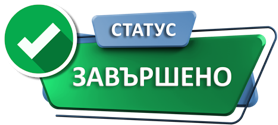
# Упражнения: Абстрактни класове и интерфейси

Можете да проверите решенията си в **Judge системата**: <https://judge.softuni.org/Contests/Practice/Index/4066>

Състояние на настоящия учебен материал:



## Фигури

**Важно**: Трябва да имате публичен клас StartUp в namespace **\_01\_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)

{

int radius = int.Parse(Console.ReadLine());

IDrawable circle = new Circle(radius);

int width = int.Parse(Console.ReadLine());

int height = int.Parse(Console.ReadLine());

IDrawable rectangle = new Rectangle(width, height);

circle.Draw();

rectangle.Draw();

}

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 3  4  5 | \*\*\*\*\*\*\*  \*\* \*\*  \*\* \*\*  \* \*  \*\* \*\*  \*\* \*\*  \*\*\*\*\*\*\*  \*\*\*\*  \* \*  \* \*  \* \*  \*\*\*\* |

### Решение

Алгоритъмът за рисуване на кръг е:

double radiusIn = this.radius - 0.4;

double radiusOut = this.radius + 0.4;

for (double y = this.radius; y >= -this.radius; y--)

{

for (double x = -this.radius; x < radiusOut; x += 0.5)

{

double value = x \* x + y \* y;

if (value >= radiusIn \* radiusIn && value <= radiusOut \* radiusOut)

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);

}

## Коли

**Важно**: Трябва да имате публичен клас StartUp в namespace \_02\_Cars.

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

Diagram

Description automatically generated

Методът **Start()** трябва да връща текста "**Engine start**". Методът **Stop()** трябва да връща "**Breaaak!**". Презаписаният метод **ToString()** трябва да връща:

* За класа **Renault**:
  + **"{цвета на колата} Renault {модел}"**
  + Резултат от метода **Start()**
  + Резултата от метода **Stop()**

Всеки от трите резултата трябва да бъде на **нов ред** във **върнатия стринг**.

* За класа **Tesla**:
  + **"{цвета на колата} Tesla {модел} with {батерия}"**
  + Резултат от метода **Start()**
  + Резултата от метода **Stop()**

Всеки от трите резултата трябва да бъде на **нов ред** във **върнатия стринг**.

Трябва да можете да използвате класа по следния начин:

Graphical user interface, text

Description automatically generated

### Примери

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

## Дефинирайте интерфейс IPerson

**Важно**: Трябва да имате публичен клас StartUp в namespace **\_03\_**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:

Text

Description automatically generated

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| Pesho  25 | Pesho  25 |

## Множество имплементации

**Важно**: Трябва да имате публичен клас StartUp в namespace-а от предишната задача **\_03\_**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:

Graphical user interface, text, application

Description automatically generated

### Примери

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| Pesho  25  9105152287  15/05/1991 | 9105152287  15/05/1991 |