Учреждение образования Федерации профсоюзов Беларуси

«Международный университет «МИТСО»

Рег. № \_\_\_\_\_\_\_\_\_\_ Кафедра информационных систем и технологий

Дата \_\_\_\_\_\_\_\_\_2020

**КУРСОВАЯ РАБОТА**

на тему \_ Учет клиентов музыкальной студии

по дисциплине \_Название\_\_Основы конструирования программ\_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
| Основные замечания: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Отметка о допуске курсовой работы к  защите:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_\_  Подпись научного руководителя:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Студент: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись)  (Ф.И.О. – полностью)  Курс \_\_1\_\_, группа \_\_\_2020\_\_\_\_  Факультет \_\_экономический\_\_\_  Специальность \_\_Информационные системы и технологии\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Научный руководитель:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (должность, ученая степень, ученое звание)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Ф.И.О. – полностью) |

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc70955691)

[1.ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 4](#_Toc70955692)

[2.ТРЕБОВАНИЕ К ПРОГРАММЕ 5](#_Toc70955693)

[3.КОНСТРУИРОВАНИЕ ПРОГРАММЫ 6](#_Toc70955694)

[4.ОПИСАНИЕ ИНТЕРФЕЙСА 7](#_Toc70955695)

[5.ВЫБОР СПОСОБА ОРГАНИЗАЦИИ ДАННЫХ 9](#_Toc70955696)

[6.РАЗРАБОТКА ПЕРЕЧНЯ ПОЛЬЗОВАТЕЛЬСКИХ МЕТОДОВ ПРОГРАММЫ 12](#_Toc70955697)

[7.ПРОГРАММНАЯ ЧАСТЬ 14](#_Toc70955698)

[ЗАКЛЮЧЕНИЕ 17](#_Toc70955699)

[СПИСОК ЛИТЕРАТУРЫ 18](#_Toc70955700)

[ПРИЛОЖЕНИЕ 19](#_Toc70955701)

# **ВВЕДЕНИЕ**

Была поставлена задача разработать программный модуль «Учет ассортимента коктейли в кафе», предназначенный для использования кафе, а именно, барменом. В базе содержатся сведения о кителях (Название, объем, состав).

Программа создана для систематизации коктейлей в кафе, быстрого просмотра/изменения и добавления новых коктейлей.

Цель разработки обеспечить работы кафе быстро и максимально эффективно, удовлетворять запросы кафе, а также должен быть возможность пополнять базу, для оптимизации процессов кафе. И неотъемлемой частью данной программы должна высокая оптимизация, чтобы пользователь мог запустить даже на самом слабом устройстве.

Задачи данного программного продукта: быстро и гибко редактироваться со стороны владельцев кафе. С точки зрения использования пользователя программа должна иметь красивый, понятный и интуитивный интерфейс чтобы было владельцу приятно пользоваться программой. Так же требуется предусмотреть возможность добавления новых функций в программе.

Для разработки программы был использован текстовый файл и язык программирования C# разработанный в среде Visual Studio компаний Microsoft.

Visual Studio-линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой Windows Forms, а также веб-сайты, веб-приложения, веб-службы, как в родном, так и в управляемом кодах для всех платформ, поддерживающих Windows, Windows Mobile, Windows Ce, .Net Framework, Xbox, Windows Phone .Net Compact Framework и Silverling.

# **1.ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ**

Была поставлена задача разработать программный модуль «Учет ассортимента коктейли в кафе», предназначенный для использования кафе, а именно, барменом. В базе содержатся сведения о кителях (Название, объем, состав).

Места общественного питания является очень распространённым видом деятельности, поэтому разрабатываемая программа должна выдержать весьма серьёзную конкуренцию.

Функции программного продукта: содержатся сведения о коктейлях (Название, объем, состав) должны иметь понятные данные и возможность обрабатывать их (добавлять/удалять, просматривать, изменять).

Данный программный продукт выгодно отличается от других подобных тем что пользуясь им, пользователь может не только просматривать коктелями, но и производить различные взаимодействия с коктейлями.

# **2.ТРЕБОВАНИЕ К ПРОГРАММЕ**

Исходя из задач, описанных выше главными требованиями к языку, на котором будет написана программы — это тонкая настройка и возможность использованием минимум вычислительных ресурсов добиваться максимум результата.

Для выполнения данных задач очень подходит такой язык программирования как с#. Так как его из-за того, что он является весьма низкоуровневым языком программирования, а значит он больше других приближен к машинному коду, из-за этого с# дает возможность оптимизировать проект так что даже на очень старых устройствах он будет работать.

Среда разработки Microsoft Visual Studio. Вид приложения – консольное. Программа построена на базе объектно-ориентированного программирования. Способ организации данных поля классов. Способ хранения данных – текстовый фалй/xml. Каждая логическая завершенная данных подзадача программы реализована в виде методов. К защите курсовой предоставляется: консольное приложение и пояснительная записка

# **3.КОНСТРУИРОВАНИЕ ПРОГРАММЫ**

В текущей главе представлено описание алгоритма (рис 3), который имеет начало алгоритма и конец, внутри алгоритма имеются циклы и иные операций.

