1. Класс Дробное число со знаком (Fractions). Число должно быть представлено двумя полями: целая часть - длинное целое со знаком, дробная часть - беззнаковое короткое целое. Реализовать арифметические операции сложения, вычитания, умножения и операции сравнения. Проверить эти методы.
2. Класс Деньги для работы с денежными суммами. Число должно быть представлено двумя полями: для рублей и для копеек. Дробная часть (копейки) при выводе на экран должна быть отделена от целой части запятой. Реализовать сложение, вычитание, деление сумм, деление суммы на дробное число, умножение на дробное число и операции сравнения. Проверить эти методы.
3. Класс Равнобочная трапеция, члены класса: координаты 4-х точек. Предусмотреть в классе конструктор и методы: проверка, является ли фигура равнобочной трапецией; вычисления и вывода сведений о фигуре: длины сторон, периметр, площадь. Продемонстрировать работу с классом: дано N трапеций, найти количество трапеций, у которых площадь больше средней площади.
4. Сделайте класс **Worker**, в котором будут следующие **public** поля - name (имя), age (возраст), salary (зарплата). Создайте объект этого класса, **затем** установите поля в следующие значения (не в \_\_construct, а для созданного объекта) - имя 'Иван', возраст 25, зарплата 1000. Создайте второй объект этого класса, установите поля в следующие значения - имя 'Вася', возраст 26, зарплата 2000. Выведите на экран сумму зарплат Ивана и Васи. Выведите на экран сумму возрастов Ивана и Васи.
5. Сделайте класс **User**, в котором будут следующие **protected** поля - name (имя), age (возраст), **public** методы setName, getName, setAge, getAge. Сделайте класс **Worker**, который наследует от класса User и вносит дополнительное **private** поле salary (зарплата), а также методы **public** getSalary и setSalary. Создайте объект этого класса 'Иван', возраст 25, зарплата 1000. Создайте второй объект этого класса 'Вася', возраст 26, зарплата 2000. Найдите сумму зарплата Ивана и Васи.
6. Сделайте класс **User**, в котором будут следующие **protected** поля - name (имя), age (возраст), **public** методы setName, getName, setAge, getAge. Сделайте класс **Driver** (Водитель), который будет наследоваться от класса User. Этот класс должен иметь следующие private поля: водительский стаж, категория вождения (A, B, C). Создайте несколько объектов класса **Driver.** Посчитать средний стаж водителей.
7. Построить систему классов для описания плоских геометрических фигур: круга, квадрата, прямоугольника. Предусмотреть методы для создания объектов, перемещения на плоскости, изменения размеров и поворота на заданный угол.
8. Построить описание класса, содержащего информацию о почтовом адресе организации. Предусмотреть возможность раздельного изменения составных частей адреса, создания и уничтожения объектов этого класса.
9. Класс Phone.  
   a) Создайте класс Phone, который содержит переменные number*,*model и weight.  
   б) Создайте три экземпляра этого класса.   
   в) Выведите на консоль значения их переменных.   
   г) Добавить в класс Phone методы: receiveCall, имеет один параметр – имя звонящего. Выводит на консоль сообщение “Звонит {name}”. getNumber – возвращает номер телефона. Вызвать эти методы для каждого из объектов.  
   д) Добавить конструктор в класс Phone, который принимает на вход три параметра для инициализации переменных класса - number, model и weight.
10. Создать класс Person, который содержит:   
    a) поля fullName*,*age.   
    б) методы move() и talk()*,*в которых просто вывести на консоль сообщение -"Такой-то  Person говорит".   
    в) Добавьте гибкий конструктор, позволяющий вызывать его в следующих вариантах  - Person() и Person(fullName, age).  
    Создайте два объекта этого класса. Один объект инициализируется конструктором Person(), другой - Person(fullName, age)*.*
11. Определить класс Reader, хранящий такую информацию о пользователе библиотеки: ФИО, номер читательского билета, факультет, дата рождения, телефон. Создать методы takeBook(), returnBook():  
    - takeBook*,*который будет принимать переменное количество названий книг. Выводит на консоль сообщение "Петров В. В. взял книги: Приключения, Словарь, Энциклопедия".

Аналогичным образом создать метод returnBook()*.*

1. Создайте пример наследования, реализуйте класс Student и класс Aspirant, аспирант отличается от студента наличием некой научной работы.  
   а) Класс Student содержит переменные: String firstName, lastName, group*.* А также double averageMark*,*содержащую среднюю оценку.  
   б) Создать переменную типа Student, которая ссылается на объект типа Aspirant.  
   в) Создать метод getScholarship() для класса Student*,*который возвращает сумму стипендии. Если средняя оценка студента равна 5, то сумма 2000 грн, иначе 1300. Переопределить этот метод в классе Aspirant.  Если средняя оценка аспиранта равна 5, то сумма 3000 грн, иначе 2000.
2. Создать классы Car , Engine  и Driver . Класс Driver содержит поля - ФИО, стаж вождения.  
   Класс Engine содержит поля - мощность, производитель.  
   Класс Car содержит поля - марка автомобиля, класс автомобиля, вес, водитель типа Driver, мотор типа Engine. Методы start()*,*stop(), turnRight()*,*turnLeft()*,* которые выводят на печать: "Поехали", "Останавливаемся", "Поворот направо" или "Поворот налево". А также метод printInfo(), который выводит полную информацию об автомобиле, ее водителе и моторе.
3. Создать класс Animal и расширяющие его классы Dog, Cat, Horse. Класс Animal содержит переменные food, location и методы makeNoise, eat, sleep. Метод makeNoise*,*например, может выводить на консоль "*Такое-то животное спит*". Dog, Cat, Horse переопределяют методы makeNoise, eat*.*Добавьте переменные в классы Dog, Cat, Horse, характеризующие только этих животных.
4. а) Создать класс Товар*,*имеющий переменные имя, цена, рейтинг*.*   
   б) Создать класс Категория, имеющий переменные имя и массив товаров. Создать несколько объектов класса Категория.

в)Вывести все товары из всех категорий , упорядочив по возрастанию цены.