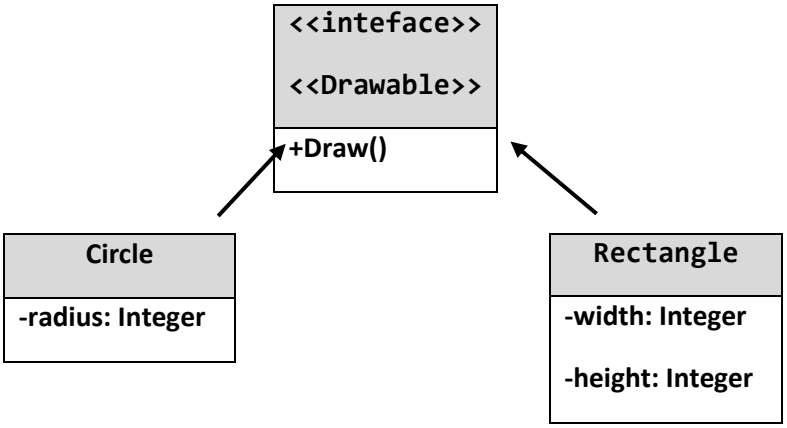
Упражнения: Интерфейси

1. Фигури

Постройте йерархията от интерфейси и класове:



Трябва да може да използвате класовете по сходен начин:

StartUp.cs

```
var radius = int.Parse(Console.ReadLine());
IDrawable circle = new Circle(radius);

var width = int.Parse(Console.ReadLine());
var height = int.Parse(Console.ReadLine());
IDrawable rect = new Rectangle(width, height);

circle.Draw();
rect.Draw();
```

Примери

Вход	Изход

3
4
5
	`   *******`
	` **       **`
	`**         **`
	`*           *`
	`**         **`
	` **       **`
	`   *******`
	`****`
	`*  *`
	`*  *`
	`*  *`
	`****`

Решение

За чертането на кръга, използвайте следния алгоритъм:

```
double r_in = this.Radius - 0.4;
double r_out = this.Radius + 0.4;

for (double y = this.Radius; y >= -this.Radius; --y)
{
    for (double x = -this.Radius; x < r_out; x += 0.5)
    {
        double value = x * x + y * y;
        if (value >= r_in * r_in && value <= r_out * r_out)
        {
            Console.Write("*");
        }
        else
        {
            Console.Write(" ");
        }
    }

    Console.WriteLine();
}
```

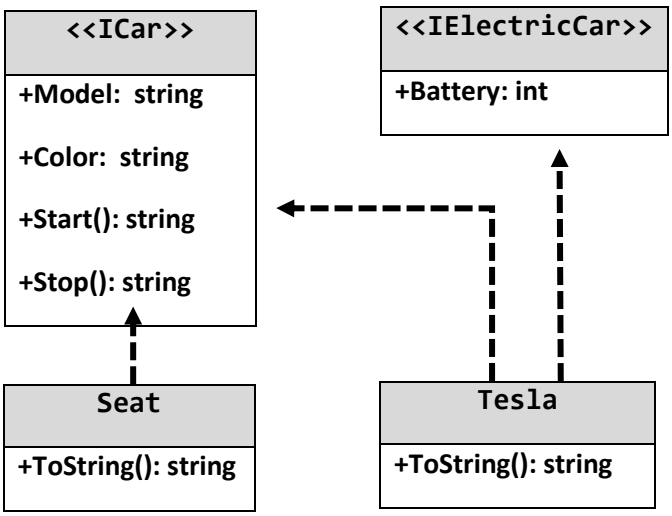
За чертането на правоъгълника използвайте следния алгоритъм:

```
public void Draw()
{
    DrawLine(this.Width, '*', '*');
    for (int i = 1; i < this.Height - 1; ++i)
        DrawLine(this.Width, '*', ' ');
    DrawLine(this.Width, '*', '*');
}

private void DrawLine(int width, char end, char mid)
{
    Console.Write(end);
    for (int i = 1; i < width - 1; ++i)
        Console.Write(mid);
    Console.WriteLine(end);
}
```

2. Коли

Постройте йерархията от интерфейси и класове:



Вашата йерархия трябва да може да се ползва със следния код:

StartUp.cs

```
ICar seat = new Seat("Leon", "Grey");
ICar tesla = new Tesla("Model 3", "Red", 2);

Console.WriteLine(seat.ToString());
Console.WriteLine(tesla.ToString());
```

Примери

Input	Output
	Grey Seat Leon Engine start Breaaak! Red Tesla Model 3 with 2 Batteries Engine start Breaaak!

