1. Создайте страницу *lab6-1.html*, на которой будет объект, например, *commonstudent*, содержащий следующие свойства:
   - ВУЗ,
   - Группа,
   - Курс (год обучения),
   - Кафедра,
   - Тип обучения (очно-заочно).

   Создайте несколько дочерних объектов (*student01, student02 … studentNN),* прототипом которых будет объект *commonstudent*. Объекты *student* должны содержать следующие свойства:
   - “ФИО”,
   - “год рождения”,
   - “пол”,
   - “платное/бюджетное обучение”.

   Объект *commonstudent* должен содержать метод подсчета текущего возраста студента (*getAge()*) (считающийся как «текущий год» – «год рождения»). Также он должен содержать метод вывода (*getInfo()*) своих свойств следующим образом:
   «Студент …(ФИО) обучается в институте … , в группе … . В данный момент он находится на … курсе .» (*this*)
   Эти методы должны вызываться из объектов-наследников (*student01, student02 … studentNN),* прототипом которых является *commonstudent.*
   Организуйте вывод на экран списка имён всех ваших объектов (*student01, student02 … studentNN)*, и кнопку, которая выводит в текстовый контейнер <p> с помощью метода *getInfo* свойства объекта, имя которого введет пользователь. (*prompt, innerHTML*)
   В качестве прототипа и производных объектов могут быть использованы любые реальные предметы и понятия, содержащие какие-то свойства:
   (Прототип: *car* со свойствами: “ количество колес”, “ количество дверей”; объекты *car01*… со свойствами “модель”, “пробег”, “год выпуска” и т.п.)
   (Прототип: *Jacket* со свойствами “материал”, “сезон”, объекты *Jacket01…* со свойствами “цвет”, “цена”, “бренд”, и т.п.)

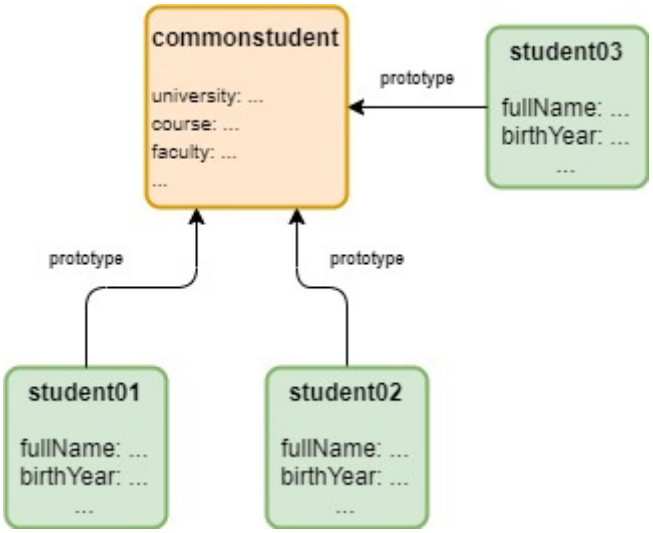

Рис. 1. Общая схема прототипирования из п.1

2. Разберите пример *Blog.html*, в котором находится блог для любителей учиться. Принцип работы данного сценария: имеется массив *blog*, содержащий объекты, формирующие записи блога. Конструктор *Blog()* создает объект записи блога с двумя свойствами – текст записи (body) и дата записи. Функция *showBlog()* выводит записи блога под атрибутом *div*.
3. Усовершенствуйте код страницы *Blog.html* следующим образом:
   - Модифицируйте способ сохранения дат, изменив тип данных переменной *date* на объект *Date()*. Формат даты для записей блога должен отображаться в виде ДД.ММ.ГГГГ.
   - Создайте функцию сортировки массива записей блога по дате их создания в обратном хронологическом порядке (самая свежая запись должна быть наверху). Данная функция должна запускаться при каждом добавлении новой записи (*sort*).
   - Создайте функцию поиска текста по всем записям блога *searchText()*. Результат поиска (запись, содержащая совпадения) должен выводиться на экран (*alert*). В случае отсутствия результатов поиска в текстовое поле *searchfail* должно выводиться сообщение об отсутствии совпадений. (*indexOf*, *toLowerCase*).
   - Напишите функцию *randomBlog()*, случайным образом выбирающую запись блога и отображающую её в отдельном окне.
   - Создайте метод *containsText()* объекта *Blog* на основе функции поиска (*this*), возвращающий true в случае, если запись блога содержит поисковую строку.
   - Создайте метод *toHTML()* объекта *Blog*, на основе функции *showBlog()*, возвращающий запись блога как отформатированный HTML-код.
   - Создайте метод *toString()* объекта *Blog*, выполняющий преобразование записей блога в строку вида: *“[ДД:ММ:ГГГГ] текст записи из body”*.
   - Для избавления от излишнего дублирования методов для каждого экземпляра объекта *Blog*, создайте прототип объекта *Blog*, который содержит три вышеупомянутых метода.
   - Создайте в прототипе объекта *Blog*, свойство signature, которое содержит подпись автора блога. Реализуйте добавление этой подписи к каждой записи блога.
   - Осуществите возможность опционального добавления картинки к записи блога (файл добавляемого изображения должен находиться в той же папке, что и файл *Blog.html*).

4. Создайте свой блог (состоящий из 10 и более записей), реализованный по данному алгоритму.
5. Используйте для оформления вашего блога фреймворк Bootstrap