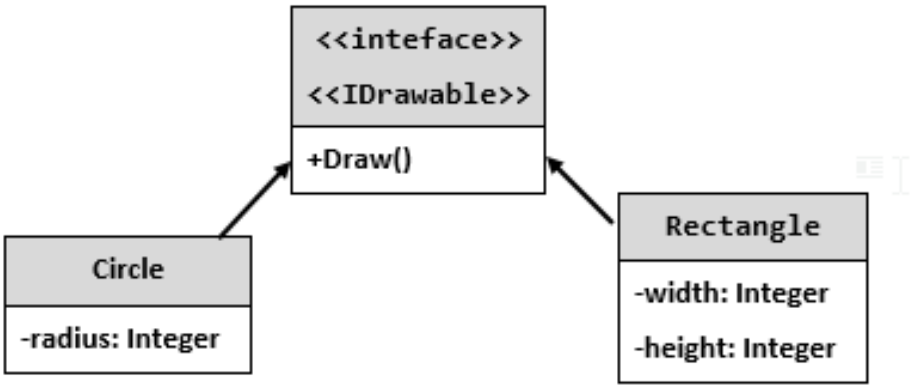
Упражнения: Абстрактни класове и интерфейси

Можете да проверите решенията си в **Judge системата**: https://judge.softuni.bg/Contests/3165/Abstract-Classes-and-Interfaces

1. Фигури

Важно: Трябва да имате публичен клас **StartUp** в namespace **Shapes**.

Създайте **йерархия** от **интерфейси** и **класове**:



И двата класа трябва да отпечатват фигурата с "*"

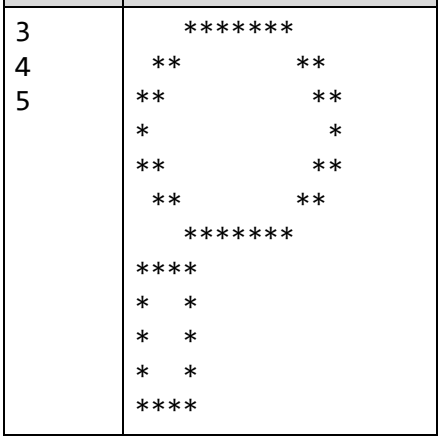
Трябва да можете да използвате класа по следния начин:

```
static void Main(string[] args)
{
    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

Алгоритъмът за рисуване на кръг е:

```
double rIn = this.radius - 0.4;
double rOut = this.radius + 0.4;
for (double y = this.radius; y >= -this.radius; --y)
{
    for (double x = -this.radius; x < rOut; x += 0.5)
    {
        double value = x * x + y * y;

        if (value >= rIn * rIn && value <= rOut * rOut)
        {
            Console.Write("*");
        }
        else
        {
            Console.Write(" ");
        }
    }
    Console.WriteLine();
}
```

Алгоритъмът за рисуване на правоъгълник е:

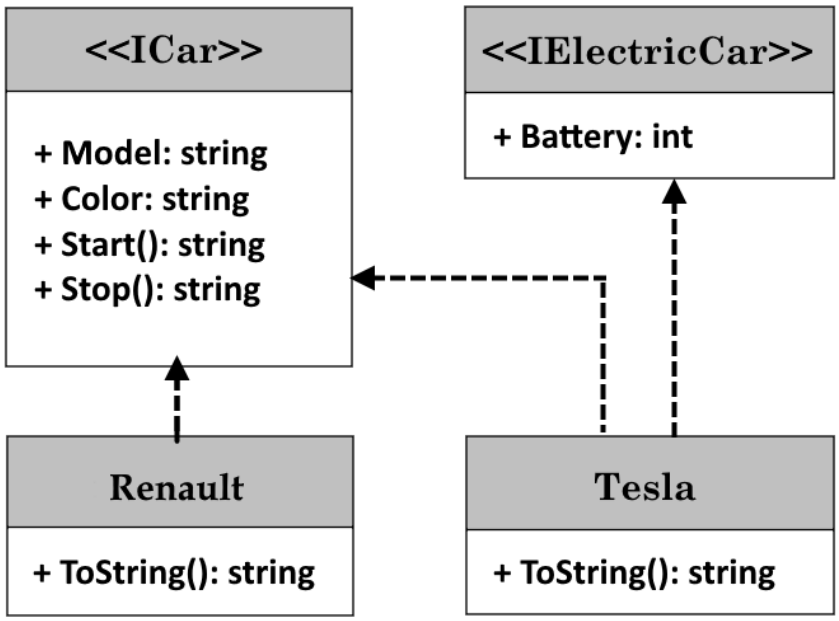
```
public void Draw()
{
    DrawLine(this.width, '*', '*');
    for (int i = 1; i < this.height - 1; ++i)
    {
        DrawLine(this.width, '*', ' ');
    }
    DrawLine(this.width, '*', '*');
}

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

Важно: Трябва да имате публичен клас **`StartUp`** в namespace **`Cars`**.

