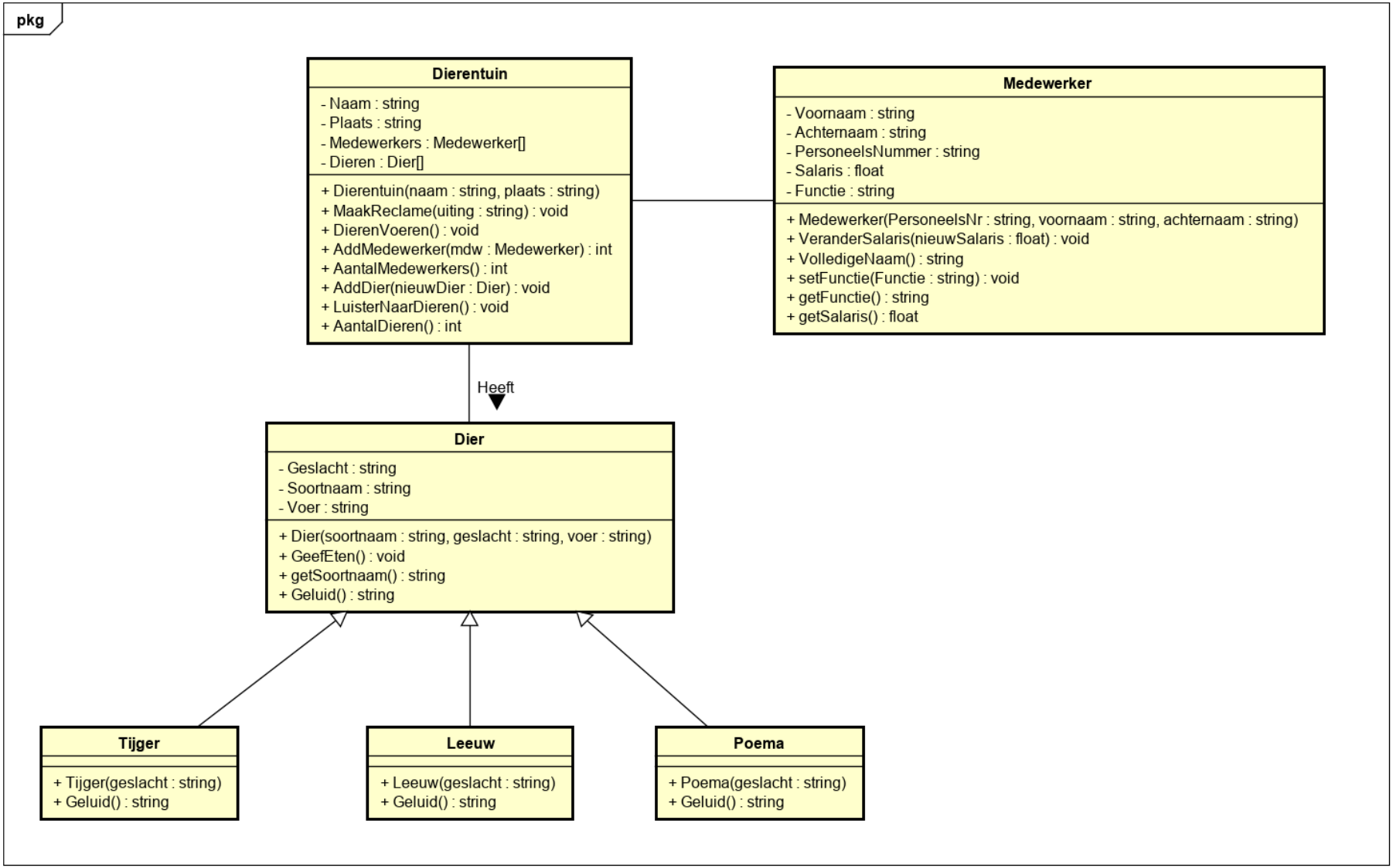
Opleiding	HBO ICT - Flex		
Titel Toetseenheid	Programmeren	**Toetsvorm**	Tentamen
Titel Toets	Programmeren	**Datum**	-
Module	Software Engineering 1 S2 2018/2019	**Examinator** **Tijdsduur**	Martin Molema 2,5 uur
Maximale score	90 punten	**Cijfer**	(90pt + 10pt) / 10
		Pagina's	7

Toegestane hulpmiddelen	Internet & Boeken

Vraag 1

Zie onderstaande klassendiagram



In bovenstaande klassendiagram worden een aantal klassen beschreven.

