НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КПІ ім. Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра програмного забезпечення комп’ютерних систем

ЛАБОРАТОРНА РОБОТА №1

з дисципліни

«Об’єктно-орієнтоване програмування»

ТЕМА: «C# .Net. Основні синтаксичні конструкції мови програмування C#»

Підготував: студент групи КП-52

Сорочинський Владислав

Перевірила:

Заболотня Тетяна Миколаївна

Київ 2016

**Мета роботи**

Ознайомитися з основами об’єктного підходу у мові С#, створенням класів, об’єктів, механізмами інкапсуляції, наслідування та поліморфізму.

**Постановка задачі**

Побудувати ієрархію класів що відтворюватимуть відношення наслідування між об’єктами реального світу. (Кількість класів >= 5). При цьому забезпечити реалізацію таких механізмів ООП:

1. Наявність у класах статичних даних. Забезпечити коректну ініціалізацію цих даних.
2. Наявність у класах змінних та методів з різними модифікаторами доступу.
3. Наявність у класах властивостей.
4. Конструктори. Створити для розроблюваних класів такі конструктори:
   * конструктор за замовчанням;
   * перевизначений конструктор;
   * приватний конструктор;
   * статичний конструктор.
5. Продемонструвати, яким чином викликаються конструктори базового та дочірнього класів.
6. Використати віртуальні та перевизначені методи.
7. Продемонструвати поліморфізм методів

