**Тема 1. Реализация «танкового сражения» с использованием ботов (4 часа)**

1. Подготовить префаб для вражеского танка в Assets из уже имеющегося танка.
2. Добавить на первый танк (не на префаб) большой (радиус ~40) сферический триггерный коллайдер, для обнаружения танка-игрока другими танками-ботами.
3. Создать новый скрипт **bot**. Объявить в нем **float** переменные **movespeed** (скорость передвижения), **rotseedtank** (скорость поворота), **rotspeedbash** (скорость поворота башни), **speedcore** (скорость вражеского снаряда). Переменные типа **Transform (bash и stvol)** для управления его башней и стволом. Переменную типа **GameObject core** для ссылки на префаб снаряда, **bool canshoot = true**, для определения, может ли в данный момент танк произвести выстрел. Переменную типа **int life = 3** для определения количества попаданий в танк-бот, после которого он уничтожится.
4. Проинициализировать данные переменные в методе **Start()** или через инспектор.
5. Добавить в скрипт бота метод **OnTriggerStay(Collider other)**, т.к. выполнять какие-либо действия он должен при нахождении в большом триггере игрока. Внутри этого метода определить, что танк находится именно в триггере игрока: **if(other.tag == “Player”)** , предварительно добавив к игроку данный тэг.
6. После определения, что танк находится в триггере игрока, нужно вычислить некоторые значения:
7. дистанцию до игрока:

**float distance = Vector3.Distance(other.transform.position, transform.position);**

1. Определить вектор направления:

**Vector3 relativPos = (other.transform.position – transform.position);**

1. И по вектору направления определить поворот:

**Quaternion newrot = Quaternion.LookRotation(relativePos);**

1. При дистанции меньше 20, бот должен двигаться и поворачиваться в направлении к игроку:

**If(distance<20)**

**{**

//поворачиваем танк с помощью интерполяции между текущим поворотом и требуемым //направлением в сторону игрока с заданной скоростью

**transform.rotation = Quaternion.Slerp(transform.rotation, newrot, Time.deltaTime \* rotspeedtank);**

//также с помощью интерполяции двигаем танк в сторону игрока, меняя значения по х и z осям

**transform.position = new Vector3(Mathf.Lerp(transform.position.x, other.transform.position.x, Time.deltaTime \* movespeed), transform.position.y, Mathf.Lerp( transform.position.z, other.transform.position.z, Time.deltaTime \* movespeed)**

**}**

1. В независимости от расстояния, но при нахождении в триггере поворачиваем башню в сторону игрока:

**bash.rotation = Quaternion.Slerp(bash.rotation, newrot, Time.deltaTime \* rotspeedbash);**

1. Для определения, повернута ли уже башня на игрока, используем метод бросания лучей:

**RaycastHit hit;** //переменная для записи объекта попадания

//выпускаем луч из башни в направлении относительно ее – вперед.

**If(Phisics.Raycast(bash.position,bash.TransformDirection(Vector3.forward), out hit))**

**{**

**If((hit.transform.tag==”Player”)&&canshoot)**

//если луч попал в коллайдер игрока и можно выстрелить

**StartCoroutine(botshoot());** //запускаем корутину для выстрела вражеского танка

**}**

1. Написать короутину для выстрела:

**IEnumerator botshoot()**

**{**

**canshoot = false;** //указываем, что теперь этот танк стрелять не может

//определяем координту для создания снаряда

**Vector3 forwardofstvol = stvol.position + stvol.TransformDirection(Vector3.forward\*4f);**

//создаем снаряд из префаба в требуемой координате

**GameObject newcore = Instantiane(core, forwardosstvol, stvol.rotation);**

//поворачиваем ядро в сторону поворота ствола

**newcore.transform.LookAt(forwardofstvol);**

**yield return new WaitForSeconds(3f);** //ждем 3 секунды (время перезарядки)

**canshoot = true;** //указываем, что можем сделать новый выстрел

**}**

1. Добавить возможность уничтожения вражеского танка, добавив метод **OnCollisionEnter (Collision collision)**. Внутри этого метода, определить по тэгу коллизию со снарядом:

**if (collision.gameObject.tag == “core”)**, а в этом условном блоке отнять счетчик жизни: **life--;** и в случае утраты всех жизней, уничтожить вражеский танк: **if(life<1) Destroy(gameObject);**

1. Поместить созданный скрипт на префаб бота.
2. Добавить взрыв ядра при попадании. Для этого открыть скрипт ядра, добавить в него ссылку на префаб взрыва (префаб взрыва найти в стандартных ассетах) **GameObject explosion**; а в метод **OnCollisionEnter** добавить строчку для создания экземпляра взрыва в точке, где находится ядро при попадании: **Instantiate(explosion, gameObject.transform);**
3. Добавить на сцену в различных местах несколько экземпляров танков-ботов и проверить, что при попадании в триггер танка-игрока они движутся и поворачиваются в его направлении, а также их башня вращается и стреляет в направлении игрока. При попадании снаряда во вражеский танк-бот 3 раза, он должен удалиться со сцены. Снаряд при попадании в танк должен взрываться.