3. Животинско царство

Както всички знаем, животните обичат да са шумни. Хората пък обичат да ги дресират. Именно с това е свързана настоящата задача. Вашата цел е да създадете:

Интерфейс IMakeNoise

- Този интерфейс трябва да съдържа сигнатурата на метод **string MakeNoise()**

Интерфейс IMakeTrick

- Този интерфейс трябва да съдържа сигнатурата на метод **string MakeTrick()**

Абстрактен клас Animal

- Този клас трябва да имплементира **IMakeNoise** и **IMakeTrick**.
- Класът трябва да съдържа полета за **name** и **age**
- Конструктор, който приема 2 параметъра – име и възраст и задава стойностите им за полетата
- Имплементация като виртуален метод на **MakeNoise()**, която да отпечатва съобщението: My name is <**name**>. I am <**years**> old.
- Имплементация като виртуален метод на **MakeTrick()**, която да отпечатва съобщението: Look at my trick!

Клас Cat

- Този клас трябва да наследява **Animal**.
- Конструктор, който приема 2 параметъра – име и възраст и извиква базовия си конструктор
- Метод **MakeNoise()**, който отпечатва съобщението: “**Meow!”**. След това извикайте **MakeNoise()** за базовия клас.
- Метод **MakeTrick()**, който отпечатва: No trick for you! I'm too lazy!

Клас Dog

- Този клас трябва да наследява **Animal**.
- Конструктор, който приема 2 параметъра – име и възраст и извиква базовия си конструктор
- Метод **MakeNoise()**, който отпечатва съобщението: Woof! След което извиква **MakeNoise()** за базовия клас
- Метод **MakeTrick()**, който отпечатва: Hold my paw, human!

4. Животинско царство 2

След като построихме първото животинско царство, сега ще добавим някои новости. Този път вашата цел е да създадете:

Интерфейс IMakeNoise

- Този интерфейс трябва да съдържа сигнатурата на метод **string MakeNoise()**

Интерфейс IMakeTrick

- Този интерфейс трябва да съдържа сигнатурата на метод **string MakeTrick()**

Интерфейс IAnimal

- Този клас трябва да имплементира **IMakeNoise** и **IMakeTrick**.
- Интерфейсът трябва да съдържа метод **Perform()**

Клас Cat

- Този клас трябва да имплементира **IAnimal**.
- Метод **MakeNoise()**, който отпечатва съобщението: "**Meow!"**.
- Метод **MakeTrick()**, който отпечатва: "**No trick for you! I'm too lazy!**"
- Метод **Perform()**, който извиква първо **MakeNoise()**, а после **MakeTrick()**

Клас Dog

- Този клас трябва да имплементира **IAnimal**.
- Метод **MakeNoise()**, който отпечатва съобщението: "**Woof!"**.
- Метод **MakeTrick()**, който отпечатва: "**Hold my paw, human!"**
- Метод **Perform()**, който извиква първо **MakeNoise()**, а после **MakeTrick()**

Клас Trainer

- Този клас трябва да има поле **IAnimal entity**.
- Конструктор, от който да се получава обекта с животното, което ще дресираме
- Метод **Make()**, който да извиква **Perform()** метода на съответното **entity**

5. Дефиниране на интерфейс IPerson

Дефинирайте интерфейс **IPerson** със свойства **Name** и **Age**. Дефинирайте клас **Citizen**, който имплементира **IPerson** и има конструктор, който взема **string** с името и **int** с възрастта.

