

1

Recursie

In dit OPO bestuderen we twee speciale datastructuren: bomen en grafen. Omdat er hiervoor nogal veel recursief geprogrammeerd moet worden, is het zinvol om aan dit principe wat extra aandacht te besteden. Hieronder vind je de oefeningen van de eerste lesweek.

Oefening 1.1



Implementeer een recursieve methode `fibonacci(int getal)` die het `getal`-de fibonacci-getal berekent (zie https://nl.wikipedia.org/wiki/Rij_van_Fibonacci). Vermoedelijk vind je een eenvoudige en korte versie die echter heel inefficiënt is. Kan je uitleggen waarom?

Oefening 1.2



Schrijf een recursieve methode `somCijfers(int getal)`. Uitvoer is de som van de cijfers van `getal`.

Voorbeeld: $234 \rightarrow 2 + 3 + 4 = 9$

Oefening 1.3



Schrijf een recursieve methode `keerOm(String str)`. Uitvoer is een string waarbij alle karakters van `str` in omgekeerde volgorde voorkomen.

Voorbeeld: `abcd` \rightarrow `dcba`

Uitbreiding: Vind twee versies van het algoritme. In de eerste versie wordt de omgekeerde string van links naar rechts opgebouwd; in de tweede versie van rechts naar links.

Oefening 1.4



Implementeer een recursieve methode `countX(String str)`. Invoer is de string `str`. Uitvoer is het aantal keer dat de letter `x` voorkomt in `str`.

Zie ook <http://codingbat.com/prob/p170371>.

Oefening 1.5



Implementeer een recursieve methode `countHi(String str)`. Invoer is de string `str`. Uitvoer is het aantal keer dat de combinatie `hi` voorkomt in `str`.

Zie ook <http://codingbat.com/prob/p184029>.

Oefening 1.6



Implementeer een recursieve methode `changeXY(String str)`. Invoer is de string `str`. Uitvoer is een nieuwe string waarin elk voorkomen van `x` vervangen wordt door `y`.

Voorbeelden:

- `changeXY("codex") → "codey"`
- `changeXY("xxhixx") → "yyhiyy"`
- `changeXY("xhixhix") → "yhiyhiy"`

Zie ook <http://codingbat.com/prob/p101372>

Oefening 1.7



Implementeer een recursieve methode `changePi(String s)`. Invoer is de string `s`. Uitvoer is een nieuwe string waarin elke deelstring `pi` vervangen wordt door `'3.14'`.

Voorbeelden:

- `changePi("xpix") → "x3.14x"`
- `changePi("pipi") → "3.143.14"`
- `changePi("pip") → "3.14p"`

Zie ook <http://codingbat.com/prob/p170924>.

Oefening 1.8



We zeggen dat de logaritme met grondtal 2 van het getal 8 gelijk is aan 3 omdat $2^3 = 2 \cdot 2 \cdot 2 = 8$. De tweelog van 256 is 8 want $2^8 = 256$. Schrijf een recursieve functie `tweelog(int x)`. Je mag uitgaan van de veronderstelling dat x een macht van 2 is.

Oefening 1.9



Schrijf een recursieve methode `findMaximum(List<double> lijst)` die het grootste getal van `lijst` teruggeeft.

Oefening 1.10



Schrijf een recursieve methode `findSubstrings(String string)` die een lijst teruggeeft met alle mogelijke combinaties van de letters van `string`. Let op: Je hoeft geen rekening te houden met de volgorde van de letters. De combinatie `abc` beschouwen we gelijk aan de combinatie `cab`. Voorbeeld:

- Mogelijke combinaties van de letters `abc` zijn: `[a, b, c, bc, ab, ac, abc]`

Oefening 1.11



Schrijf een recursieve functie `aantalKaarten(int n)` die berekent hoeveel kaarten er nodig zijn voor een kaartenhuisje van n verdiepingen. Een kaartenhuisje wordt opgebouwd zoals getoond in figuur 1.1



Figuur 1.1 Tekening bij opgave 1.11

Voorbeeld:

1 Recursie

- een kaartenhuisje van 1 verdieping = 2 kaarten
- een kaartenhuisje van 2 verdiepingen = 7 kaarten
- een kaartenhuisje van 3 verdiepingen = 15 kaarten
- een kaartenhuisje van 12 verdiepingen = 222 kaarten
- een kaartenhuisje van 20 verdiepingen = 610 kaarten

Meer oefenmateriaal nodig?

<http://codingbat.com/java/Recursion-1>

en <http://codingbat.com/java/Recursion-2>.