HO GENT	**Hogeschool Gent – Faculteit Bedrijf en organisatie FBO** **Toegepaste Informatica 1e jaar – Academiejaar 2019 - 2020** **OEFENINGEN OO Programmeren I – Hoofdstuk 6_8_deel1**

1. Menu-applicatie

Schrijf een Java-applicatie waarin je gebruik maakt van een methode **toonMenuEnGeefKeuze** die het onderstaande menu toont en de keuze van de gebruiker vraagt.

```
run:
Menu:
1) Begroeting
2) Huidige datum
3) Naar hexadecimaal
0) Stoppen
Geef uw keuze:
```

Bij een ongeldig antwoord wordt het menu opnieuw getoond. Een geldig antwoord wordt als een int waarde door de methode **toonMenuEnGeefKeuze** teruggeven aan de oproeper (methode **main**). De **main** methode van de applicatie zorgt ervoor dat de gebruiker telkens opnieuw (nadat de gekozen taak is uitgevoerd) een keuze uit het menu kan maken totdat er voor 0 gekozen wordt. Je voorziet voor elke taak een methode, nl: **geefBegroeting**, **toonHuidigeDatum** en **zetOmNaarHexadecimaal**. Voor het omzetten naar een hexadecimaal getal dient eerst (in de main-methode) nog een getal te worden ingegeven. Methode **toonResultaat** brengt een String op het scherm.

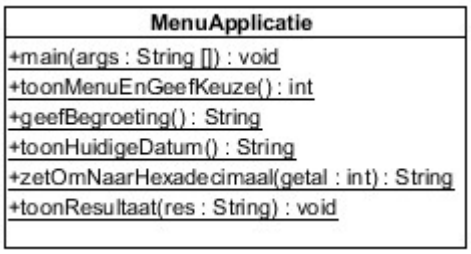
<u>Mogelijke uitvoer van toonResultaat()</u>

keuze 1:
```
Hallo ....
```

keuze 2 (zie appendix I):
```
woensdag, 13 september, 2017
```

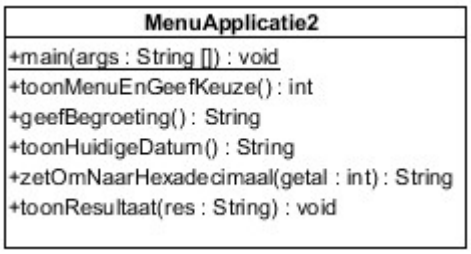
keuze 3 (zie appendix I):
```
Geef een getal: 11
hexadecimaal van 11 is B
```

Het ontwerp van de klasse ziet er uit als volgt:



Extra (Zorg eerst dat je oefening volledig werkt!)

Pas de code aan zodanig dat je code overeenkomt met volgend ontwerp.
Wat moet je precies aanpassen? Zorg dat je dit goed begrijpt!



2. Achterstevoren

In de **main**-methode lees je een geheel getal in. Dit getal wordt meegegeven als parameter van de methode **draaiOm**, die het getal achterstevoren teruggeeft. Dit nieuwe getal wordt tenslotte getoond.

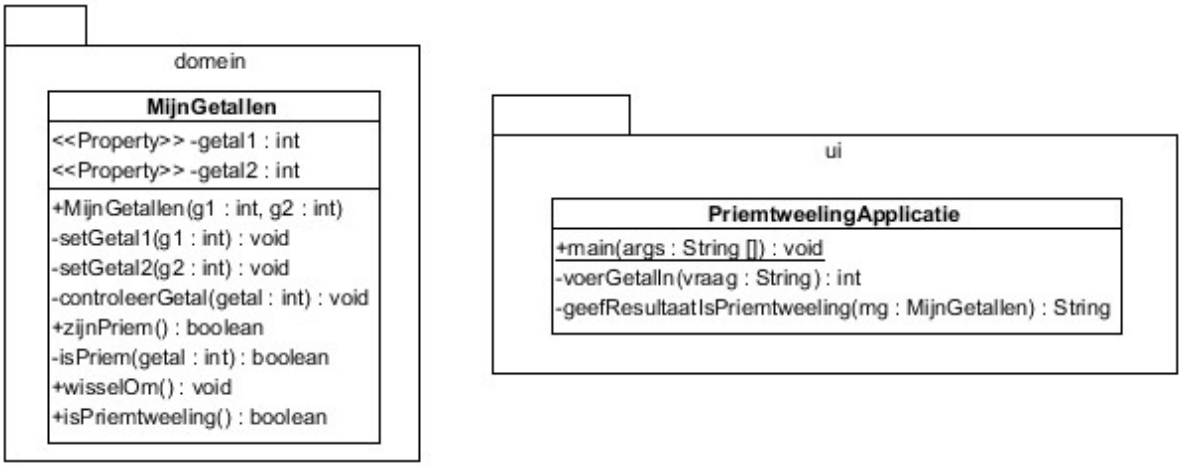
De methode **draaiOm** werkt **enkel met wiskundige bewerkingen** en geeft een getal terug dat gelijk is aan het oorspronkelijke getal van achter naar voor gelezen.

