

Tipe assessering/
Type of assessment: Semesterprojek
Semester project
Modulekode/
Module code: CMPG111
Modulebeskrywing/
Module description: Inleiding tot rekenaarwese en programmering /
Introduction to computing and programming
Eksaminator(e)/
Examiner(s): Mnr. D.P. Snyman
Ms. N. Dladlu
Mrs. T. Eksteen

Kwalifikasie/ B.Sc.; B.Ing.;
Qualification: B.Com.
Tydsduur/ -
Duration:
Maks/ 100
Max:
Datum/ 14 Junie 2018
Date: 14 June 2018

Vraag 1.1 [10]
Bepaalde-iterasie lusse, keuse-strukture en rekenkunde

*Skep 'n **Vraag1.1.py**-lêer en voltooi die onderstaande program*

FizzBuzz is 'n eenvoudige programmerings-probleem wat algemeen in programmeerder-werksonderhoude gebruik word om kandidate se logiese redeneringsvermoë te toets.

Gegewe 'n reeks waardes, skryf elke waarde na die skerm, maar vir veelvoude van drie vertoon "Fizz" in plaas van die getal en vir die veelvoude van vyf vertoon "Buzz". Vir getalle wat veelvoude van beide drie en vyf is vertoon "FizzBuzz".

Skryf 'n program wat die getalle vertoon, soos hierbo beskryf, van 1 tot by 'n waarde wat die gebruiker nomineer.

*Wenk: Gebruik die **modulus** operator % om te bepaal of 'n getal 'n veelvoud van drie/vyf is.*

Modulus gee die res na heelgetaldeling.

Sien Figuur 1 vir voorbeeldafvoer.

Question 1.1 [10]
Definite iteration loops, decision structures and arithmetic

*Create a **Question1.1.py** file and complete the program below*

FizzBuzz is a simple programming problem that is commonly used during programmer employment interviews to test a candidate's logical reasoning.

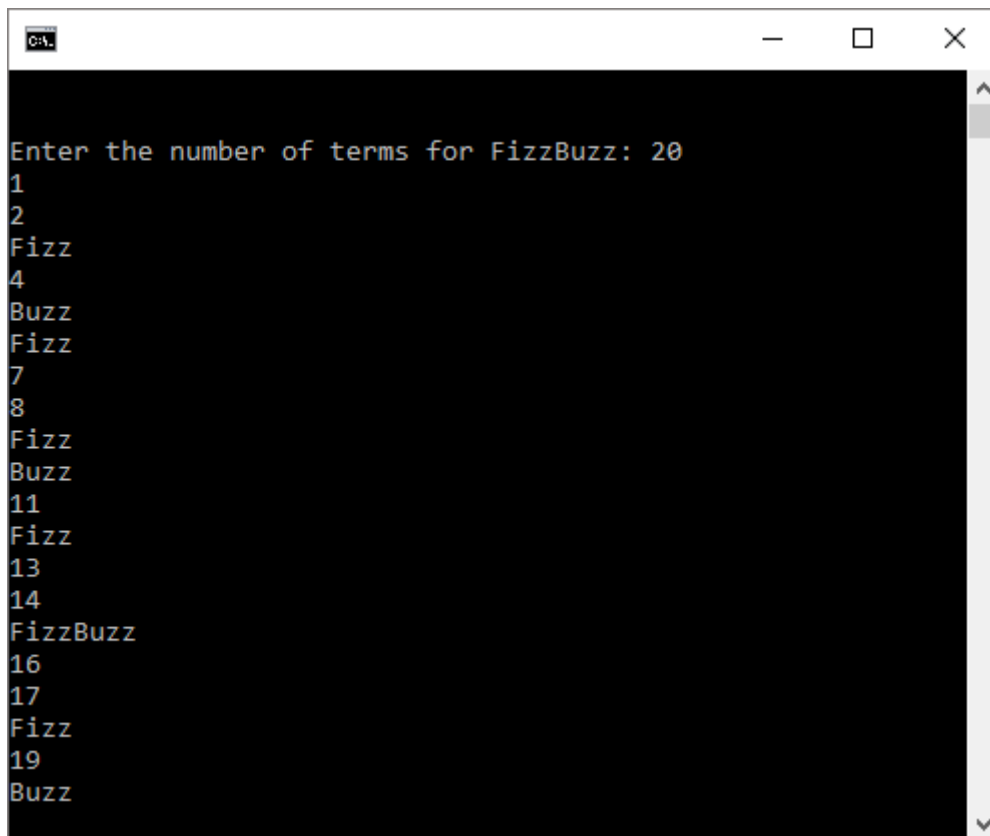
Given a series of values, write every value to the screen, but for multiples of three display "Fizz" instead of the number and for the multiples of five display "Buzz". For numbers which are multiples of both three and five display "FizzBuzz".

