

## Opgave 1: 'Lettervolgorde'

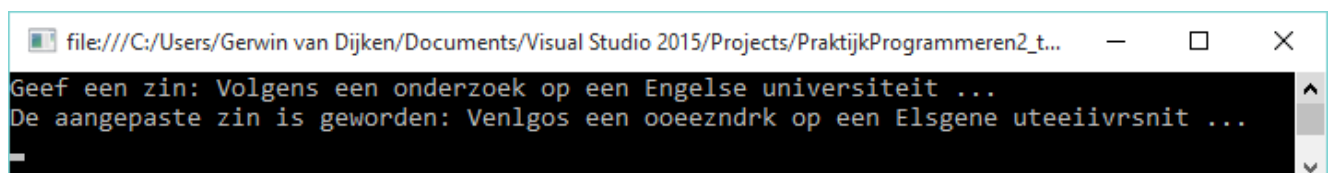
(25 punten)

Volgens een onderzoek op een Engelse universiteit maakt het niet uit in welke volgorde de letters in een woord staan, het enige wat belangrijk is, is dat de eerste en de laatste letter op de juiste plaats staan. De rest van de letters mogen willekeurig geplaatst worden en je kunt vervolgens gewoon lezen wat er staat. Dit komt omdat we niet elke letter op zich lezen maar het woord als geheel.

In deze opdracht gaan we de woorden van een zin (die de gebruiker invoert) husselen, en kijken of bovenstaande klopt. Blijft de zin leesbaar (voor anderen)?

### Aan de slag:

0. Maak een Console project aan met de naam '**Opgave1**', en geef als solution naam op 'PraktijkProgrammeren2-Tentamen'.  
(de overige opgaven worden ook als projecten in deze solution aangemaakt, zie later)
1. Vraag in de Start-methode aan de gebruiker om een zin in te voeren en sla deze op.
2. Maak een methode met signatuur "`string` HusselZinWoorden(`string` zin)". Deze methode ontvangt een zin en geeft deze zin terug waarbij de woorden gehusseld zijn. Ga als volgt te werk:
  - a. roep de Split-methode aan op de zin met een enkele spatie als parameter (`zin.Split(' ');`), je krijgt hierbij een array met woorden (`string[]`) terug;
  - b. doorloop deze array en roep voor elk woord de methode "HusselWoord" aan (zie hieronder). Plak elk woord dat HusselWoord teruggeeft achter elkaar.
  - c. Retourneer de aan elkaar geplakte zin als resultaat;
3. Maak een methode met signatuur "`string` HusselWoord(`string` woord)" die van een ontvangen woord een gehusselde versie teruggeeft. Ga als volgt te werk:
  - a. als het ontvangen woord 3 letters of minder bevat, dan valt er niets te husselen; geef hetzelfde woord terug als resultaat (de punten b t/m f dan niet verwerken);
  - b. maak een lokale variabele 'nieuwWoord' en zet de eerste letter van het (ontvangen) woord hier in;
  - c. maak een variabele 'restWoord' en plaats daarin het middelste gedeelte van het (ontvangen) woord; gebruik hierbij `Substring(...)`;
  - d. zolang het restwoord nog letters bevat:
    - i. neem een willekeurige letter uit het restwoord (random index), en voeg deze letter toe aan het nieuwe woord;
    - ii. verwijder vervolgens deze letter uit het restwoord (gebruik hierbij: `restWoord.Remove(index, 1)` en sla het resultaat hiervan op in `restWoord`);
  - e. voeg de laatste letter van het (ontvangen) woord toe aan het nieuwe woord;
  - f. geef het nieuwe woord terug als return-waarde;
4. Roep vanuit de Start-methode de methode "HusselZinWoorden" aan met de ingevoerde tekst (van de gebruiker) als parameter, en toon het resultaat (zie voorbeeld hieronder).



```
file:///C:/Users/Gerwin van Dijken/Documents/Visual Studio 2015/Projects/PraktijkProgrammeren2_t...
Geef een zin: Volgens een onderzoek op een Engelse universiteit ...
De aangepaste zin is geworden: Venlgos een ooeezndrk op een Elsgene uteeiivrsnit ...
```

