Разработка консольного приложения с элементами ООП

Интернет магазин продуктов

Выполнил студент группы ПКС-25:

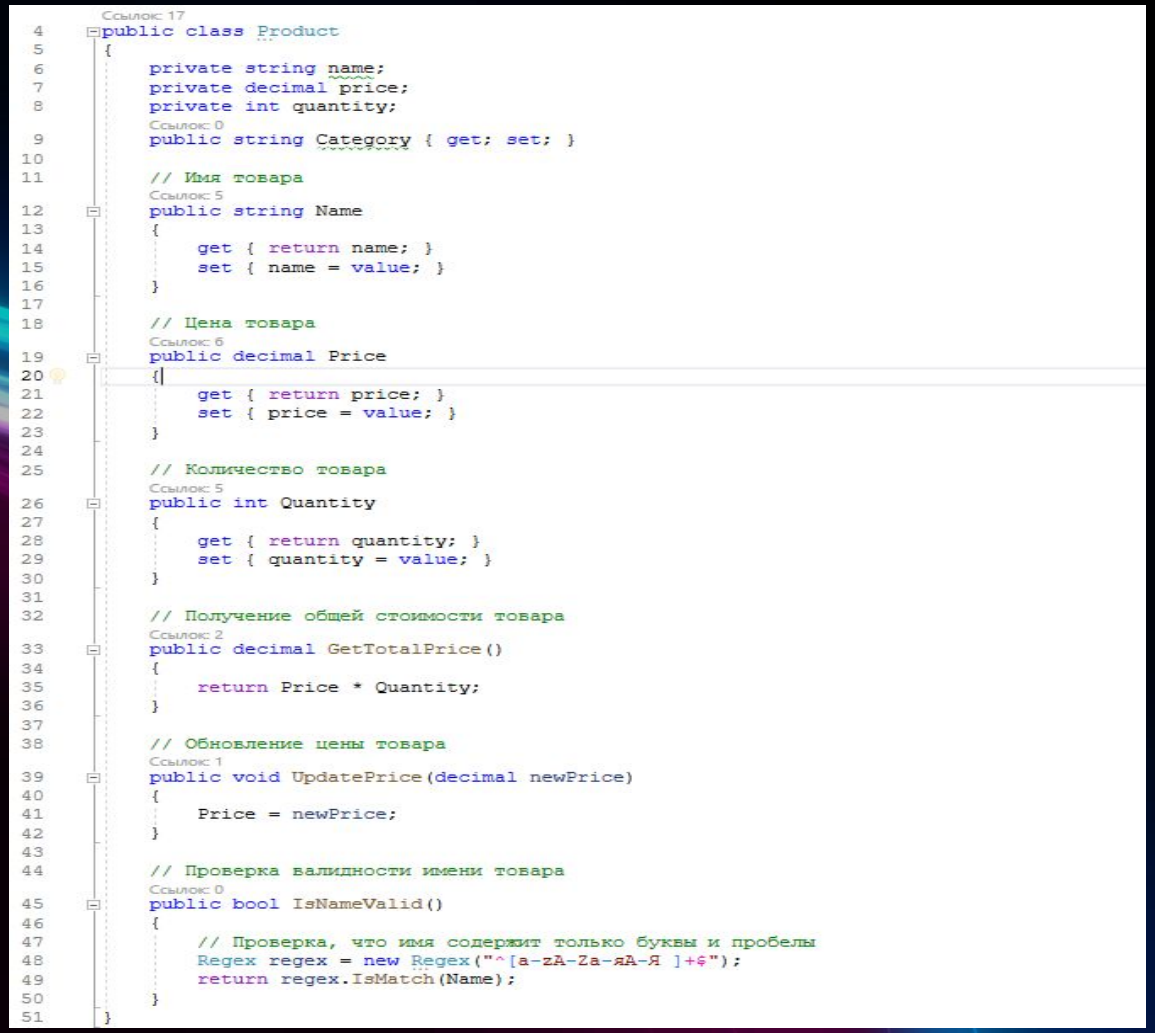
Гусев Александр

Классы

- Продукты
- Корзина
- Пользователь

Класс Продукты

- Класс Product представляет собой название товара, имеющую количество и цену за одну единицу товара. Конструктор, принимающий название, количество и цену в качестве параметров.



```
Ссылок: 17
public class Product
{
    private string name;
    private decimal price;
    private int quantity;
    Ссылок: 0
    public string Category { get; set; }

    // Имя товара
    Ссылок: 5
    public string Name
    {
        get { return name; }
        set { name = value; }
    }

    // Цена товара
    Ссылок: 6
    public decimal Price
    {
        get { return price; }
        set { price = value; }
    }

    // Количество товара
    Ссылок: 5
    public int Quantity
    {
        get { return quantity; }
        set { quantity = value; }
    }

    // Получение общей стоимости товара
    Ссылок: 2
    public decimal GetTotalPrice()
    {
        return Price * Quantity;
    }

    // Обновление цены товара
    Ссылок: 1
    public void UpdatePrice(decimal newPrice)
    {
        Price = newPrice;
    }

    // Проверка валидности имени товара
    Ссылок: 0
    public bool IsNameValid()
    {
        // Проверка, что имя содержит только буквы и пробелы
        Regex regex = new Regex("^[a-zA-Za-яА-Я ]+$");
        return regex.IsMatch(Name);
    }
}
```

Класс Корзина

Класс Cunt расширяет функциональность списка продуктов, предоставляя методы для работы с корзиной товаров.

Класс "Корзина" унаследован от класса Product и предоставляет дополнительные функции для управления содержимым корзины.

```
public class Cart : List<Product>
{
    // Метод для получения общей стоимости корзины
    Ссылок: 1
    public decimal GetTotalPrice()
    {
        decimal total = 0;
        foreach (var product in this)
        {
            total += product.GetTotalPrice();
        }
        return total;
    }

    // Метод для просмотра содержимого корзины
    Ссылок: 1
    public void ViewCart()
    {
        Console.WriteLine("Товары в корзине:");

        foreach (var product in this)
        {
            decimal totalPrice = product.GetTotalPrice();
            Console.WriteLine($"Название: {product.Name}, Цена: {product.Price}, Количество: {product.Quantity}, Общая цена: {totalPrice}");
        }

        decimal totalCartPrice = GetTotalPrice();
        Console.WriteLine($"ИТОГО: {totalCartPrice}");
    }
}
```

Класс Пользователь

