//Kevin Kangethe

//Part B: Minimal Coding Example

#include <iostream>

#include <string>

using namespace std;

//Prive creature class holding string(name), and float (health)

class Creature{

private:

string name;

float health;

public:

//Constructor to creat Creature

Creature(string creatureName, float creatureHP)

: name(creatureName), health(creatureHP){

//Display the creation of of creation and health point

cout << "Creature " << name << "has been created with "

<< health << " health points \n" << endl;

}

//Destructor

~Creature (){

//Display the destrution of the creature clas

cout << "Creature " << name << "is being destoryed \n" << endl;

}

//Display the name and health

void display () const {

cout << "Create Name " << name << " Health: " << health << endl;

}

};

int main (){

Creature creature1 ("Godzilla " , 100000); //Create Goblin object

creature1.display(); // Object will be destoryed

return 0;

}

The Creature class uses a constructor to initialize the creature's name and health when an object is created, and it displays a message confirming creation. A destructor is used to print a message when the object is destroyed, which happens automatically when it goes out of scope at the end of the main() function. This demonstrates basic object lifecycle management in C++, where constructors initialize and destructors clean up.