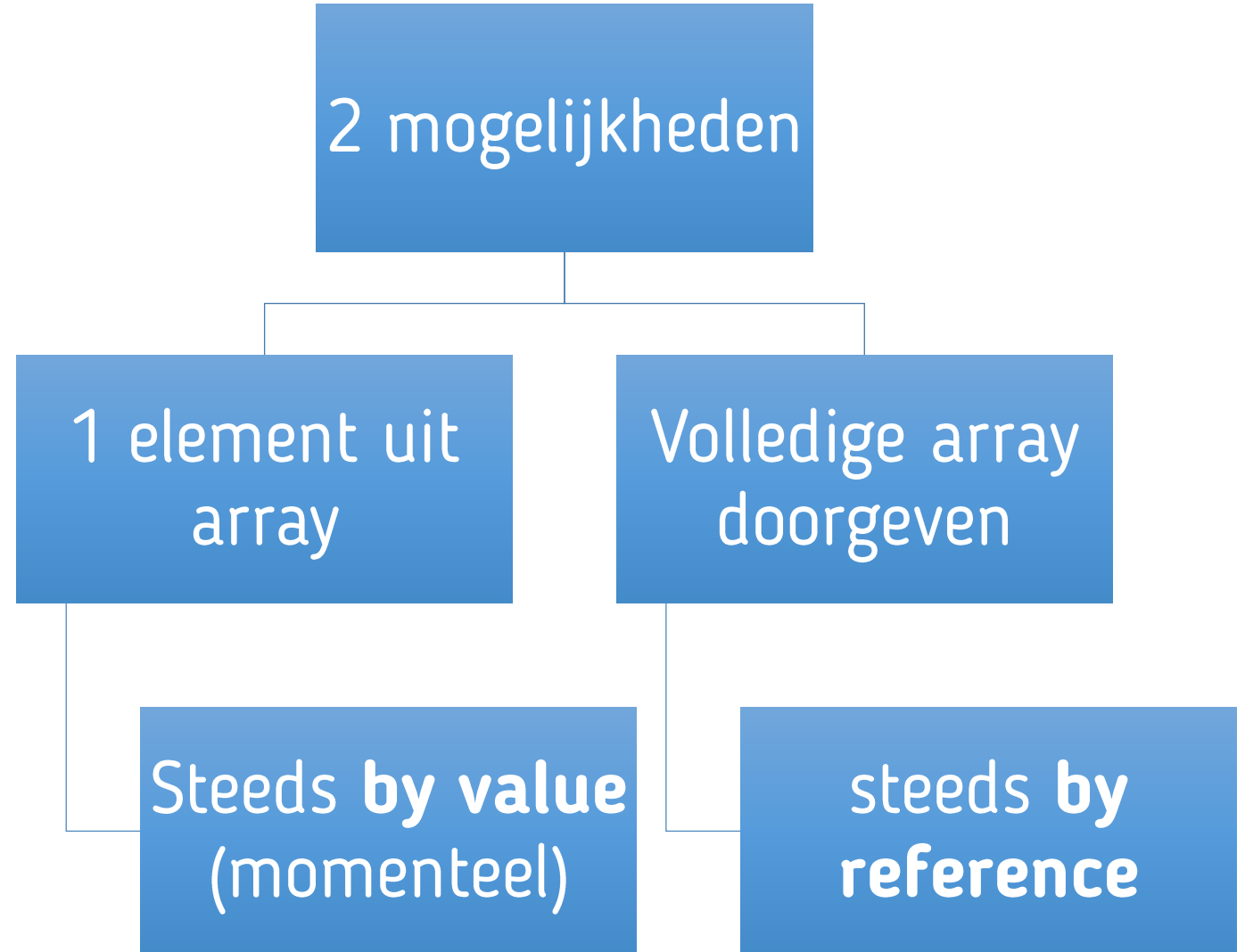


6. Arrays en methoden

H8. Arrays



Arrays doorgeven



Enkel element uit array doorgeven...



Zoals we gewoon
zijn

Alsof het om
een gewone
variabele of
literal gaat



Element wordt by
value meegegeven

Kopie van
waarde
wordt
doorgegeven

Voorbeeld

- Onze bestaande methoden kan je hier voor gebruiken
- Stel, volgende verdubbelingsmethode:

```
static void DoubleValue(int v)
{
    v *= 2;
    Console.WriteLine($"\\n\\nValue of passed parameter doubled in method = {v}\\n");
}
```