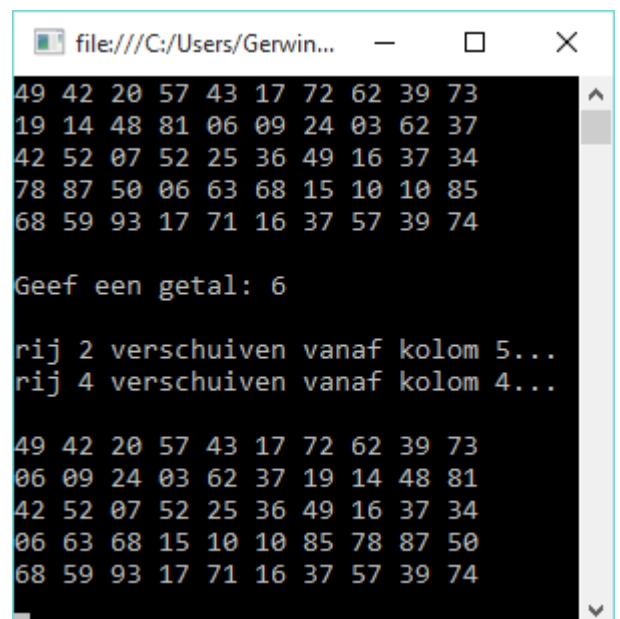
## Opgave 2: 'Schuifmatrix'

(25 punten)

In deze opgave gaan we een matrix met random getallen bewerken; dat gebeurt door het verschuiven van bepaalde rijen in de matrix.

### Aan de slag:

0. Voeg aan solution 'PraktijkProgrammeren2-Tentamen' een Console project toe met naam 'Opgave2'.
1. Maak een 2-dimensionale array aan genaamd "matrix" (in de Start-methode); geef deze matrix 5 rijen en 10 kolommen.
2. Implementeer een methode "VulMatrix" die de 2-dimensionale array ontvangt als parameter. Vul in deze methode de array met random getallen tussen 1 (inclusief) en 100 (exclusief). Roep deze methode vanuit de Start-methode aan.
3. Implementeer een methode "ToonMatrix" die de 2-dimensionale array ontvangt als parameter. Deze methode toont de array zoals in onderstaand voorbeeld te zien is. Roep deze methode vanuit de Start-methode aan.
4. Vraag de gebruiker vervolgens (in de Start-methode) om een zoekgetal (`int`). Dit getal gaan we straks gebruiken om de rijen te verschuiven (zie hieronder).
5. Implementeer een methode "`void VerschuifMatrix(int[,] matrix, int zoekgetal)`". Van elke rij willen we dat het zoekgetal (indien aanwezig) vooraan komt te staan. Ga als volgt te werk: doorloop alle rijen en zoek in elke rij naar het zoekgetal (door de kolommen te doorlopen); stop met zoeken in een rij als het getal gevonden is. Als het zoekgetal in een rij gevonden is, roep dan de methode "VerschuifRij" aan (zie hieronder) die er voor zorgt dat de betreffende rij verschoven wordt.
6. Implementeer een methode "`void VerschuifRij(int[,] matrix, int rij, int kolom)`". Deze methode verschuift de getallen in de opgegeven rij (2<sup>e</sup> parameter) zodanig dat het getal in de aangegeven kolom (3<sup>e</sup> parameter) vooraan komt te staan. Ga als volgt te werk: maak een tijdelijke (1-dimensionale) array aan met dezelfde breedte als de matrix. Kopieer nu de getallen uit de matrix-rij naar de tijdelijke array; begin bij de tijdelijke array vanaf 0, en bij de matrix-rij vanaf 'kolom' te indexeren. Nadat alle getallen naar de temp-array zijn gekopieerd, kopieer ze dan weer terug naar de matrix-rij (vanaf index 0).
7. Roep vanuit de Start-methode de methode "VerschuifMatrix" aan en vervolgens de methode "ToonMatrix" om de verschuivingen zichtbaar te maken, zie voorbeeld hiernaast.



```
file:///C:/Users/Gerwin...  -  □  ✕  
49 42 20 57 43 17 72 62 39 73  
19 14 48 81 06 09 24 03 62 37  
42 52 07 52 25 36 49 16 37 34  
78 87 50 06 63 68 15 10 10 85  
68 59 93 17 71 16 37 57 39 74  
  
Geef een getal: 6  
  
rij 2 verschuiven vanaf kolom 5...  
rij 4 verschuiven vanaf kolom 4...  
  
49 42 20 57 43 17 72 62 39 73  
06 09 24 03 62 37 19 14 48 81  
42 52 07 52 25 36 49 16 37 34  
06 63 68 15 10 10 85 78 87 50  
68 59 93 17 71 16 37 57 39 74
```

## Opgave 3: 'Verkiezingen'

(40 punten)



PvdA



SP.

