**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Саратовский государственный технический университет**

**имени Гагарина Ю.А.»**

Институт прикладных информационных технологий и коммуникаций

Направление «Информационные системы и технологии»

Кафедра «Прикладные информационные технологии»

**КОНТРОЛЬНАЯ РАБОТА**

по дисциплине «Теория алгоритмов»

Выполнил студент группы

б1-ИФСТипу-21

заочной формы обучения

Рамазанов Павел Андреевич

Номер зачетной книжки 202583

Проверил: старший преподаватель кафедры ПИТ

Пудиков А.С.

Саратов, 2022

**ЗАДАЧА «PIZZA TIME»**

* Смоделируйте работу пиццерии в вашем приложении.
* К сожалению, в вашей пиццерии ещё не работает доставка.
* Две ключевые сущности: пользователь и пиццерия взаимодействуют через заказ и пиццу. Пользователь делает заказ в заведении, после чего ждет оповещения о том, что пицца готова. Например – его имя высветится на табло. После этого пользователь сам забирает пиццу.
* Подумайте, как может выглядеть такое приложение? Какой минимум данных необходим для реализации?
* Плюсом будет хранение заказов – например, в формате JSON, или в текстовом файле, или БД (но не обязательно).
* Наличие консольного интерфейса обязательно. Можете продемонстрировать работу с помощью базовых Console.WriteLine(…) и Console.ReadLine(…).
* Обязательно наличие вменяемой архитектуры, прописанных сущностей и базового взаимодействия.

**КРИТЕРИИ ОЦЕНКИ И ДЕТАЛИ**

Оценка за экзаменационный проект будет выставлена по следующим критериям:

* 5 – выполнен законченный проект на одну из заданных тем, группа успешно показала и объяснила исходный код проекта;
* 4 – представлен недоделанный проект на одну из заданных тем, либо проект выполнен, но группа не смогла объяснить исходный код проекта;
* 3 – проект не представлен к сдаче.

**РЕШЕНИЕ ЗАДАЧИ**

Для того чтобы решение соответствовало техническому заданию (в нашем случае – заданию контрольной работы), для начала нужно создать классы, которые будут взаимодействовать между собой в главном классе программы – **Program**. Этими классами будут: **Pizza** (класс для создания сущностей «Пицца»), **Pizzeria** (класс для взаимодействия сущности «Пиццерия» с сущностями «Пицца» и «Клиент), **Consumer** (класс для взаимодействия с сущностями «Пицца» и «Пиццерия»). На рисунке 1 представлен процесс объявления вышеуказанных классов.

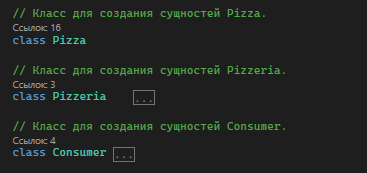


Рисунок 1 – Объявление классов

Теперь необходимо описать каждый из классов.

Класс **Pizza**. Так как этот класс только создает свои экземпляры, для него достаточно объявить конструктор, который будет принимать строку в качестве наименования названия пиццы. Также этот конструктор должен проверить входную строку на **null** или пустое значение. Если таковое имеется, то CLR C# выдает ошибку.

Помимо конструктора, нужно учесть принцип инкапсуляции ООП, поэтому к конструктору класса Pizza добавляется свойство с методами **set и get**.Код показан на рисунке 2.

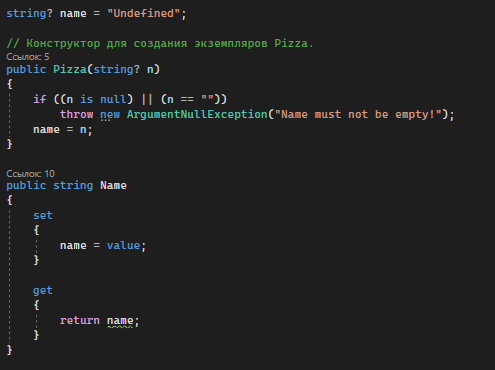


Рисунок 2 – Объявление конструктора и «сеттера», «геттера» для класса Pizza

Класс **Pizzeria**. Для начала нужно объявить в классе конструктор и «сеттер» с «геттером», аналогично классу **Pizza** (рисунок 3).

