## Лабораторная работа 4. Абстрактные классы и интерфейсы

**Конструктор класса – это специальный метод, который создает экземпляр класса, называется как класс и не имеет типа возвращаемого значения.**

Например:

classCoOrds

{

publicint x, y;

// конструктор

publicCoOrds()

{

x = 0;

y = 0;

}

// конструктор с двумя аргументами

publicCoOrds(intx, inty)

{

this.x = x;

this.y = y;

}

}

staticvoid Main()

{

CoOrdsp1 = newCoOrds();// определениеэкземпляракласса

CoOrds p2 = newCoOrds(5, 3);

// выводрезультатов

Console.WriteLine("CoOrds #1 at {0}", p1);

Console.WriteLine("CoOrds #2 at {0}", p2);

Console.ReadKey();

}

**ЗАДАНИЕ**

**#4A.**

Создайте класс Angle, включающий следующие три поля:

* типа int для числа градусов,
* типа float для числа минут
* типа char для указания направления (N, S, E или W).

Создайте конструктор класса по умолчанию (без параметров) со значениями 0,0, ‘S’.

Создайте конструктор класса, который принимает три аргумента.

Создайте метод, позволяющий ввести координату точки, направление, в котором она измеряется.

Создайте метод, выводящий на экран значение этой координаты.

В функции main() определите экземпляр (объект) класса вызовом конструктора с параметрами и выведите значение экземпляра на экран.

В функции main() определите экземпляр (объект) класса вызовом конструктора без параметров, вызовите метод, запрашивающий значение координаты; вызовите метод для отображения введенного значения на экране.

Выполнение:

Код программы:

using System;

namespace Program

{

    public class Angle

    {

        int angles;

        float minutes;

         char napravlenie;

        public Angle()

        {

            angles = 0;

            minutes = 0;

            napravlenie = 'S';

        }

        public Angle(int angles, float minutes, char napravlenie)

        {

            this.angles = angles;

            this.minutes= minutes;

            this.napravlenie= napravlenie;

        }

        public void EnterCoords()

        {

            try{

                System.Console.Write("Enter angles here: ");

                int.TryParse(Console.ReadLine(), out angles);

                System.Console.Write("Enter minutes here: ");

                float.TryParse(Console.ReadLine(), out  minutes);

                System.Console.Write("Enter napravlenie here: ");

                char.TryParse(Console.ReadLine(), out napravlenie);

            }

            catch

            {

                System.Console.WriteLine("error");

            }

        }

        public override string ToString()

        {

            return $"angle: {angles}\nminutes: {minutes}\nnapravlenie: {napravlenie}";

        }

    }

    public class Lab\_4

    {

        public static void Main()

        {

            Angle a = new Angle(12,10,'S');

            System.Console.WriteLine(a);

            Angle b = new Angle();

            b.EnterCoords();

            System.Console.WriteLine(b);

        }

    }

}

Вывод на консоль:

Изображение выглядит как текст

Автоматически созданное описание

**#4B.**

Постройте UML-диаграмму классов и создайте проект, в котором содержатся (рис.1):

1. Абстрактный класс с именем HotDrink, у которого имеются

* методы типа stringDrink() (выпить), AddMilk() (добавить молоко), AddSugar() (добавить сахар);
* защищенные поля sugar, milk (значения по умолчанию);
* свойства для этих полейMilk и Sugar.

2. Интерфейс с именем ICup, у которого имеются

* методы Refill() (наполнить повторно), Wash() (вымыть),
* свойства Type (тип) и Capacity (Объем).

3. Класс с именем CupOfCoffee, который является наследником класса HotDrink, поддерживает интерфейс ICup и обладает дополнительным свойством BeanType (тип зерен).

4. Класс с именем CupOfTea, который является производным от класса HotDrink, поддерживает интерфейс ICup и обладает дополнительным свойством LeafType (тип чая).

5. Класс с функцией Main(), а также с функциейProcessCup, которая получает в качестве параметра объект одного из классов CupOfCoffee или CupOfTea, по выбору пользователя (чай или кофе). Далее функция должна вызывать все методы и свойстваабстрактного класса и интерфейса для любого объекта, переданного ей в качестве параметра (т.е. в качестве формального параметра используется абстрактный тип).

