

# C# Programmeren 1

ENUMS EN STRUCTS

# Inhoud

1.Enumeraties (enum)

1.Structuren (struct)





# Enumeraties

Types beperkt tot een voorgedefinieerde reeks van waarden

# Enumeraties

- **Enumeraties** in C# zijn types waarvan de mogelijke waarden beperkt worden tot een voorgedefinieerde lijst van waarden
  - Bv de dagen van de week
  - Declaratie gebeurt via het keyword **enum** in C#
  - Gebruikt enkel waarden uit de voorgedefinieerde lijst
  - Onderliggend type is standaard int en onderliggende waarden zijn:  
0 voor het eerste element, 1 voor het 2de, 2 voor het 3de,...

```
public enum Kleur { Rood, Groen, Blauw, Zwart }  
//...  
Kleur kleur = Kleur.Rood;  
Console.WriteLine(kleur.ToString()); // Rood  
kleur = (Kleur) 3;
```

# Enumeraties

- Onderliggend type van enum kan gespecificeerd worden (bv: Byte, short, int, long)
- Onderliggende waarden van elementen kunnen worden gedefinieerd
- Voorbeeld:

```
enum Kleur: byte
{
    Rood = 1,
    Groen = 2,
    Blauw = 4,
    Zwart = 0
}
```

# Enumeraties – Voorbeeld 1

```
enum Dag
{
    Maandag = 1,
    Dinsdag = 2,
    Woensdag = 3,
    Donderdag = 4,
    Vrijdag = 5,
    Zaterdag = 6,
    Zondag = 7
};
//...
Dag vandaag = (Dag)4;
Console.WriteLine("Vandaag is " + vandaag.ToString());
Dag day = Dag.Monday;
Console.WriteLine("Maandag = " + (int)day);
Console.WriteLine("Zondag = " + (int)Dag.Zondag);
```

# Enumeraties – Voorbeeld 2

```
enum VerkeersLicht
{
    Groen,
    Oranje,
    Rood
}

static void Main(string[] args)
{
    VerkeersLicht verkeerslicht = VerkeersLicht.Oranje;
    //...
    switch (verkeerslicht)
    {
        case VerkeersLicht.Rood:
            Console.WriteLine("STOPPEN!");
            break;
        case VerkeersLicht.Groen:
            Console.WriteLine("DOORRIJDEN!");
            break;
        case VerkeersLicht.Oranje:
            Console.WriteLine("GAS BIJGEVEN?");
            break;
    }
}
```

# Enumeraties – Voorbeeld 3

```
enum Dag { Zaterdag, Zondag, Maandag, Dinsdag, Woensdag, Donderdag, Vrijdag };

public static void Main()
{
    Dag dag = typeof(Dag);

    Console.WriteLine("dagen van de week en waarden:");

    foreach (string s in Enum.GetNames(dag))
        Console.WriteLine("{0,-11}= {1}", s, Enum.Format(dag, Enum.Parse(dag, s), "d"));
}
```

Enum.Format:

<https://docs.microsoft.com/en-us/dotnet/api/system.enum.format?view=netcore-3.1>



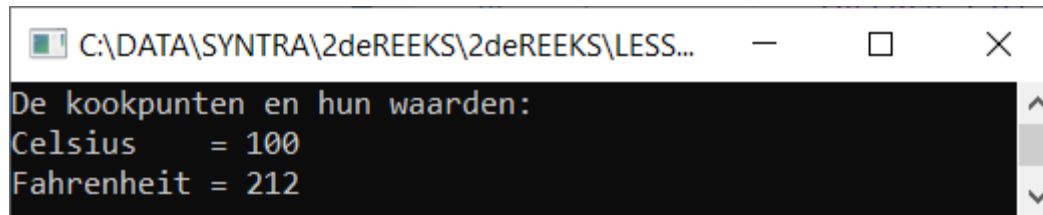


# Enumeraties

Demo

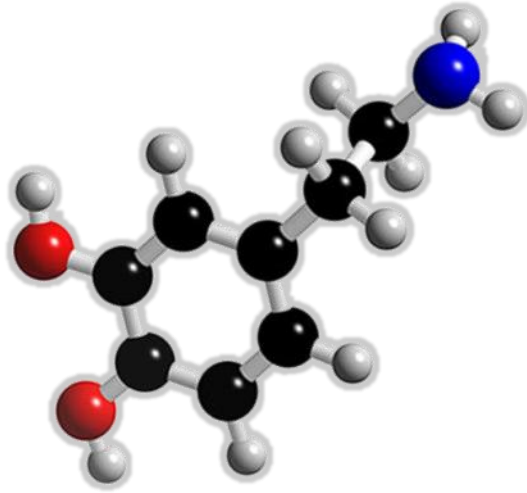
# Oefening Enumeratie

**Definieer een enum KookPunt met 2 elementen Celsius met waarde 100 en Fahrenheit met waarde 212. En schrijf dan het volgende naar de console. Gebruik een foreach lus om de gegevens naar de console te schrijven**

