**ПРАКТИЧНА РОБОТА № 8**

**«ІНТЕРФЕЙСИ»**

**ВАРІАНТ №5**

**Мета**: Закріпити вміння реалізовувати класи, методи та конструктори. Навчитися створювати інтерфейси.

Ієрархія для завдання: Організація, страхова компанія, нафтогазова компанія, завод.

Повний лістинг коду:

using System;

using System.Collections.Generic;

namespace prac3\_console

{

public interface IInfo

{

public void GetInfo();

}

public interface IPayday

{

public void Pay();

}

public abstract class Organization

{

public string name;

public string industry;

public string area\_served;

public Organization(string name, string industry, string area\_served)

{

this.name = name;

this.industry = industry;

this.area\_served = area\_served;

}

}

public class Factory : Organization, IPayday, IInfo

{

public string head;

public int number\_of\_employees;

public string city;

public void GetInfo()

{

Console.WriteLine($"Factory: \n\tName of organisation: {name}");

Console.WriteLine("\tName: {0}\n\tHead: {1}\n\tNumber of employees: {2}\n\tCity: {3}\n",

name, head, number\_of\_employees, city);

}

public void Pay()

{

Console.WriteLine($"{number\_of\_employees} employees was paid their salary");

}

public Factory(string name, string industry, string area\_served, string head, string city, int number\_of\_employees) : base(name, industry, area\_served)

{

this.head = head;

this.city = city;

this.number\_of\_employees = number\_of\_employees;

}

}

public class Insurance : Organization, IInfo

{

public int client\_number;

public string speciality;

public void GetInfo()

{

Console.WriteLine($"Insurance: \n\tName of organisation: {name}");

Console.WriteLine("\tNumber of clients: {0}\n\tSpecialization: {1}\n", client\_number, speciality);

}

public Insurance(string name, string industry, string area\_served, int client\_number, string speciality) : base(name, industry, area\_served)

{

this.client\_number = client\_number;

this.speciality = speciality;

}

}

public class BuildingCompany : Organization, IPayday, IInfo

{

public string speciality;

public string city;

public int number\_of\_employees;

public void GetInfo()

{

Console.WriteLine($"Building Company: \n\tName of organisation: {name}");

Console.WriteLine("\tName: {0}\n\tSpeciality: {1}\n\tNumber of employees: {2}\n\tCity: {3}\n",

name, speciality, number\_of\_employees, city);

}

public void Pay()

{

Console.WriteLine($"{number\_of\_employees} employees was paid their salary");

}

public BuildingCompany(string name, string industry, string area\_wide, string speciality, string city, int number\_of\_employees) : base(name, industry, area\_wide)

{

this.speciality = speciality;

this.city = city;

this.number\_of\_employees = number\_of\_employees;

}

}

internal class Program

{

static void Main(string[] args)

{

Action action = new Action(() => Console.WriteLine("Delegate work started\n"));

Factory f = new Factory("Oil Co.", "oil", "Country", "Popolski", "Kyiv", 900);

BuildingCompany bc = new BuildingCompany("Building Co.", "skys", "City", "Houses", "Odesa", 1900);

Insurance ins = new Insurance("Insurance Co. ", "Money", "City", 300, "Cars");

action += f.Pay;

action += f.GetInfo;

action += bc.Pay;

action += bc.GetInfo;

action += ins.GetInfo;

action.Invoke();

}

}

}

Описано 2 інтерфейси IInfo та IPayday . У першому створений метод GetInfo() у другому Pay().

public interface IInfo

{

public void GetInfo();

}

public interface IPayday

{

public void Pay();

}

Клас Organization є абстрактним класом. В ньому описано декілька змінних для організацій та конструктор за цими змінними.

public abstract class Organization

{

public string name;

public string industry;

public string area\_served;

public Organization(string name, string industry, string area\_served)

{

this.name = name;

this.industry = industry;

this.area\_served = area\_served;

}

}

Клас Factory наслідує клас Organization та реалізовує інтерфейси IPayday, IInfo. Викликає обидва методи цих інтерфейсів та має конструктор базований на констрикторі основного класу.

public class Factory : Organization, IPayday, IInfo

{

public string head;

public int number\_of\_employees;

public string city;

public void GetInfo()

{

Console.WriteLine($"Factory: \n\tName of organisation: {name}");

Console.WriteLine("\tName: {0}\n\tHead: {1}\n\tNumber of employees: {2}\n\tCity: {3}\n",

name, head, number\_of\_employees, city);

}

public void Pay()

{

Console.WriteLine($"{number\_of\_employees} employees was paid their salary");

}

public Factory(string name, string industry, string area\_served, string head, string city, int number\_of\_employees) : base(name, industry, area\_served)

{

this.head = head;

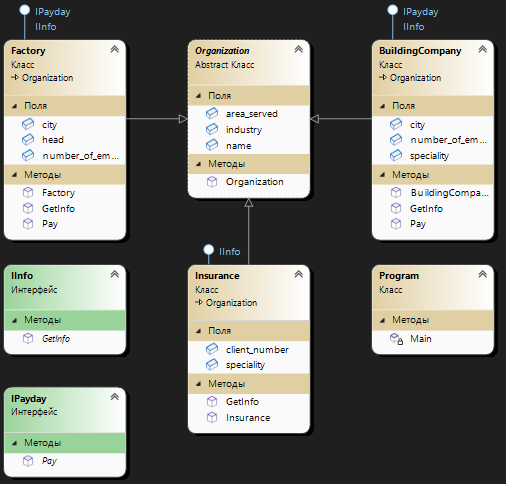
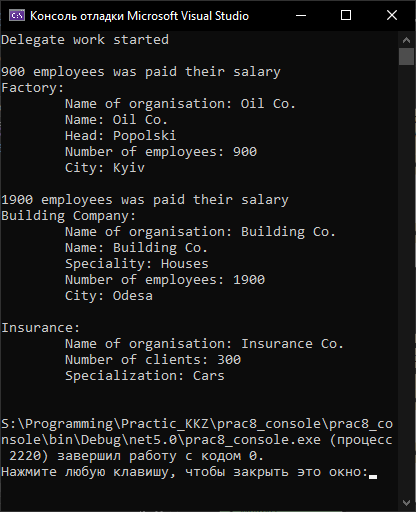
this.city = city;

this.number\_of\_employees = number\_of\_employees;

}

}

Класи BuildingCompany та Insurance мають таку ж реалізацію як і клас Factory.



**Висновок:** ми закріпили вміння реалізовувати класи, методи та конструктори. Навчилися створювати інтерфейси.