Первым пунктом идёт выбор операции, за счет введение цифры от 1 до 5 и последующей за вводом проверки.

Следующий шаг — это работа непосредственно с программой в зависимости от выбранных данных. Добавление данных, вывод данных,удаление коктейлей, сортировка, изменение данных.

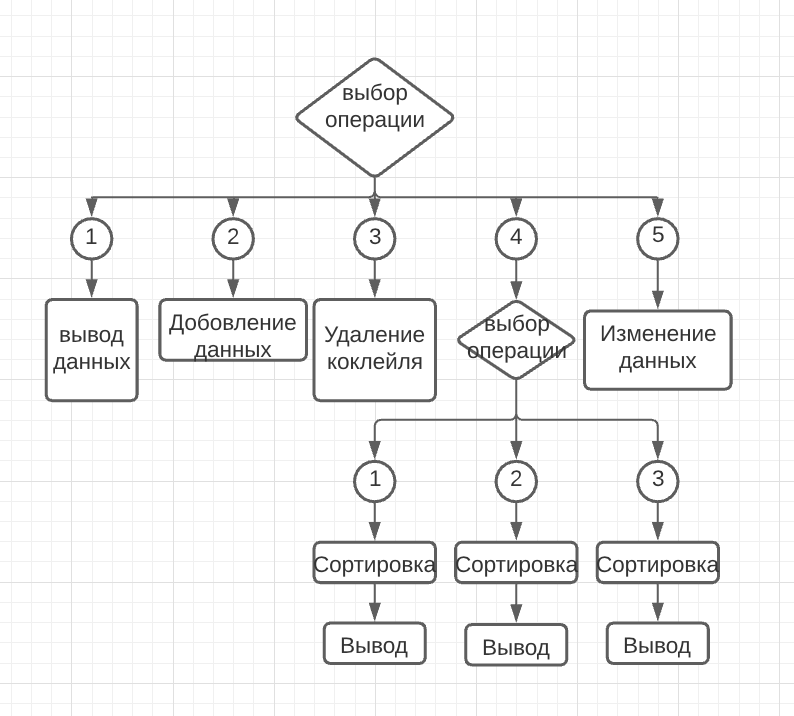


Рис 3

# **4.ОПИСАНИЕ ИНТЕРФЕЙСА**

При старте программы происходит выбор желаемой операции (рис.1)

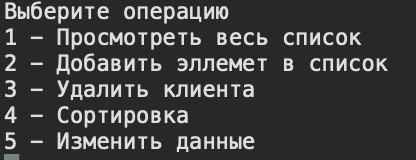


Рис. 4.1

При выборе первого пункта на консоль выводиться все данные из файла (рис. 2).

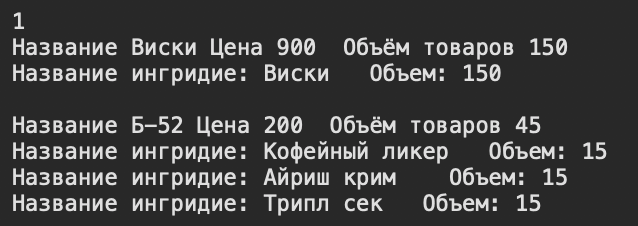


Рис.4.2

При выборе второго пункта пользователю необходимо поочерёдно вводить данные (рис 3). Любой коктейль состоит из ингредиентов. И суть добавление заключается в перечислении ингредиентов.

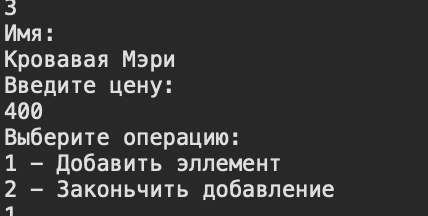


Рис. 4.3

При выборе третьего пункта вначале на консоль выводиться количество элементов. И необходимо выбрать элемент, входящий в область указанного значения. (рис 4)

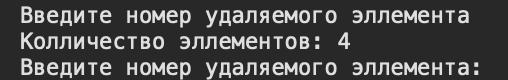


Рис.4.4

В случае же если значение указано неверно, то выводится следующая надпись (рис.5).



Рис.4.5

При выборе четвертого пункта происходит последующий выбор, по какому пункту будет происходить сортировка(рис.6), а потом отсортированные данные выводятся на консоль.

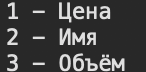


Рис.4.6

Изменение данных происходит схожим образам с добавление и удалением (рис 4). Вначале надо выбрать необходимый элемент. После этого происходит последующая последовательная изменение данных(рис.7).