Write a program that displays the numbers, as described above, from 1 to a value that is nominated by the user.

*Hint: Use the **modulus** operator % to determine whether a number is a multiple of three/five.*

Modulus gives the remainder after integer division.

See Figure 1 for example output.



```
C:\>  
Enter the number of terms for FizzBuzz: 20  
1  
2  
Fizz  
4  
Buzz  
Fizz  
7  
8  
Fizz  
Buzz  
11  
Fizz  
13  
14  
FizzBuzz  
16  
17  
Fizz  
19  
Buzz
```

Figuur 1 / Figure 1

Vraag 1.2

[15]

Skep 'n **Vraag1.2.py**-lêer en voltooi die onderstaande program.

Enkripsie word algemeen gebruik om boodskappe op die internet te verdoesel. 'n Caesar-kode maak 'n skuif van al die karakters in 'n string (gebaseer op hul ASCII-waardes, sien Tabel 1.2), bv.

hello → mjqqt

Die voorbeeld toon 'n verskuiwing met 'n afstand van **5** karakters, d.i. $h_{(ASCII:104)} \rightarrow m_{(ASCII:109)}$

Skryf 'n Python-program wat die gebruiker vra om 'n reël gewone teks en die afstandwaarde in te voer en 'n geënkripteerd teks lewer deur 'n Caesar-kode te gebruik. Die ASCII-waardes wissel van 0 tot 127. Die program moet vir enige vertoonbare karakters werk.

Sien Figuur 1.2 vir voorbeeld afvoer.

Question 1.2

[15]

Create a **Question1.2.py** file and complete the program below.

Encryption is commonly used to disguise messages on the internet. A Caesar cipher performs a shift of all of the characters in a string (based on their ASCII values, see Table 1.2), e.g.

hello → mjqqt

The example shows a shift with a distance of **5** characters, i.e. $h_{(ASCII:104)} \rightarrow m_{(ASCII:109)}$

Write a Python program that ask the user to input a line of plaintext and the distance value and outputs an encrypted text using a Caesar cipher, with the ASCII values range from 0 through 127.

The program should work for any printable characters.

See Figure 1.2 for example output.

```

Enter a message: hello sipho your pin is 2365
Enter the distance value: 5
The encoded message is: mjqqt%xnumt%~tzw%uns%nx%78;:

```

Figuur 1.2 / Figure 1.2

| ASCII-Tabel / ASCII Table | | | | | | | | | | | | | | | |
|---------------------------|-----|----|-----|----|----|----|---|----|---|----|---|-----|---|-----|-----|
| 0 | NUL | 16 | DLE | 32 | SP | 48 | 0 | 64 | @ | 80 | P | 96 | ` | 112 | p |
| 1 | SOH | 17 | DC1 | 33 | ! | 49 | 1 | 65 | A | 81 | Q | 97 | a | 113 | q |
| 2 | STX | 18 | DC2 | 34 | " | 50 | 2 | 66 | B | 82 | R | 98 | b | 114 | r |
| 3 | ETX | 19 | DC3 | 35 | # | 51 | 3 | 67 | C | 83 | S | 99 | c | 115 | s |
| 4 | EOT | 20 | DC4 | 36 | \$ | 52 | 4 | 68 | D | 84 | T | 100 | d | 116 | t |
| 5 | ENQ | 21 | NAK | 37 | % | 53 | 5 | 69 | E | 85 | U | 101 | e | 117 | u |
| 6 | ACK | 22 | SYN | 38 | & | 54 | 6 | 70 | F | 86 | V | 102 | f | 118 | v |
| 7 | BEL | 23 | ETB | 39 | ' | 55 | 7 | 71 | G | 87 | W | 103 | g | 119 | w |
| 8 | BS | 24 | CAN | 40 | (| 56 | 8 | 72 | H | 88 | X | 104 | h | 120 | x |
| 9 | HT | 25 | EM | 41 |) | 57 | 9 | 73 | I | 89 | Y | 105 | i | 121 | y |
| 10 | LF | 26 | SUB | 42 | * | 58 | : | 74 | J | 90 | Z | 106 | j | 122 | z |
| 11 | VT | 27 | ESC | 43 | + | 59 | ; | 75 | K | 91 | [| 107 | k | 123 | { |
| 12 | FF | 28 | FS | 44 | , | 60 | < | 76 | L | 92 | \ | 108 | l | 124 | |
| 13 | CR | 29 | GS | 45 | - | 61 | = | 77 | M | 93 |] | 109 | m | 125 | } |
| 14 | SO | 30 | RS | 46 | . | 62 | > | 78 | N | 94 | ^ | 110 | n | 126 | ~ |
| 15 | SI | 31 | US | 47 | / | 63 | ? | 79 | O | 95 | _ | 111 | o | 127 | DEL |

Tabel 1.2 / Table 1.2

Vraag 2.1 [20]

Skep 'n **Vraag2.1.py**-lêer en voltooi die onderstaande program.