Създайте **йерархия** от **интерфейси** и **класове**:



Трябва да можете да използвате класа по следния начин:

```
static void Main(string[] args)
{
    ICar renault = new Renault("Duster", "Grey");
    ICar tesla = new Tesla("Model 3", "Red", 2);

    Console.WriteLine(renault.ToString());
    Console.WriteLine(tesla.ToString());
}
```

Примери

Изход
Grey Renault Duster Engine start Breaaak! Red Tesla Model 3 with 2 Batteries Engine start Breaaak!

3. Дефинирайте интерфейс IPerson

Важно: Трябва да имате публичен клас **StartUp** в namespace **PersonInfo**.

Дефинирайте **интерфейс IPerson** със свойства за **Name** и **Age**. Дефинирайте клас **Citizen**, който имплементира **IPerson** и има **конструктор**, който приема:

- **name – string**
- **Age - int**

Създайте нов **Person** по следния начин:

```
string name = Console.ReadLine();
int age = int.Parse(Console.ReadLine());
IPerson person = new Citizen(name, age);
Console.WriteLine(person.Name);
Console.WriteLine(person.Age);
```

Примери

Вход	Изход
Pesho 25	Pesho 25

4. Множество имплементации

Важно: Трябва да имате публичен клас **StartUp** в namespace **PersonInfo**.

Използвайки кода от предишната задача, дефинирайте **интерфейс IIdentifiable** със свойство **string Id** и **интерфейс IBirthable** със свойство **string Birthdate**. Имплементирайте ги в класа **Citizen**. Презапишете конструктора на **Citizen**, за да приема новите параметри.

Трябва да можете да използвате класа по следния начин:

```
static void Main(string[] args)
{
    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);
    Console.WriteLine(identifiable.Id);
    Console.WriteLine(birthable.Birthdate);
}
```

Примери

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

5. Телефония

Имате бизнес – **производство** на **телефони**. Само че нямате софтуерни инженери, така че се обаждате на приятели и ги молите за помощ при създаването на софтуер. Те се съгласяват и започвате да работите по проекта. Той се състои от **два главни модела** - **Smartphone** и **StationaryPhone**. Всеки смартфон трябва да има функционалности да **звъни на други телефони** и да **достъпва интернет**. **StationaryPhone** могат **само да звънят на други телефони**.

Оказва се, че приятелите ви са много заети, така че решавате да напишете кода сами. Изискванията са следните:

Трябва да имате **модел `Smartphone`** и две отделни функционалности на смартфона ви – да **звъни на други телефони** и да **достъпва интернет**.

Трябва да имате и **модел `StationaryPhone`** и функционалност, която има – да **звъни на други телефони**.

Трябва да имате **два класа** и **два интерфейса**.

Вход

- Първи ред: **телефонни номера**, на които да се обаждате (**`string`**), разделени с интервал.
- Втори ред: **сайтове**, които да посетите (**`string`**), разделени с интервал.

Изход

- Първо **звъннете на всички телефонни номера** в последователността, в която сте ги получили,а след това отворете **всички сайтове** в оригиналната последователност.
- Функционалността за звънене трябва да отпечатва на конзолата на кой номер звъните:
  - Ако номерът е дълъг 10 цифри, звъните от смартфона и отпечатвате: **`Calling... {number}`**
  - Ако номерът е дълъг 7 цифри, звъните от стационарния телефон и отпечатвате: **`Dialing... {number}`**
- Функционалността с браузъра трябва да отпечатва кой сайт е отворен:

  `Browsing: {site}!`
- Ако има число в някой от URL-ите на сайтовете, отпечатайте **"Invalid URL!"** и продължете с останалите сайтове.
- Ако в някой телефонен номер има символ, различен от цифра, отпечатайте **"Invalid number!"** и продължете с останалите номера

Забележки

- Телефонните номера винаги ще бъдат с дължина от 7 или 10 символа.

Примери

Вход	Изход
0882134215 0882134333 0899213421 0558123 3333123 http://softuni.bg http://youtube.com http://www.g00gle.com	Calling... 0882134215 Calling... 0882134333 Calling... 0899213421 Dialing... 0558123 Dialing... 3333123 Browsing: http://softuni.bg! Browsing: http://youtube.com! Invalid URL!

6. Граничен контрол

Пренасяме се в бъдещето и вие сте владетел на тоталитарно дистопично общество, обитавано от **граждани** и **роботи**. Тъй като се страхувате от потенциални бунтове, решавате да имплементирате стриктен контрол върху това кой влиза във вашия град. Вашите войници проверяват **Id**-тата на всеки, който влиза и излиза от града.

До получаване на команда "**End**" на всеки ред ще имате информация или за **гражданин**, или за **робот**, който иска да влезе в града, в следния формат: **"{name} {age} {id}"** за **граждани** и **"{model} {id}"** за **роботи**. След получаване на команда "**End**", ще получите **едно число**, което представлява **последните цифри на фалшивите id**-та. **Всички граждани** и **роботи**, чиито **id**-та завършват с тези цифри, трябва да бъдат **задържани**.

Изходът от програмата ви трябва да се състои от **id**-тата на всички **задържани** граждани и роботи **на отделен ред** в последователността, в която сте ги приели.