Vraag 1a (theorie) (4 + 3 + 3 = 10 punten)

1) Bij de klasse ‘Dier’ staat een methode ‘Dier’. Hoe heet deze type methode en wat is het doel?
2) De klassen Tijger, Leeuw en Poema zijn gerelateerd aan de klasse ‘Dier’. Hoe noemen we de
    a) de klasse **Dieren,** in relatie tot de klassen ‘Tijger’, ‘Leeuw’ en ‘Poema’?
    b) de klassen ‘Tijger’, ‘Leeuw’ en ‘Poema’, in relatie tot de klasse ‘Dieren’?

Vraag 1b (10 punten)

Bouw de basis van de klassen uit het bovenstaande klassendiagram met C# via een *Console Application*. Toelichting

- alle variabelen hebben de beperkte zichtbaarheid **‘private**’.
- een dier heeft een soortnaam (“Leeuw”, “Tijger” etc)
- De dieren staan op een dieet, de waardes voor ‘voer’ zijn:
    - De Leeuw eet alleen Antilope
    - De Tijger eet alleen Nijlpaard
    - De Poema eet alleen Giraffe
- In de klasse “**Dierentuin**” staat “**Dieren[]**”. Dit duidt op een lijst van dieren. Zelfde geldt voor Medewerkers.

De functies ga je in de volgende vragen implementeren.

Vraag 2 (36 punten)

De dierentuinen zijn inmiddels geopend. We gaan met de basis van de klassen in de dierentuin verder.

2a: Dierentuinen maken (5 punten)

Maak twee dierentuinen: **Gaia Zoo** in Maastricht en **Wildlands** in Emmen.

2b: Medewerkers (5 punten)

In elke dierentuin zijn een aantal verschillende medewerkers actief:

- bewakers
- kaartverkopers
- medewerker personeelszaken

Maak voor elke dierentuin van elke functie **minstens** 2 medewerkers aan. Na het aanmaken geef je elke medewerker een (fictief) salaris. Daarvoor gebruik je `'VeranderSalaris()'`. Een medewerker koppel je aan de dierentuin via ‘AddMedewerker()’. Deze functie geeft het huidige aantal medewerkers terug.

2c: Eten geven mogelijk maken (8 punten)

Maak van elke soort (leeuw, tijger, poema) **minimaal** twee dieren en geef ze te eten. Het geslacht mag je zelf bepalen. Het eten geven doe je via de functie '`DierenVoeren()`'. Deze functie roept per dier in de dierentuin de functie '`GeefEten()`' aan.

Dieren koppel je aan de dierentuin via '`AddDier()`'.

De functie '`GeefEten()`' zorgt ervoor dat de bewaker onderstaande acties uitvoert door een tekst op de console weer te geven

1. de deur van het hok open doet
2. het eten neerlegt (noem het soort eten dat gebruikt wordt)
3. de deur weer dicht doet

Voorbeeld output:

```
Open hok Leeuw
Geef eten:Antilope
Sluit hok Leeuw

Open hok Tijger
Geef eten:Nijlpaard
Sluit hok Tijger
```

2d: Maak geluid! (8 punten)

Elk dier maakt natuurlijk ook geluid. Nadat je de dieren eten hebt gegeven maken ze hun geluid. Schrijf de functie '`LuisterNaarDieren()`'. Deze roept per dier de functie '`Geluid()`' aan. Deze functie geeft het geluid terug die het dier maakt.
Noem per dier dat je tegenkomt de soortnaam en het geluid dat deze maakt. Let op: deze functie De output ziet er bijvoorbeeld als volgt uit:

```
Leeuw:nogal harde brul
Leeuw:nogal harde brul
Leeuw:nogal harde brul
Tijger:brul
Tijger:brul
Poema:grom
Poema:grom
```

2e: Maak reclame (10 punten)

Implementeer de functie '`MaakReclame()`'. Deze schrijft een wervende tekst op de console. Deze bestaat minimaal de volgende elementen:

- de naam van het park
- de plaats van het park
- de meegegeven reclame uiting
- het aantal medewerkers in het park
- het aantal dieren **per soort**

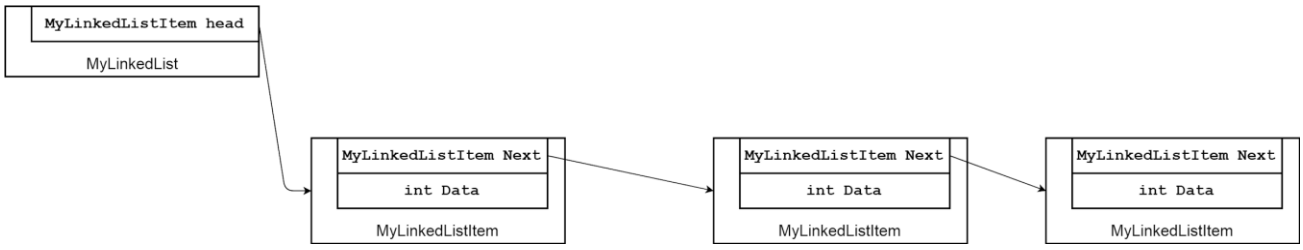
Voorbeeld output:

```
Komt allen naar dierentuin Gaia Zoo te Maastricht
Met 6 medewerkers maken wij er een onvergetelijke dag van
Wij hebben de volgende dieren in onze dierentuin:
3 van soort Leeuw
4 van soort Tijger
2 van soort Poema
```

Vraag 3 (19 punten)

Voor deze vraag wordt een klasse `'MyLinkedList'` gebruikt die in de solution van Visual Studio zit. Je moet een aantal opdrachten uitvoeren die getest kunnen worden met NUnit. De solution is te vinden op BlackBoard. De klasse `'MyLinkedList'` is verborgen maar is wel bruikbaar.

De enkelvoudige gelinkte lijst wordt gebruikt om een lijst van integers te kunnen opslaan. Hiervoor wordt een klasse gebruikt met de naam `'MyLinkedListItem'`. Dit werkt als volgt:



Een lijst van het type '`MyLinkedList`' bevat een verwijzing naar een **lijst** met gegevens. De start van deze lijst is opgenomen in de variabele '`head`'. De lijst bestaat uit items van het type '`MyLinkedListItem`'. In een item van dat type zit een getal ('`Data`') en een verwijzing naar het volgende element ('`Next`'). Als er geen volgend item is, dan heeft '`Next`' de waarde NULL.

Tekst omdraaien (10+9 punten)

Getallen en letters kun je in C# door elkaar gebruiken, mits je enkele aanhalingstekens gebruikt. Zo zijn het getal 65 en de hoofdletter 'A' uitwisselbaar. De gelinkte lijst bevat alleen getallen, maar die kunnen we dus interpreteren als letters.

Opdracht

Maak een functie met twee varianten die de getallen in de lijst omzet in een **tekst**, en deze **omdraait** (zie verderop hoe je dat doet). De twee varianten:

a) een oplossing met een normale lus : '`CreateReverseTextLoop()`' (**10 punten**)
b) een recursieve oplossing: '`CreateReverseTextRecursive()`' (**9 punten**)

Je vind deze functies in het bestand '`StudentLinkedList.cs`'. De klasse '`StudentLinkedList`' erf van de klasse '`MyLinkedList`'.

Deze twee functies hebben beiden een eigen test-functie.

a) `TestList2Recursief`
b) `TestList2Loop`

Daarbij gaan we uit van de volgende omzetting van getal naar letter:
1 = A
2 = B
3 = C
…
…
25 = Y
26 = Z

Je kunt dit doen met de volgende berekening in C#:

```
char Letter = (char)('A' + x -1);
```

Waarbij x het getal tussen 1 en 26 is.

Een voorbeeld:
Stel in de lijst staan de volgende getallen:

1,2,3,4,5,1

Dan kunnen we dit omzetten naar

ABCDEA
Vervolgens draaien we dit om naar:

AEDCBA

De uitvoer van de functie '`CreateReverseTextRecursive([1,2,3,4,5,1])`' is dus AEDCBA
De uitvoer van de functie '`CreateReverseTextLoop([1,2,3,4,5,1])`' is dus AEDCBA

Nog een voorbeeld:
De uitvoer van de functie '`CreateReverseTextRecursive([1,1,1,2])`' is dus BAAA
De uitvoer van de functie '`CreateReverseTextLoop([1,1,1,2])`' is dus BAAA
De uitvoer van de functie '`CreateReverseTextRecursive([7,1,1,25])`' is dus ZAAG
De uitvoer van de functie '`CreateReverseTextLoop([7,1,1,25])`' is dus ZAAG

Vraag 4 (15 punten)

In de Visual Studio Solution zit in het bestand '`Program.cs`' één functie (`BerekenSomRecursiveOdd`) die je moet afmaken. Deze kun je testen met NUnit tests.

In deze opgave moet je een **recursieve** oplossing bedenken om een som te berekenen van alle oneven getallen in een array met integers. Schrijf je uitkomst in de functie '`BerekenSomRecursiveOdd()`' en test deze met '`TestBerekenSomRecursiveOdd()`'.

Bij een lege lijst geeft de functie de uitkomst nul.

----------EINDE VAN HET TENTAMEN----------

Puntenverdeling en controle

Vraag	Onderdeel	Sub	Punten	Som
1	a	1	4	20
		2	6	
	b		10	
2	a		5	36
	b		5	
	c		8	
	d		8	
	e		10	
3	a	1	10	19
		2	9	
4	a		15	15

TOTAAL 90 + 10 start punten = 100