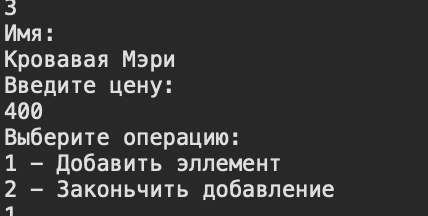


Рис.7

В случае же если выбран неверный пункт, то на консоль выводятся надпись (рис. 8.).



Рис 8

# 

## 

# **5.ВЫБОР СПОСОБА ОРГАНИЗАЦИИ ДАННЫХ**

Программа состоит одного класса.

Класс **Cocktail**:

Переменные: **string** Name, **int** Prise, **List<Ingredient>** ingredients, **int** ResultVolume– публичные переменные, которые служат в качестве информации об коктейлях.

Counter – статическая переменная служащая для определения общее количество коктейлей.

Методы:

Add **(**рис 5.1.**)** – данный метод является частью класса Cocktail. Он необходим для заполнения или редактирование данных коктейлей. Работа данного метода заключается в заполнении полей класса данными с консоли (Name, Prise, ingredients, ResultVolume). В методе используются механизмы ветвления и циклы, для реализации алгоритма вариации возраста. Данный метод не возвращает значений.

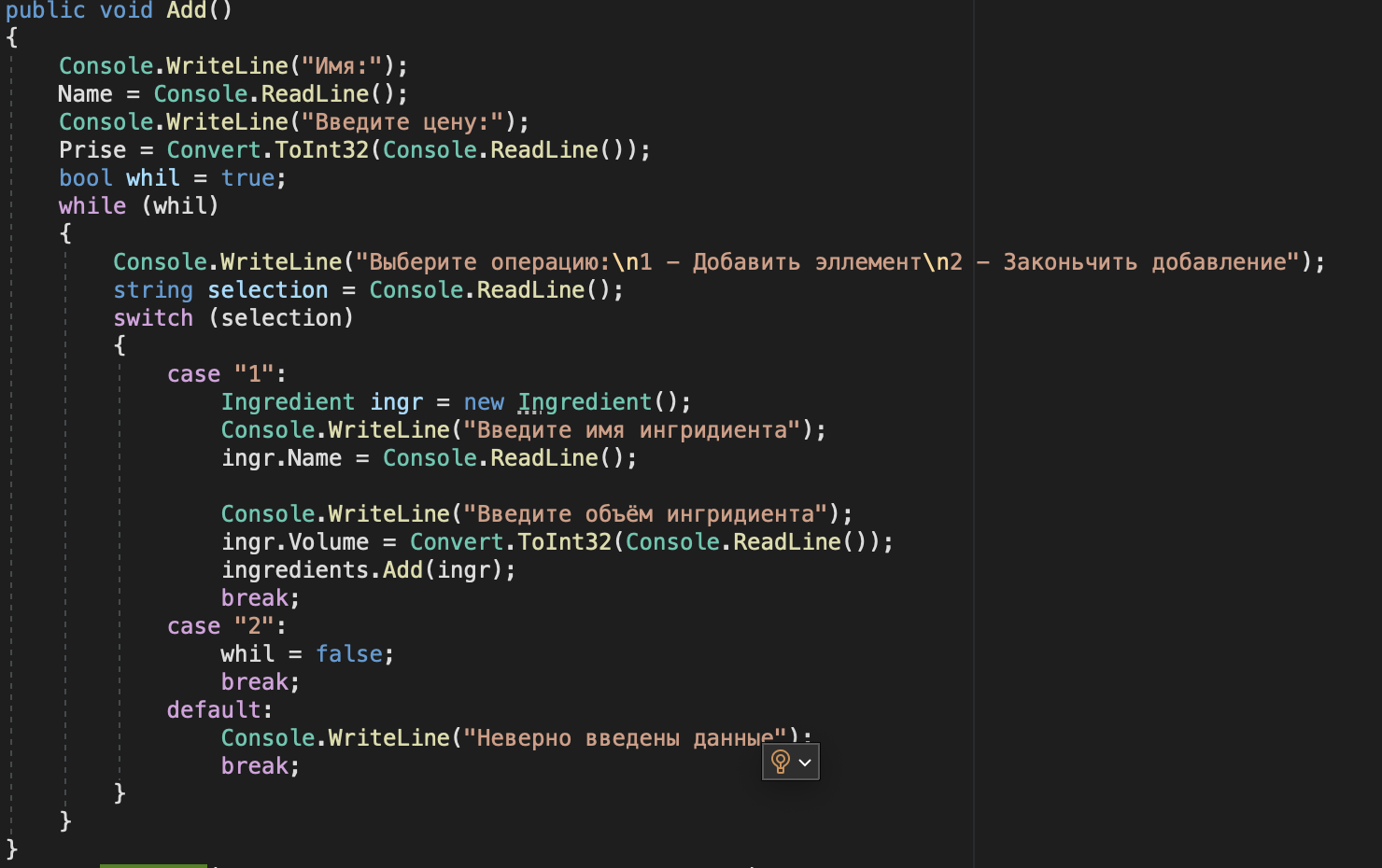


рис 5.1.

override ToString(рис 5.2) – данный метод является перегруженный метод из стандартной библиотеки. Он необходим для представления пользовательских типов данных в строковом виде. В данной программе этот метод перегружен для вывода переменныx: Name, Prise, ingredients, ResultVolume. Данный метод возвращает строковое значение.