Примери

Вход	Изход
Pesho 22 9010101122 MK-13 558833251 MK-12 33283122 End 122	9010101122 33283122
Toncho 31 7801211340 Penka 29 8007181534 IV-228 999999 Stamat 54 3401018380 KKK-666 80808080 End 340	7801211340

7. Рожден ден

Известен факт е, че хората празнуват рождените си дни, а понякога празнуват рождените дни и на своите домашни любимци.

Разширете програмата от предишната задача, като добавите **рождените дни** на гражданите и създадете клас **Pet**. Всеки домашен любимец има **name** и **birthdate**. Разпределете повтарящата се функционалност в интерфейси и ги имплементирайте в класовете си.

До получаване на команда "**End**", всеки ред ще съдържа информация в един от следните формати:

- "**Citizen <name> <age> <id> <birthdate>**" за **гражданин**
- "**Robot <model> <id>**" за **робот**
- "**Pet <name> <birthdate>**" за **домашен любимец**

След получаване на команда "**End**" ще получите **едно число**, което представлява **конкретна година**. Вашата задача е да отпечатате **всички рождени дни** (и за граждани, и за домашни любимци) от тази година във формата **ден/месец/година** в реда, в който сте ги получили.

Примери

Вход	Изход
Citizen Pesho 22 9010101122 10/10/1990 Pet Sharo 13/11/2005 Robot MK-13 558833251 End 1990	10/10/1990
Citizen Stamat 16 0041018380 01/01/2000 Robot MK-10 12345678 Robot PP-09 00000001 Pet Topcho 24/12/2000 Pet Kosmat 12/06/2002 End 2000	01/01/2000 24/12/2000
Robot VV-XYZ 11213141 Citizen Penka 35 7903210713 21/03/1979 Citizen Kane 40 7409073566 07/09/1974 End 1975	<empty output>

8. Недостатъчна храна

Вашето тоталитарно общество страда от недостиг на храна и се появяват много бунтовници. Разширете кода от предишната задача с нова функционалност, за да разрешите този проблем.

Дефинирайте клас **`Rebel`**, който има **`name (име)`**, **`age (възраст)`** и **`group`** (**`група`**)**.** Имената са **уникални** – няма да има двама бунтовници/граждани с едно и също име.

Дефинирайте интерфейс **`IBuyer`**, който дефинира метод **`BuyFood()`** и свойство **`int Food`**. Имплементирайте интерфейса **`IBuyer`** в класовете **`Citizen`** и **`Rebel`**. И бунтовниците, и гражданите започват с **0 Food**. Когато бунтовник купи храна, **Food** се увеличава с **5**, а когато гражданин купи храна, **Food** се увеличава с **10**.

Вход

1. Първи ред – цяло число **N** – общият брой на хората
2. На следващите **N** реда – информация в един от следните формати:
   - "**<name> <age> <id> <birthdate>**" за гражданин (**`Citizen`**)

- "**<name> <age><group>**" за бунтовник (**Rebel**)

3. След получаване на **N** реда, до получаване на команда "**End**", ще получавате **имена на хора**, които са **купили храна**, всеки на нов ред.

Важно: Не всички имена ще бъдат валидни – в случай че получите невалидно име, игнорирайте данните.

Изход

Изходът се състои от **един ред**, на който трябва да отпечатате **общото количество купена храна**.

Примери

Вход	Изход
2 Pesho 25 8904041303 04/04/1989 Stancho 27 WildMonkeys Pesho Gosho Pesho End	20
4 Stamat 23 TheSwarm Toncho 44 7308185527 18/08/1973 Joro 31 Terrorists Penka 27 881222212 22/12/1988 Jiraf Jo ro Jiraf Joro Stamat Penka End	20

9. Експлицитни интерфейси

Създайте два интерфейса **IResident** и **IPerson**. **IResident** трябва да има **name (име)**, **country (държава)** и метод **GetName()**. **IPerson** трябва да има **name (име)**, **age (възраст)** и метод **GetName()**.

Създайте клас **Citizen**, който имплементира и **IResident**, и **IPerson** и експлицитно декларирайте, че методът **GetName()** на **IResident** трябва да **връща** "**Mr/Ms/Mrs** " преди името, докато метода **GetName()** на **IPerson** трябва да връща **само името**.

Вход

До получаване на команда "**End**" ще получавате информация за **Citizen** (всеки на отделен ред). Всеки гражданин ще бъде във формата "**<name> <country> <age>**". Създайте съответната инстанция на **Citizen** и отпечатайте неговия **GetName()** от **IPerson** и неговия **GetName()** от **IResitent**.

Примери

Вход	Изход
PeshoPeshev Bulgaria 20 End	PeshoPeshev Mr/Ms/Mrs PeshoPeshev
JoroJorev Bulgaria 33 EricAnderson GreatBritain 28 PeterArmstrong USA 19 End	JoroJorev Mr/Ms/Mrs JoroJorev EricAnderson Mr/Ms/Mrs EricAnderson PeterArmstrong Mr/Ms/Mrs PeterArmstrong