Voorbeeld uitvoer:

```
run:
Geef een getal: 1234
Het getal achterstevoren is: 4321
BUILD SUCCESSFUL (total time: 2 seconds)

Geef een getal: 2567652
Het getal achterstevoren is: 2567652
BUILD SUCCESSFUL (total time: 8 seconds)
```

3. Herschrijf oefening 4 van H4.



Als p en q beide priemgetallen zijn en q = p + 2, dan wordt het paar (p,q) een priemtweeling genoemd (bv. (3,5)).

Maak een applicatie waarin 2 getallen worden ingevoerd (**methode voerGetalIn tweemaal oproepen. Het ingevoerde getal moet strikt positief zijn**). Deze 2 getallen worden gebruikt om een object te maken van de klasse MijnGetallen.

Via de methode isPriemtweeling gaan we na of de 2 getallen een priemtweeling vormen (**methode geefResultaatIsPriemtweeling: geeft een String met het resultaat terug**).
Het resultaat wordt op het scherm getoond in de main-methode.

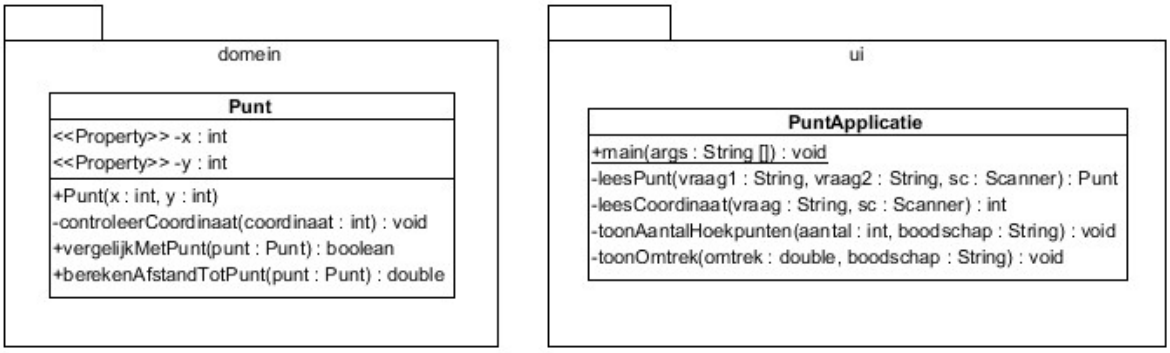
Methoden applicatie:

- main
- voerGetalIn
- geefResultaatIsPriemtweeling

Hergebruik de domeinklasse MijnGetallen

- Herschrijf de setters van getal1 en getal2 zodat deze gebruik maken van een **hulpmethode controleerGetal** die de controle op het positief zijn van de getallen doet
- Herschrijf de methode isPriemtweeling (maak hierbij gebruik van de methodes wisselOm en zijnPriem!)
- Hulpmethode wisselOm: wisselt de waarden om van de attributen
- Hulpmethode zijnPriem: controleert of de getallen uit de attributen al dan niet priemgetallen zijn.
- Hulpmethode isPriem: controleert of een getal (meegegeven als parameter) al dan niet een priemgetal is.

4. Puntapplicatie



Maak een domeinklasse **Punt** met volgende gegevens:

- Attributen: x en y (integers)
- Methoden :
  - constructor met 2 parameters
  - getters & setters (elke coördinaat moet >=0)
  - methode vergelijkMetPunt: bepaalt of het gegeven Punt dezelfde x- en y-coördinaat heeft als het punt dat meegegeven werd als parameter
  - methode geefAfstandTotPunt: berekent de afstand tussen het gegeven Punt (veronderstel met coördinaten (x1, y1)) en het punt dat als parameter (veronderstel met coördinaten (x2,y2)) werd meegegeven volgens de formule:

$$afstand = \sqrt{(x2 - x1)^2 + (y2 - y1)^2}$$

Maak de applicatieklasse **PuntApplicatie** met de volgende methoden:

- main
- leesPunt: leest een punt in adhv 2 vragen om 1 coördinaat in te lezen
- leesCoordinaat: leest een coördinaat in adhv de meegegeven vraag
- toonAantalHoekpunten: toont het aantal hoekpunten
- toonOmtrek: toont de omtrek met 2 cijfers na de komma

In de main-methode wordt eerst het beginHoekpunt ingelezen via de methode leesPunt (die tweemaal leesCoordinaat oproept om het x- en y-coördinaat van het punt in te lezen). Dit hoekpunt wordt ook gekopieerd in een variabele vorigHoekpunt om straks de afstand tussen dit punt en het volgende punt te kunnen berekenen.

Daarna wordt in een lus telkens een nieuw hoekpunt ingelezen (telkens met leesPunt) en het aantalHoekpunten en de omtrek tot op dat punt worden bijgehouden. Hiervoor dient het aantalHoekpunten met 1 verhoogd te worden en bij de omtrek de nieuwe zijde te worden bijgeteld. Voor dit laatste gebruik je de methode berekenAfstandTotPunt die voor de zijde de afstand tussen vorigHoekpunt en hoekpunt moet bepalen. Tenslotte vervang je het vorigHoekpunt door het huidig hoekpunt. Dit blijf je herhalen zolang het nieuwe hoekpunt verschillend is van het beginHoekpunt, wat je kan testen met de methode vergelijkMetPunt.

Voorbeeld uitvoer applicatie: