HO
GENT

H6 Methodes - next level - Oefeningen

Table of Contents

1. Doelstellingen . . . 1
2. Oefeningen . . . 1
    2.1. BTW-nummer . . . 1
    2.2. Temperatuurconversie . . . 1
    2.3. Priemgetal . . . 3
    2.4. Breuk . . . 4
    2.5. Stappenteller . . . 5
    2.6. Datum . . . 8
    2.7. Arraybewerkingen . . . 9
    2.8. Temperatuur . . . 11
    2.9. Studenten . . . 12
    2.10. Lotto . . . 15
    2.11. Datum . . . 17
    2.12. Vierkant . . . 19

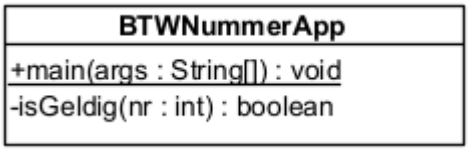
1. Doelstellingen

- Oefening 2.1 t.e.m. 2.4: Het gebruik van static en non-static methodes inoefenen in een applicatieklasse
- Oefening 2.5: Het gebruik van static en non-static methodes inoefenen in een applicatie met een domeinklasse
- Oefening 2.6 t.e.m. 2.8, 2.11 en 2.12: Het gebruik van static en non-static methodes met arrays inoefenen in een applicatie met een domeinklasse
- Oefening 2.9 en 2.10: Het gebruik van static en non-static methodes met arrays inoefenen in een applicatieklasse

2. Oefeningen

> Bij de oefeningen met een domeinklasse is telkens een **testklasse** voorhanden zodat je de code van het domein kan testen alvorens aan de implementatie van de applicatie te beginnen. Deze testklasses vind je op Chamilo.

2.1. BTW-nummer



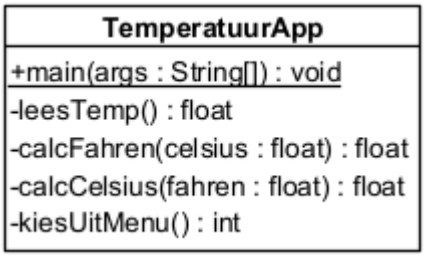
Een BTW-nummer is altijd van de vorm ABC.PQR.XYZ. Het is geldig als de som van de rest van de deling van het getal ABCPQRX door 97, en het getal YZ terug 97 is.

Voorbeeld:
BTW-nummer 200.858.492 is geldig, want 2008584/97 = 20707 rest 5 en 5 + 92 = 97.
Controleer of een ingevoerd BTW-nummer geldig is. Dit gebeurt met behulp van de methode `boolean isGeldig (int nr)`. Schrijf hiervoor een applicatie.

```
Geef een btw-nummer van de vorm ABC.PQR.XYZ zonder de puntjes: 123456789
123456789 is een ongeldig btw-nummer
```

```
Geef een btw-nummer van de vorm ABC.PQR.XYZ zonder de puntjes: 200858492
200858492 is een geldig btw-nummer
```

2.2. Temperatuurconversie



Maak een applicatie met het volgende menu:

1) Omzetten naar Fahrenheit
2) Omzetten naar Celcius
3) Stoppen

Zorg ervoor dat bij elke invoer van de gebruiker dit menu duidelijk zichtbaar is. Kiest de gebruiker voor 1 of 2 dan wordt vervolgens een temperatuur gevraagd. Het programma wordt afgesloten bij invoer van drie.

De methode **kiesUitMenu** drukt het menu af, vraagt om een cijfer in te geven (invoercontrole) én geeft het gekozen cijfer terug. De applicatie blijft dit menu telkens opnieuw tonen totdat de gebruiker kiest voor optie 3 (Stoppen).

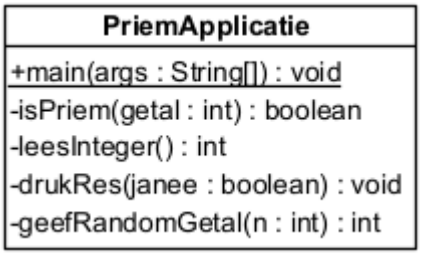
Voor de conversie maak je gebruik van de methode `float calcFahren(float celsius)` of `float calcCelsius(float fahren)`.

De formule: F = 32 + (C * 9)/5.

Let op : het berekende resultaat moet steeds met één cijfer na de komma getoond worden.

```
Menu
1) Omzetten naar Fahrenheit
2) Omzetten naar Celcius
3) Stoppen
Wat kies je? 4
Ongeldige keuze
Wat kies je? 0
Ongeldige keuze
Wat kies je? 1
Geef een temperatuur: 25
25,0 in Celsius komt overeen met 77,0 in Fahrenheit
Menu
1) Omzetten naar Fahrenheit
2) Omzetten naar Celcius
3) Stoppen
Wat kies je? 2
Geef een temperatuur: 77
77,0 in Fahrenheit komt overeen met 25,0 in Celsius
Menu
1) Omzetten naar Fahrenheit
2) Omzetten naar Celcius
3) Stoppen
Wat kies je? 3
```

2.3. Priemgetal



Schrijf een applicatie die opgesplitst is in 5 methodes:

- geefRandomGetal: bepaalt een randomgetal tussen 0 en n
- isPriem: gaat na of een geheel getal een priemgetal is
- leesInteger: leest een positief geheel getal n in (+ invoercontrole)
- drukRes: drukt af of het randomgetal een priemgetal is
- main-methode:
  - vraagt om een geheel getal n via de methode leesInteger,
  - roept vervolgens de methode geefRandomGetal aan om een randomgetal te bepalen tussen 0 en n,
  - controleert via de methode isPriem of dit getal een priemgetal is
  - en drukt vervolgens het resultaat af (via de drukRes-methode)

```
Geef een strikt positief geheel getal: -1
Geef een strikt positief geheel getal: 12
Het random bepaalde getal tussen 0 en 12 is 3
Dit getal is een priemgetal

Geef een strikt positief geheel getal: 30
Het random bepaalde getal tussen 0 en 30 is 21
Dit getal is geen priemgetal
```

2.4. Breuk

Lees de teller en de noemer in van een breuk. Bereken de grootste gemene deler van beide waarden en vereenvoudig de breuk, m.a.w. deel teller en noemer door de gevonden grootste gemene deler.

Voor de berekening van de grootste gemene deler kan je het algoritme van Euclides gebruiken:

- ggd(x,y) = x als y = 0
- ggd(y,x%y) als y <> 0

Voor de teller van de breuk mag om het even welk geheel getal gebruikt worden; de noemer mag niet 0 zijn (invoercontrole). De methode berekenGrootsteGemeneDeler is recursief!

Je toont de grootst gemene deler en de (indien mogelijk) vereenvoudigde breuk.

```
Geef een geheel getal voor de teller van de breuk: 8
Geef een geheel getal verschillend van nul voor de noemer van de breuk: 0
Geef een geheel getal verschillend van nul voor de noemer van de breuk: 6
De grootste gemene deler van 8 en 6 is 2
We kunnen de breuk 8 / 6 dus vereenvoudigen tot 4 / 3
```

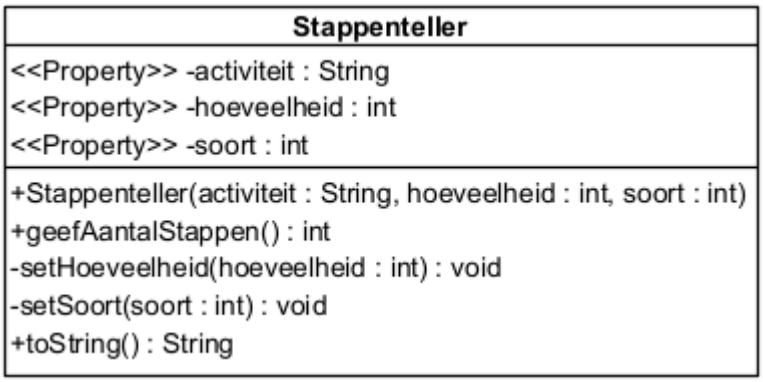
```
Geef een geheel getal voor de teller van de breuk: -55
Geef een geheel getal verschillend van nul voor de noemer van de breuk: -140
De grootste gemene deler van -55 en -140 is -5
We kunnen de breuk -55 / -140 dus vereenvoudigen tot 11 / 28
```

```
Geef een geheel getal voor de teller van de breuk: 8
Geef een geheel getal verschillend van nul voor de noemer van de breuk: 15
De grootste gemene deler van 8 en 15 is 1
We kunnen de breuk 8 / 15 dus niet vereenvoudigen
```

2.5. Stappenteller

We maken een applicatie om het aantal stappen dat we op één dag hebben gezet uit te rekenen. Hierbij is het de bedoeling dat we per uur een activiteit kunnen opgeven plus het aantal stappen dat we hierbij gezet hebben. Bovendien moeten we rekening houden met hoogteverschillen: als we klimmen dan mogen we de stappen dubbel tellen, maar als we een dalende beweging maken, dan tellen onze stappen maar voor de helft mee.

Domeinklasse:



In de domeinklasse Stappenteller komen **drie attributen**: activiteit (een tekst die aangeeft wat je gedaan hebt), hoeveelheid (een geheel getal dat bepaalt hoeveel stappen je daarbij gezet hebt) en soort (het soort activiteit, aangeduid met 1 voor een activiteit op vlakke grond, 2 voor een activiteit bergop en 3 voor een activiteit bergaf).

De **constructor** gebruikt de setters om de parameterwaarden te controleren en - indien mogelijk - in te stellen.

De **getters** van de attributen geven de bijhorende waarde terug. Voor de **setter** van activiteit is geen controle nodig, de overige setters controleren eerst of de opgegeven waarde voldoet, daarna wordt de waarde (eventueel) ingesteld. Indien de parameterwaarde van de setter niet voldoet, wordt een foutmelding gegooid. Voor het attribuut hoeveelheid moet de waarde tussen 0 (niet inbegrepen) en 1000 (inbegrepen) liggen. Voor het attribuut soort mag enkel 1, 2 of 3 ingevuld worden.

De extra methode **geefAantalStappen** berekent het aantal stappen als volgt: voor soort 1 is het aantal stappen hetzelfde als de hoeveelheid, voor soort 2 is het aantal stappen het dubbel van de hoeveelheid en voor soort 3 bedraagt het eigenlijke aantal stappen slechts de helft van de hoeveelheid.

De **tekstweergave** van een Stappenteller-object wordt opgebouwd als volgt: de activiteit in een veldbreedte van 40, gevolgd door het aantal stappen in een veldbreedte van 10.

User interface:

In de applicatieklasse StappentellerApp zijn er 3 methodes aanwezig.

De static methode **main** roept eerst de methode voerUurIn twee keer aan om het begin- en einduur van de dag op te vragen. Het beginuur mag tussen 0 en 23 liggen, het interval waartoe het einduur moet behoren, is [beginuur, 23]. Vervolgens wordt er voor elk uur tussen begin en einde een stappenteller gemaakt met de methode maakStappenteller. Hiermee kan je dan het aantal effectieve stappen bepalen. Je houdt ook het totaal aantal stappen bij. De uitvoerstring wordt in de main opgebouwd en uiteindelijk getoond in een overzichtelijk tabelformaat.

De non-static methode **voerUurIn** stelt een vraag aan de gebruiker en leest een uur tussen 2 opgegeven grenzen in. De gebruiker krijgt de grenswaarden te zien, zodat hij een juiste waarde kan invoeren. Wanneer het uur toch niet aan de voorwaarden voldoet, wordt het steeds opnieuw opgevraagd.

