cost, disc);

break;

}

case "Куртка":

{

purchases[index] = new Jacket(name, cost, disc);

break;

}

case "Джинсы":

{

purchases[index] = new Jeans(name, cost, disc);

break;

}

case "Рубашка":

{

purchases[index] = new Shirt(name, cost, disc);

break;

}

case "Шорты":

{

purchases[index] = new Shorts(name, cost, disc);

break;

}

case "Кроссовки":

{

purchases[index] = new Sneakers(name, cost, disc);

break;

}

case "Носки":

{

purchases[index] = new Socks(name, cost, disc);

break;

}

case "Топ":

{

purchases[index] = new Top(name, cost, disc);

break;

}

default:

{

purchases[index] = new NewProducts(name, cost, disc);

break;

}

}

}

/// <summary>

/// Edits the specified name.

/// </summary>

/// <param name="name">The name.</param>

/// <param name="cost">The cost.</param>

/// <param name="disc">The disc.</param>

/// <param name="i">The i.</param>

public void Edit(string name, double cost, Dicount disc, int i)

{

purchases[i].Name = name;

purchases[i].Cost = cost;

purchases[i].dicount = disc;

}

public Dicount SelectedDiscount(string discount)

{

Dicount disc;

switch (int.Parse(discount))

{

case 0: disc = new Dicount(); return disc;

case 10: disc = new Dicount10(); return disc;

case 25: disc = new Dicount25(); return disc;

default: disc = new MyDiscount(int.Parse(discount)); return disc;

}

}

}

}

Edit.xaml.cs

using System;

using System.Windows;

using System.Windows.Controls;

using Products;

using Products.Discount;

using System.Text.RegularExpressions;

using System.Windows.Input;

namespace Wpf

{

/// <summary>

/// Логика взаимодействия для Edit.xaml

/// </summary>

/// <seealso cref="System.Windows.Window" />

/// <seealso cref="System.Windows.Markup.IComponentConnector" />

public partial class Edit : Window

{

/// <summary>

/// The main window

/// </summary>

private MainWindow mainWindow;

/// <summary>

/// Initializes a new instance of the <see cref="Edit"/> class.

/// </summary>

/// <param name="mainWindow">The main window.</param>

public Edit(MainWindow mainWindow)

{

this.mainWindow = mainWindow;

InitializeComponent();

Initializer();

}

private void Initializer()

{

try

{

if (mainWindow.Table.SelectedItem == null)

{

MessageBox.Show("Укажите продукт");

}

else

{

Purchases purchases = mainWindow.Table.SelectedItem as Purchases;

products1.Text = purchases.Name;

discount1.Text = Convert.ToString(purchases.Percent);

price.Text = Convert.ToString(purchases.Cost);

}

}

catch(Exception ex)

{

MessageBox.Show(Convert.ToString(ex));

}

}

/// <summary>

/// Handles the SelectionChanged event of the ComboBox control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="SelectionChangedEventArgs"/> instance containing the event data.</param>

private void ComboBox\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

}

/// <summary>

/// Handles the 1 event of the ComboBox\_SelectionChanged control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="SelectionChangedEventArgs"/> instance containing the event data.</param>

private void ComboBox\_SelectionChanged\_1(object sender, SelectionChangedEventArgs e)

{

}

/// <summary>

/// Handles the TextChanged event of the price control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="TextChangedEventArgs"/> instance containing the event data.</param>

private void price\_TextChanged(object sender, TextChangedEventArgs e)

{

}

/// <summary>

/// Handles the Click event of the Button control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="RoutedEventArgs"/> instance containing the event data.</param>

private void Button\_Click(object sender, RoutedEventArgs e)

{

try

{

if (mainWindow.Table.SelectedItem == null)

{

MessageBox.Show("Укажите продукт");

}

else if (Convert.ToDouble(price.Text) <= 0)

{

MessageBox.Show("Цена не может быть отрицательной или равной 0");

}

else if (Convert.ToDouble(discount1.Text) < 0 || Convert.ToDouble(discount1.Text) > 99)

{

MessageBox.Show("Некорректная скидка");

