### Кам’янець-Подільський національний університет імені Івана Огієнка

### КАФЕДРА КОМП’ЮТЕРНИХ НАУК

### Навчальна дисципліна «Об’єктно-орієнтоване програмування»

### ЛАБОРАТОРНА РОБОТА #0105

## Тема:

## Інтерфеси

Варіант №1

**Виконав**:  
студент 1-го курсу  
групи KNms1-B23  
Осадчук В. А.

**Прийняла**:  
старший викладач,  
Мястковська М.О.

Кам’янець-Подільський – 2023

1. **Короткі теоретичні відомості.**

Інтерфейс – іменований набір абстрактних членів. Інтерфейс схожий на клас, але надає лише специфікацію, а не реалізацію для своїх членів. Специфічні члени, що визначаються інтерфейсом, залежать від того, яку точно поведінку він моделює. Іншими словами, інтерфейс виражає поведінку, яку заданий клас або структура може обрати для підтримки. Інтерфейс (інтерфейсний тип) володіє наступними особливостями:

• всі члени інтерфейсу є неявно абстрактними; на противагу цьому, клас може мати як абстрактні члени, так і конкретні члени з реалізацією;

• клас (або структура) може реалізовувати декілька інтерфейсів; на противагу цьому клас може бути успадкований лише від одного класу, а структура взагалі не підтримує успадкування.

Опис інтерфейсу схожий на опис класу, але при цьому ніякої реалізації не надається, оскільки всі члени інтерфейсу неявно абстрактні. Всі ці члени обов’язково мають бути реалізовані класами і структурами, які включають даний інтерфейс.

Інтерфейс може містити лише методи, властивості, події та індексатори, що відповідає членам класу, які можуть бути абстрактними. Проте інтерфейс, що встановлюється абстрактним класом має суттєве обмеження: члени, що визначені абстрактним батьківським класом, підтримуються лише дочірніми класами. Тому класи, що не мають спільного батьківського класу, окрім System.Object, не можуть реалізовувати спільні інтерфейси. Крім того, класи можуть мати лише один батьківський клас, а реалізовувати можуть декілька інтерфейсів.

Члени інтерфейсу завжди неявно public, і для них не можна оголошувати модифікатори доступу. Реалізація інтерфейсу означає надання реалізації public для всіх його членів. Жоден з членів інтерфейсу не може бути оголошений як static.

Щоб розширити функціональність класу (або структури) за рахунок підтримки інтерфейсів, необхідно додати в його визначення список потрібних інтерфейсів, розділених комами, причому прямий батьківський клас (якщо він відмінний від System.Object) має бути першим.

Для визначення, чи підтримує даний тип той чи інший інтерфейс, можна скористатися ключовим словом as. Якщо об'єкт може бути інтерпретований як зазначений інтерфейс, то повертається посилання на інтерфейс, а якщо ні, то посилання null. Помилки часу виконання не виникне.

Завдяки тому, що інтерфейси – допустимі типи .NET, можна будувати методи, що приймають інтерфейси в якості параметрів. Інтерфейси можуть також використовуватися як значення методів, що повертаються.

1. **Повні умови завдань.**

**Задача 1:**

Описати інтерфейс IShape з членами:

1) метод, що виводить на екран тип фігури;

2) метод, що виводить на екран площу фігури;

3) властивість, що відповідає за перший лінійний розмір фігури.

Описати інтерфейс IColoredShape, що успадковується від інтерфейсу IShape та містить властивість, що відповідає за колір фігури, та метод, що виводить на екран колір фігури.

Описати клас Circle, та ColoredCircle що реалізовують інтерфейси IShape та IColoredShape відповідно.

Створити не менше трьох екземплярів кожного класу, сформувати з них один масив.

Створити метод, який для кожного елемента масиву виконує його методи в залежності від інтерфейсу, що реалізується класом об’єкта.

**Задача 2:**

Визначити інтерфейс Printable, який містить метод void print().

Визначити клас Book, який реалізує інтерфейс Printable.

Визначити клас Magazine, який реалізує інтерфейс Printable.

Створити масив типу Printable, який міститиме книги та журнали.

У циклі пройти масивом і викликати метод print() кожного об'єкта.

Створити статичний метод printMagazines(Printable[] printable) у класі Magazine, який виводить на консоль назви лише журналів.

Створити статичний метод printBooks(Printable[] printable) у класі Book, який виводить на консоль назви лише книг. Використовуємо оператор instanceof.

