Practicum OO — lente 2015 **Opgave 7**

1 Veeltermen

Veeltermen of polynomen (*polynomials* in het Engels) worden niet enkel in de wiskunde maar ook in andere wetenschapsgebieden, zoals de natuurkunde, toegepast. Zo'n veelterm bestaat uit een som van *termen*, waarbij iedere term van de vorm cx^n is. Hierin is de *factor* c een reëel getal ongelijk aan 0 en de *exponent* n een natuurlijk getal. Alle exponenten in de veelterm zijn verschillend. De hoogste exponent die in de veelterm voorkomt noemt men wel de *graad* van de veelterm. Bijvoorbeeld $3x^4 - 4x^3 + 2x - 8$ is een veelterm van graad 4.

2 Leerdoelen

In deze opgave vragen we je een Java klasse voor veeltermen te ontwerpen, te implementeren én te testen. Na afloop van deze opdracht ben je in staat om:

- De `ArrayList`-klasse te gebruiken om andere samengestelde datatypes te implementeren;
- met behulp van *(list)iterators* de elementen van zo'n samengesteld datatype te inspecteren, nieuwe elementen toe te voegen of (overbodige) elementen te verwijderen;
- zogenaamde 'unit tests' te bedenken en uit te voeren vóór, tijdens en na de ontwikkeling van je programma.

3 Instructie

Bestudeer allereerst de onderdelen die op college zijn besproken en raadpleeg eventueel de online Java API Specificaties van de `ArrayList`-klasse en de (`List`) `Iterator`-interface. De `ArrayList`-klasse implementeert de `List`-interface die de methodes `iterator` en `listIterator` bevat. Zorg ervoor, dat je (op papier) al een ontwerp van je veelterm operaties hebt gemaakt, alvorens je aanschuift achter de pc om je code in te voeren en uit te testen.

4 Probleemschets

Een veelterm wordt gerepresenteerd door een klasse genaamd `Polynomial`. Omdat een veelterm willekeurig veel termen kan bevatten gebruiken we `List`s om de termen op te slaan. We staan hierbij geen termen toe waarvan de factor 0 is en ook meerdere termen met dezelfde exponent moeten worden uitgesloten. Bovendien blijkt het handig te zijn om termen in volgorde van oplopende exponenten op te slaan.

Om je op weg te helpen vinden jullie op Blackboard al een raamwerk van de klasse `Polynomial` welke gebruik maakt van de klasse `Term`. De `Polynomial` klasse bevat o.a. drie constructoren: één (zonder argumenten) voor het aanmaken van een 'lege veelterm' (de zogenaamde *nulveelterm*), één copy constructor en een constructor die 'n string als parameter meekrijgt waarin de termen alle (als een reeks van afwisselende factoren en exponenten) zijn aangegeven. Bekijk de voorgegeven code om te begrijpen hoe dit in z'n werk gaat. Zoals je kunt zien maakt deze constructor gebruik van de *statische* hulpmethode `scanTerm` uit de klasse `Term`. De precieze werking van deze hulpmethode is misschien niet meteen duidelijk, maar dat is voor het maken van de opdracht ook niet noodzakelijk. Beide voorgegeven klassen kun je als basis voor je verdere implementatie gebruiken.

In dit raamwerk zijn eveneens al de headers voorgegeven van de standaard `equals` methode en de operatoren `+`, $-$, $*$ en `/`, waarmee resp. twee veeltermen kunnen worden opgeteld, afgetrokken, vermenigvuldigd en gedeeld. De deling is slechts voor de volledigheid toegevoegd. Je hoeft deze alleen te implementeren als je nog een extra uitdaging zoekt nadat je de rest van de opdracht al hebt afgerond. Deze operaties zijn voorbeelden van zogenaamde *symbolische manipulaties*, bijvoorbeeld $(x^3 + 2x^2) + (x^3 - 2x^2 + x) = 2x^3 + x$. In tegenstelling tot C++ kent Java geen *operator overloading*. Dat betekent dat je in Java niet je eigen versie van bijvoorbeeld de + kunt definiëren. Verder kun je in deze opdracht kunt het beste ieder van deze operatoren *destructief* implementeren. Bijvoorbeeld, de optelling `plus` met volgende signatuur:

```
public void plus( Polynomial p )
```

dient de als parameter meegegeven polynoom `p` bij het huidige object op te tellen. Naast de bodies van deze operaties ontbreekt ook het commentaar (in JavaDoc-formaat). Dit dien je in je uitwerking natuurlijk toe te voegen.

Deelopdrachten

1. Maak in NetBeans (of Eclipse) een project en voeg daar de voorgegeven bestanden aan toe. Hierbij is het handig om `Polynomial` en `Term` in een apart package te plaatsen, zeg `polynomials`. Genereer meteen een testpackage voor `polynomials` (in NetBeans: **Tools** $\rightarrow$ Create/Update Tests).

2. Bedenk voor iedere operatie/methode voorbeelden die karakteristiek zijn voor met name de randgevallen. Bijvoorbeeld kunnen bij de optelling termen verdwijnen doordat hun coëfficient 0 geworden is. Controleer of jouw implementatie hier rekening mee houdt. Relateer ook de verschillende operaties aan elkaar in de tests: aftrekken is hetzelfde als vermenigvuldigen met -1 en optellen. Controleer bovendien de eigenschap dat vermenigvuldiging distribueert over optelling. Sommige operatoren zijn *commutatief* en/of *associatief*. Test dit eveneens.

3. Implementeer je test-cases door aan de gegenereerde testklassen hiervoor geschikte testcode toe te voegen.

4. Implementeer nu de volgende operaties:

   (a) De methode `toString` toe die een tekstrepresentatie van de veelterm oplevert.

   (b) De operatoren `plus`, `minus` en `times`. Probeer in je implementatie het gebruik van indices om termen uit de lists te selecteren te vermijden door enkel met (list)iterators te werken.

   (c) Een methode `apply` die de waarde van een veelterm voor een gegeven argument (een voor x ingevulde waarde) oplevert. Dus voor voor de veelterm $p = 3x^4 - 4x^3 + 2x - 8$ geldt dat $p.\texttt{apply}(2) = 48 - 32 + 4 - 8 = 12$.

5 Producten

Als producten moet alle `.java` bestanden ingeleverd worden. Nogmaals, in je uitwerking dien je je te houden aan de basisregels van JavaDoc. Dat betekent dat er per klasse minstens een regel commentaar is waarin de bedoeling van die klasse staat, dat er een `@author` is voor de auteur(s) van de klasse (inclusief studentnummers), en voor iedere methode naast een korte omschrijving van de functie, een `@param` voor de argumenten van deze methode een `@return` voor het resultaat. Het ontbreken van elementaire documentatie kan door de assistenten worden beschouwd als een niet serieuze inleverpoging!

6 Inleveren van je producten

Vóór zondag 22 maart, 11:00 uur, via Blackboard. Lever alle `.java` files uit je project in (dus ook de bestanden die zijn voorgegeven).