}

else

{

Dicount disc = mainWindow.listOfPurchases.SelectedDiscount(discount1.Text);

string product = products1.Text;

double price1 = Convert.ToDouble(price.Text);

mainWindow.listOfPurchases.Edit(product, price1, disc, mainWindow.Table.SelectedIndex);

mainWindow.Update();

Close();

}

}

catch

{

MessageBox.Show("Заполните все ячейки корректно");

}

}

}

}

Add.xaml.cs

using Products;

using Products.Discount;

using System;

using System.Windows;

using System.Windows.Controls;

namespace Wpf

{

/// <summary>

/// Логика взаимодействия для Add.xaml

/// </summary>

/// <seealso cref="System.Windows.Window" />

/// <seealso cref="System.Windows.Markup.IComponentConnector" />

public partial class Add : Window

{

/// <summary>

/// The main window

/// </summary>

private MainWindow mainWindow;

/// <summary>

/// Initializes a new instance of the <see cref="Add" /> class.

/// </summary>

/// <param name="mainWindow">The main window.</param>

public Add(MainWindow mainWindow)

{

this.mainWindow = mainWindow;

InitializeComponent();

}

/// <summary>

/// Handles the SelectionChanged event of the ComboBox control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="SelectionChangedEventArgs" /> instance containing the event data.</param>

private void ComboBox\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

}

/// <summary>

/// Handles the Click event of the Button control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="RoutedEventArgs" /> instance containing the event data.</param>

private void Button\_Click(object sender, RoutedEventArgs e)

{

try

{

if (Convert.ToDouble(price.Text) <= 0)

{

MessageBox.Show("Цена не может быть отрицательной или равной 0");

}

else if(Convert.ToDouble(discount1.Text) < 0 || Convert.ToDouble(discount1.Text) > 99)

{

MessageBox.Show("Некорректная скидка");

}

else

{

Dicount disc = mainWindow.listOfPurchases.SelectedDiscount(discount1.Text);

string product = products1.Text;

double price1 = Convert.ToDouble(price.Text);

mainWindow.listOfPurchases.Add(product, price1, disc);

mainWindow.Update();

Close();

}

}

catch

{

MessageBox.Show("Заполните все ячейки корректно");

}

}

/// <summary>

/// Handles the 1 event of the ComboBox\_SelectionChanged control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="SelectionChangedEventArgs" /> instance containing the event data.</param>

private void ComboBox\_SelectionChanged\_1(object sender, SelectionChangedEventArgs e)

{

}

/// <summary>

/// Handles the TextChanged event of the price control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="TextChangedEventArgs" /> instance containing the event data.</param>

private void price\_TextChanged(object sender, TextChangedEventArgs e)

{

}

}

}

MainWindow.xaml.cs

using Products;

using System;

using System.Collections.Generic;

using System.Windows;

using System.Windows.Controls;

namespace Wpf

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

/// <seealso cref="System.Windows.Window" />

/// <seealso cref="System.Windows.Markup.IComponentConnector" />

public partial class MainWindow : Window

{

/// <summary>

/// The array products

/// </summary>

public ListOfPurchases listOfPurchases;

/// <summary>

/// Initializes a new instance of the <see cref="MainWindow" /> class.

/// </summary>

public MainWindow()

{

InitializeComponent();

listOfPurchases = new ListOfPurchases();

}

/// <summary>

/// Handles the SelectionChanged event of the DataGrid control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="SelectionChangedEventArgs" /> instance containing the event data.</param>

private void DataGrid\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

}

/// <summary>

/// Handles the SelectionChanged event of the ComboBox control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="SelectionChangedEventArgs" /> instance containing the event data.</param>

private void ComboBox\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

}

/// <summary>

/// Handles the Click event of the Button control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="RoutedEventArgs" /> instance containing the event data.</param>

private void Button\_Click(object sender, RoutedEventArgs e)

{

Add add = new Add(this);

add.Show();

}

/// <summary>

/// Handles the 1 event of the Button\_Click control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="RoutedEventArgs" /> instance containing the event data.</param>

private void Button\_Click\_1(object sender, RoutedEventArgs e)

{

try

{

if (listOfPurchases.TotalCost() == "Цена без скидки: 0\nЦена со скидкой: 0")

{

MessageBox.Show("Заполните список покупок");

}

else MessageBox.Show(listOfPurchases.TotalCost());

}

catch (Exception ex)