CDA

D66

ChristenUnie

GROENLINKS

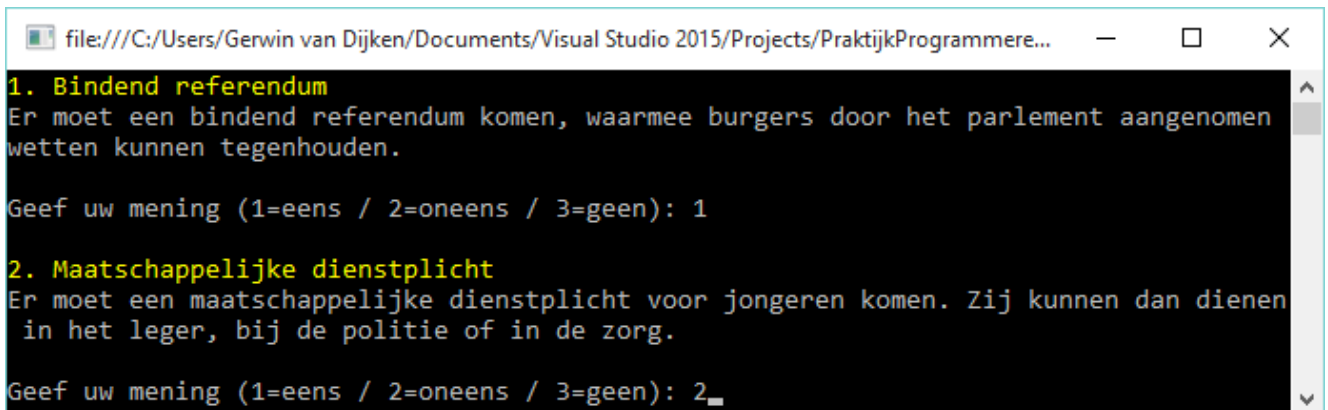
Met de Tweede kamerverkiezingen net achter de rug, gaan we in deze laatste opdracht een eigen 'kieswijzer' implementeren. Er wordt hierbij een aantal stellingen voorgelegd aan de gebruiker die bij elke stelling moet aangeven of hij/zij het daarmee eens/oneens is. Nadat alle stellingen voorgelegd zijn aan de gebruiker, gaat het programma deze antwoorden van de gebruiker vergelijken met de antwoorden van elke partij (alleen bovenstaande 8 partijen doen mee in deze kieswijzer...).

De stellingen (30x) worden uit een bestand gelezen ("stellingen.txt"); elke stelling bestaat 2 regels: eerst de titel en daarna de tekst van de stelling. De partijen (8x) worden ook uit een bestand gelezen ("partijen.txt"); elke partij bestaat 2 regels: eerst de naam van de partij en daarna de antwoorden van deze partij (op alle stellingen) in de vorm van een string van 30 karakters (1=eens, 2=oneens, 3=geen mening).

## Aan de slag:

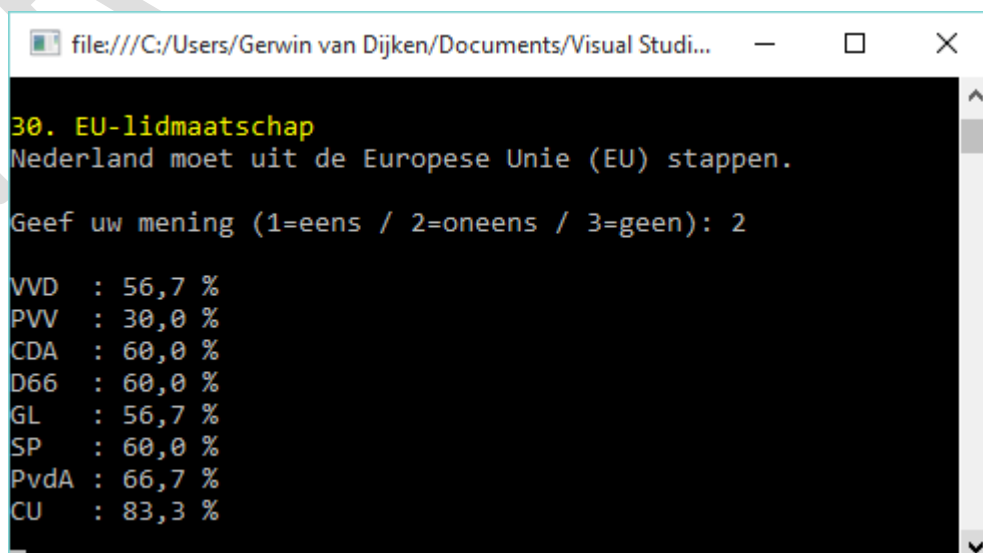
0. Voeg aan solution 'PraktijkProgrammeren2-Tentamen' een Console project toe met naam '**Opgave3**'. Download de 2 bestanden ("stellingen.txt" en "partijen.txt") en plaats deze in jouw projectdirectory.
1. Maak een **class** "Stelling" met de velden 'titel' (**string**) en 'tekst' (**string**) en plaats deze in een apart bestand genaamd "Stelling.cs". Maak een **class** "Partij" met de velden 'naam' (**string**) en 'antwoorden' (**string**) en plaats deze in een apart bestand genaamd "Partij.cs".
2. Maak een methode met signatuur "**List<Stelling>** LeesStellingen(**string** bestand)". In deze methode worden alle stellingen gelezen (uit bestand "stellingen.txt"). Ga als volgt te werk: maak een lijst aan waarin stellingen (type **Stelling**) geplaatst kunnen worden; zolang er nog regels niet gelezen zijn, maak een nieuwe stelling aan, lees zowel de titel als de tekst, en plaats deze stelling in de lijst. Zodra alle regels ingelezen zijn, retourneer de lijst met stellingen.
3. Maak een methode met signatuur "**List<Partij>** LeesPartijen(**string** bestand)". In deze methode worden alle partijen gelezen (uit bestand "partijen.txt"). Ga als volgt te werk: maak een lijst aan waarin partijen (type **Partij**) geplaatst kunnen worden; zolang er nog regels niet gelezen zijn, maak een nieuwe partij aan, lees zowel de naam als de antwoorden, en plaats deze partij in de lijst. Zodra alle regels ingelezen zijn, retourneer de lijst met partijen.
4. Lees vanuit de Start-methode alle stellingen (via methode LeesStellingen), en lees alle partijen (via methode LeesPartijen). Stop het programma als er geen stellingen in de (ontvangen) stellingen-lijst staan of als er geen partijen in de (ontvangen) partijen-lijst staan.

