

## Opdracht 1

We gaan in deze opdracht een boekhandel-applicatie maken. In de boekhandel worden naast boeken ook magazines en CD's verkocht.

Maak een abstracte class "BookStoreItem" en plaats hierin de volgende properties:

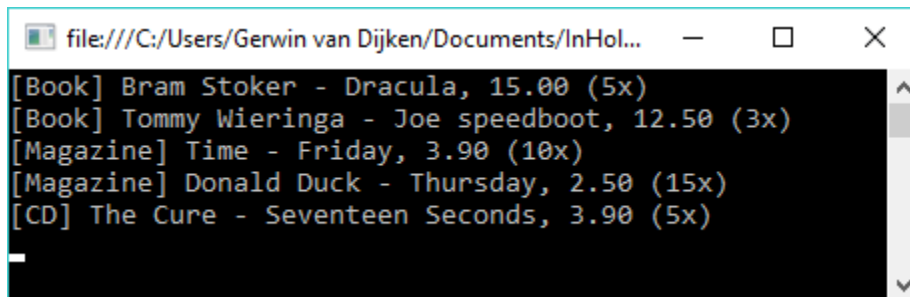
```
public string Title { get; set; }  
public float Price { get; set; }  
public int NumberOfItems { get; set; }
```

Deze informatie (titel, prijs, aantal) wordt via de constructor van de class BookStoreItem meegegeven.

Maak 3 afgeleide classes van BookStoreItem: Book, Magazine en CD. Geef een boek een extra (string) property "Author", een magazine een extra (DayOfWeek) property "ReleaseDay", en een CD een extra (string) property "Artist".

Implementeer een class "BookStore" met daarin een lijst met "BookStoreItem"-objecten. Zorg dat deze lijst alleen door de class BookStore gebruikt kan worden (niet benaderbaar van buitenaf). Voeg de volgende 2 methoden toe aan class BookStore: "Add" (om een item aan de lijst toe te voegen) en "PrintAllItems" (om alle items te printen).

Test de code door vanuit het hoofdprogramma (Start methode) een boekhandel aan te maken, daaraan verschillende boeken, magazines en CD's toe te voegen, en deze (hele handel) vervolgens te printen. Hierbij moeten ook de item-specifieke eigenschappen geprint worden (bv auteur van een boek en artiest van een CD).



```
file:///C:/Users/Gerwin van Dijken/Documents/InHol...  
[Book] Bram Stoker - Dracula, 15.00 (5x)  
[Book] Tommy Wieringa - Joe speedboot, 12.50 (3x)  
[Magazine] Time - Friday, 3.90 (10x)  
[Magazine] Donald Duck - Thursday, 2.50 (15x)  
[CD] The Cure - Seventeen Seconds, 3.90 (5x)
```

*Probeer een object aan te maken van class "BookStoreItem". Welke foutmelding krijg je?*

## Opdracht 2

We gaan in deze opdracht een Stack implementeren. Een stack is een data-structuur waarin items geplaatst kunnen worden, waarbij het laatst toegevoegde item er als eerste weer uitgehaald wordt (en het eerste item als laatste). Deze volgorde wordt ook aangeduid met LIFO (Last In First Out) of FILO (First In Last Out).

Een voorbeeld waarbij een stack gebruikt wordt is het stack-geheugen in computer programma's; bij elke methode-aanroep worden de parameters en de lokale variabelen (van de methode) op de stack geplaatst. Zodra de methode afgehandeld is, worden deze items weer van de stack verwijderd. Hier geldt dus ook, de items van de laatste methode-aanroep worden als eerste weer verwijderd.

Van een Stack willen we de volgende operaties kunnen uitvoeren (*ongeacht hoe de stack geïmplementeerd is*):

1. `void Push(int value)` → plaatst een nieuw item (int) op de stack; throw een exception als de stack vol is
2. `int Pop()` → retourneert het laatst geplaatste item (int) op de stack terug (en verwijdert deze van de stack); throw een exception als de stack leeg is
3. `bool Contains(int value)` → retourneert true als een specifiek item (int) op de stack aanwezig is, anders false
4. `int Count { get; }` → retourneert het aantal items op de stack
5. `bool IsEmpty { get; }` → retourneert true als de stack leeg is, anders false

