**Лабораторна робота №1**

**Тема:** Вивчення породжуючих шаблонів проектування.

**Мета:** Набути практичних навичок в використанні породжуючих шаблонів використання.

***Завдання:***

Вивчити та реалізувати шаблони проектування згідно варіанту. Оформити звіт, у якому описати предметну область та проблему, яку вирішуватимете за допомогою шаблона проектування, здійснити опис шаблона проектування, діаграма класів шаблону, реалізація та приклад використання програми(результати роботи).

***Варіант 6***

* Шаблон прототип.
* Шаблом фабричного методу.

***Хід роботи***

***Фабричний метод – використовується для того щоб швидко створювати об’єкти класів. У випадку мови C# створюэться за допомогою використання інтерфейсів. Код:***

interface Developer

{

void WritesCode();

}

interface DeveloperFactroy

{

Developer createDeveloper();

}

class CppDeveloperFactory : DeveloperFactroy

{

public Developer createDeveloper()

{

return new CppDeveloper();

}

}

class JavaDeveloperFactory : DeveloperFactroy

{

public Developer createDeveloper()

{

return new JavaDeveloper();

}

}

class CppDeveloper : Developer

{

public void WritesCode()

{

Console.WriteLine("C++ developer writes c++ code...");

}

}

class JavaDeveloper : Developer

{

public void WritesCode()

{

Console.WriteLine("Java developer writes Java code...");

}

}

class Program

{

static bool EquelsIgnoreCase(String a,String b)

{

return String.Equals(a, b, StringComparison.OrdinalIgnoreCase);

}

static DeveloperFactroy createDeveloperBylang(String lang)

{

if (EquelsIgnoreCase(lang, "java")) { return new JavaDeveloperFactory(); }

else if (EquelsIgnoreCase(lang, "c++")) { return new CppDeveloperFactory(); }

else { throw new Exception(lang+" is unknown lang"); }

}

static void Main(string[] args)

{

DeveloperFactroy df = createDeveloperBylang("c++");

Developer dev = df.createDeveloper();

dev.WritesCode();

df = createDeveloperBylang("java");

dev = df.createDeveloper();

dev.WritesCode();

Console.Read();

}

}

***Прототип – використовуєтсья для більшої зручності при роботі з об’єктами класу, основна суть – глибоке копіювання об’єкту з подальшою кастомізацією. У випадку мови C# реалізується за допомогою серіалізації, так як цей об’єкт повністью копіює граф усіх данних. Код:***

static class ExtensionMethods

{

public static T DeepCopy<T>(this T self)

{

if (!typeof(T).IsSerializable) throw new ArgumentException("Type must be serializable.");

if (ReferenceEquals(self, null)) return default(T);

var formatter = new BinaryFormatter();

using (var stream = new MemoryStream())

{

formatter.Serialize(stream, self);

stream.Seek(0, SeekOrigin.Begin);

return (T)formatter.Deserialize(stream);

};

}

}

[Serializable]

public class Stats

{

public int Power;

public int Agility;

public int Intelegence;

public override string ToString()

{

return $"Power: {Power},Agility: {Agility},Intelegence: {Intelegence}";

}

}

[Serializable]

public class Charapter

{

public string Name;

public string Gameclass;

public int Level;

public Stats Status;

public override string ToString()

{

return $"Name: {Name},Gameclass: {Gameclass},Level: {Level} " + Status.ToString();

}

}

class Program

{

static void Main(string[] args)

{

var bases = new Charapter

{

Gameclass = "Warrior", Level = 1, Status = new Stats { Agility = 5, Power =5, Intelegence =5 }

};

var pristochka = bases.DeepCopy();

pristochka.Name = "Pristo4ka";

var Vasya\_Lol = bases.DeepCopy();

Vasya\_Lol.Name = "Vasya\_Lol";

var This\_IS\_BURIIIICH = bases.DeepCopy();

This\_IS\_BURIIIICH.Name = "This\_IS\_BURIIIICH";

This\_IS\_BURIIIICH.Gameclass = "SPARTANEC!";

var Sanyok\_ok\_ok = bases.DeepCopy();

Sanyok\_ok\_ok.Name = "Sanyok\_ok\_ok";

Sanyok\_ok\_ok.Gameclass = "Archer!";

Sanyok\_ok\_ok.Level = 25;

Console.WriteLine(pristochka);

Console.WriteLine(Vasya\_Lol);

Console.WriteLine(This\_IS\_BURIIIICH);

Console.WriteLine(Sanyok\_ok\_ok);

Console.Read();

}

}

Висновок: На цій лабораторній я набув практичних навичок в роботі з породжуючими шаблонами проектування. Та закріпив навички програмування на мові С#.