5. Maak een methode met signatuur "`string` DoorloopStellingen(`List<Stelling>` stellingen)". Deze methode laat alle stellingen zien en leest per stelling de mening van de gebruiker. Ga als volgt te werk: doorloop de lijst (bv via een foreach-lus) en print van elke stelling de titel en de tekst; vraag bij elke stelling de mening van de gebruiker. De gebruiker voert alleen een getal in: 1=eens, 2=oneens, 3=geen mening (zie voorbeeld hieronder); dit antwoord wordt steeds toegevoegd aan een string. Nadat alle stellingen verwerkt zijn, wordt deze string met alle antwoorden (30 karakters lang) geretourneerd.



```
file:///C:/Users/Gerwin van Dijken/Documents/Visual Studio 2015/Projects/PraktijkProgrammere...  
1. Bindend referendum  
Er moet een bindend referendum komen, waarmee burgers door het parlement aangenomen  
wetten kunnen tegenhouden.  
  
Geef uw mening (1=eens / 2=oneens / 3=geen): 1  
  
2. Maatschappelijke dienstplicht  
Er moet een maatschappelijke dienstplicht voor jongeren komen. Zij kunnen dan dienen  
in het leger, bij de politie of in de zorg.  
  
Geef uw mening (1=eens / 2=oneens / 3=geen): 2_
```

6. Maak een methode met signatuur "`double` BepaalMatchPercentage(`string` gebruiker, `Partij` partij)". Deze methode bepaalt in hoeverre de gebruiker (1<sup>e</sup> parameter) een match heeft met een partij (2<sup>de</sup> parameter). Ga als volgt te werk: doorloop alle antwoorden in de string gebruiker (30 karakters met de karakters '1', '2' en '3') en vergelijk deze met de antwoorden van de partij (dus karakter voor karakter). Retourneer aan het einde de verhouding van het aantal gelijke antwoorden gedeeld door het aantal stellingen \* 100%.
7. Maak een methode met signatuur "`void` VergelijkPartijen(`string` gebruiker, `List<Partij>` partijen)". Deze methode doorloopt de lijst met partijen en bepaalt van elke partij in hoeverre de gebruiker daarmee een match heeft; dit gebeurt uiteraard via een aanroep van methode BepaalMatchPercentage (zie hierboven). Toon van elke partij de naam en het "match-percentage".
8. We moeten alleen nog de Start-methode uitbreiden. Roep de methode DoorloopStellingen aan en vervolgens VergelijkPartijen. Het eindresultaat is dan als hieronder weergegeven.



```
file:///C:/Users/Gerwin van Dijken/Documents/Visual Studi...  
30. EU-lidmaatschap  
Nederland moet uit de Europese Unie (EU) stappen.  
  
Geef uw mening (1=eens / 2=oneens / 3=geen): 2  
  
VVD : 56,7 %  
PVV : 30,0 %  
CDA : 60,0 %  
D66 : 60,0 %  
GL : 56,7 %  
SP : 60,0 %  
PvdA : 66,7 %  
CU : 83,3 %
```