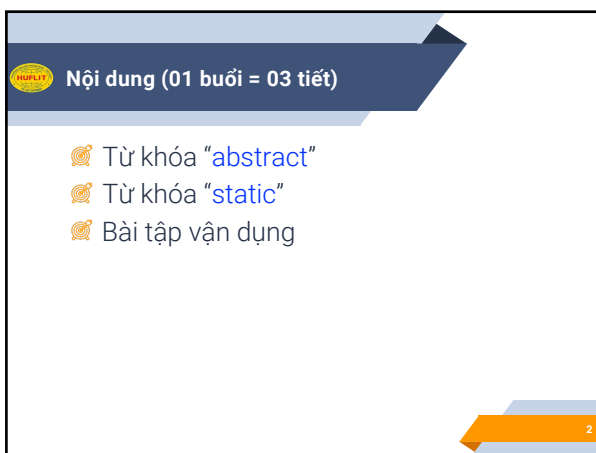




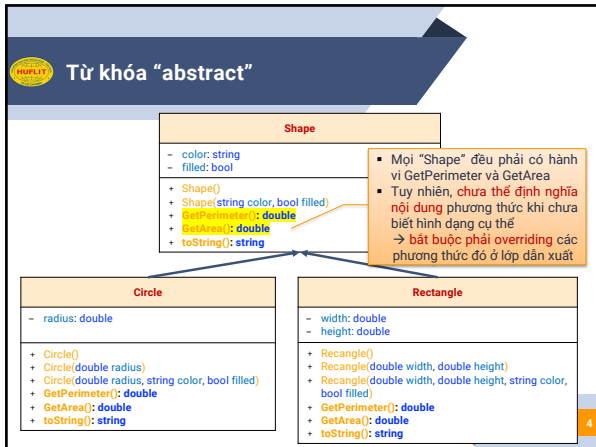
1



2



3



4

Từ khóa "abstract"

- Từ khóa **abstract** dùng để khai báo một thành phần mà phần cài đặt chi tiết của nó có thể còn thiếu hoặc chưa hoàn chỉnh
- Từ khóa **abstract** có thể dùng cho **class**, **method**, **property**, (**indexer** và **event**)

5

Từ khóa "abstract"

- Lớp Trừu tượng (**abstract class**):
 - abstract class** được định nghĩa chỉ để dùng làm lớp cơ sở cho các lớp khác kế thừa
 - KHÔNG** thể tạo đối tượng (object) của **abstract class**
 - Các thành phần **abstract** bên trong **abstract class** phải được hiện thực ở các **non-abstract class** kế thừa từ **abstract class**

```

abstract class BaseClass
{
    //...
}

class Program
{
    static void Main(string args)
    {
        BaseClass obj = new BaseClass();
    }
}
    
```

6

Từ khóa "abstract"

- Phương thức Trừu tượng (*abstract method*):
 - ☐ **abstract method** là phương thức chỉ có phần khai báo (prototype) mà không có nội dung → khai báo method kết thúc bằng dấu ";" và không có cặp dấu "{}"
 - ☐ **abstract method** phải được khai báo bên trong **abstract class**

```
abstract class BaseClass
{
    abstract void MyMethod();
}
```
 - ☐ **abstract method** bắt buộc phải được hiện thực (overriding) ở các **non-abstract class** kế thừa từ **abstract class**

7

7

Từ khóa "abstract"

```
public abstract class Shape
{
    // Attributes
    private string color;
    private bool filled;

    // Constructors
    public Shape() {}
    public Shape(string color, bool filled)
    {
        this.color = color;
        this.filled = filled;
    }

    // Methods
    public abstract double getPerimeter();
    public abstract double getArea();
    public virtual void toString() { . . . }
}
```

8

8

Từ khóa "abstract"

```
public class Circle : Shape
{
    // Attributes
    private double radius;

    // Constructors
    public Circle(double radius)
    {
        this.radius = radius;
    }
    public Circle(double radius, string color, bool filled) : base(color, filled)
    {
        this.radius = radius;
    }

    // Methods
    public override double getPerimeter() { return 3.14 * 2 * radius; }
    public override double getArea() { return 3.14 * radius * radius; }
    public override void toString() { . . . }
}
```

9

9

Từ khóa "abstract"

```

public class Rectangle : Shape
{
    // Attributes
    private double width;
    private double height;

    // Constructors
    public Rectangle(double width, double height)
    {
        this.width = width;
        this.height = height;
    }
    public Rectangle(double width, double height, string color, bool filled) :
        base(color, filled)
    {
        this.width = width;
        this.height = height;
    }

    // Methods
    public override double GetPerimeter() { return 2 * (width + height); }
    public override double GetArea() { return width * height; }
    public override void toString() { . . . }
}
    
```

