

Assignment 1: 'Letter order'

(25 points)

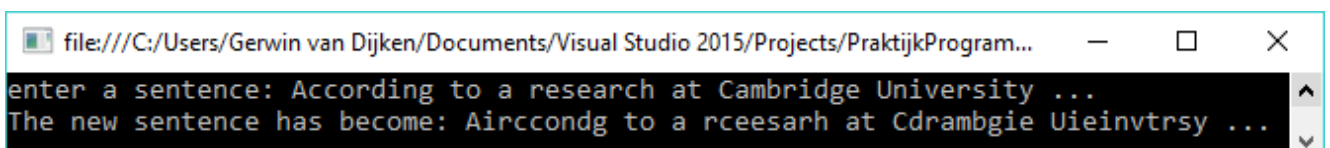
"Aoccdrnig to a rscheearch at Cmabrigde Uinervtisy, it deosn't mtttaer in waht oredr the ltteers in a wrod are, the olny iprmoatnt tihng is taht the frist and lsat ltteers be at the rghit plcae. The rset can be a toatl mses and you can sitll raed it wouthit porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe."

[According to a research at Cambridge University, it doesn't matter in what order the letters in a word are, the only important thing is that the first and last letter be at the right place. The rest can be a total mesh and you can still read it without problem. This is because the human does not read every letter by itself, but the word as a whole.]

In this assignment we will shuffle the words in a sentence, and see if it's true that the order of the letters doesn't matter. Will the shuffled sentence still be readable?

Let's get started:

0. Create a Console project with name '**Assignment1**', and give the solution the name '**Programming2-TestExam**'.
(the other assignments will also be added as projects to this solution, see later)
1. In de Start method ask the user to enter a sentence and save this text (in a variable).
2. Create a method with signature "`string ShuffleWords(string sentence)`". This method receives a sentence and returns this sentence with each word shuffled.
Follow the next procedure:
 - a. use the Split method using a space (`sentence.Split(' ');`) to split the sentence into a word array (`string[]`);
 - b. iterate over this word array and call for each word method "ShuffleWord" (see next section). Add each returned word (of ShuffleWord) to a result variable;
 - c. return the concatenated sentence as result of this method;
3. Create a method with signature "`string ShuffleWord(string word)`" that returns a shuffled version of a word. Follow the next procedure:
 - a. if the received word contains 3 letters or less, then the word cannot be shuffled; return the same word as result (*skip items b..f*);
 - b. create a local variable 'newWord' and give it the first letter of the (received) word;
 - c. create a local variable 'remainingWord' and store the middle part of the (received) word; use `Substring(...)`;
 - d. while the remaining word contains letters:
 - i. pick a random letter from the remaining word (random index), and add this letter to the new word;
 - ii. remove this letter from the remaining word (`remainingWord.Remove(index, 1)`, and store the result of Remove back in remainingWord);
 - e. add the last letter of the (received) word to the new word;
 - f. return the new word as result;
4. Call from the Start the method "ShuffleWords" with the entered text (from the user) as parameter, and display the result (see example below).



```
file:///C:/Users/Gerwin van Dijken/Documents/Visual Studio 2015/Projects/PraktijkProgram...
enter a sentence: According to a research at Cambridge University ...
The new sentence has become: Airccondg to a rceesarh at Cdrambgie Uieinvtrsy ...
```

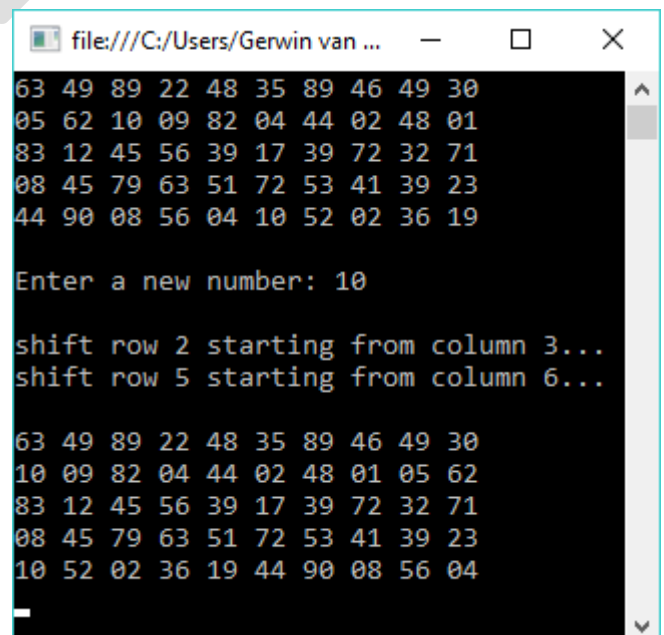
Assignment 2: 'Shift matrix'

(25 point)

In this assignment we will manipulate a matrix with random numbers, by shifting certain rows.

