Uwagi co do struktury.

1. Powinno się zastosować Object Pooling aby zmniejszyć obciążenie pamięci. Należałoby wprowadzić Object Pooling jako temat lekcji.
2. Pociski nie znikają samoczynnie. Jeżeli w nic nie trafią. Należałoby wprowadzić timer usuwający pociski. Wstępnie wprowadzono timer z Destroy Object, ale docelowo powinno się to zintegrować z Object Pooling.
3. Przeciwnicy powinni być zgrupowani jako Prefab aby zmiany w prefabie powodowały zmiany we wszystkich obiektach. W przeciwnym wypadku może wystąpić błąd tak jak opisany poniżej.
4. Player Controller powinien być używany do poruszania się postaci a osobno powinien być skrypt odpowiedzialny za strzelanie.
5. W zadaniu włączono do skryptu Player Controller funkcje odpowiedzialne za tracenie życia, zyskiwanie życia, warunki zwycięstwa i strzelanie. Każda z nich powinna być w osobnej klasie.
6. Destroy(GetComponent<PlayerLook>()); spowoduje problemy gdy będziemy w przyszłości chcieli zrobić Respawn postaci. Lepiej go dezaktywować komendą enabled=false a potem uruchomić ponownie w osobnej metodzie Respawn.
7. Dlaczego nie zastosowano wbudowanego systemu INPUT ?
8. W grze brakuje grawitacji. Została ona dodana bez używania Rigidbody, ale zamiast Character Controllera można też wprowadzić Rigidbody i stworzyć customowy skrypt do poruszania postacią.
9. Nie ma potrzeby dodawania komponentu Character Controller do przeciwników jeżeli nie ma skryptu który ich porusza.
10. Zmienna HP jest niepotrzebna ponieważ dotknięcie przeciwnika powoduje natychmiastową śmierć.
11. Nie ma menu pozwalającego wyłączyć grę ani opcji restart po śmierci lub zwycięstwie.

Błędy podstawowe

1. Błąd który powoduje że postać ginie na samym początku – należy zainicjować w metodzie Start życie postaci ponieważ w przeciwnym razie ginie ona od razu.
2. Wprowadzono komendę **Destroy(this)** która usuwa Player **Controller** na początku rozgrywki.
3. Skrypt Bullet nie zawiera komendy określającej co ma się dziać kiedy pocisk znajdzie się w odległości 1 jednostki od Collidera.
4. W scrypcie Bullet znajduje się using System.Security.Cryptography mimo, iż nie jest stosowane szyfrowanie.
5. Metoda setDirection powinna być nazwana SetDirection według reguł nazewnictwa.
6. W Skrypcie Player Controller są:

using System.Collections;

using System.Collections.Generic;

using System.Security.Cryptography;

using UnityEditor;

mimo, że nie są rzeczywiście używane.

1. Jeden przeciwnik ma Rigidbody, ale nie ma Tagu Enemy, drugi ma Tag Enemy ale żaden z nich nie ma Collidera
2. W Player Controller float health nie jest używany spoza skryptu więc nie musi być publiczny.
3. Używane jest gameObject.tag == x zamiast gameObject.CompareTag(„x”)
4. Zamiast wpisywać stałą wartość odległości od Collidera w skrycie PlayerController lepiej użyć zmiennej.
5. Zrefaktorowano skrypt PlayerLook. Nie ma potrzeby wprowadzania zmiennej Transform dla Playera ponieważ wystarczy użycie komendy **transform** by wykryć transform obiektu, do którego doczepiony jest dany komponent.
6. Nie ma możliwości użycia umiejętności przypisanej do prawego przycisku myszki ponieważ zakodowano na stałe wartości z których wynika, że postać zginie zanim będzie mogła wykorzystać tą umiejętność.
7. W chwili obecnej nie ma obiektów umożliwiający leczenie a zatem nie jest potrzebny kod dotyczący leczenia. Został on wykomentowany.
8. Rotację x (czyli góra dół) można zablokować w przedziale -90, 90 używając funkcji xRot = Mathf.Clamp(xRot, -90, 90);