**Задача 3:**

Створити клас Shop, що містить масив із розмірами одягу (XXS, XS, S, M, L). Клас містить метод getDescription, який повертає рядок "Дорослий розмір".

Перевизначити метод getDescription - для константи XXS метод має повертати рядок "Дитячий розмір".

Також клас має містити числове значення euroSize (32, 34, 36, 38, 40), що відповідає кожному розміру. Створити конструктор, який приймає на вхід euroSize.

Створити інтерфейси "Чоловічий Одяг" з методом "одягти Чоловіка" та "Жіночий Одяг" з методом "одягти Жінку".

Створити абстрактний клас Одяг, що містить змінні – розмір одягу, вартість, колір.

Створити класи спадкоємці Одягу - Футболка (реалізує інтерфейси "Чоловічий Одяг" та "Жіночий Одяг"), Штани (реалізує інтерфейси "Чоловічий Одяг" та "Жіночий Одяг"), Спідниця (реалізує інтерфейси "Жіночий Одяг"), Краватка (реалізує інтерфейси " Чоловічий одяг").

Створити масив, що містить усі типи одягу.

Створити клас Ательє, що містить методи одягнути Жінку, одягнути Чоловіка, на вхід яких надходитиме масив, що містить всі типи одягу.

Метод одягу Жінку виводить на консоль всю інформацію про жіночий одяг. Те ж саме для методу одягнути Чоловіка.

1. **Лістинг програми.**

**Задача 1:**

using System;

public interface IShape

{

void DisplayType();

void DisplayArea();

double Size { get; set; }

}

public interface IColoredShape : IShape

{

string Color { get; set; }

void DisplayColor();

}

public class Circle : IShape

{

public Circle(double size)

{

Size = size;

}

public double Size { get; set; }

public void DisplayType()

{

Console.WriteLine("Це коло");

}

public void DisplayArea()

{

double area = Math.PI \* Size \* Size;

Console.WriteLine($"Площа кола: {area}");

}

}

public class ColoredCircle : IColoredShape

{

public ColoredCircle(double size, string color)

{

Size = size;

Color = color;

}

public double Size { get; set; }

public string Color { get; set; }

public void DisplayType()

{

Console.WriteLine("Це коло");

}

public void DisplayArea()

{

double area = Math.PI \* Size \* Size;

Console.WriteLine($"Площа кола: {area}");

}

public void DisplayColor()

{

Console.WriteLine($"Колір кола: {Color}");

}

}

class Program

{

static void Main()

{

IShape[] shapes = new IShape[]

{

new Circle(5.0),

new ColoredCircle(3.0, "червоний"),

new Circle(7.5),

new ColoredCircle(4.0, "синій"),

new ColoredCircle(6.0, "зелений")

};

foreach (var shape in shapes)

{

shape.DisplayType();

shape.DisplayArea();

if (shape is IColoredShape coloredShape)

{

coloredShape.DisplayColor();

}

Console.WriteLine();

}

}

}

**Задача 2:**

using System;

public interface Printable

{

void Print();

}

public class Book : Printable

{

private string title;

public Book(string title)

{

this.title = title;

}

public void Print()

{

Console.WriteLine("Книга: " + title);

}

public static void PrintBooks(Printable[] printable)

{

Console.WriteLine("Список книг:");

foreach (Printable item in printable)

{

if (item is Book)

{

item.Print();

}

}

}

}

public class Magazine : Printable

{

private string title;

public Magazine(string title)

{

this.title = title;

}

public void Print()

{

Console.WriteLine("Журнал: " + title);

}

public static void PrintMagazines(Printable[] printable)

{

Console.WriteLine("Список журналів:");

foreach (Printable item in printable)

{

if (item is Magazine)

{

item.Print();

}

}

}

}

public class Program

{

public static void Main(string[] args)

{

Printable[] printableItems = new Printable[]

{

new Book("Книга 1"),

new Magazine("Журнал 1"),

new Book("Книга 2"),

new Magazine("Журнал 2")

};

foreach (Printable item in printableItems)

{

item.Print();

}

Magazine.PrintMagazines(printableItems);

Book.PrintBooks(printableItems);

}

}

**Задача 3:**

using System;

public enum ClothingSize

{

XXS, XS, S, M, L

}

interface IMensClothing

{

void DressMan();

}

interface IWomensClothing

{

void DressWoman();

}

abstract class Clothing

{

protected ClothingSize size;

protected double price;

protected string color;

public Clothing(ClothingSize size, double price, string color)

{

this.size = size;

this.price = price;

this.color = color;

}

public abstract string Description();

}

class TShirt : Clothing, IMensClothing, IWomensClothing

{

public TShirt(ClothingSize size, double price, string color)

: base(size, price, color)

{

}

public override string Description()

{

return "Футболка: Розмір " + size + ", Колір " + color + ", Вартість " + price + " євро";

}

public void DressMan()

{

Console.WriteLine("Чоловік одягнув футболку.");

}

public void DressWoman()

{

Console.WriteLine("Жінка одягнула футболку.");

}

}

class Pants : Clothing, IMensClothing, IWomensClothing

{

public Pants(ClothingSize size, double price, string color)

: base(size, price, color)

{

}

public override string Description()

{

return "Штани: Розмір " + size + ", Колір " + color + ", Вартість " + price + " євро";

}

public void DressMan()

{

Console.WriteLine("Чоловік одягнув штани.");

}

public void DressWoman()

{

Console.WriteLine("Жінка одягнула штани.");

}

}

class Skirt : Clothing, IWomensClothing

{

public Skirt(ClothingSize size, double price, string color)

: base(size, price, color)

{

}

public override string Description()

{

return "Спідниця: Розмір " + size + ", Колір " + color + ", Вартість " + price + " євро";

}

public void DressWoman()

{

Console.WriteLine("Жінка одягнула спідницю.");

}

}

class Tie : Clothing, IMensClothing

{

public Tie(ClothingSize size, double price, string color)

: base(size, price, color)

{

}

public override string Description()

{

return "Краватка: Розмір " + size + ", Колір " + color + ", Вартість " + price + " євро";

}

public void DressMan()

{

Console.WriteLine("Чоловік одягнув краватку.");

}

}

class Shop

{

private ClothingSize[] clothingSizes;

private int euroSize;

public Shop(ClothingSize[] clothingSizes, int euroSize)

{

this.clothingSizes = clothingSizes;

this.euroSize = euroSize;

}

public string GetDescription()

{

if (euroSize == 32)

{

return "Дитячий розмір";

}

else

{

return "Дорослий розмір";

}

}

}

class Atelier

{

public void DressWoman(IWomensClothing[] womensClothing)

{

Console.WriteLine("Одягнути жінку:");

foreach (var clothing in womensClothing)

{

clothing.DressWoman();

}

}

public void DressMan(IMensClothing[] mensClothing)

{

Console.WriteLine("Одягнути чоловіка:");

foreach (var clothing in mensClothing)

{

clothing.DressMan();

}

}

}

class Program

{

static void Main(string[] args)

{

ClothingSize[] clothingSizes = { ClothingSize.XXS, ClothingSize.XS, ClothingSize.S, ClothingSize.M, ClothingSize.L };

Shop shop = new Shop(clothingSizes, 36);

TShirt tShirt = new TShirt(ClothingSize.M, 20.0, "Чорний");

Pants pants = new Pants(ClothingSize.L, 40.0, "Сірий");

Skirt skirt = new Skirt(ClothingSize.S, 30.0, "Рожевий");

Tie tie = new Tie(ClothingSize.M, 15.0, "Синій");

IWomensClothing[] womensClothing = { tShirt, skirt };

IMensClothing[] mensClothing = { tShirt, pants, tie };

Atelier atelier = new Atelier();

atelier.DressWoman(womensClothing);

atelier.DressMan(mensClothing);

Console.WriteLine("Розмір одягу в магазині: " + shop.GetDescription());

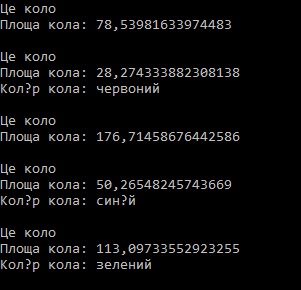
Console.ReadLine();

}

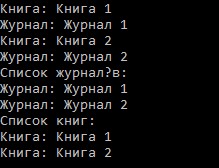
}

1. **Результати роботи програм (копії вікон програми).**

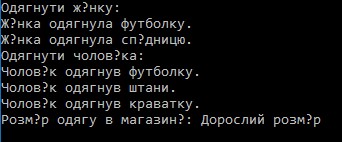
**Задача 1:**

****

**Задача 2:**

****

**Задача 3:**

****