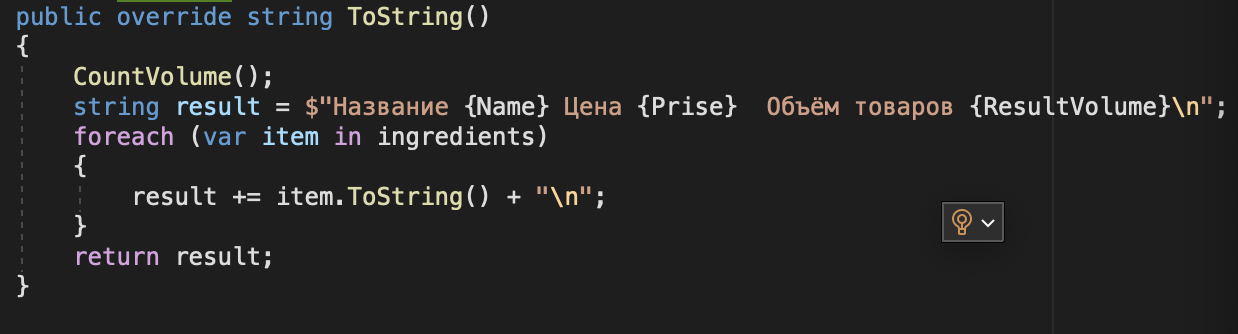
ъ

рис 5.2

Так же в классе реализован конструктор без параметров. Его главная и единственная задача считать количество созданных экземпляров класса. Так как чтение из файла объекта класса происходит через конструктора без параметров.

Так же в данном классе реализован конструктор копирования. С его помощью возможно присваивать данные полям объекта класса(рис 5.3)

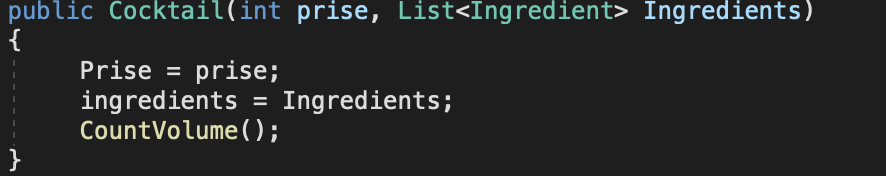
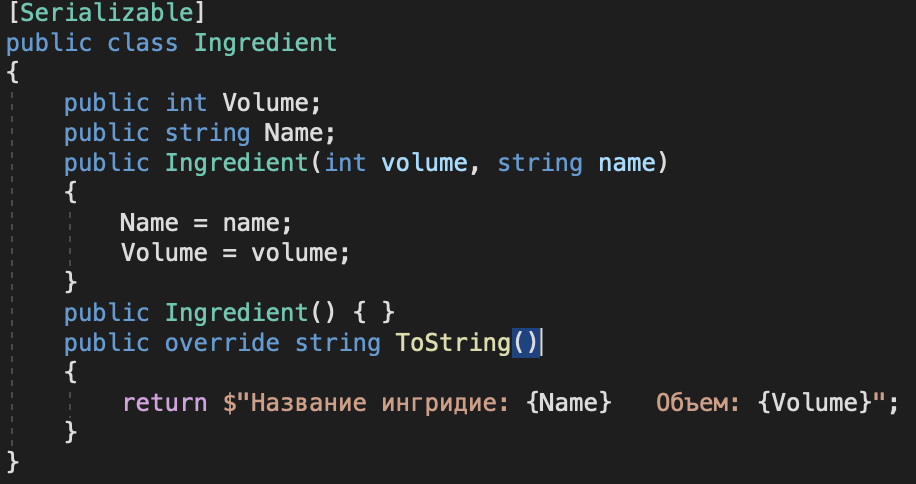


Рис 5.3

Класс **Ingredient** (рис 5.4.):



Переменные: **string** Name, **int** Volume – публичные переменные, которые служат в качестве информации об ингредиенте коктейля.

override ToString(рис 5.2) – данный метод является перегруженный метод из стандартной библиотеки. Он необходим для представления пользовательских типов данных в строковом виде. В данной программе этот метод перегружен для вывода переменныx: Name, Volume. Данный метод возвращает строковое значение.

Так же в данном классе реализован конструктор копирования. С его помощью возможно присваивать данные полям объекта класса.