Класс "Пользователь" (User.cs) представляет собой модель пользователя с основными свойствами, такими как имя и адрес электронной почты.

Класс "Пользователь" содержит свойства Name и Email, которые предоставляют доступ к имени и адресу электронной почты пользователя соответственно.

```
1   public class User
2   {
3       // Свойство для имени пользователя
        Ссылок: 3
4       public string Name { get; set; }
5
6       // Свойство для адреса электронной почты пользователя
        Ссылок: 4
7       public string Email { get; set; }
8
9       // Пустой конструктор класса User
        Ссылок: 1
10      public User() { }
11
12      // Конструктор класса User с параметрами name и email
        Ссылок: 0
13      public User(string name, string email)
14      {
15          Name = name;
16          Email = email;
17      }
18      //перегрузка метода, вызывает 1 функцию с разными действиями
        Ссылок: 0
19      public User(string email)
20      {
21          Email = email;
22      }
23  }
24
```

Методы класса Продукты

- Метод для добавления товара:

```
// Метод для добавления товара
Ссылок: 1
static void AddProduct(DataManager<Product> productManager, Cart cart)
{
    Console.Write("Введите название товара: ");
    string name = Console.ReadLine();

    Console.Write("Введите цену товара: ");
    decimal price = decimal.Parse(Console.ReadLine());

    Console.Write("Введите количество товара: ");
    int quantity = int.Parse(Console.ReadLine());

    Product product = new Product { Name = name, Price = price, Quantity = quantity };
    productManager.AddObject(product);

    Console.WriteLine("Товар успешно добавлен.");
```

Методы класса Продукты

- Метод для удаления товара:

```
90      // Метод для удаления товара
        Ссылок: 1
91      static void RemoveProduct(DataManager<Product> productManager)
92      {
93          Console.Write("Введите индекс товара для удаления: ");
94          string input = Console.ReadLine();
95
96          if (int.TryParse(input, out int index))
97          {
98              List<Product> products = productManager.GetObjects();
99              if (index >= 0 && index < products.Count)
100             {
101                 Product product = products[index];
102                 productManager.RemoveObject(product);
103
104                 Console.WriteLine("Товар успешно удален.");
105             }
106             else
107             {
108                 Console.WriteLine("Некорректный индекс.");
109             }
110         }
111         else
112         {
113             Console.WriteLine("Некорректный ввод. Введите целое число в качестве индекса.");
114         }
115     }
```