De non-static methode **maakStappenteller** maakt een Stappenteller aan voor een bepaald uur en geeft deze terug. Om de Stappenteller te kunnen maken, worden achtereenvolgens de activiteit (wat?), het aantal stappen (hoeveel?) en de soort activiteit opgevraagd. Het aantal stappen kan tussen 1 en 1000 liggen (grenzen inbegrepen), de soort is 1 voor een activiteit op vlakke grond, 2 voor een activiteit bergop (telt dubbel!) en 3 voor een activiteit bergaf (telt maar voor de helft!). Bij foute invoer wordt de invoer opnieuw gevraagd. Met behulp van de 3 ingevoerde gegevens wordt de Stappenteller gemaakt.

```
Geef het uur waarop je bent opgestaan (tussen 0 en 23): -1
Geef het uur waarop je bent opgestaan (tussen 0 en 23): 24
Geef het uur waarop je bent opgestaan (tussen 0 en 23): 11
Geef het uur waarop je bent gaan slapen (tussen 11 en 23): 10
Geef het uur waarop je bent gaan slapen (tussen 11 en 23): 20
Wat heb je om 11u gedaan? opstaan - trap naar beneden
Hoeveel stappen heb je gezet (max. 1000)? 39
Geef 1 voor een activiteit op vlakke grond,
2 voor een activiteit bergop en 3 voor een activiteit bergaf.
Welk soort activiteit was dit? 3
Wat heb je om 12u gedaan? brunch klaarmaken
Hoeveel stappen heb je gezet (max. 1000)? 27
Geef 1 voor een activiteit op vlakke grond,
2 voor een activiteit bergop en 3 voor een activiteit bergaf.
Welk soort activiteit was dit? 1
Wat heb je om 13u gedaan? brunch eten
Hoeveel stappen heb je gezet (max. 1000)? 5
Geef 1 voor een activiteit op vlakke grond,
2 voor een activiteit bergop en 3 voor een activiteit bergaf.
Welk soort activiteit was dit? 1
Wat heb je om 14u gedaan? wandeling maken
Hoeveel stappen heb je gezet (max. 1000)? 732
Geef 1 voor een activiteit op vlakke grond,
2 voor een activiteit bergop en 3 voor een activiteit bergaf.
Welk soort activiteit was dit? 1
Wat heb je om 15u gedaan? boek lezen
Hoeveel stappen heb je gezet (max. 1000)? 11
Geef 1 voor een activiteit op vlakke grond,
2 voor een activiteit bergop en 3 voor een activiteit bergaf.
Welk soort activiteit was dit? 1
Wat heb je om 16u gedaan? schoonmaken
Hoeveel stappen heb je gezet (max. 1000)? 478
Geef 1 voor een activiteit op vlakke grond,
2 voor een activiteit bergop en 3 voor een activiteit bergaf.
Welk soort activiteit was dit? 1
Wat heb je om 17u gedaan? avondeten klaarmaken
Hoeveel stappen heb je gezet (max. 1000)? 18
Geef 1 voor een activiteit op vlakke grond,
2 voor een activiteit bergop en 3 voor een activiteit bergaf.
Welk soort activiteit was dit? 1
```

```
Wat heb je om 18u gedaan? avondmaal eten
Hoeveel stappen heb je gezet (max. 1000)? 9
Geef 1 voor een activiteit op vlakke grond,
2 voor een activiteit bergop en 3 voor een activiteit bergaf.
Welk soort activiteit was dit? 1
Wat heb je om 19u gedaan? tv kijken
Hoeveel stappen heb je gezet (max. 1000)? 12
Geef 1 voor een activiteit op vlakke grond,
2 voor een activiteit bergop en 3 voor een activiteit bergaf.
Welk soort activiteit was dit? 1
Wat heb je om 20u gedaan? gaan slapen - trap omhoog
Hoeveel stappen heb je gezet (max. 1000)? 39
Geef 1 voor een activiteit op vlakke grond,
2 voor een activiteit bergop en 3 voor een activiteit bergaf.
Welk soort activiteit was dit? 2

 UUR                              ACTIVITEIT  #STAPPEN

  11              opstaan - trap naar beneden        19
  12                        brunch klaarmaken        27
  13                              brunch eten         5
  14                          wandeling maken       732
  15                               boek lezen        11
  16                              schoonmaken       478
  17                     avondeten klaarmaken        18
  18                            avondmaal eten         9
  19                                 tv kijken        12
  20               gaan slapen - trap omhoog        78

                                      TOTAAL      1389
```

2.6. Datum

Maak een applicatie die meerdere data van de vorm DDMM controleert, met behulp van een controlemethode. DD staat voor het nummer van de dag in de maand, MM staat voor het maandnummer. Het nummer van de dag kan soms uit slechts 1 cijfer bestaan, het maandnummer bestaat altijd uit 2 cijfers.

Na het ingeven van een datum, krijg je te zien of dit een geldige of ongeldige datum is.

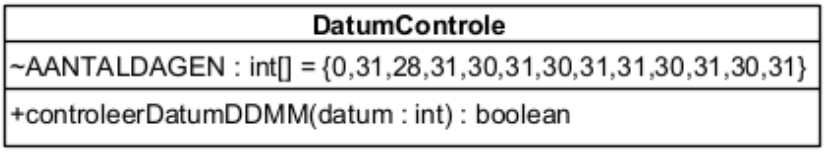
Nadien krijg je terug een kans om een datum in te geven, tenzij je als datum 1313 ingaf bij de vorige beurt. Bij het ingeven van 1313 stopt de applicatie (zonder aan te geven dat dit een ongeldige datum is).

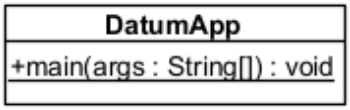
Gebruik bij de controle de onderstaande lokale constante array. Deze houdt per maand het maximum aantal dagen bij.

```
final int AANTALDAGEN[] = {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
```

Gegeven:

- volgende klassediagrammen voor domein en cui:





- de testklasse voor de domeinklasse: zie Chamilo
- voorbeelduitvoer:

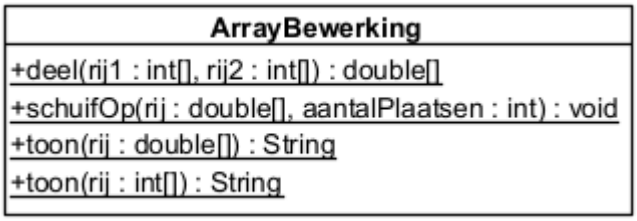
```
Geef datum (DDMM)(STOP met 1313): 1213
De datum 1213 is ongeldig
Geef datum (DDMM)(STOP met 1313): 811
De datum 811 is geldig
Geef datum (DDMM)(STOP met 1313): 14
De datum 14 is ongeldig
Geef datum (DDMM)(STOP met 1313): 2902
De datum 2902 is ongeldig
Geef datum (DDMM)(STOP met 1313): 2802
De datum 2802 is geldig
Geef datum (DDMM)(STOP met 1313): 12323
De datum 12323 is ongeldig
Geef datum (DDMM)(STOP met 1313): 1313
```

2.7. Arraybewerkingen

Maak volgende applicatie die een aantal arraybewerkingen uitvoert. Op deze manier kan je dus ook testen of je H5 ivm arrays goed onder de knie hebt!

Gegeven:

- volgende klassediagrammen:



- de testklasse voor de domeinklasse ArrayBewerking (zie Chamilo)

Gevraagd:

- Werk de klasse ArrayBewerking, die een aantal bewerkingen met arrays bevat, uit:
  - Methode deel:
    Deze methode krijgt twee even lange rijen (rij1 en rij2) met gehele getallen als parameter door. De methode geeft een derde rij (ook weer even lang) terug, waarin het quotiënt van de

respectievelijke elementen van rij1 en rij2 staat.
Het eerste element van rij1 wordt bijvoorbeeld gedeeld door het eerste element van rij2 en wordt in het eerste element van rij3 als een decimaal getal (double), bewaard.
Vb. Stel dat rij1 = {2, 7, 1, 9, 4} en rij2 = {3, 8, 6, 4, 10}, dan is rij3 gelijk aan { 2/3, 7/8, 1/6, 9/4, 4/10 }

- Methode schuifOp:
  Deze methode roteert de getallen in de array met een aantal plaatsen.
  Vb. rij3 = { 2/3, 7/8, 1/6, 9/4, 4/10 } en het aantal plaatsen is 2 dan worden de elementen twee plaatsen naar rechts verplaatst, dus we verkrijgen rij3 = { 9/4, 4/10, 2/3, 7/8, 1/6 }
- Methode toon met als parameter een rij kommagetallen:
  Deze methode geeft een geformateerde tekst terug van de getallen in de array.
  De getallen worden telkens genoteerd met 2 cijfers na de komma.
  De getallen worden rechts gealigneerd in een veld met een breedte van 10 posities.
- Methode toon met als parameter een rij gehele getallen:
  Deze methode geeft een geformateerde tekst terug van de getallen in de array.
  De getallen worden rechts gealigneerd in een veld met een breedte van 10 posities.