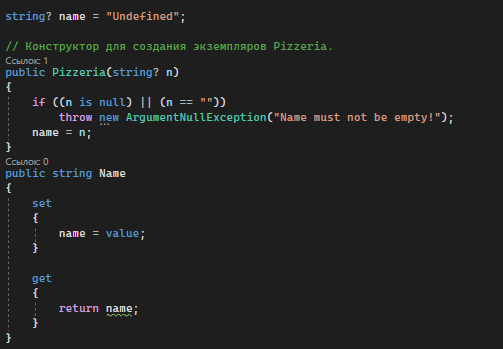


Рисунок 3 – Объявление конструктора и «сеттера», «геттера» для класса Pizzeria

Класс **Consumer**. По аналогии с двумя предыдущими классами необходимо объявить конструктор и «геттер», «сеттер (рисунок 4).

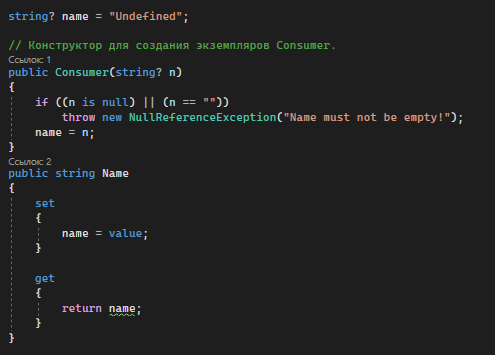


Рисунок 4 – Объявление конструктора и «сеттера», «геттера» для класса Consumer

Для взаимодействия с классами **Pizza** и **Consumer** внутри класса **Pizzeria** требуется описать следующие методы:

1. Получение названий пицц (названий экземпляров класса **Pizza**).
2. Уведомление о приготовлении купленной пиццы.
3. Экран процесса приготовления пиццы.
4. Сообщение, выводимое на экран, о выдаче готовой пиццы.

Заданные методы описаны на рисунке 5.

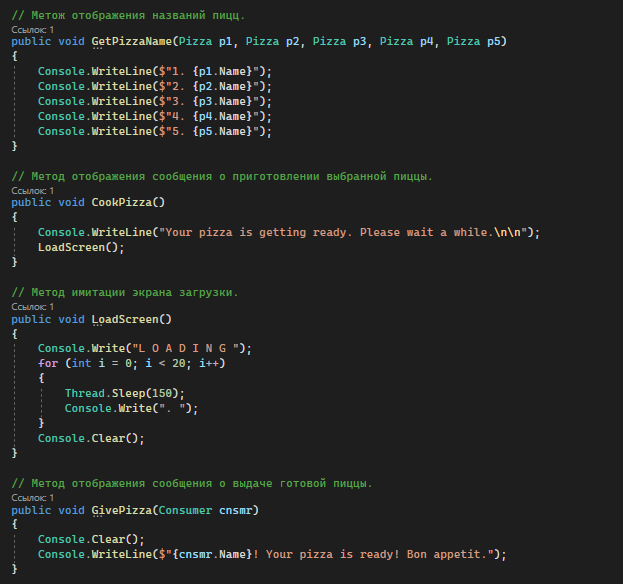


Рисунок 5 – Методы класса Pizzeria

Чтобы класс экземпляры класса **Consumer** корректно работали, внутри класса нужно описать пару методов:

1. Список действий, которые может выполнять экземпляр класса (купить пиццу / выйти из пиццерии).
2. Действие – забрать пиццу после приготовления.

Создание методов представлено на рисунке 6.

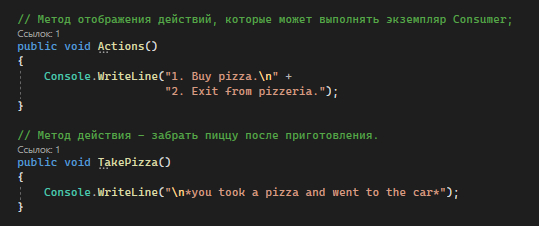


Рисунок 6 – Методы класса Consumer

Класс **Program**. Основная логика программы описана в главном классе **Program**, который аккуратно разбит на несколько методов, в том числе и исполняемый метод **Main()**.

Чтобы упростить читаемость программного кода, отдельные процедуры выполнения программы вынесены в отдельные методы (как, например, создание экземпляров классов **Pizza, Pizzeria** и **Consumer**), что представлено на рисунке 7.

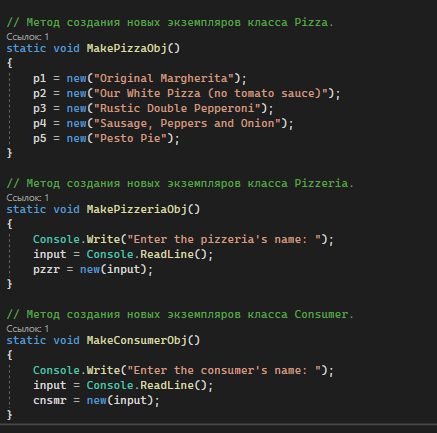


Рисунок 7 – Методы класса Program для создания объектов

Так же в отдельных методах описана логика программы (рис. 8, 9).