{

MessageBox.Show(Convert.ToString(ex));

}

}

/// <summary>

/// Handles the 2 event of the Button\_Click control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="RoutedEventArgs" /> instance containing the event data.</param>

private void Button\_Click\_2(object sender, RoutedEventArgs e)

{

try

{

if (listOfPurchases.AllCostByType(List.Text) == "Цена без скидки: 0\nЦена со скидкой: 0")

{

MessageBox.Show("Нет товара");

}

else MessageBox.Show(listOfPurchases.AllCostByType(List.Text));

}

catch (Exception ex)

{

MessageBox.Show(Convert.ToString(ex));

}

}

/// <summary>

/// Handles the 3 event of the Button\_Click control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="RoutedEventArgs" /> instance containing the event data.</param>

private void Button\_Click\_3(object sender, RoutedEventArgs e)

{

try

{

if (listOfPurchases.SubmittedUnits() == "0")

{

MessageBox.Show("Нет покупок со скидками");

}

else MessageBox.Show(listOfPurchases.SubmittedUnits());

}

catch (Exception ex)

{

MessageBox.Show(Convert.ToString(ex));

}

}

/// <summary>

/// Updates this instance.

/// </summary>

public void Update()

{

List<Purchases> list = new List<Purchases>();

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

{

list.Add(new Purchases()

{

Name = listOfPurchases.purchases[i].Name,

Cost = listOfPurchases.purchases[i].Cost,

dicount = listOfPurchases.purchases[i].dicount,

});

}

Table.Items.Refresh();

Table.AutoGenerateColumns= false;

Table.ItemsSource = list;

}

/// <summary>

/// Handles the 4 event of the Button\_Click control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="RoutedEventArgs" /> instance containing the event data.</param>

private void Button\_Click\_4(object sender, RoutedEventArgs e)

{

try

{

if (listOfPurchases.AllCountByType(List.Text) == "0")

{

MessageBox.Show("Нет товара");

}

else MessageBox.Show(listOfPurchases.AllCountByType(List.Text));

}

catch (Exception ex)

{

MessageBox.Show(Convert.ToString(ex));

}

}

/// <summary>

/// Handles the 5 event of the Button\_Click control.

/// </summary>

/// <param name="sender">The source of the event.</param>

/// <param name="e">The <see cref="RoutedEventArgs"/> instance containing the event data.</param>

private void Button\_Click\_5(object sender, RoutedEventArgs e)

{

Edit edit = new Edit(this);

edit.Show();

}

}

}

Dicount10

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Products.Dicount

{

/// <summary>

/// Dicount10

/// </summary>

/// <seealso cref="Products.Dicount" />

/// <seealso cref="Products.Discount" />

public class Dicount10 : Dicount

{

/// <summary>

/// Initializes a new instance of the <see cref="Discount10" /> class.

/// </summary>

public Dicount10()

{

SetDicount(10);

}

}

}

Dicount25

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Products.Dicount

{

/// <summary>

/// Dicount25

/// </summary>

/// <seealso cref="Products.Dicount" />

public class Dicount25 : Dicount

{

/// <summary>

/// Initializes a new instance of the <see cref="Dicount25"/> class.

/// </summary>

public Dicount25()

{

SetDicount(25);

}

}

}

Dicount

namespace Products.Dicount

{

/// <summary>

/// Dicount

/// </summary>

public class Dicount

{

/// <summary>

/// Gets the percent.

/// </summary>

/// <value>

/// The percent.

/// </value>

public int Percent { get; private set; }

/// <summary>

/// Initializes a new instance of the <see cref="Dicount" /> class.

/// </summary>

public Dicount()

{

Percent = 0;

}

/// <summary>

/// Sets the dicount.

/// </summary>

/// <param name="i">The i.</param>

public void SetDicount(int i)

{

Percent = i;

}

}

}

MyDiscount

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Products.Discount

{

/// <summary>

/// MyDiscount

/// </summary>

/// <seealso cref="Products.Dicount" />

public class MyDiscount : Dicount

{

/// <summary>

/// Initializes a new instance of the <see cref="MyDiscount"/> class.

/// </summary>

/// <param name="i">The i.</param>

public MyDiscount(int i)

{

SetDicount(i);

}

}

}