- Na het succesvol testen van de klasse ArrayBewerking, schrijf je ook de klasse InvoerApp:
  - Methode invoerRij:
    Deze methode zal de array, meegegeven als parameter, met 5 strikt positieve gehele getallen opvullen. De gebruiker dient deze getallen in te geven (invoer controleren en eventueel foutboodschappen weergeven).
  - Methode main:
    - Initialiseer een constante array rij1 met de gehele getallen 2, 7, 1, 9 en 4
    - Maak een tweede rij (rij2) aan en vul die op via de methode invoerRij
    - Roep de methode deel uit de domeinklasse op. Geef rij1 en rij2 als parameters door en bewaar het resultaat in rij3.
    - Roep de methode schuifOp uit de domeinklasse op. Geef als parameter rij3 mee en genereer een random getal voor het aantal plaatsen (dit getal behoort tot het interval [1,lengte van de array - 1])
- Zorg voor volgende invoer/uitvoer:

```
Geef strikt positief getal nr 1 in: 6
Geef strikt positief getal nr 2 in: 5
Geef strikt positief getal nr 3 in: 2
Geef strikt positief getal nr 4 in: -9
Geef strikt positief getal nr 4 in: 9
Geef strikt positief getal nr 5 in: 1
Rij1 =          2         7         1         9         4
Rij2 =          6         5         2         9         1
Rij3 =       0,33      1,40      0,50      1,00      4,00

Rij3 na verschuiving met 1 plaats =       4,00      0,33      1,40      0,50      1,00
```

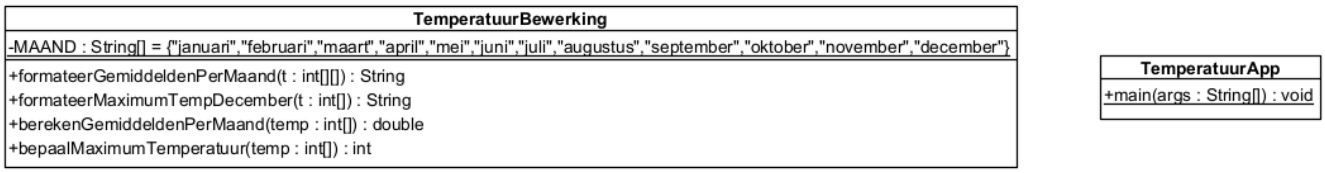
OF

```
Geef strikt positief getal nr 1 in: 1
Geef strikt positief getal nr 2 in: 2
Geef strikt positief getal nr 3 in: 3
Geef strikt positief getal nr 4 in: 4
Geef strikt positief getal nr 5 in: -6
Geef strikt positief getal nr 5 in: 6
Rij1 =          2         7         1         9         4
Rij2 =          1         2         3         4         6
Rij3 =       2,00      3,50      0,33      2,25      0,67

Rij3 na verschuiving met 2 plaatsen =       2,25      0,67      2,00      3,50      0,33
```

2.8. Temperatuur

Maak een programma dat een aantal berekeningen doet op basis van de volgende resultaten bij temperatuurswaarnemingen in België:

```
int temp[][] = {{4, -3, 2, 0, -5}, {-1, 8, 3}, {11, 3, 7, 9},
{10, 14, 7, 5}, {10, 9, 17, 14, 21}, {18, 10, 24, 27, 14, 22},
{17, 32, 27, 24, 25}, {31, 28, 22, 30, 17}, {22, 24, 17, 14, 12, 11},
{12, 14, 9, 7, 12}, {7, 11, 14, 11, 6, 0, 7}, {3, -1, -3, 5}};
```



Klasse TemperatuurBewerking:

- Definieer een array maand met de namen van de maanden als attribuut.
- methode berekenGemiddeldenPerMaand:
  Deze methode heeft 1 parameter (1-dimensionele array met integers). De methode berekent het gemiddelde van alle getallen in deze array en geeft dit gemiddelde als een kommagetal terug.
- methode bepaalMaximumTemperatuur:
  Deze methode heeft 1 parameter (1-dimensionele array met integers). De methode bepaalt de maximumwaarde uit deze array en geeft dit geheel getal terug.
- methode formateerGemiddeldenPerMaand:
  Deze methode heeft 1 parameter (2-dimensionele array met integers). De methode geeft de gemiddelden per maand als String terug. Maak hierbij gebruik van de methode berekenGemiddeldenPerMaand.
- methode formateerMaximumTempDecember: Deze methode heeft 1 parameter (1-dimensionele array met integers). De methode geeft de maximale temperatuur in de maand december als String terug. Maak hierbij gebruik van de methode berekenMaximumTemperatuur.

Let op: maak bij de methodes hierboven geen gebruik van de wetenschap dat een jaar 12 maanden telt en ga het aantal waarnemingen per maand in de temp array niet tellen, aangezien daar ook andere waarden (meer of minder) hadden kunnen staan en het programma dan nog steeds moet werken.
Geef voor de gemiddelden 2 cijfers na de komma weer.

Klasse TemperatuurApp:

In de main-methode definieer je de lokale variabele temp (zoals hierboven voorgesteld). Zorg voor deze uitvoer:

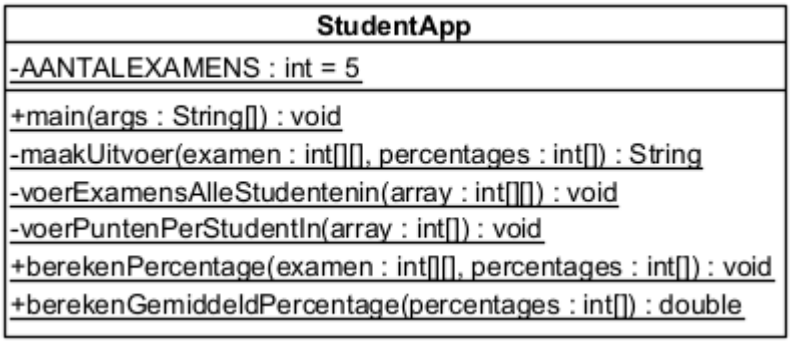
```
De gemiddelde temperatuur is:

In januari: -0,40
In februari: 3,33
In maart: 7,50
In april: 9,00
In mei: 14,20
In juni: 19,17
In juli: 25,00
In augustus: 25,60
In september: 16,67
In oktober: 10,80
In november: 8,00
In december: 1,00

De maximum temperatuur in december is: 5
```

De tekst in het rode kader is het resultaat van de aanroep van de methode formateerGemiddeldenPerMaand, die in het blauwe kader komt van de aanroep van formateerMaximumTempDecember.