Методы класса Продукты

- Метод для просмотра всех товаров:

```
117     // Метод для просмотра всех товаров
        Ссылок: 1
118     static void ViewProducts(DataManager<Product> productManager)
119     {
120         Console.WriteLine("Товары:");
121
122         foreach (Product product in productManager.GetObjects())
123         {
124             Console.WriteLine($"Название: {product.Name}, Цена: {product.Price}, Количество: {product.Quantity}");
125         }
126     }
127
```

Методы класса Продукты

- Метод для обновления цены товара:

```
128     // Метод для обновления цены товара
        Ссылок: 1
129     static void UpdateProductPrice(DataManager<Product> productManager)
130     {
131         Console.Write("Введите индекс товара для обновления цены: ");
132         string input = Console.ReadLine();
133
134         if (int.TryParse(input, out int index))
135         {
136             List<Product> products = productManager.GetObjects();
137             if (index >= 0 && index < products.Count)
138             {
139                 Product product = products[index];
140
141                 Console.Write("Введите новую цену товара: ");
142                 string priceInput = Console.ReadLine();
143                 if (decimal.TryParse(priceInput, out decimal newPrice))
144                 {
145                     product.UpdatePrice(newPrice);
146                     Console.WriteLine("Цена товара успешно обновлена.");
147                 }
148                 else
149                 {
150                     Console.WriteLine("Некорректный ввод. Введите число в качестве цены товара.");
151                 }
152             }
153             else
154             {
155                 Console.WriteLine("Некорректный индекс.");
156             }
157         }
158         else
159         {
160             Console.WriteLine("Некорректный ввод. Введите целое число в качестве индекса.");
161         }
162     }
163
```

Методы класса Корзина

- Метод для получения общей стоимости корзины:

```
6       // Метод для получения общей стоимости корзины
        Ссылок: 1
7       public decimal GetTotalPrice()
8       {
9           decimal total = 0;
10          foreach (var product in this)
11          {
12              total += product.GetTotalPrice();
13          }
14          return total;
15      }
16
```

Методы класса Корзина

- Метод для просмотра содержимого корзины:

```
17      // Метод для просмотра содержимого корзины
        Ссылок: 1
18      public void ViewCart()
19      {
20          Console.WriteLine("Товары в корзине:");
21
22          foreach (var product in this)
23          {
24              decimal totalPrice = product.GetTotalPrice();
25              Console.WriteLine($"Название: {product.Name}, Цена: {product.Price}, Количество: {product.Quantity}, Общая цена: {totalPrice}");
26          }
27
28          decimal totalCartPrice = GetTotalPrice();
29          Console.WriteLine($"ИТОГО: {totalCartPrice}");
30      }
31  }
```

Методы класса Пользователь

- Метод для добавления пользователя:

```
166     // Метод для добавления пользователя
        Ссылок: 1
167     static void AddUser(DataManager<User> userManager)
168     {
169         Console.Write("Введите имя пользователя: ");
170         string name = Console.ReadLine();
171
172         Console.Write("Введите адрес электронной почты пользователя: ");
173         string email = Console.ReadLine();
174
175         User user = new User { Name = name, Email = email };
176         userManager.AddObject(user);
177
178         Console.WriteLine("Пользователь успешно добавлен.");
179     }
180
```

Методы класса Пользователь

- Метод для удаления пользователя

```
// Метод для удаления пользователя
Ссылок: 1
static void RemoveUser(DataManager<User> userManager)
{
    Console.Write("Введите индекс пользователя для удаления: ");
    int index = int.Parse(Console.ReadLine());

    List<User> users = userManager.GetObjects();
    if (index >= 0 && index < users.Count)
    {
        User user = users[index];
        userManager.RemoveObject(user);

        Console.WriteLine("Пользователь успешно удален.");
    }
    else
    {
        Console.WriteLine("Некорректный индекс.");
    }
}
```

Методы класса Пользователь

- Метод для просмотра всех пользователей

```
201     // Метод для просмотра всех пользователей
        Ссылок: 1
202     static void ViewUsers(DataManager<User> userManager)
203     {
204         Console.WriteLine("Пользователи:");
205
206         foreach (User user in userManager.GetObjects())
207         {
208             Console.WriteLine($"Имя: {user.Name}, Email: {user.Email}");
209         }
210     }
```

Перегрузка для класса Пользователь

- Данная перегрузка перегружает метод, добавляя возможность пользователю ввести только имя, и код ошибку не выдаст:

```
12      // Конструктор класса User с параметрами name и email
        Ссылок: 0
13      public User(string name, string email)
14      {
15          Name = name;
16          Email = email;
17      }
18      //перегрузка метода, вызывает 1 функцию с разными действиями
        Ссылок: 0
19      public User(string email)
20      {
21          Email = email;
22      }
23  }
```