Let's get started:

0. Add to solution 'Programming2-TestExam' a Console project with name '**Assignment2**'.
1. Create a 2-dimensional array with name "matrix" (in the Start method) with 5 rows and 10 columns.
2. Implement a method "FillMatrix" that receives the 2-dimensional array as parameter. In this method fill the complete array with random numbers between 1 (inclusive) and 100 (exclusive). Call this method from the Start method.
3. Implement a method "DisplayMatrix" that receives the 2-dimensional array as parameter. This method displays the array (as in the example below). Call this method from the Start method.
4. Ask the user (in the Start method) to enter a search number (`int`). This number will be used to shift the rows.
5. Implement a method "`void ShiftMatrix(int[,] matrix, int number)`". If a row contains the number then this number is shifted to the front of the row.
Follow the next procedure: process each row and search for the number (by iterating over the columns); stop searching if the number is found. If the number has been found in a row, then call method "ShiftRow" (see next item), passing the row and column (of the number found) as parameters.
6. Implement a method "`void ShiftRow(int[,] matrix, int row, int column)`". This method shifts the numbers in the given row (2nd parameter) to the left so the number in the given column (3rd parameter) will be positioned at the front of the row.
Follow the next procedure: create a temporary (1-dimensional) array with the same width as the matrix. Copy all numbers from the matrix-row to the temporary array; start indexing the temporary array at 0, and the matrix-row at 'column'. After all numbers have been copied to the temporary array, copy them back to the matrix-row (start indexing at 0).
7. Call method "ShiftMatrix" from the Start method and then method "DisplayMatrix" to see the shifted matrix, see example to the right.



```
file:///C:/Users/Gerwin van ...
63 49 89 22 48 35 89 46 49 30
05 62 10 09 82 04 44 02 48 01
83 12 45 56 39 17 39 72 32 71
08 45 79 63 51 72 53 41 39 23
44 90 08 56 04 10 52 02 36 19

Enter a new number: 10

shift row 2 starting from column 3...
shift row 5 starting from column 6...

63 49 89 22 48 35 89 46 49 30
10 09 82 04 44 02 48 01 05 62
83 12 45 56 39 17 39 72 32 71
08 45 79 63 51 72 53 41 39 23
10 52 02 36 19 44 90 08 56 04
```

Assignment 3: 'Elections'

(40 points)



PvdA



SP.

CDA

D66

ChristenUnie

GROENLINKS

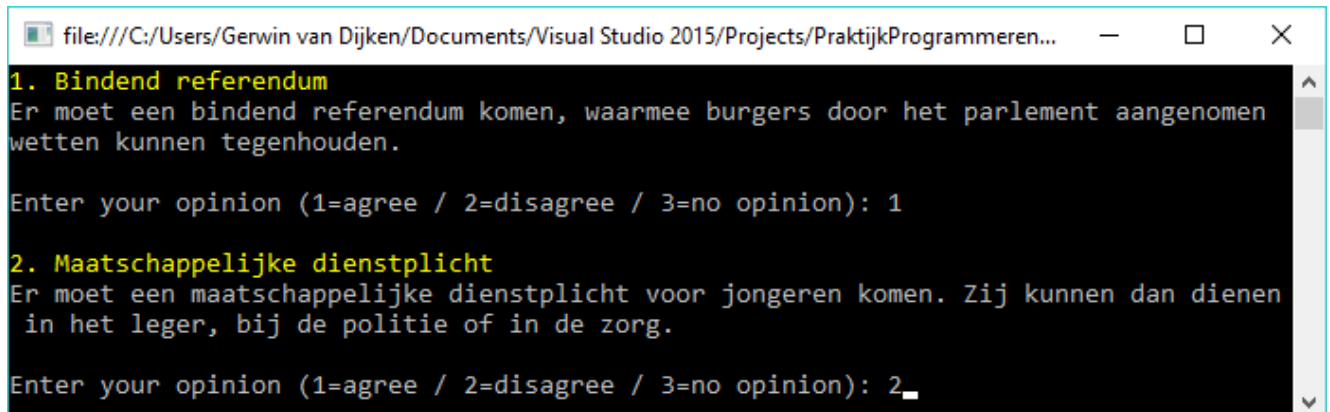
In this last assignment we will implement an election application that will help civilians to vote on 'the right party'. A number of premises (subjects) are presented to the user, and each time the user must indicate that he/she agrees or disagrees with the premise. After all premises are presented to the user, the program will compare the answers of the user to the answers of several parties (*only the 8 parties in the banner above are joining...*).

The premises (30x) are read from a file ("permises.txt"); for each premise there are 2 lines in the file: first the title and then the text of the premise. The parties (8x) are also read from a file ("parties.txt"); for each party there are 2 lines in the file: first the name of the party and then the answers of this party (to all 30 premises): a string with 30 characters (1=agree, 2=disagree, 3=no opinion).