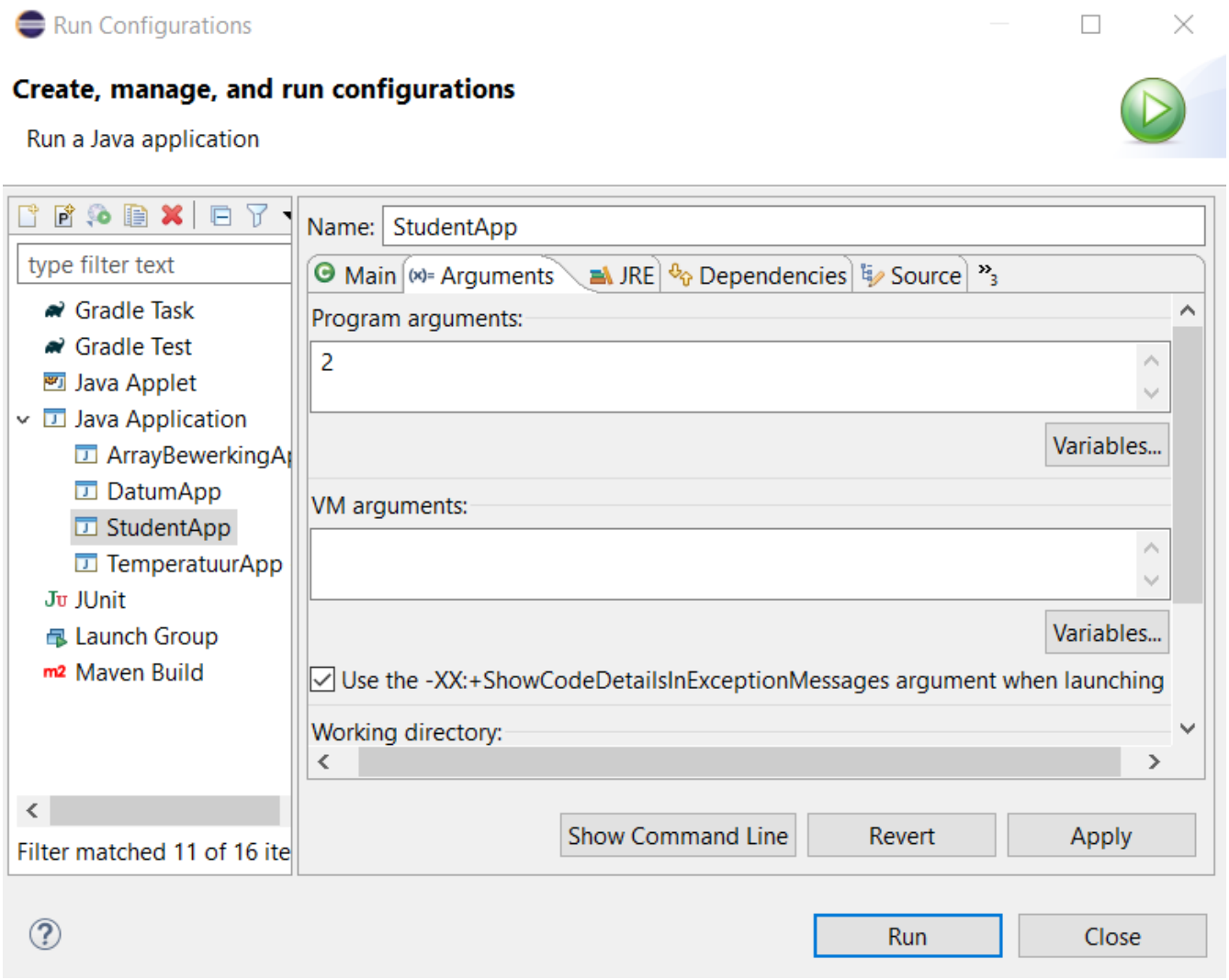
2.9. Studenten

Schrijf een applicatie die voor een aantal studenten de punten, een gemiddelde en de tekst "geslaagd" of "niet geslaagd" uitschrijft.

Alle code komt in één klasse StudentApp – zie ook onderstaande UML!



Als je de applicatie start, geef je een programmaparameter mee. Dit cijfer legt het aantal studenten vast.



Dit aantal studenten (hier 2) dient maximum 5 examens af te leggen.
Per student wordt ingegeven hoeveel examens deze student aflegt en welke punten (op 20) hij haalt op die examens (met invoercontrole).
Een student is geslaagd indien hij op elk afgelegd examen 10 of meer heeft behaald en indien zijn percentage 60 of meer is.
Per student wordt afgebeeld: zijn punten, zijn percentage en "geslaagd" of "niet geslaagd".
Het gemiddelde van de percentages van de drie studenten wordt ook weergegeven met twee cijfers na de komma.

Schrijf hiervoor de benodigde code.

Het aantal studenten wordt als programmaparameter meegegeven (in het voorbeeld: 2 studenten).
Toon een foutboodschap als deze programmaparameter ontbreekt.

```
Geen parameter ingegeven!
```

De applicatie bestaat uit volgende methodes:

- main:
  - Maak een array voor de punten van alle examens en een tweede array voor de percentages

per student.

- Roep de methode voerExamensAlleStudentenin aan om alle punten in te voeren.
- Roep de methode berekenPercentage aan om adhv de punten de percentages per student te berekenen
- Gebruik tenslote de methode maakUitvoer om alle uitvoer in één grote String te zetten die je dan kan printen.

- voerExamensAlleStudentenIn:
  Vraag per student het aantal afgelegde examens op (met invoercontrole!), maak een array die de punten kan bevatten van deze student en roep de methode voerPuntenPerStudentIn aan.
- voerPuntenPerStudentIn: geef de punten van één student in (met invoercontrole!)
- berekenPercentage: bereken het percentage per student en hou deze percentages bij in een array
- maakUitvoer:
  Maak een uitvoerString met daarin per student zijn volgnummer en zijn punten en op de volgende regel het percentage en de tekst "geslaagd" of "niet geslaagd". Zie ook onderstaande voorbeelduitvoer.
- berekenGemiddeldPercentage: Bereken het gemiddelde van de percentages van alle studenten

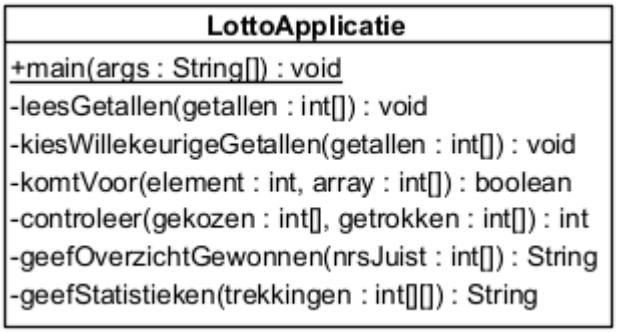
Bij het uitvoeren van de applicatie wordt voor elke student gevraagd hoeveel examens hij aflegt. Als de eerste student bijvoorbeeld 3 examens aflegt, krijg je onmiddellijk de kans om voor die 3 examens de punten in te geven. Zorg voor een juiste nummering van de examens. Dit herhaal je voor het aantal studenten meegegeven via de programmaparameter.

