using System;

using System.Collections.Generic;

using System.IO;

using System.Runtime.Serialization.Formatters.Binary;

namespace SamuraiGame

{

[Serializable]

public class Inventory

{

public Dictionary<string, int> Items = new Dictionary<string, int>();

public Inventory()

{

Items["food"] = 0;

Items["medicine"] = 0;

Items["weapon"] = 0;

}

public void AddItem(string item, int quantity = 1)

{

if (Items.ContainsKey(item))

{

Items[item] += quantity;

}

else

{

Console.WriteLine("Invalid item.");

}

}

public bool UseItem(string item)

{

if (Items.ContainsKey(item) && Items[item] > 0)

{

Items[item]--;

return true;

}

return false;

}

}

[Serializable]

public class Hero

{

public string Name;

public string ClassType;

public int Health;

public int Mana;

public int Level;

public int Experience;

public int Attack;

public int Defense;

public Inventory Inventory;

public int MaxHealth;

public Hero(string name, string classType)

{

Name = name;

ClassType = classType;

Health = 100;

Mana = 100;

Level = 1;

Experience = 0;

Inventory = new Inventory();

MaxHealth = 100;

if (classType == "warrior")

{

Attack = 20;

Defense = 15;

}

else if (classType == "wizard")

{

Attack = 15;

Mana = 150;

}

else if (classType == "archer")

{

Attack = 18;

Defense = 10;

}

}

public void LevelUp()

{

if (Experience >= 100 \* Level)

{

Level++;

Experience = 0;

Attack += 5;

Defense += 3;

MaxHealth += 20;

Health = MaxHealth;

Console.WriteLine($"{Name} leveled up to level {Level}!");

}

}

public void TakeDamage(int damage)

{

Health -= Math.Max(0, damage - Defense);

if (Health <= 0)

Health = 0;

Console.WriteLine($"{Name} took {damage} damage. Health: {Health}/{MaxHealth}");

}

public void Heal(int amount)

{

Health = Math.Min(MaxHealth, Health + amount);

Console.WriteLine($"{Name} healed {amount} health. Health: {Health}/{MaxHealth}");

}

public void AttackEnemy(Enemy enemy)

{

Random random = new Random();

int damage = random.Next(Attack - 5, Attack + 5);

if (random.NextDouble() < 0.1) // 10% chance for critical hit

{

damage \*= 2;

Console.WriteLine("Critical hit!");

}

Console.WriteLine($"{Name} attacks {enemy.Name} for {damage} damage!");

enemy.TakeDamage(damage);

}

}

[Serializable]

public class Enemy

{

public string Name;

public int Health;

public int Attack;

public int Defense;

public Enemy(string name, int health, int attack, int defense)

{

Name = name;

Health = health;

Attack = attack;

Defense = defense;

}

public void TakeDamage(int damage)

{

Health -= Math.Max(0, damage - Defense);

Console.WriteLine($"{Name} took {damage} damage. Health: {Health}");

}

public void AttackHero(Hero hero)

{

Random random = new Random();

int damage = random.Next(Attack - 5, Attack + 5);

hero.TakeDamage(damage);

}

}

[Serializable]

public class NPC

{

public string Name;

public string Dialogue;

public NPC(string name, string dialogue)

{

Name = name;

Dialogue = dialogue;

}

public void Talk()

{

Console.WriteLine($"{Name}: {Dialogue}");

}

}

public class Game

{

public Hero PlayerHero;

public void Start()

{

Console.WriteLine("Enter your hero's name:");

string name = Console.ReadLine();

Console.WriteLine("Choose your class (warrior, wizard, archer):");

string classChoice = Console.ReadLine().ToLower();

PlayerHero = new Hero(name, classChoice);

bool playing = true;

while (playing)

{

DisplayMenu();

string choice = Console.ReadLine();

switch (choice)

{

case "1":

FightEnemy();

break;

case "2":

UsePotion();

break;

case "3":

TalkToNPC();

break;

case "4":

SaveGame();

break;

case "5":

playing = false;

break;

default:

Console.WriteLine("Invalid option.");

break;

}

}

}

public void DisplayMenu()

{

Console.WriteLine($"Level: {PlayerHero.Level} | Health: {PlayerHero.Health} | Mana: {PlayerHero.Mana}");

Console.WriteLine("1. Attack an enemy");

Console.WriteLine("2. Use healing potion");

Console.WriteLine("3. Talk to NPC");

Console.WriteLine("4. Save game");

Console.WriteLine("5. Exit");

Console.Write("Choose an action: ");

}

public void FightEnemy()

{

Enemy enemy = new Enemy("Bandit", 50, 10, 2);

PlayerHero.AttackEnemy(enemy);

}

public void UsePotion()

{

if (PlayerHero.Inventory.UseItem("medicine"))

{

PlayerHero.Heal(50);

}

else

{

Console.WriteLine("You don't have any medicine.");

}

}

public void TalkToNPC()

{

NPC npc = new NPC("Old Man", "Seek the sword of the ancient samurai!");

npc.Talk();

}

public void SaveGame()

{

FileStream fs = new FileStream($"{PlayerHero.Name}\_save.dat", FileMode.Create);

BinaryFormatter formatter = new BinaryFormatter();

formatter.Serialize(fs, PlayerHero);

fs.Close();

Console.WriteLine("Game saved!");

}

public static Hero LoadGame(string fileName)

{

if (File.Exists(fileName))

{

FileStream fs = new FileStream(fileName, FileMode.Open);

BinaryFormatter formatter = new BinaryFormatter();

Hero loadedHero = (Hero)formatter.Deserialize(fs);

fs.Close();

Console.WriteLine("Game loaded!");

return loadedHero;

}

else

{

Console.WriteLine("No saved game found.");

return null;

}

}

}

class Program

{

static void Main(string[] args)

{

Game game = new Game();

game.Start();

}

}

}