Maak een interface (IStack) aan met bovenstaande methoden/properties. Maak een class aan (`ArrayStack`) die deze IStack interface implementeert, gebruikmakende van een array om de stack-items in op te slaan. Geef het maximaal aantal items dat op de stack geplaatst kan worden mee als constructor-parameter. Onderstaand programma kun je gebruiken om jouw implementatie te testen.

```
void Start()
{
    IStack myStack = new ArrayStack(50);
    AddValues(myStack);
    ProcessValues(myStack);
}

void AddValues(IStack stack)
{
    Random rnd = new Random();
    for (int i = 0; i < 10; i++)
    {
        int value = rnd.Next(100);
        stack.Push(value);
        Console.WriteLine($"pushed {value},
                           new count: {stack.Count}");
    }
}

void ProcessValues(IStack stack)
{
    while (!stack.IsEmpty)
    {
        int value = stack.Pop();
        Console.WriteLine($"popped {value}, new count: {stack.Count}");
    }
}
```

```
pushed 76, new count: 1
pushed 4, new count: 2
pushed 39, new count: 3
pushed 38, new count: 4
pushed 0, new count: 5
pushed 87, new count: 6
pushed 30, new count: 7
pushed 68, new count: 8
pushed 28, new count: 9
pushed 33, new count: 10

popped 33, new count: 9
popped 28, new count: 8
popped 68, new count: 7
popped 30, new count: 6
popped 87, new count: 5
popped 0, new count: 4
popped 38, new count: 3
popped 39, new count: 2
popped 4, new count: 1
popped 76, new count: 0
```

Voeg zelf een test-methode 'CheckValues' toe die de methode 'Contains' test.

## Opdracht 3

Gegeven zijn de volgende interfaces (elk in een eigen bestand):

```
public interface IPencil {
    bool CanWrite { get; } // determines if the pencil can still write
    void Write(string message); // writes characters of the message
    void AfterSharpening(); // the pencil is made 'new' (so it can write 'max' again)
}

public interface IPencilSharpener
{
    void Sharpen(IPencil pencil);
}
```

Maak een class "Pencil" die de interface "IPencil" implementeert en een class "PencilSharpener" die de interface "IPencilSharpener" implementeert.

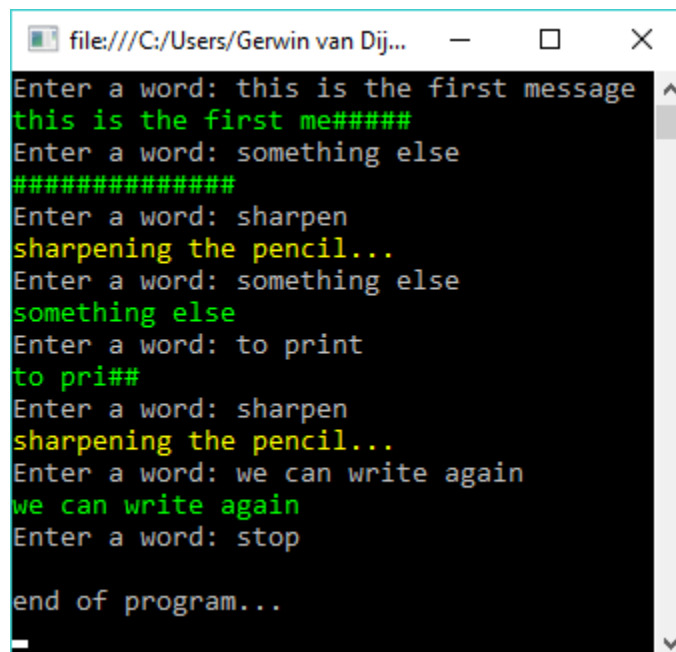
Voeg de volgende member toe aan class Pencil:

```
private int maxToWrite; // number of characters to write with a sharpened pencil
private int nrOfCharsWritten; // number of written characters
```

De implementatie zal zodanig moeten zijn dat het potlood niet in staat is een boodschap volledig te schrijven als deze te lang is; na een bepaald aantal geschreven karakters zal het potlood stomp zijn en zal een '#' geschreven worden i.p.v. het te schrijven karakter.

Schrijf een eenvoudig programma dat berichten leest van de gebruiker totdat het woord "stop" wordt ingevoerd. Elk ingevoerd bericht moet geschreven worden met de (aangemaakte) potlood. Als de gebruiker het woord "sharpen" invoert, dan moet de potlood geslepen worden door de (aangemaakte) poloodslipper.

Een voorbeeld van de uitvoer van dit programma zie je hieronder (*de potlood kan steeds 20 karakters schrijven*):



```
file:///C:/Users/Gerwin van Dij...
Enter a word: this is the first message
this is the first me#####
Enter a word: something else
#####
Enter a word: sharpen
sharpening the pencil...
Enter a word: something else
something else
Enter a word: to print
to pri##
Enter a word: sharpen
sharpening the pencil...
Enter a word: we can write again
we can write again
Enter a word: stop

end of program...
```