Задачи.

1 – Создать класс TestConstructor с одним конструктором без параметров, при создании объекта от класса должна выходить текст “Hi I am run my constructor first” или “Привет я запустил свой конструктор первым”.

2 – Создать класс TestField, 1 - класс должен иметь одно поле с типом string с модификатором доступа private, 2 – У конструктора класса должен быть один параметр с типом string. 3 – Метод Show с типом void. При создании объекта от класса TestField переданный текст в параметре объекта должна отображаться при вызове метода Show. **(Для того что бы решить данную задачу смотрите прошлую лекцию Lesson-6.cs)**

3 – Создать класс TextLibrary без конструктора, задача класса отображать данные разных типов используя один метод Show **(Используйте подход, перегруженный метод)**

4 – Создать класс User, у класса должны быть следующие поля: string name, string address, byte age, модификаторы доступа всех полей должны быть private. У конструктора класса должны быть 3 параметра string \_name, string \_address, byte \_age. У класса должны быть методы GetName(), GetAddress(), GetAge() все методы должны возвращать переданные значений. **(Что бы тестировать работает ли класс, используйте уже созданный класс TextLibrary)**

Шпаргалка:

TextLibrary text = new TextLibrary();

User user = new User(“Иванов Иван”, “г. Бишкек”, 25); text.Show(user.GetName()); text.Show(user.GetAddress()); text.Show(user.GetAge());

5 – Создать класс Bank у класса должны быть закрытые поля string account, decimal sum, у конструктора класса должны быть 2 параметра string \_account, decimal \_sum, и методы GetAccount(), GetSum().**(Что бы тестировать работает ли класс, используйте уже созданный класс TextLibrary)**

Техническое задание.

Требуется создать класс, который будет выдавать кредит, у класса должны быть следующие поля **string receiverName, string receiverAddress, string receiverAge, string receiverAccount, decimal receiverSum**, у конструктора класса должны быть следующие параметры: **string \_receiverName, string \_receiverAddress, string \_receiverAge, string \_receiverAccount, decimal \_receiverSum** У класса должен быть один метод PrintCheck() без параметров и без возвращающего типа, используйте void. Что бы выдать кредит нужно использовать уже готовые классы **User** и **Bank.**

Шпаргалка:

User user = new User(“Иванов Иван”, “г. Бишкек”, 25); string userName = user.GetName(); string userAddress = user.GetAddress(); string userAge = user.GetAge();

Bank bank = new Bank(“1111234567892222”, 2500); string userAccount = user.GetAccount(); decimal sum = bank.GetSum();

Creditor credit = new Creditor(userName, userAdress, userAge, userAccount, sum); credit.PrintCheck();

При вызове метода PrintCheck() результат должен быть следующим

ФИО: Иванов Иван Адрес клиента: г. Бишкек Счет: 1111234567892222 Сумма кредита: 2500

Во время печати чека должны быть проверки если возраст клиента меньше 18лет, то результат должен быть следующим. Извините, но мы не можем выдать кредит, вам еще нет 18, приходите через 5лет, что бы получить кредит. Что бы выводить данные в методе PrintCheck() используйте уже готовый класс **TextLibrary.**

**public void PrintCheck() {**

**TextLibrary text = new TextLibrary();**

**}**