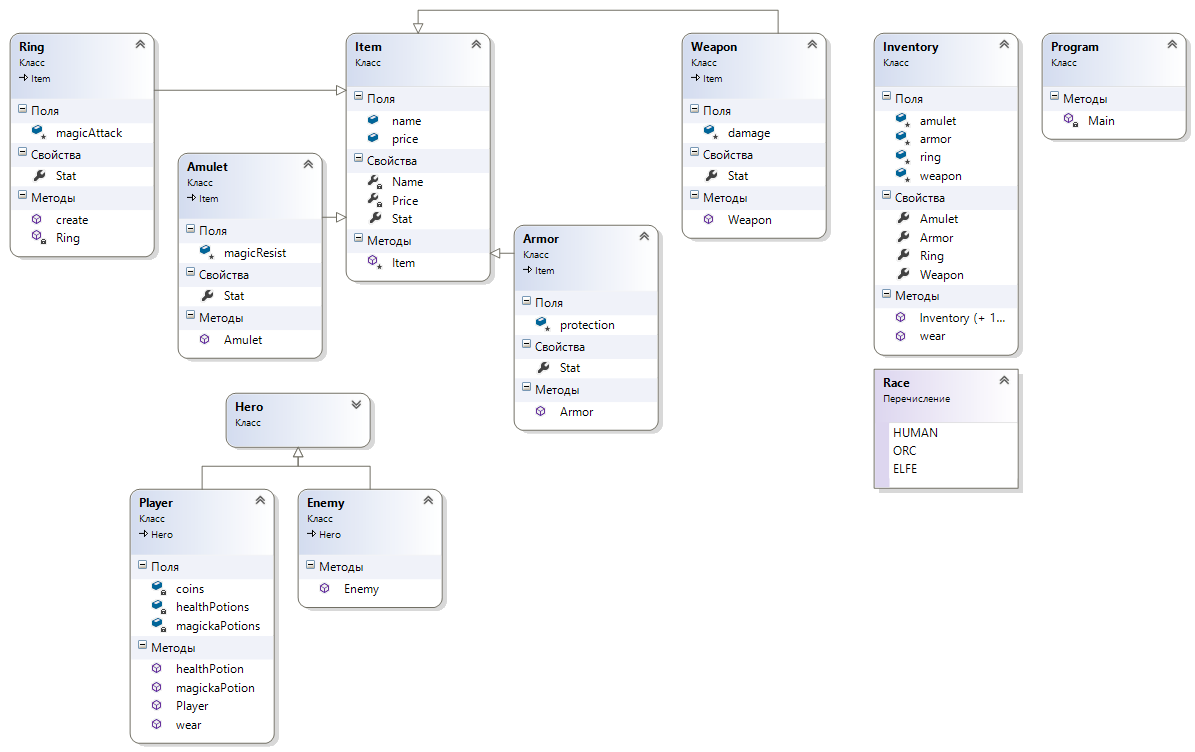
**Лістинг програми**

Класи Hero, Enemy, Player

|  |
| --- |
| Hero.cs |
| ﻿using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace lab1  {  public enum Race  {  HUMAN = 0,  ORC,  ELFE  }  public class Hero  {  public string Name;  protected Race race;  public double health, magicka, weaponDamage, magicDamage;  protected int maxHealth, maxMagicka;  public Inventory inventory;  protected static int level;  static Hero()  {  level = 1;  }  public Hero(string name, Race race)  {  this.Name = name;  this.race = race;  if (race == Race.HUMAN)  {  this.maxHealth = 35;  this.maxMagicka = 30;  this.weaponDamage = 6;  this.magicDamage = 5;  }  if (race == Race.ORC)  {  this.maxHealth = 35;  this.maxMagicka = 25;  this.weaponDamage = 7;  this.magicDamage = 4;  }  if (race == Race.ELFE)  {  this.maxHealth = 30;  this.maxMagicka = 35;  this.weaponDamage = 4;  this.magicDamage = 7;  }  this.health = this.maxHealth;  this.magicka = this.maxMagicka;  }  public virtual void levelUp(){level++;}  public virtual void meleeAttack(Hero target){  return;  }  public virtual void magicAttack(Hero target)  {  return;  }  }  public class Player: Hero  {  int coins;  int healthPotions;  int magickaPotions;  public Player (string name, Race race): base (name, race)  {  inventory = new Inventory();  healthPotions = 3;  magickaPotions = 3;  coins = 100;  }  public void wear (Item item) {  inventory.wear(item);  }  public void healthPotion()  {  if (healthPotions > 0)  health += 15;  if (health > maxHealth)  health = maxHealth;  }  public void magickaPotion()  {  if (magickaPotions > 0)  magicka += 15;  if (magicka > maxMagicka)  magicka = maxMagicka;  }  }  public class Enemy: Hero  {  public Enemy(string name, Race race)  : base(name, race)  {  Weapon weapon = new Weapon("Sword", 50, 1.15);  Armor armor = new Armor("Armor", 30, 0.9);  Ring ring = Ring.create("Habr la bra");  Amulet amulet = new Amulet("Amulet", 45, 0.8);  inventory = new Inventory(weapon, armor, ring, amulet);  }  }  } |

Класи Weapon, Armor, Ring, Amulet успадковуються від Item і містять віртуальну властивість.

|  |
| --- |
| Inventory.cs |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace lab1  {  public class Inventory  {  protected Weapon weapon;  protected Armor armor;  protected Ring ring;  protected Amulet amulet;  public Weapon Weapon {  get { return weapon; }  set { if (value.name != "empty")  weapon = value; }  }  public Armor Armor  {  get { return armor; }  set { if (value.name != "empty")  armor = value; }  }  public Ring Ring  {  get { return ring; }  set { if (value.name != "empty")  ring = value; }  }  public Amulet Amulet  {  get { return amulet; }  set { if (value.name != "empty")  amulet = value; }  }  public Inventory ()  {  weapon = new Weapon ("empty", 0, 1);  armor = new Armor("empty", 0, 1);  ring = Ring.create("null");  amulet = new Amulet("empty", 0, 1);  }  public Inventory(Weapon weapon, Armor armor, Ring ring, Amulet amulet)  {  this.weapon = weapon;  this.armor = armor;  this.ring = ring;  this.amulet = amulet;  }  public void wear(Item item)  {  if (item is Weapon) weapon = (Weapon) item;  if (item is Armor) armor = (Armor) item;  if (item is Ring) ring = (Ring)item;  if (item is Amulet) amulet = (Amulet)item;  }  }  public class Item  {  public string name;  public int price;  protected Item(string name, int price)  {  this.name = name;  if (price >= 0)  this.price = price;  else {  Console.WriteLine("Error, unavailable price");  return;  }  }  int Price  {  get { return price; }  }  string Name  {  get { return name; }  }  public virtual double Stat { get; set; }  }  public class Weapon: Item  {  protected double damage;  public Weapon(string name, int price, double damage): base (name, price)  {  this.damage = damage;  }    public override double Stat  {  get {return this.damage;}  }  }  public class Armor: Item  {  protected double protection;  public Armor(string name, int price, double protection): base(name, price)  {  this.protection = protection;  }  public override double Stat  {  get { return this.protection; }  }  }  public class Ring: Item  {  protected double magicAttack;  private Ring(string name, int price, double magicAttack)  : base(name, price)  {  this.magicAttack = magicAttack;  }  public static Ring create(string spell) {  if (spell == "null")  return new Ring("empty", 0, 1);  if (spell == "Habr la bra")  return new Ring("Aprentice ring", 20, 1.1);  else Console.WriteLine("You don`t know a right spell, mortal\n");  return null;  }  public override double Stat  {  get { return magicAttack; }  }  }  public class Amulet: Item{  protected double magicResist;  public Amulet(string name, int price, double magicResist)  : base(name, price)  {  this.magicResist = magicResist;  }  public override double Stat  {  get { return magicResist; }  }  }  } |

**Діаграма класів**