Создание страницы выбора

```
6       static void Main(string[] args)
7       {
8           DataManager<Product> productManager = new DataManager<Product>();
9           DataManager<User> userManager = new DataManager<User>();
10          Cart cart = new Cart();
11
12          while (true)
13          {
14              Console.WriteLine("                                    Добро пожаловать в интернет магазин продуктов");
15              Console.WriteLine();
16              Console.WriteLine("1. Добавить товар");
17              Console.WriteLine("2. Удалить товар");
18              Console.WriteLine("3. Просмотреть все товары");
19              Console.WriteLine("4. Обновить цену товара");
20              Console.WriteLine("5. Добавить пользователя");
21              Console.WriteLine("6. Удалить пользователя");
22              Console.WriteLine("7. Просмотреть всех пользователей");
23              Console.WriteLine("8. Просмотреть корзину");
24              Console.WriteLine("9. Выйти");
25              Console.WriteLine();
26              Console.Write("Введите ваш выбор: ");
27              string choice = Console.ReadLine();
28
29              Console.WriteLine();
```

Обработка ошибок

```
29          Console.WriteLine();
30
31          switch (choice)
32          {
33              case "1":
34                  AddProduct(productManager, cart);
35                  break;
36              case "2":
37                  RemoveProduct(productManager);
38                  break;
39              case "3":
40                  ViewProducts(productManager);
41                  break;
42              case "4":
43                  UpdateProductPrice(productManager);
44                  break;
45              case "5":
46                  AddUser(userManager);
47                  break;
48              case "6":
49                  RemoveUser(userManager);
50                  break;
51              case "7":
52                  ViewUsers(userManager);
53                  break;
54              case "8":
55                  cart.ViewCart();
56                  break;
57              case "9":
58                  return;
59              default:
60                  Console.WriteLine("Некорректный выбор. Пожалуйста, попробуйте снова.");
61                  break;
62          }
63
64          Console.WriteLine();
65      }
```

```
// Метод для удаления товара
Ссылок: 1
static void RemoveProduct(DataManager<Product> productManager)
{
    Console.Write("Введите индекс товара для удаления: ");
    string input = Console.ReadLine();

    if (int.TryParse(input, out int index))
    {
        List<Product> products = productManager.GetObjects();
        if (index >= 0 && index < products.Count)
        {
            Product product = products[index];
            productManager.RemoveObject(product);

            Console.WriteLine("Товар успешно удален.");
        }
        else
        {
            Console.WriteLine("Некорректный индекс.");
        }
    }
    else
    {
        Console.WriteLine("Некорректный ввод. Введите целое число в качестве индекса.");
    }
}
```

Обработка ошибок

```
128     // Метод для обновления цены товара
        Ссылок: 1
129     static void UpdateProductPrice(DataManager<Product> productManager)
130     {
131         Console.Write("Введите индекс товара для обновления цены: ");
132         string input = Console.ReadLine();
133
134         if (int.TryParse(input, out int index))
135         {
136             List<Product> products = productManager.GetObjects();
137             if (index >= 0 && index < products.Count)
138             {
139                 Product product = products[index];
140
141                 Console.Write("Введите новую цену товара: ");
142                 string priceInput = Console.ReadLine();
143                 if (decimal.TryParse(priceInput, out decimal newPrice))
144                 {
145                     product.UpdatePrice(newPrice);
146                     Console.WriteLine("Цена товара успешно обновлена.");
147                 }
148                 else
149                 {
150                     Console.WriteLine("Некорректный ввод. Введите число в качестве цены товара.");
151                 }
152             }
153             else
154             {
155                 Console.WriteLine("Некорректный индекс.");
156             }
157         }
158         else
159         {
160             Console.WriteLine("Некорректный ввод. Введите целое число в качестве индекса.");
161         }
162     }
```

```
179     // Метод для удаления пользователя
        Ссылок: 1
180     static void RemoveUser(DataManager<User> userManager)
181     {
182         Console.Write("Введите индекс пользователя для удаления: ");
183         int index = int.Parse(Console.ReadLine());
184
185         List<User> users = userManager.GetObjects();
186         if (index >= 0 && index < users.Count)
187         {
188             User user = users[index];
189             userManager.RemoveObject(user);
190
191             Console.WriteLine("Пользователь успешно удален.");
192         }
193         else
194         {
195             Console.WriteLine("Некорректный индекс.");
196         }
197     }
```

Спасибо за внимание!