Enkel element doorgeven

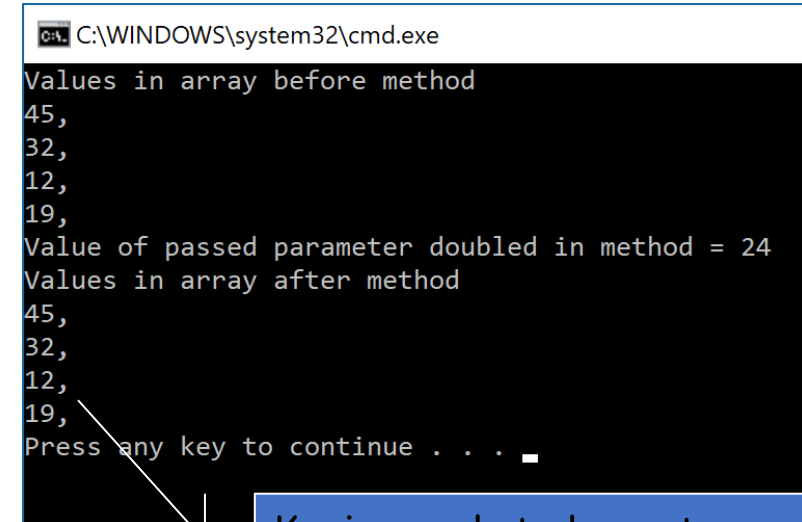
```
static void Main(string[] args)
{
    int[] ages = { 45, 32, 12, 19 };

    Console.WriteLine("Values in array before method");
    for (int i = 0; i < ages.Length; i++)
    {
        Console.Write($"{ages[i]},");
    }

    DoubleValue(ages[2]);

    Console.WriteLine("Values in array after method");
    for (int i = 0; i < ages.Length; i++)
    {
        Console.Write($"{ages[i]},");
    }
}

static void DoubleValue(int v)
{
    v *= 2;
    Console.WriteLine($"{v}\nValue of passed parameter doubled in method = {v}");
}
```



```
C:\WINDOWS\system32\cmd.exe
Values in array before method
45,
32,
12,
19,
Value of passed parameter doubled in method = 24
Values in array after method
45,
32,
12,
19,
Press any key to continue . . .
```

Kopie van het element op plek 2 werd aangepast, niet het element zelf in de array

Hele array doorgeven



Je kan een volledige array
doorgeven



Deze worden **ALTIJD**
by reference
doorgegeven

De methode krijgt
het adres naar de
effectieve array;
en kan dus de
originele array
aanpassen!

Methode signatuur indien array kan ontvangen worden

- De lengte mag niet in signatuur staan!

```
static void DoubleArray(int[] v)
{
    for (int i = 0; i < v.Length; i++)
    {
        v[i] *= 2;
    }
}
```

- Verplicht je om .Length in de methode te gebruiken en dus algemeen bruikbare methoden te schrijven
- **Don't do this!** => `static void DoubleArray(int[] v, int length)`

Voorbeeld van volledige array doorgeven

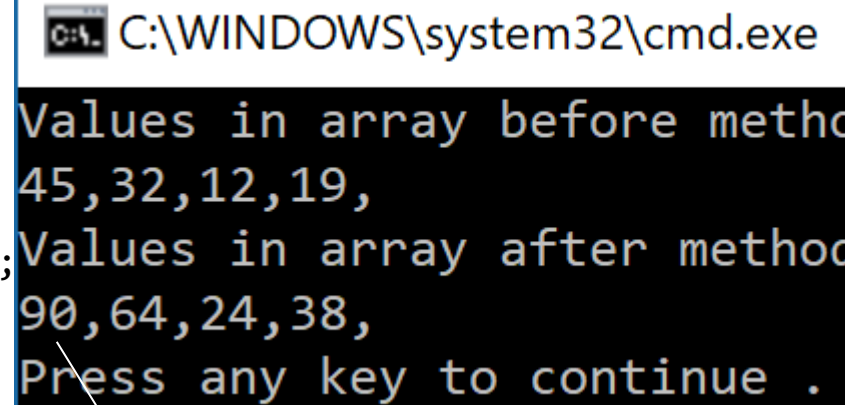
```
static void Main(string[] args)
{
    int[] ages = { 45, 32, 12, 19 };

    Console.WriteLine("Values in array before method");
    for (int i = 0; i < ages.Length; i++)
    {
        Console.Write($"{ages[i]},");
    }

    DoubleArray(ages);

    Console.WriteLine("\nValues in array after method");
    for (int i = 0; i < ages.Length; i++)
    {
        Console.Write($"{ages[i]},");
    }
}

private static void DoubleArray(int[] v)
{
    for (int i = 0; i < v.Length; i++)
    {
        v[i] *= 2;
    }
}
```