Let's get started:

0. Add to solution 'Programming2-TestExam' a Console project with name '**Assignment3**'. Download the 2 files ("permises.txt" and "parties.txt") and save them in your projectdirectory.
1. Create a `class` "Premise" with fields 'title' (`string`) and 'text' (`string`) and save it in a separate file "Premise.cs". Create a `class` "Party" with fields 'name' (`string`) and 'answers' (`string`) and save it in a separate file "Party.cs".
2. Create a method with signature "`List<Premise> ReadPremises(string filename)`". In this method all premises are read (from file "permises.txt").
Follow the next procedure: create a list for storing premises (type `Premise`); while there are more lines in the file to read, create a new `Premise`, read title and text from the file, and add the new premise to the list. When all lines have been read from the file, return the list with premises.
3. Create a method with signature "`List<Party> ReadParties(string filename)`". In this method all parties are read (from file "parties.txt").
Follow the next procedure: create a list for storing parties (type `Party`); while there are more lines in the file to read, create a new `Party`, read name and answer from the file, and add the new party to the list. When all lines have been read from the file, return the list with parties.
4. Read from the Start method all premises (via method `ReadPremises`), and read all parties (via method `ReadParties`). Stop the program if there are no premises in the (received) list or if there are no parties in the (received) list.

5. Create a method with signature "`string ProcessPremises(List<Premise> premises)`". This method displays all premises and reads the opinion of the user for every premise. Follow the next procedure: process the list (with a for-loop or foreach-loop) and print for each premise the title and text; ask the user to enter his/her opinion on the given premise. The user only enters a number: 1=agree, 2=disagree, 3=no opinion (see *example below*); everytime the answer is added to a string. After processing all premises, return this string with all user-answers (30 characters long).



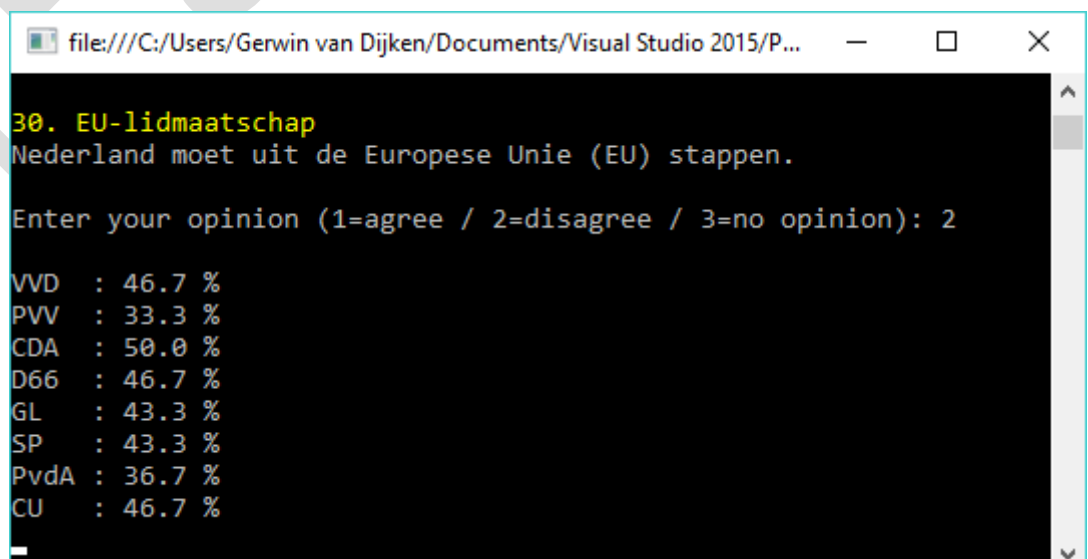
```
file:///C:/Users/Gerwin van Dijken/Documents/Visual Studio 2015/Projects/PraktijkProgrammeren...
1. Bindend referendum
Er moet een bindend referendum komen, waarmee burgers door het parlement aangenomen
wetten kunnen tegenhouden.

Enter your opinion (1=agree / 2=disagree / 3=no opinion): 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.

Enter your opinion (1=agree / 2=disagree / 3=no opinion): 2
```

6. Create a method with signature "`double DetermineMatch(string user, Party party)`". This method determines the match (percentage) of the user (1st parameter) with a party (2nd parameter). Follow the next procedure: process all answers in the string user (30 characters containing '1', '2' and '3') and compare these with the answers of the party (one by one). At the end return the proportion of the number of same answers divided by the total number of premises * 100%.
7. Create a method with signature "`void CompareParties(string user, List<Party> parties)`". This method processes the list with parties and determines for each party the match (percentage) of the user; of course this will be done by calling method DetermineMatch (see item 6). Display for every party the name and the "match percentage".
8. Now we only need to extend the Start method; call method ProcessPremises and then CompareParties. The end result of the program is shown below.



```
file:///C:/Users/Gerwin van Dijken/Documents/Visual Studio 2015/P...
30. EU-lidmaatschap
Nederland moet uit de Europese Unie (EU) stappen.

Enter your opinion (1=agree / 2=disagree / 3=no opinion): 2

VVD : 46.7 %
PVV : 33.3 %
CDA : 50.0 %
D66 : 46.7 %
GL : 43.3 %
SP : 43.3 %
PvdA : 36.7 %
CU : 46.7 %
```