Het eindresultaat ziet eruit als volgt:

```
Hoeveel examens heeft student 1 afgelegd (max = 5): 6
Hoeveel examens heeft student 1 afgelegd (max = 5): 3
Examen 1 (op 20): -9
Examen 1 (op 20): 22
Examen 1 (op 20): 18
Examen 2 (op 20): 10
Examen 3 (op 20): 14
Hoeveel examens heeft student 2 afgelegd (max = 5): 2
Examen 1 (op 20): 6
Examen 2 (op 20): 155
Examen 2 (op 20): 15
Student 1      18      10      14
percentage : 70   geslaagd
Student 2       6      15
percentage : 52   niet geslaagd

gemiddelde : 61,00
```

2.10. Lotto

In de LottoApplicatie mag een speler 6 getallen tussen 1 en 45 opgeven. Daarna zal het systeem 1000 trekkingen simuleren en controleren hoeveel juiste nummers de speler voorspeld had. Het systeem toont dan een overzicht van het aantal keer dat de speler op de 1000 trekkingen 0, 1, … 6 getallen juist had. Tenslotte wordt ook nog getoond hoeveel keer elke lottobal getrokken werd.



Schrijf hiervoor volgende methodes:

- De **main**-methode wordt gebruikt om de andere methodes aan te roepen, zodat het programmaverloop er net zo uitziet als in onderstaande uitvoer. Om de gekozen en getrokken getallen te laten zien, kan je gebruik maken van de static methode toString uit de Arrays-klasse.
- De methode **leesGetallen** vult de meegegeven array op. Hierbij wordt telkens aan de gebruiker een getal gevraagd tussen 1 en 45. Als het getal niet aan deze voorwaarde voldoet, wordt het getal opnieuw gevraagd. Daarna wordt gecontroleerd of het getal reeds in de array zat met behulp van de methode **komtVoor**. Alleen als het nog niet voorkwam, mag het getal toegevoegd worden in de array! Tenslotte worden alle getallen uit de array nog gesorteerd via de methode sort van de Arrays-klasse.
- De methode **kiesWillekeurigeGetallen** vult eveneens de meegegeven array op, maar hierbij wordt telkens een willekeurig getal bepaald tussen 1 en 45. Als dit getal echter al voorkomt in de array (zie methode **komtVoor**), dan moet het getal ook hier opnieuw gekozen worden! Als het getal nog niet voorkwam in de array, mag het er aan toegevoegd worden. Tenslotte worden alle getallen uit de array nog gesorteerd via de methode sort van de Arrays-klasse.
- In de methode **komtVoor** wordt nagekeken of een bepaald getal in de opgegeven array voorkomt. Het resultaat is true (getal komt al voor) of false (getal komt nog niet voor).
- De methode **controleer** kijkt na hoeveel getallen uit de eerste array (met de door de gebruiker gekozen getallen) voorkomen in de tweede array (met de door het systeem getrokken getallen) en geeft dit aantal terug.
- De methode **geefOverzichtGewonnen** geeft een String terug met per aantal juiste nummers het aantal keer dat dit aantal voorkwam. Zie ook onderstaand voorbeeld van de uitvoer.
- In de methode **geefStatistieken** wordt een String opgebouwd met de statistieken per lottobal. Van elke bal moet getoond worden hoeveel keer hij op alle trekkingen samen is getrokken. Zie ook bijgevoegd voorbeeld van de uitvoer.

Voorbeeld in- en uitvoer:

Invoer lottogetallen:

```
Kies je 1e lottogetal (1-45): 7
Kies je 2e lottogetal (1-45): 23
Kies je 3e lottogetal (1-45): 49
Kies je 3e lottogetal (1-45): 44
Kies je 4e lottogetal (1-45): -3
Kies je 4e lottogetal (1-45): 3
Kies je 5e lottogetal (1-45): 17
Kies je 6e lottogetal (1-45): 34
Gekozen getallen: [3, 7, 17, 23, 34, 44]
```

Trekkingen:

```
Trekking 1 - Getrokken getallen: [3, 10, 13, 26, 28, 32] - Aantal juiste: 1
Trekking 2 - Getrokken getallen: [5, 7, 10, 35, 40, 42] - Aantal juiste: 1
Trekking 3 - Getrokken getallen: [6, 12, 15, 22, 36, 40] - Aantal juiste: 0
Trekking 4 - Getrokken getallen: [3, 18, 22, 32, 36, 40] - Aantal juiste: 1
Trekking 5 - Getrokken getallen: [12, 19, 20, 27, 35, 36] - Aantal juiste: 0
Trekking 6 - Getrokken getallen: [4, 5, 16, 17, 28, 45] - Aantal juiste: 1
Trekking 7 - Getrokken getallen: [2, 5, 26, 32, 37, 43] - Aantal juiste: 0
Trekking 8 - Getrokken getallen: [3, 6, 12, 21, 33, 45] - Aantal juiste: 1
Trekking 9 - Getrokken getallen: [4, 8, 26, 35, 37, 44] - Aantal juiste: 1
Trekking 10 - Getrokken getallen: [1, 7, 8, 30, 40, 45] - Aantal juiste: 1
```

...

```
Trekking 991 - Getrokken getallen: [4, 9, 13, 14, 22, 28] - Aantal juiste: 0
Trekking 992 - Getrokken getallen: [4, 7, 10, 12, 20, 29] - Aantal juiste: 1
Trekking 993 - Getrokken getallen: [2, 11, 22, 30, 39, 45] - Aantal juiste: 0
Trekking 994 - Getrokken getallen: [3, 8, 13, 33, 37, 38] - Aantal juiste: 1
Trekking 995 - Getrokken getallen: [6, 15, 26, 30, 34, 43] - Aantal juiste: 1
Trekking 996 - Getrokken getallen: [8, 14, 24, 30, 34, 36] - Aantal juiste: 1
Trekking 997 - Getrokken getallen: [1, 8, 9, 15, 33, 37] - Aantal juiste: 0
Trekking 998 - Getrokken getallen: [7, 12, 23, 29, 34, 35] - Aantal juiste: 3
Trekking 999 - Getrokken getallen: [3, 13, 16, 23, 28, 38] - Aantal juiste: 2
Trekking 1000 - Getrokken getallen: [4, 12, 33, 35, 38, 44] - Aantal juiste: 1
```

Aantal keer x nummers juist:

```
408 keer 0 nummers juist
424 keer 1 nummers juist
148 keer 2 nummers juist
19 keer 3 nummers juist
1 keer 4 nummers juist
0 keer 5 nummers juist
0 keer 6 nummers juist
```

Statistieken per bal:

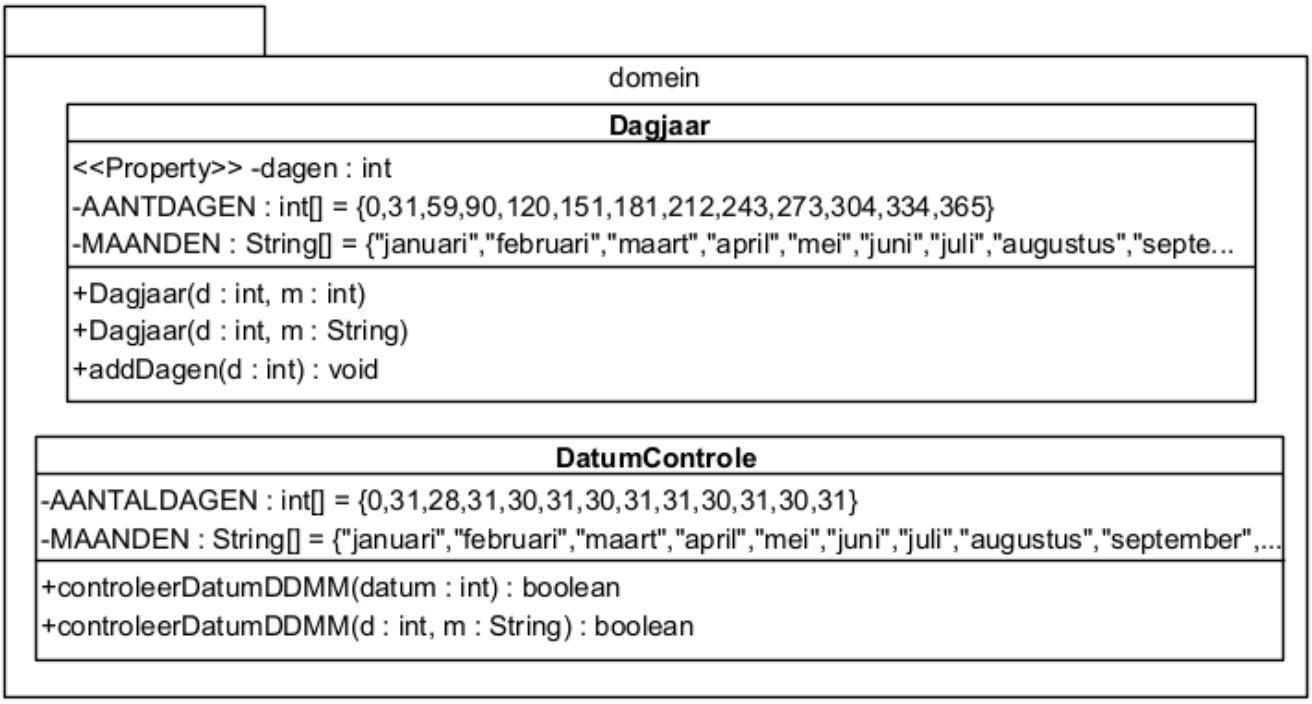
```
Bal 1 werd 136 keer getrokken
Bal 2 werd 128 keer getrokken
Bal 3 werd 147 keer getrokken
Bal 4 werd 145 keer getrokken
Bal 5 werd 140 keer getrokken
```

...

```
Bal 41 werd 155 keer getrokken
Bal 42 werd 121 keer getrokken
Bal 43 werd 118 keer getrokken
Bal 44 werd 111 keer getrokken
Bal 45 werd 137 keer getrokken
```

2.11. Datum

Schrijf een applicatie, die een datum in 2021 omzet naar het aantal dagen verlopen sinds 1 januari 2021. Bijvoorbeeld: 20/2 wordt 51 (= 31 + 20) dagen.



We gebruiken daarvoor de klasse **Dagjaar**:

- Attribuut: dagen = aantal dagen sinds 1 januari 2021
- Methoden:
  - 2 constructoren :
    1. de parameters zijn 2 integers, die de dag en de maand voorstellen
    2. de parameters zijn 1 integer en 1 string, bv. 3 en december

       De parameters moeten niet meer gecontroleerd worden op geldigheid!!!
       De twee constructoren zorgen ervoor dat het attribuut dagen de goede waarde krijgt, deze wordt berekend adhv de parameters. Gebruikt hiervoor de constante attributen. (zie onder)

  - de functie addDagen(int d), om het aantal dagen te kunnen verhogen (als object.dagen > 365 wordt, dan object.dagen = object.dagen – 365)
  - de functie getDagen(), die de waarde van het attribuut dagen teruggeeft

- De klasse heeft ook de volgende constante attributen, die gebruikt worden in de constructoren:

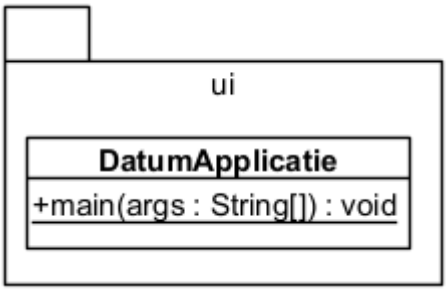
```
private final String[] MAANDEN = {"januari","februari","maart","april",
"mei","juni","juli","augustus","september","oktober","november","december"};
private final int[] AANTALDAGEN = {0,31,59,90,120,151,181,212,243,273,304,334,365};
```

We hergebruiken ook de klasse **DatumControle** uit oefening 2.6:

Neem deze klasse uit oefening 2.6 en vul deze verder aan met een tweede controleerDatumDDMM()-methode (parameters zijn DD (= int) en maand (= String)), die de eerste methode gebruikt!

In de testklasse voor DatumControle vind je 4 extra tests die je voor deze oefening uit commentaar moet halen!

Tenslotte maken we de ui-klasse **DatumApplicatie**:



We laten de gebruiken een datum invoeren in de vorm DD MM of DD maand. Deze wordt dan eerst gecontroleerd op geldigheid via de corresponderende methode controleerDatumDDMM(). Indien geldig, dan wordt de data in een object van de klasse Dagjaar gestockeerd.