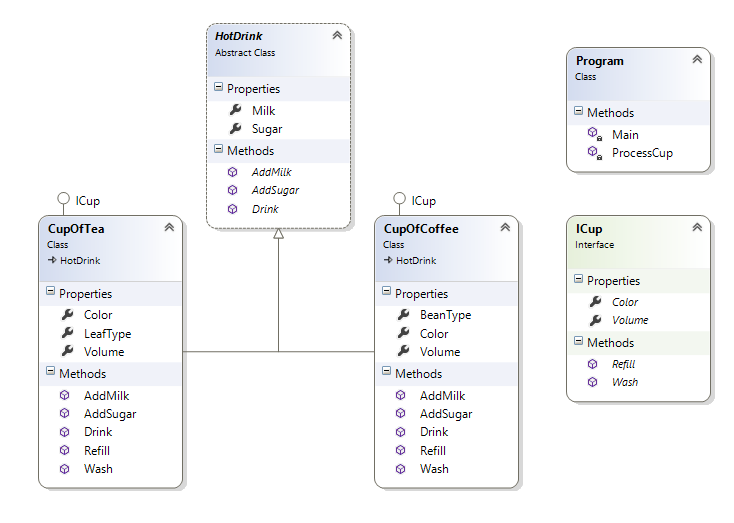


Рис.1. Диаграмма классов проекта

Пример вывода на консоль:

* красным обозначен ввод значений с клавиатуры;
* синим обозначен вывод работы переопределенных абстрактных методов и свойств;
* зеленым обозначен вывод работы реализованных методов и свойств интерфейса.

Изображение выглядит как текст

Автоматически созданное описание

Рис.2. Пример вывода на консоль

Выполнение:

Код программы:

using System;

namespace Program{

    public abstract class HotDrink

    {

        private int shugar = 3;

        private int milk = 3;

        public int Shugar{set{

            if ((value > 0) && (value < 10))

            {

                shugar = value;

            }

            else shugar = 3;

        }

        get{

            return shugar;

        }}

        public int Milk{set{

            if ((value > 0) && (value < 10))

            {

                milk = value;

            }

            else milk = 3;

        }

        get{

            return milk;

        }}

        public void stringDrink(){System.Console.WriteLine("Выпить");}

        public void AddMilk(int count){Milk = count;}

        public void AddSugar(int count){Shugar = count;}

    }

    public interface ICup

    {

         string Type{get;set;}

        public double Capacity{get;set;}

        void Refill();

        void Wash();

    }

    public class CupOfCoffee : HotDrink, ICup

    {

        private string type = "пластик";

        private double capacity = 0.2;

        private string beanType = "арабика";

        public string Type { get{return type;}

                            set{if ((value == "пластик") || (value == "стекло")){

                                type = value;

        }

        else type = "пластик";

            }} //ТИП СТАКАНА

        public double Capacity { get{return capacity;}

                            set{if ((value == (double)0.2) || (value == (double)0.3)){

                                capacity = value;

        }

        else capacity = 0.2;

            }} // ОБЪЕМ

        public string BeanType { get{return beanType;}

                            set{if ((value == "робуста") || (value == "арабика")){

                                beanType = value;

        }

        else beanType = "арабика";

            }} // ТИП ЗЕРЕН

        public void Refill()

        {

            System.Console.WriteLine($"Повторить кофе объемом {Capacity} мл");

        }

        public void Wash()

        {

            System.Console.WriteLine($"Вымыть {Type} чашку с кофе");

        }

        new public void stringDrink()

        {

            System.Console.WriteLine($"В кофе добавлен сахар: {Shugar}\nВ кофе добавлено молоко: {Milk}\nПолучите кофе: {BeanType}");

        }

    }

    public class CupOfTea : HotDrink, ICup

    {

        private string type = "пластик";

        private double capacity = 0.2;

        private string leafType = "черный";

        public string Type { get{return type;}

                            set{if ((value == "пластик") || (value == "стекло")){

                                type = value;

        }

        else type = "пластик";

            }} //ТИП СТАКАНА

        public double Capacity { get{return capacity;}

                            set{if ((value == (double)0.2) || (value == (double)0.3)){

                                capacity = value;

        }

        else capacity = 0.2;

            }} // ОБЪЕМ

        public string LeafType { get{return leafType;}

                            set{if ((value == "черный") || (value == "зеленый")){

                                leafType = value;

        }

        else leafType = "черный";