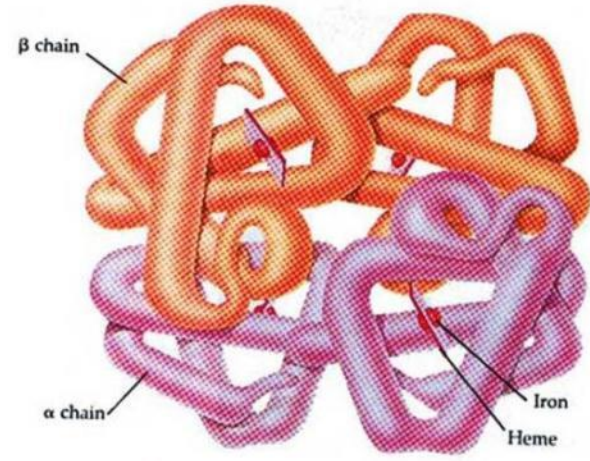


```
C:\DATA\SYNTRA\2deREEKS\2deREEKS\LESS...
De kookpunten en hun waarden:
Celsius      = 100
Fahrenheit   = 212
```

The screenshot shows a Windows console window with a title bar containing the file path 'C:\DATA\SYNTRA\2deREEKS\2deREEKS\LESS...'. The console output displays the text 'De kookpunten en hun waarden:' followed by two lines: 'Celsius = 100' and 'Fahrenheit = 212'. The text is left-aligned and uses a monospaced font.



(a) Collagen



(b) Hemoglobin

# Structuren

**Wat zijn Structuren? Wanneer ze te gebruiken?**

# Structuren

- **Structuren** in C# zijn gelijkaardig aan classes
  - Structures zijn **value types** (bevatten waarde)
  - Classes zijn **reference types** (verwijzing naar waarde)
- Structuren worden meestal gebruikt om gegevensstructuren bij te houden, zonder andere functionaliteit
- Structuren kunnen fields, properties,... bevatten
- Voorbeeld van structure
  - **System.DateTime** – een struct waarin datum en tijd worden bijgehouden

# Structuren

- Ideaal voor “licht-gewicht” structuren, zoals DateTime, Point, Rectangle,...
- Zoals bij alle value types, een variabele van struct types bevatten hun eigen data en worden bewaard op de execution stack Bijgevolg:
  - Een variabele van struct types kan **niet null** zijn
  - 2 struct variabelen kunnen niet refereren naar dezelfde object

# Structuren – Voorbeeld

```
public struct Boek
{
    public decimal Prijs;
    public string Titel;
    public string Auteur;
}

static void Main(string[] args)
{
    Boek book = new Boek();
    book.Prijs = 10.55m;
    book.Titel = "C# Programming Basics";
    book.Auteur = "Addison W.";
}
```

# Structuren – Voorbeeld

```
struct Kaart
{
    Kleur Kaart_Kleur; //enum te definiëren
    Rang Kaart_Rang;   //enum te definiëren
    public Kaart(Kleur kleur, Rang rang) {
        //Constructor van Kaart
        Kaart_Kleur = kleur;
        Kaart_Rang = rang;
    }
    public void PrintKaart() //methode
    {
        Console.WriteLine(Kaart_Kleur.ToString() + " " +
Kaart_Rang.ToString());
    }
}
```

# Oefening struct en enum

## Oefening enum en struct:

Schrijf een struct `Kaart`.

Een kaart heeft:

- een **kleur** (harten, ruiten, klaveren of schoppen) en (definieer enum `Kleur`)
- een **rang** (2, 3, 4, 5, 6, 7, 8, 9, 10, boer, dame, heer, aas). (definieer enum `Rang`)

Maak een constructor die een willekeurige kaart aanmaakt.

Maak een methode `PrintKaart()` die de kleur en de rang van een kaart weergeeft.

Maak een methode `PrintAlleKaarten()` die de kleur en de rang van alle kaarten weergeeft. Gebruik een array van `Kaart`