За да се тества успешно задача, добавете следния код към **Main** метода си:

```
public static void Main(string[] args)
{
    Type personInterface = typeof(Citizen).GetInterface("IPerson");
```

```
    PropertyInfo[] properties = personInterface.GetProperties();
    Console.WriteLine(properties.Length);
    string name = Console.ReadLine();
    int age = int.Parse(Console.ReadLine());
    IPerson person = new Citizen(name, age);
    Console.WriteLine(person.Name);
    Console.WriteLine(person.Age);
}
```

Ако сте дефинирали интерфейса и сте го имплементирали правилно, тестовете би трябвало да минат.

Примери

Вход	Изход
Pesho 25	2 Pesho 25

6. Множествена имплементация

Използвайки кода от предната задача, дефинирайте интерфейс **IIdentifiable** със свойство **Id** от тип **string** и интерфейс **IBirthable** със свойство **Birthdate** от тип **string** и ги имплементирайте в клас **Citizen**. Пренапишете конструктора, така че да приема новите параметри.

За да се тества успешно задача, добавете следния код към **Main** метода си:

```
public static void Main(string[] args)
{
    Type identifiableInterface = typeof(Citizen).GetInterface("IIdentifiable");
    Type birthableInterface = typeof(Citizen).GetInterface("IBirthable");
    PropertyInfo[] properties = identifiableInterface.GetProperties();
    Console.WriteLine(properties.Length);
    Console.WriteLine(properties[0].PropertyType.Name);
    properties = birthableInterface.GetProperties();
    Console.WriteLine(properties.Length);
    Console.WriteLine(properties[0].PropertyType.Name);
    string name = Console.ReadLine();
    int age = int.Parse(Console.ReadLine());
    string id = Console.ReadLine();
    string birthdate = Console.ReadLine();
    IIdentifiable identifiable = new Citizen(name, age,id, birthdate);
    IBirthable birthable = new Citizen(name, age, id, birthdate);
}
```

Ако сте дефинирали интерфейса и сте го имплементирали правилно, тестовете би трябвало да минат.

Примери

Вход	Изход
Pesho 25 9105152287 15/05/1991	1 String 1 String

7. Телефония

Имате бизнес – **производство на мобилни телефони**. Но нямате софтуерни разработчици, затова се обаждате на няколко приятели и ги молите за помощ. Те вече са се съгласили и сте започнали работа. Проекта се състои от един главен **модел – Смартфон**. Всеки смартфон трябва да има функционалности за **свързване с други телефони** и **достъпване на Интернет.**

Тези ваши приятели са доста заети, затова решавате да напишете кода сам. Ето го вашето задължително задание:

Имате **модел** - **Смартфон** и две отделни функционалности – **обаждане** и **достъпване на Интернет.** Накрая трябва да получите **един клас** и **два интерфейса.**

Вход

Входът се задава от конзолата. Той съдържа два реда:

- **Първи ред**: **телефонни номера** (като низ), разделени с интервали.
- **Втори ред: сайтове** (като низ), разделени от интервали.

Изход

- Първо трябва да се **обадите на всички номера** според реда на въвеждането им, а след това **да посетите всички сайтове** в реда на въвеждането им
- Функционалността за обаждане на телефоните отпечатва на конзолата съобщение в следния формат: **`Calling... <номер>`**
- Функционалността за посещение на сайт отпечатва на конзолата съобщение в следния формат: **`Browsing: <сайт>!`**
- Ако има число в списъка с URL адреси, отпечатайте: **"Invalid URL!"** и продължете нататък с останалите URL адреси.
- Ако има знак различен от цифра в телефонните номера, отпечатайте: **"Invalid number!"** и продължете към следващия номер.

Ограничения

- Всеки URL трябва да съдържа само букви и символи (**Не са позволени цифри** в URL адресите)

Примери

Вход	Изход

`0882134215 0882134333 08992134215 0558123 3333 1`
`http://mon.bg http://youtube.com http://www.g00gle.com`
	`Calling... 0882134215`
	`Calling... 0882134333`
	`Calling... 08992134215`
	`Calling... 0558123`
	`Calling... 3333`
	`Calling... 1`
	`Browsing: http://mon.bg!`
	`Browsing: http://youtube.com!`
	`Invalid URL!`