1. класс – это сборная структура свойств (характеристик), определяющая общие признаки для каких-либо предметов (объектов). Объект – один экземпляр, например, класса, обладающий своими определенными значениями свойств и методов. Класс – многоэтажный дом (свойства: этажность, тип конструкции, год постройки, тип проекта планировок и т.д), объект – любой конкретный дом на улице (у него можно описать все свойства, определенные в данном классе)
2. Школа: учителя (ФИО, предмет, стаж, квалификация, выпуск, образование), ученики (ФИО,класс, пол, направление обучения, ср. балл), администрация (ФИО, отдел, квалификация, должность, функции)

Заказ покупателя(дата, товар, способ доставки, способ оплаты), запрос на сайте на консультацию врача (дата, тема, способ связи, контактные данные), заявка в техподдержку(имя, тема, вопрос/проблема, дата, доп. файлы – фото, видео)

1. конструктор – это спец. функция, которая предназначена для создания определенной конфигурации – набора свойств и методов- и использования ее для дальнейшего наполнения созданных объектов в соответствии с данной конфигурацией. конструктор- эквивалент типов данных(классов) в некоторых языках, в js можно пользоваться встроенными конструкторами для примитивов, тривиальных данных, сложных типов и типом функция.

с помощью function (параметр1, парматер2, …) {} создается анонимная функция-конструктор для объекта типа Object. В тело функции можно добавлять свойства, затем создавать экземпляры по данному конструктору. если создавать переменные с помощью сокращенных способов создания переменных: литерал, сокращенных выражений, без оператора new, то к данной переменной нельзя добавлять новые свойства и методы. С конструктором используется ключевое слово this, которое позволяет обращаться именно к данному объекту в процессе создания свойств.

1. в первом случае ничего не выведет, т.к. изначально не был указан параметр в скобках, и функция не знает, что и откуда брать

в двух следующих случаях будет ошибка, т.к. мы вызываем bike как функцию, а это конструктор, и т.к. он не записан в теле объекта и объект не создан по этому конструктору, они не могут быть связаны и использоваться друг с другом.

1. Статические свойства задают какую-либо общую характеристику для всего класса в целом, а не для отдельных объектов, например, какой-либо устоявшийся факт – температура кипения воды. при создании экземпляра данного класса это свойство унаследуется в указанном значении, обратится к нему с помощью оператора this нельзя, только через обращение к классу: console.log(имякласса.имястатики)
2. создадим класс Tea, одним из его методов можно установить функцию добавления сахара, другим-уменьшение, это будут сеттеры, т.к. при обращении к этим методам через вновь созданный объект данного класса мы можем регулировать количество сахара на определенный шаг, заданный сеттером. Напрямую регулировать количество сахара нельзя, это обращение не перезапишет установленное значение, т.к. внешне присвоить значение переменной, входящей в внутренние свойства объекта нельзя. Создав функцию, которая выводит итоговое значение количества сахара, мы можем обращаться к объекту, т.е. вызывать геттер – данные о состоянии какого-либо свойства.
3. будет выведено Object, т.к. в JS нельзя создавать новые типы данных классами, а только данные типа Object.
4. код выведет исходные данные без города, т.к. мы попытались создать сеттер извне, безотносительно данного объекта. если вместо 6 и 7 строки написать person.city = "amsterdam"; тогда свойство добавится, т.к. мы обратились к свойствам через сам объект