# **6.РАЗРАБОТКА ПЕРЕЧНЯ ПОЛЬЗОВАТЕЛЬСКИХ МЕТОДОВ ПРОГРАММЫ**

**void** Print (рис 6.1) — это функция, состоящая из одной переменной типа IEnumerable<Cocktail> - это коллекцию которую метод будет выводить на консоль. Работа функции построена последовательно, в ней не предусмотрен циклический метод. Функция дает пользователю вывести список коктейлей на консоль. Данная функция не возвращает значений.

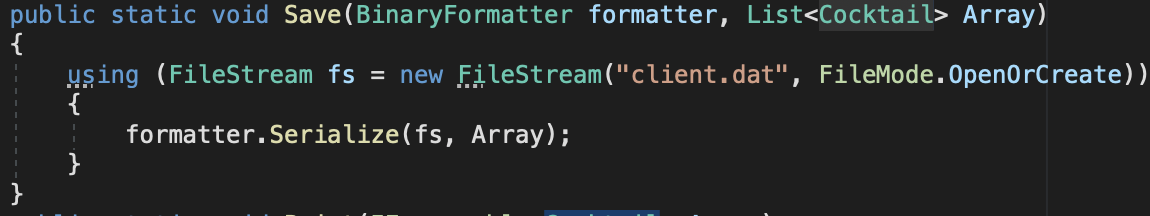


рис 6.1

**void** Save (рис 6.2) — это функция, состоящая из двух переменных BinaryFormatter – предназначенный для записи данных в .dat -файл и List< Cocktail> - непосредственно коллекция для которая будет записываться в файл. Данныя функция необходима для сохранения списка в файл. Для оптимальной работы потока ввода в файл была использована конструкция using. Так как при её использовании нет необходимости следить за закрытием потока. Работа функции построена последовательно. Данная функция не возвращает значений.

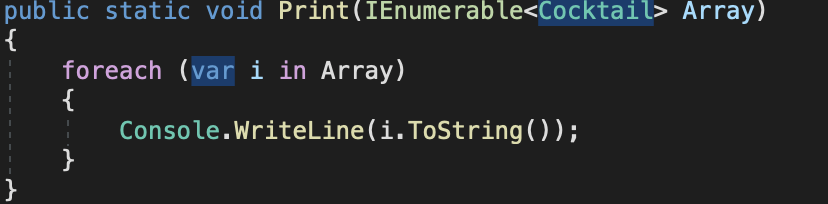


рис 6.2

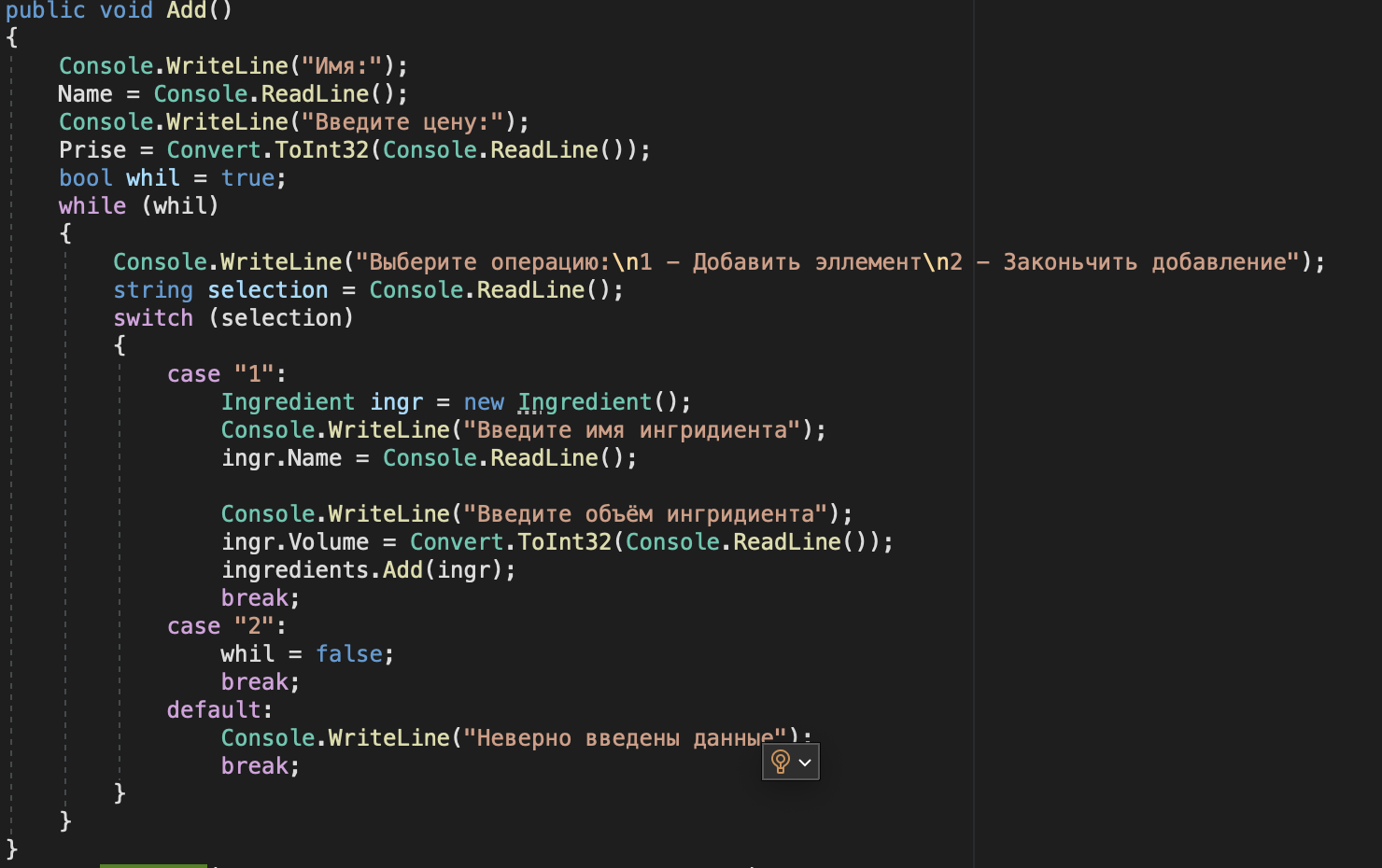
**void** Add (рис 6.3) -данный метод является частью класса Cocktail. Он необходим для заполнения или редактирование данных коктейлей. Работа данного метода заключается в заполнении полей класса данными с консоли (Name, Prise, ingredients, ResultVolume). В методе используются механизмы ветвления и циклы, для реализации алгоритма вариации возраста. Данный метод не возвращает значений.