            }} // ТИП ЧАЯ

        public void Refill()

        {

            System.Console.WriteLine($"Повторить чай объемом {Capacity} мл");

        }

        public void Wash()

        {

            System.Console.WriteLine($"Вымыть {Type} чашку с чаем");

        }

        new public void stringDrink()

        {

            base.stringDrink();

            System.Console.WriteLine($"В чай добавлен сахар: {Shugar}\nВ чай добавлено молоко: {Milk}\nПолучите чай: {LeafType}");

        }

    }

    public class Program

    {

        public static void ProcessCup(HotDrink drink)

        {

                   Type type = drink.GetType();

            if(type.Equals(typeof(CupOfCoffee)))

            {

                CupOfCoffee coffee =  (CupOfCoffee)Convert.ChangeType(drink, typeof(CupOfCoffee));

                coffee.stringDrink();

                coffee.Refill();

                coffee.Wash();

            }

            else if (type.Equals(typeof(CupOfTea)))

            {

                CupOfTea tea=  (CupOfTea)Convert.ChangeType(drink, typeof(CupOfTea));

                tea.stringDrink();

                tea.Refill();

                tea.Wash();

            }

        }

        public static void Main(){

            System.Console.Write("Выберите напиток: кофе(1) или чай(2): ");

            int choise = Convert.ToInt32(Console.ReadLine());

            dynamic drink = null;

            if (choise == 1){

                drink = new CupOfCoffee();

                System.Console.WriteLine("\n\nТип зерен: арабика или робуста (арабика по умолч.)");

                System.Console.WriteLine("Сахар: 0 .. 5 (по умолч. 3");

                System.Console.WriteLine("Молоко: 0 .. 10 (по умолч. 3");

                System.Console.WriteLine("Тип стакана: пластик или стекло (по умолч. пластик");

                System.Console.WriteLine("Объем: 0,2 или 0,3 (по умолч. 0,2");

                System.Console.Write("\nТип зерен: ");

                drink.BeanType = Console.ReadLine();

            }

            else if(choise == 2)

            {

                drink = new CupOfTea();

                System.Console.WriteLine("\n\nТип чая: черный или зеленый (черный по умолч.)");

                System.Console.WriteLine("Сахар: 0 .. 5 (по умолч. 3");

                System.Console.WriteLine("Молоко: 0 .. 10 (по умолч. 3");

                System.Console.WriteLine("Тип стакана: пластик или стекло (по умолч. пластик");

                System.Console.WriteLine("Объем: 0,2 или 0,3 (по умолч. 0,2");

                System.Console.Write("\nТип зерен: ");

                drink.LeafType = Console.ReadLine();

            }

            else{

                Environment.Exit(0);

            }

                System.Console.Write("Молоко: ");

                int milk = Convert.ToInt32(Console.ReadLine());

                drink.AddMilk(milk);

                System.Console.Write("Сахар: ");

                int shugar = Convert.ToInt32(Console.ReadLine());

                drink.AddSugar(shugar);

                System.Console.Write("Тип стакана: ");

                drink.Type = Console.ReadLine();

                System.Console.Write("Объем (мл): ");

                drink.Capacity = Convert.ToDouble(Console.ReadLine());

            System.Console.WriteLine("--------------------------------");

            ProcessCup(drink);

        }

    }

}

Запуск программы:

Изображение выглядит как текст

Автоматически созданное описание

**#4C.**Модифицируйте приведенную ниже структуру, включив в нее функцию, которая возвращает суммарную стоимость заказа:

structorders

{

publicstringitemname; //наименование

publicintunitCount; //числоединиц

publicdoubleunitCost; //стоимостьоднойединицы

}

В Mainсоздайте объект структуры, вызовите метод и выведите значение на экран.

Выполнение:

Код программы:

using System;

namespace Program

{

    struct orders

    {

    public string itemname;  //наименование

    public int unitCount;       //числоединиц

    public double unitCost; //стоимостьоднойединицы

    public double Summ()

    {

        return  unitCost \* unitCount;

    }

}

    public class Program

    {

        public static void Main()

        {

            orders order = new orders();

            order.itemname = "ORDERRRRRRR";

            order.unitCost = 10.5;

            order.unitCount = 10;

            System.Console.WriteLine($"Ордер: {order.itemname} суммарно стоит: {order.Summ()}");

        }

    }

}

Запуск программы:

