```cpp
// Fill out your copyright notice in the Description page of Project Settings.

#pragma once

#include "CoreMinimal.h"
#include "GameFramework/Character.h"
#include "enemy1.generated.h"
UENUM(BlueprintType)
enum class EEnemyMovementStatus :uint8
{
    EMS_Idle UMETA(DeplayName = "Idle"),
    EMS_MoveToTarget UMETA(DeplayName = "MoveToTarget"),
    EMS_Attacking UMETA(DeplayName = "Attacking"),
    EMS_MAX UMETA(DeplayName = "MAX"),
    EMS_Dead UMETA(DeplayName = "Dead")
};

UCLASS()
class PARK_API Aenemy1 : public ACharacter
{
    GENERATED_BODY()

public:
    // Sets default values for this character's properties
    Aenemy1();
    UPROPERTY(VisibleAnywhere, BlueprintReadonly, Category = "Movement")
        EEnemyMovementStatus EnemyMovementStatus;
    FORCEINLINE void SetEnemyMovementStatus(EEnemyMovementStatus Status) { EnemyMovementStatus=Status; }
    FORCEINLINE EEnemyMovementStatus GetEnemyMovementStatus() { return EnemyMovementStatus; }


    UPROPERTY(VisibleAnywhere,BlueprintReadOnly,category="AI")
    class USphereComponent* Agrosphere;

    UPROPERTY(VisibleAnywhere, BlueprintReadOnly, category = "AI")
    USphereComponent* Combatsphere;

    UPROPERTY(VisibleAnywhere, BlueprintReadOnly, category = "AI")
    class AAIController* Aicontroller;

    UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "Combat")
        TSubclassOf<UDamageType>DamageTypeclass;
    UPROPERTY(EditAnywhere, BlueprintReadWrite, category = "AI")
        float Health;
    UPROPERTY(EditAnywhere, BlueprintReadWrite, category = "AI")
        float MaxHealth;
    UPROPERTY(EditAnywhere, BlueprintReadWrite, category = "AI")
        float Damage;

```

```cpp
	bool bhasvalidtarget;

	FTimerHandle AttackTimer;

	UPROPERTY(EditAnywhere,BlueprintReadOnly,Category="Combat")
	float AttackMinTime;
	UPROPERTY(EditAnywhere, BlueprintReadOnly, Category = "Combat")
	float AttackMaxTime;
	FTimerHandle DeathTimer;
	UPROPERTY(EditAnywhere,BlueprintReadWrite,Category="Combat")
	float DeathDelay;

	void disappear();

	UPROPERTY(VisibleAnywhere,BlueprintReadWrite,Category="Combat")
		class UBoxComponent* CombatCollision;

	UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "AI")
		class UParticleSystem* HitParticles;
	UPROPERTY(EditAnywhere, BlueprintReadWrite, Category = "AI")
		class USoundCue* HitSound;

	UFUNCTION(BlueprintCallable)
	void Deathend();

	void die(AActor *Causer);

	virtual float TakeDamage(float DamageAmount, struct FDamageEvent const& DamageEvent, class AController* EventInstigator, AActor* DamageCauser) override;

	bool alive();
protected:
	// Called when the game starts or when spawned
	virtual void BeginPlay() override;

public:
	// Called every frame
	virtual void Tick(float DeltaTime) override;

	// Called to bind functionality to input
	virtual void SetupPlayerInputComponent(class UInputComponent* PlayerInputComponent) override;

	UFUNCTION()
		virtual void AgroSphereOnOverLapbegin(UPrimitiveComponent* OverlappedComponent, AActor* OtherActor, UPrimitiveComponent* OtherComp, int32 otherbodyindex, bool bFromSweep, const FHitResult& SweepResult);
	UFUNCTION()
		virtual void AgroSphereOnOverLapend(UPrimitiveComponent* OverlappedComponent, AActor* OtherActor, UPrimitiveComponent* OtherComp, int32 otherbodyindex);
```

```cpp

    UFUNCTION()
        virtual void CombatSphereOnOverLapbegin(UPrimitiveComponent* OverlappedComponent, AActor* OtherActor, UPrimitiveComponent* OtherComp, int32 otherbodyindex, bool bFromSweep, const FHitResult& SweepResult);
    UFUNCTION()
        virtual void CombatSphereOnOverLapend(UPrimitiveComponent* OverlappedComponent, AActor* OtherActor, UPrimitiveComponent* OtherComp, int32 otherbodyindex);

    UFUNCTION(BlueprintCallable)
    void MoveToTarget(class AMaincharacter* Target);

    UPROPERTY(VisibleAnywhere,BlueprintReadWrite,Category="AI")
    bool bOverlappingCombatSphere;


    UPROPERTY(VisibleAnywhere,BlueprintReadWrite,Category="AI")
    AMaincharacter* CombatTarget;

    UPROPERTY(EditAnyWhere,BlueprintReadWrite,Category="Combat")
    class UAnimMontage* CombatMontage;

    UFUNCTION()
        void combatOnOverLapbegin(UPrimitiveComponent* OverlappedComponent, AActor* OtherActor, UPrimitiveComponent* OtherComp, int32 otherbodyindex, bool bFromSweep, const FHitResult& SweepResult);

    UFUNCTION()
        void combatOnOverLapend(UPrimitiveComponent* OverlappedComponent, AActor* OtherActor, UPrimitiveComponent* OtherComp, int32 otherbodyindex);

    UFUNCTION(Blueprintcallable)
        void activatecollision();

    UFUNCTION(Blueprintcallable)
        void deactivatecollision();



    void attack();
    UFUNCTION(Blueprintcallable)
    void attackend();

    UPROPERTY(visibleAnywhere,BlueprintReadOnly,Category="Combat")
    bool bAttacking;
};

```