рис 6.3

override **string** ToString (рис 6.4) – данный метод является перегруженный метод из стандартной библиотеки. Он необходим для представления пользовательских типов данных в строковом виде. В данной программе этот метод перегружен для вывода переменныx: Name, Age, Number, Musical\_Instrument. Данный метод возвращает строковое значение.

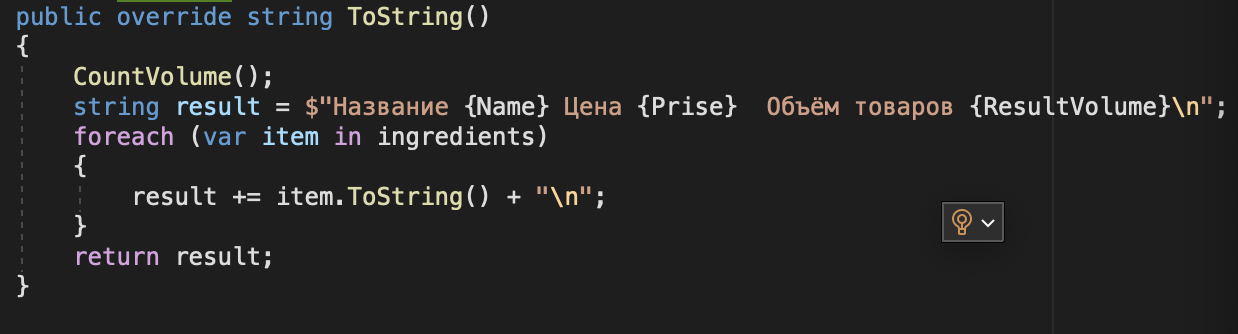
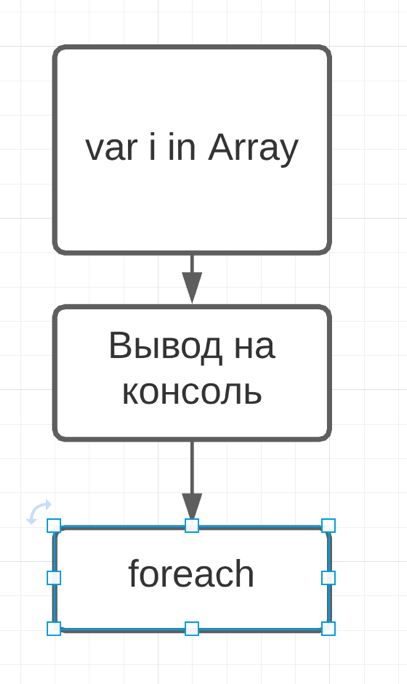


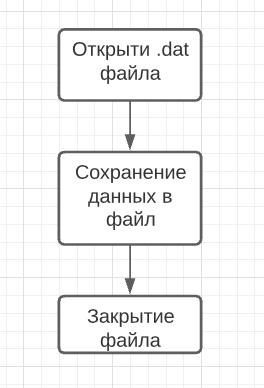
рис 6.4

# **7.ПРОГРАММНАЯ ЧАСТЬ**

void Print()



void Save()



Вначале происходит считывание данных из файла и заполнение данных в список. В случае же возникновение каких-либо ошибок они обрабатываются и выводиться на консоль.