10

10

2

TỪ KHÓA "static"

11

11

Từ khóa "static"

- Từ khóa **static**: dùng để khai báo một thành phần tĩnh, tồn tại độc lập, không phụ thuộc vào một đối tượng cụ thể
- Ví dụ khi định nghĩa lớp nhân viên:
 - ☐ Mỗi nhân viên sẽ có hệ số lương khác nhau
 - ☐ Lương nhân viên = hệ số lương x **lương cơ bản** (*áp dụng chung cho toàn bộ nhân viên*) → khai báo **lương cơ bản** là thuộc tính **static**
- Từ khóa **static** có thể dùng cho **class**, **attribute**, **method**, **property**, **operator**, **constructor** và (**event**)

12

12

Từ khóa "static"

- static attribute:

```

public class Staff
{
    public static double baseSalary;
    public double coefficient;
    //...
    public double Pay()
    {
        return baseSalary * coefficient;
    }
}
    
```

```

public class Program
{
    static void Main(string args)
    {
        Staff.baseSalary = 4000000;
        Staff obj = new Staff();
        obj.coefficient = 3.2;
        obj.Pay();
    }
}
    
```

- baseSalary tồn tại không phụ thuộc vào các object của class Staff
- Truy cập static attribute thông qua class name

13

13

Từ khóa "static"

- static method:

```

public class Fraction
{
    private int numerator;
    private int denominator;
    //...
    public static Fraction Add(Fraction f1, Fraction f2)
    {
        ...
    }
}
    
```

```

public class Program
{
    static void Main(string args)
    {
        Fraction f1 = new Fraction();
        Fraction f2 = new Fraction();
        Fraction sum = Fraction.Add(f1, f2);
    }
}
    
```

- Truy cập static method thông qua class name
- Cần phải truyền đủ 2 đối tượng Fraction để thực hiện phép cộng

14

14

Từ khóa "static"

- static class: tất cả thành phần của class đều static → thường dùng để tạo các lớp tiện ích (không cần tạo đối tượng của lớp đó)

```

public static class Math
{
    public static double Pow(double x, double y)
    {
        return ...;
    }
    public static double Abs(double value)
    {
        return ...;
    }
    ...
}
    
```

```

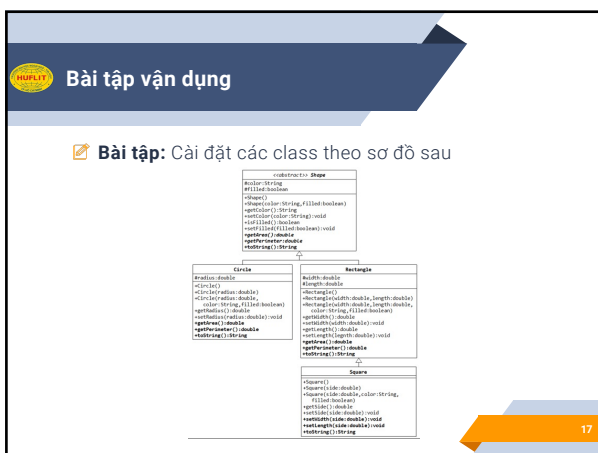
public class Program
{
    static void Main(string args)
    {
        double n = Math.Pow(2, 3);
        double x = Math.Abs(-5);
    }
}
    
```

15

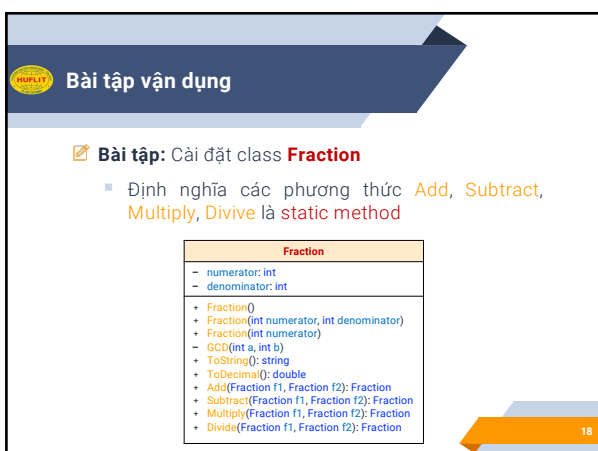
15



16



17



18



19
