Тестовое задание.

Игрок управляет “космическим кораблем”, который представляет собой равнобедренный треугольник. Корабль может лететь вперед, назад, влево и вправо по нажатию на клавиши управления. Клавиши управления должны быть настраиваемыми в редакторе. Вершинный угол “корабля” - это его “нос”. Нос корабля всегда смотрит на указатель мыши.

Также корабль может стрелять снарядами (форма по желанию) по нажатию на клавишу space. Снаряды всегда летят прямо. В случайных местах (за экраном) спавнятся “метеориты”, имеющие случайное направление движения и скорость в определенном диапазоне. Метеориты представляют собой круглые объекты. При попадания снарядов в метеориты, последние получают урон. Если количество “хп” у метеоритов становится меньше нуля, то метеорит уничтожается, а игроку зачисляется определенное количество очков. Количество очков зависит от “хп” метеорита и его скорости. Если метеорит сталкивается с кораблем, то корабль уничтожается и игра заканчивается.

В игре должно быть несколько уровней. Для каждого уровня задано количество очков, которые необходимо набрать игроку, чтобы завершить уровень. В каждом последующем уровне количество спавнящихся “метеоритов” должно увеличиваться. Также с уровнями должна повышаться скорость метеоритов и количество их “хп”. При выполнении условий для завершения уровня должен загружаться новый уровень.

В каждом уровне при достижении определенного количества очков урон оружия корабля повышается. Также меняется цвет снарядов. Уровень оружия корабля сохраняется от уровня к уровню.

Значения множителей для скорости метеоритов, их “хп” и количества, зависящие от уровня должны быть редактируемыми через редактор (например храниться в scriptable object, или в xml, но не должны задаваться в префабе). То же касается количества очков, необходимых для “апгрейда” оружия, цвета снарядов и урона оружия. Все значения должны редактироваться независимо друг от друга.

Задание должно работать на версии Unity 2017+. Нежелательно использование легаси компонентов.

При выполнении задания желательно продемонстрировать следующее:
- использование паттернов проектирования;
- использование принципов SOLID;
- использование внедрения зависимостей (можно использовать сторонние фреймворки);
- избегать использования “статики”.

Итоговый проект можно передать в виде ссылки на архив с проектом (например на google disk), или в виде ссылки на открытый репозиторий (bitbucket, github etc).