Затем происходит выбор операции:

1. Просмотреть весь список
2. Добавить элемент в список
3. Удалить коктейль
4. Сортировка
5. Изменить данные

Далее при помощи конструкции switch происходит переход к выбранной операции.

1: Циклом foreach происходит итерация по списку. При каждой итерации происходит вызов перегруженного метода ToString. Который выводит в консоль данные каждого коктейля (Name, Prise, ingredients, ResultVolume).

2:Вначале создаётся новый(пустой) коктейль. Далее вызывается метод класса Cocktail Add.

Далее происходит добавление коктейлей в список, а затем происходит сохранение дополненного списка в файл при помощи функции Save – который принимает 2 параметра (Переменная для сохранения в файл и список коктейлей).

3: Удаление происходит схожим образом с добавлением из-за сходства данных алгоритмов и появилась необходимость создание функции Save.

Удаление элементов происходит по номеру коктейля в списке. Вначале на консоль выводится количество элементов чтобы пользователю было удобнее взаимодействовать в программе. А далее пользователь вводит желаемый элемент для удаления. Так же приходит проверка данных на валидность. Т.е проверка являются ли вводимые данные числом и не выходит ли данное число из области допустимых значений.

4: Сортировка данных происходит при помощи библиотеки Linq по следующим параметрам: Название, цена, объём. После же сортировки элементов происходит вывод отсортированного списка на консоль.

5: Изменение происходит схожим образом с удалением из-за сходства данных алгоритмов и появилась необходимость создание функции Save. Так как изменение происходит так же с выбором изменяемого элемента и последующей проверкой введенного номера на волосность. Вот только не производит создание нового элемента, а из выбранного элемента списка происходит вызов метода Add.

# **ЗАКЛЮЧЕНИЕ**

Исходя из задач, описанных выше главными требованиями к языку, на котором будет написана программы — это тонкая настройка и возможность использованием минимум вычислительных ресурсов добиваться максимум результата.

Были реализованы задачи:

1)Быстро и гибко редактироваться со стороны владельцев кафе студии.

2)Программный продукт имеет красивый, понятный и интуитивный интерфейс и владельцу приятно пользоваться программой.

3)Так же предусмотрено возможность добавления новых функций в программе.

Цель была выполнены, картотека имеет следующие преимущества:

1)Работа кафе происходит быстро и максимально эффективно.

2)Удовлетворяет запросы кафе.

2)Также реализована возможность пополнять базу, для оптимизации процессов кафе.

3)Реализована высокая оптимизация, что-то привело к возможности пользователем запускать приложение даже на самом слабом устройстве.

Данный программный продукт выгодно отличается от других подобных тем что, пользуясь им, пользователь может не только просматривать коктейли, но и производить различные взаимодействия с самими коктейлями.

Таким образом программа работает без сбоев и все задачи были выполнены.

# **СПИСОК ЛИТЕРАТУРЫ**

1. Багласова, Т.Г. Методические указания по оформлению курсовых и дипломных проектов / Т.Г. Багласова, К.О. Якимович. – Минск: КБП, 2013. – 29 c.
2. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. Приемы объектно-ориентированного проектирование / Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. – СПб: Питер, 2015. – 368с.
3. Коплиен Дж. Мультипарадигменное проектирование для С++ / Коплиен Дж. – Питер, 2005.
4. Страуструп Б. Программирование. Принципы и практика с использованием C++ / Страуструп Б. – 2-е изд. – Вильямс, 2016.
5. Model-View-Presenter [Электронный ресурс]. – Википедия, 2020. – Режим доступа: https://ru.wikipedia.org/wiki/Model-View-Presenter. – Дата доступа: 22.04.2020.
6. SFML [Электронный ресурс]. – SFML, 2020. – Режим доступа: https://www.sfml-dev.org. – Дата доступа: 22.04.2020.
7. SFML [Электронный ресурс]. – Википедия, 2020. – Режим доступа: https://ru.wikipedia.org/wiki/SFML. – Дата доступа: 22.04.2020.
8. Visual Studio 2019 [Электронный ресурс]. – Microsoft, 2020. – Режим доступа: https://visualstudio.microsoft.com/ru/vs. – Дата доступа: 22.04.2020.

# **ПРИЛОЖЕНИЕ**

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Runtime.Serialization;

using System.Runtime.Serialization.Formatters.Binary;

namespace Angrey

{

class Program

{

static void Main(string[] args)

{

List<Cocktail> cocktails = new List<Cocktail>();

BinaryFormatter formatter = new BinaryFormatter();

try

{

using (FileStream fs = new FileStream("client.dat", FileMode.OpenOrCreate))

{

cocktails = (List<Cocktail>)formatter.Deserialize(fs);

}

}

catch (SerializationException ex)

{

Console.WriteLine(ex.Message);

}

while (true)

{

Console.WriteLine("Выберите операцию");

Console.WriteLine("1 - Просмотреть весь список");

Console.WriteLine("2 - Добавить эллемет в список");

Console.WriteLine("3 - Удалить коктейль");

