ООП часть 1

1. Класс Employee. Делайте в одном проекте.
    a. Создайте базовый класс Employee с полями Name и Salary, а также методом CalculateBonus(), который возвращает 10% от зарплаты. Создайте производный класс Manager, который переопределяет метод CalculateBonus(), чтобы возвращать 20% от зарплаты. Создайте объекты обоих классов, вызовите метод и выведите результат на экран.
    b. Расширьте класс Employee, добавив конструктор, принимающий имя и зарплату. В классе Manager добавьте дополнительное поле TeamSize. Реализуйте конструктор в классе Manager, который вызывает конструктор базового класса с помощью base и устанавливает значение TeamSize. В методе CalculateBonus() учитывайте размер команды: если команда больше 5 человек, бонус должен увеличиваться на 5%.
    c. Создайте новый класс Contractor, унаследованный от Employee, который добавляет поле HourlyRate и скрывает метод CalculateBonus(), делая его зависящим от отработанных часов. Реализуйте метод CalculateBonus(int hoursWorked). Создайте список сотрудников, включая Contractor, и реализуйте приведение типов, чтобы рассчитать бонусы для всех сотрудников, независимо от типа.
2. Создайте библиотеку классов. Создайте там интерфейс ILogger с методоми Trace(string), Info(string), Debug(string), Warning(string), Error(string), Fatal(string), а также метод Log(string, LogLevel). Добавьте классы FileLogger и ConsoleLogger, реализующие этот интерфейс. Реализованные методы должны писатьв консоль/файл сообщения типа “<dateTime> | <class> | <logLevel> | <message>”. Используйте эти классы для логирования операций в предыдущих заданиях.
3. Класс Animal. Делайте в одном проекте.
    a. Создайте базовый класс Animal с полями Name и Age. Добавьте методы Eat() и Sleep(), которые выводят соответствующие сообщения, например, "Animal is eating" и "Animal is sleeping". Создайте производные классы Dog и Cat, унаследованные от Animal.
    b. Добавьте в класс Animal виртуальный метод MakeSound(), который выводит "Some generic animal sound". Переопределите этот метод в классах Dog и Cat, чтобы они выводили "Woof!" и "Meow!" соответственно. Проверьте работу метода через объекты каждого класса.
    c. Добавьте в класс Animal конструктор, принимающий параметры name и age. В классах Dog и Cat реализуйте конструкторы, которые вызывают конструктор базового класса с использованием ключевого слова base и добавляют специфическую логику, например, вывод сообщения о создании конкретного животного.
    d. Создайте новый класс Parrot, унаследованный от Animal, который добавляет поле Color и скрывает метод MakeSound(), выводя сообщение "Parrot is talking". Реализуйте метод MakeSound(string words), который принимает строку и выводит её на экран, как будто попугай повторяет

слова. Создайте список животных, включающий Dog, Cat, и Parrot, и реализуйте приведение типов, чтобы вызвать скрытые и переопределенные методы для каждого типа животных.

e. Создайте интерфейс IFlyable с методом Fly(). Реализуйте этот интерфейс в классе Parrot, чтобы метод Fly() выводил сообщение "Parrot is flying". Создайте ещё один класс Eagle, который также реализует интерфейс IFlyable, но не наследуется от Animal. В методе Fly() класса Eagle выводите "Eagle is soaring high". Напишите программу, которая создает массив объектов типа IFlyable, включает в него объекты Parrot и Eagle, и вызывает метод Fly() для каждого из них.