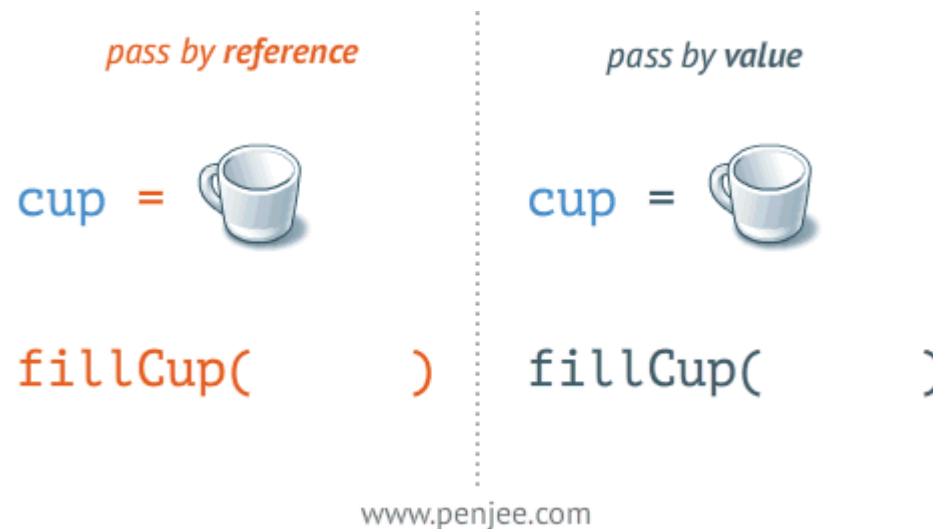


```
C:\WINDOWS\system32\cmd.exe
Values in array before method
45,32,12,19,
Values in array after method
90,64,24,38,
Press any key to continue .
```

Aanpassingen in de array
(by reference) hebben
dus effect op de origineel
meegegeven array

Samenvatting

- 2 mogelijkheden bij arrays en methoden
1. Enkel element uit array=> steeds by **value**
 2. Volledige array meegeven=> steeds by **reference**



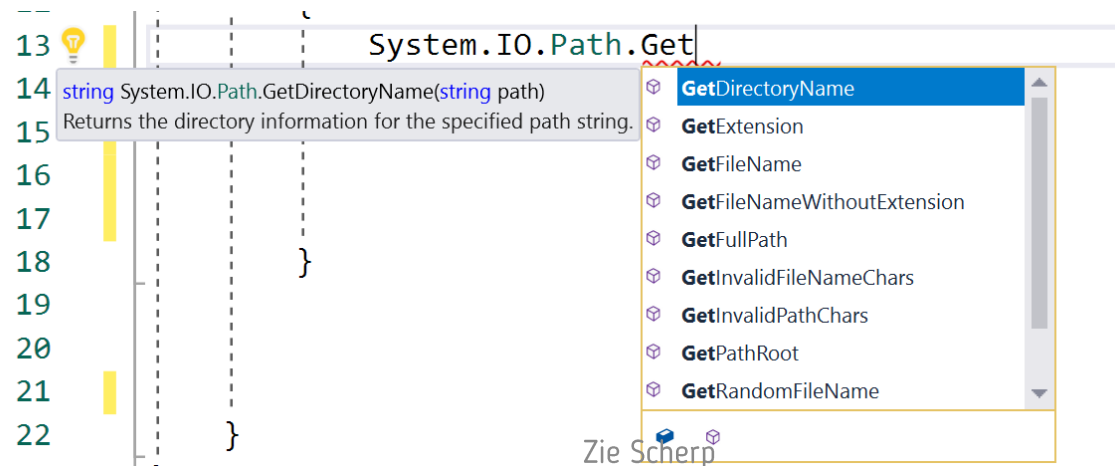
Animatie bekijken in pdf?:

<https://ericlbarnesblog.files.wordpress.com/2016/06/pass-by-reference-vs-pass-by-value-animation.gif>

System.Environment werkt ook met arrays!

```
string[] drives = Environment.GetLogicalDrives();  
string[] dirs= System.IO.Directory.GetDirectories(@"c:\");  
string[] files = System.IO.Directory.GetFiles(@"c:\temp");
```

- Pro-tip: Kijk ook eens naar de werking van System.IO.Path.Getxxxx(); methoden!



Array als returntype

- Geef gewoon aan dat er array terugkomt:

```
static int[] MaakArray(int lengte, int beginwaarde)
{
    int[] resultArray = new int[lengte];
    for (int i = 0; i < lengte; i++)
    {
        resultArray[i] = beginwaarde;
    }
    return resultArray;
}
```

- Voorbeeld aanroep:

```
int[] mijnNieuweArray= MaakArray(4,666);
```

7. Startup parameters

H8. Arrays