Console.WriteLine("4 - Сортировка");

Console.WriteLine("5 - Изменить данные");

string selection = Console.ReadLine();

switch(selection)

{

case "1":

foreach (var item in cocktails)

{

Console.WriteLine(item.ToString());

}

break;

case "2":

Cocktail cocktail = new Cocktail();

cocktail.Add();

cocktails.Add(cocktail);

Save(formatter, cocktails);

break;

case "3":

Console.WriteLine($"Введите номер удаляемого эллемента\nКолличество эллементов: {cocktails.Count}");

int Number;

while (true)

{

int result;

Console.WriteLine("Введите номер удаляемого эллемента: ");

string NumberStr = Console.ReadLine();

bool check = int.TryParse(NumberStr, out result);

if (check)

{

Number = Convert.ToInt32(NumberStr);

if (Number > cocktails.Count || Number < 1)

Console.WriteLine("Номер находиться вне обсасти допустимых значений");

else

break;

}

else

Console.WriteLine("Номер введён неправильно!");

}

cocktails.RemoveAt(Number - 1);

File.Delete("client.xml");

Save(formatter, cocktails);

break;

case "4":

Console.WriteLine("Выберите позитицию для сортировки");

Console.WriteLine("1 - Цена");

Console.WriteLine("2 - Имя");

Console.WriteLine("3 - Объём");

string choice = Console.ReadLine();

switch (choice)

{

case "1":

IEnumerable<Cocktail> sortedPrise = cocktails.OrderBy(i => i.Prise);

Print(sortedPrise);

break;

case "2":

IEnumerable<Cocktail> sortedName = cocktails.OrderBy(i => i.Name);

Print(sortedName);

break;

case "3":

IEnumerable<Cocktail> sortedResultVolume = cocktails.OrderBy(i => i.ResultVolume);

Print(sortedResultVolume);

break;

default:

Console.WriteLine("Вы выбрали неверный пунтк");

break;

}

break;

case "5":

int number = 0;

while (true)

{

int result;

Console.WriteLine($"Введите номер изменяемого эллемента\nКолличество эллементов: {cocktails.Count}");

string NumberStr = Console.ReadLine();

bool check = int.TryParse(NumberStr, out result);

if (check)

{

number = Convert.ToInt32(NumberStr);

if (number > cocktails.Count || number < 1)

Console.WriteLine("Номер находиться вне обсасти допустимых значений");

else

break;

}

else

Console.WriteLine("Номер введён неправильно!");

}

cocktails[number - 1].Add();

File.Delete("client.xml");

Save(formatter, cocktails);

break;

default:

Console.WriteLine("Данные введены не верно");

break;

}

}

}

public static void Save(BinaryFormatter formatter, List<Cocktail> Array)

{

using (FileStream fs = new FileStream("client.dat", FileMode.OpenOrCreate))

{

formatter.Serialize(fs, Array);

}

}

public static void Print(IEnumerable<Cocktail> Array)

{

foreach (var i in Array)

{

Console.WriteLine(i.ToString());

}

}

}

}

using System;

using System.Collections.Generic;

namespace Angrey

{

[Serializable]

public class Cocktail

{

public string Name;

public int Prise;

List<Ingredient> ingredients = new List<Ingredient>();

public int ResultVolume;

public void CountVolume()

{

ResultVolume = 0;

foreach (var i in ingredients)

ResultVolume += i.Volume;

}

public void Add()

{

Console.WriteLine("Имя:");

Name = Console.ReadLine();

Console.WriteLine("Введите цену:");

Prise = Convert.ToInt32(Console.ReadLine());

bool whil = true;

while (whil)

{

Console.WriteLine("Выберите операцию:\n1 - Добавить эллемент\n2 - Законьчить добавление");

string selection = Console.ReadLine();

switch (selection)

{

case "1":

Ingredient ingr = new Ingredient();

Console.WriteLine("Введите имя ингридиента");

ingr.Name = Console.ReadLine();

Console.WriteLine("Введите объём ингридиента");

ingr.Volume = Convert.ToInt32(Console.ReadLine());

ingredients.Add(ingr);

break;

case "2":

whil = false;

break;

default:

Console.WriteLine("Неверно введены данные");

break;

}

}

}

public Cocktail(int prise, List<Ingredient> Ingredients)

{

Prise = prise;

ingredients = Ingredients;

CountVolume();

}

public Cocktail() { }

public override string ToString()

{

CountVolume();

string result = $"Название {Name} Цена {Prise} Объём товаров {ResultVolume}\n";

foreach (var item in ingredients)

{

result += item.ToString() + "\n";

}

return result;

}

}

}

using System;

namespace Angrey

{

[Serializable]

public class Ingredient

{

public int Volume;

public string Name;

public Ingredient(int volume, string name)

{

Name = name;

Volume = volume;

}

public Ingredient() { }

public override string ToString()

{

return $"Название ингридие: {Name} Объем: {Volume}";

}

}

}