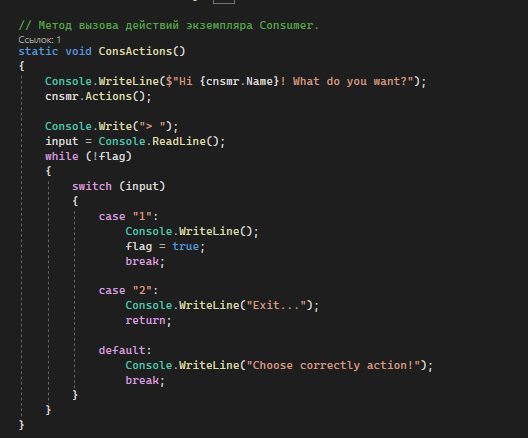


Рисунок 8 – Взаимодействие класса Program с классом Consumer через метод ConsActions()

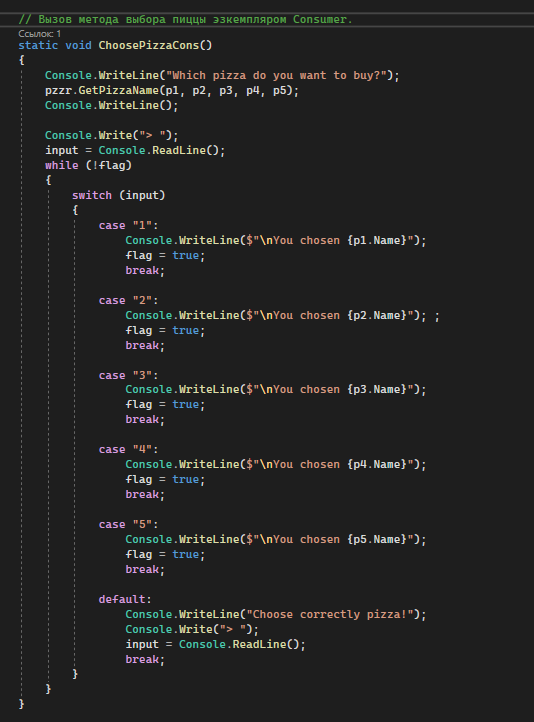


Рисунок 9 – Описание метода ChoosePizzaCons() (главное задание контрольной работы)

На рисунке 10 представлен метод вызова уведомлений о приготовлении пиццы.

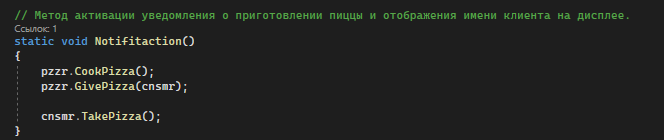


Рисунок 11 – Метод Notification()

Все методы вызываются из главного метода **Main()**. Ссылки на объекты классов **Pizza, Pizzeria** и **Consumer**, переменная для получения строки, введенной с клавиатуры (**input**) и переменная флага (**flag**) являются глобальными (рисунок 11).

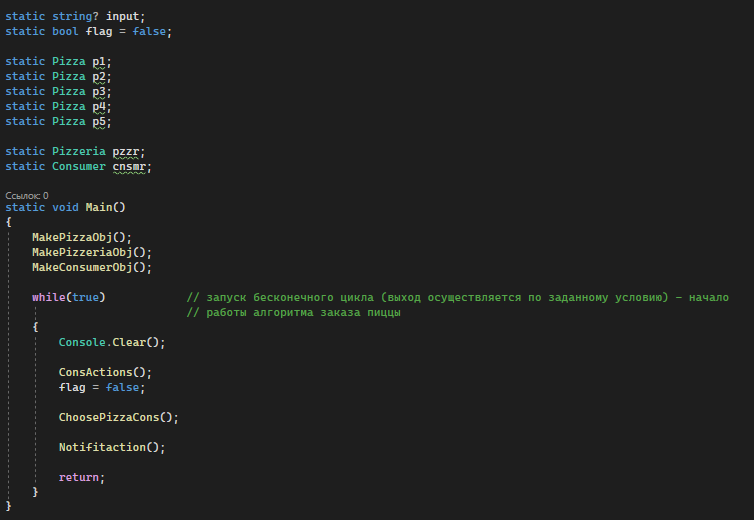


Рисунок 11 – Метод Main()