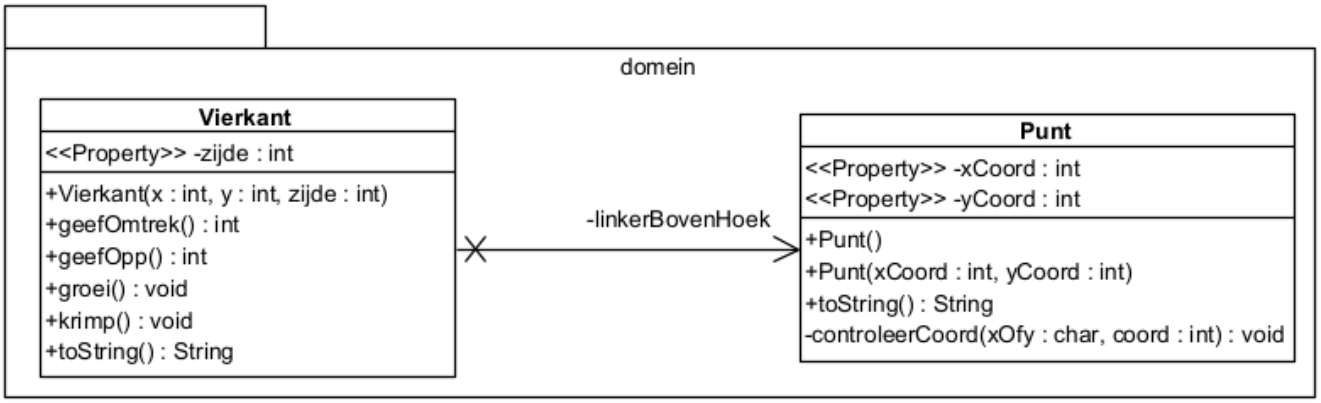
Vervolgens krijgt de gebruiker de kans om een invoer te doen van een aantal dagen (positief getal, controle!) en dit wordt via de methode addDagen toegevoegd aan het attribuut dagen. Via getDagen() kunnen we de uitvoer verzorgen per correcte datum (toon telkens het resultaat voor en na de verhoging).

Mogelijke uitvoer:

```
Geef dag in: 32
Geef maand in: 10
De datum 32 10 is ongeldig

Geef dag in: 13
Geef maand in: 13
De datum 13 13 is ongeldig

Geef dag in: 27
Geef maand in: setember
De datum 27 setember is ongeldig
```

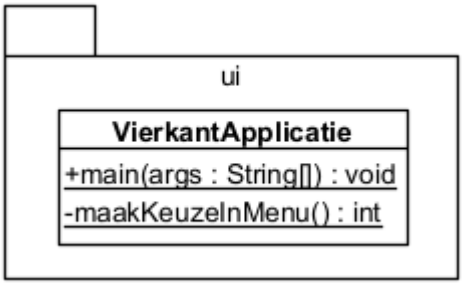
```
Geef dag in: 25
Geef maand in: 3
De datum 25 3 is geldig.
De datum in dagen: 84.
Hoeveel dagen wens je toe te voegen? 100
De datum in dagen NA de toevoeging = 184

Geef dag in: 3
Geef maand in: 12
De datum 3 12 is geldig.
De datum in dagen: 337.
Hoeveel dagen wens je toe te voegen? 50
De datum in dagen NA de toevoeging = 22
```

2.12. Vierkant



- Maak een domeinklasse **Punt** met:
  - Attributen:
    - x-coördinaat
    - y-coördinaat
  - Methodes:
    - constructor met 2 parameters
    - constructor zonder parameters, roept de andere constructor op met defaultwaarden 1
    - getter voor de x-coördinaat
    - getter voor de y-coördinaat
    - setters voor x- en y-coördinaat
    - controleermethode: wordt vanuit beide setters aangeroepen om te controleren of de coördinaat > 0 is, anders wordt een exception gegooid met een passende boodschap, waarin vermeld wordt om welke coördinaat het gaat
    - toString(): omzetten van een Punt-object in een String van de vorm (x,y)
- Maak een tweede domeinklasse **Vierkant** met:
  - Attributen:

        - linkerBovenHoek (een Punt-object)
        - zijde (geheel getal)
    - Methoden:
        - constructor met 3 parameters (x, y en zijde – zie testklasse)
        - setter voor zijde: zijde moet altijd tussen 1 en 500 liggen (grenzen inbegrepen), anders wordt een exception gegooid met een passende boodschap
        - getter voor de linkerbovenhoek (returntype = Punt)
        - getter voor de zijde
        - geefOmtrek(): berekent en geeft de omtrek terug
        - geefOpp(): berekent en geeft de oppervlakte terug
        - groei(): laat de zijde 5 eenheden groeien (blijf controleren op de voorwaarde, als de zijde niet meer kan groeien met 5 eenheden blijft de huidige waarde behouden!)
        - krimp(): idem als vorige, maar verklein nu de zijde met 5 eenheden
        - toString(): zie uitvoer
- Maak een kleine applicatie die de functionaliteit van de domeinklassen test:



    - Laat de gebruiker eerst de linkerbovenhoek (x- en y-coördinaat) en de lengte van de zijde van het vierkant ingeven.
    - Toon dan alle gegevens van het vierkant
    - Zorg vervolgens voor een menu met de volgende opties:
        - Krimpen van het bestaande vierkant
        - Groeien van het bestaande vierkant
        - Een nieuwe linkerbovenhoek definiëren
        - Applicatie afsluiten
    - Wanneer de gebruiker één van deze opties kiest, tonen we onmiddellijk de resultaten, bijvoorbeeld:

```
Het vierkant heeft:
Zijn linkerbovenhoek op: (50,70)
Een zijde van: 80
Een omtrek van: 320
Een oppervlakte van: 6400
```

Merk op dat we voor alle menu-opties telkens weer starten met het oorspronkelijke vierkant!

Voorbeelduitvoer:

```
Geef de x-coördinaat van de linkerbovenhoek in: 10
Geef de y-coördinaat van de linkerbovenhoek in: 20
De zijden van het vierkant hebben ook een lengte. Geef deze lengte in: 10
Het vierkant heeft:
Zijn linkerbovenhoek op: (10, 20)
Een zijde van: 10
Een omtrek van: 40
Een oppervlakte van: 100
Menu:
1) Krimpen van het bestaande vierkant
2) Groeien van het bestaande vierkant
3) Een nieuwe linkerbovenhoek definiëren
4) Applicatie afsluiten
1
Het vierkant heeft:
Zijn linkerbovenhoek op: (10, 20)
Een zijde van: 5
Een omtrek van: 20
Een oppervlakte van: 25
Menu:
1) Krimpen van het bestaande vierkant
2) Groeien van het bestaande vierkant
3) Een nieuwe linkerbovenhoek definiëren
4) Applicatie afsluiten
3
Geef de x-coördinaat van de linkerbovenhoek in: 5
Geef de y-coördinaat van de linkerbovenhoek in: 15
Het vierkant heeft:
Zijn linkerbovenhoek op: (5, 15)
Een zijde van: 10
Een omtrek van: 40
Een oppervlakte van: 100
Menu:
1) Krimpen van het bestaande vierkant
2) Groeien van het bestaande vierkant
3) Een nieuwe linkerbovenhoek definiëren
4) Applicatie afsluiten
4
```