Skryf 'n program wat herhaaldelik vir die gebruiker vra om 'n reële waarde in te lees. Die program moet aanhou waardes aanvaar totdat die gebruiker 'n negatiewe waarde ingelees het.

Sodra die gebruiker 'n negatiewe waarde inlees moet 'n opsomming aan die gebruiker vertoon word met:

1. Die hoeveelheid waardes wat ingelees is;
2. die hoogste waarde;
3. die laagste waarde; en
4. die gemiddelde waarde (afgerond tot 3 desimale plekke).

Sien Figuur 2.1 vir voorbeeldafvoer.

Question 2.1 [20]

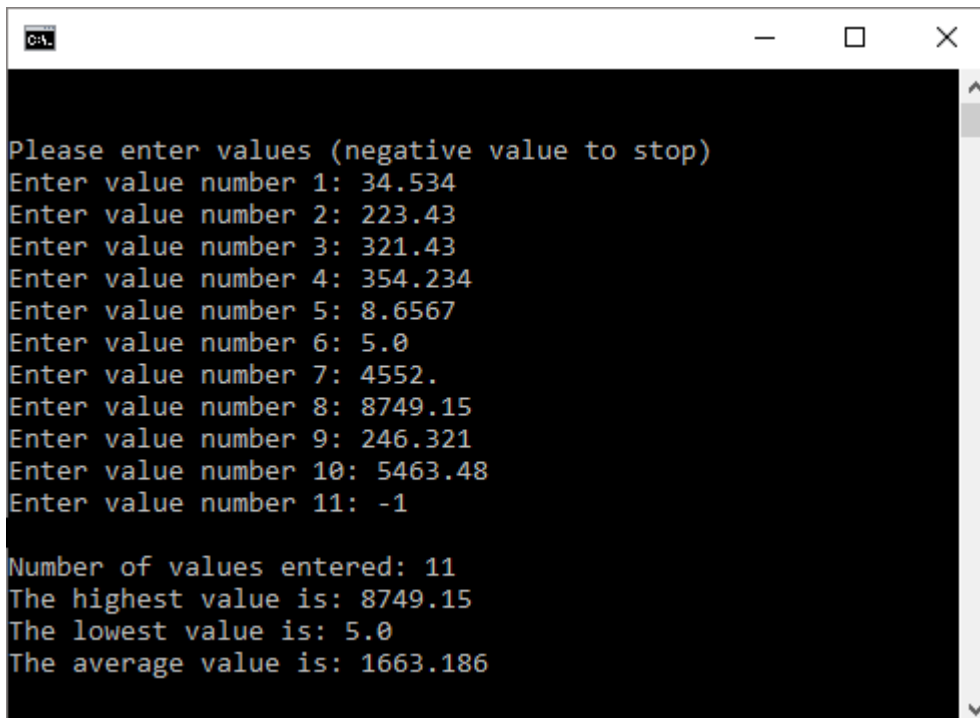
Create a **Question2.1.py** file and complete the program below.

Write a program that repeatedly asks the user to enter real values. The program must keep accepting values until a negative value is entered.

As soon as the user enters a negative value, a summary must be shown with:

1. The number of values entered;
2. the highest value;
3. the lowest value; and
4. the average value (rounded to 3 decimal places).

See Figure 2.1 for example output.



```
Please enter values (negative value to stop)
Enter value number 1: 34.534
Enter value number 2: 223.43
Enter value number 3: 321.43
Enter value number 4: 354.234
Enter value number 5: 8.6567
Enter value number 6: 5.0
Enter value number 7: 4552.
Enter value number 8: 8749.15
Enter value number 9: 246.321
Enter value number 10: 5463.48
Enter value number 11: -1

Number of values entered: 11
The highest value is: 8749.15
The lowest value is: 5.0
The average value is: 1663.186
```

Figuur 2.1 / Figure 2.1

Vraag 2.2 [15]

Skep 'n **Vraag2.2.py**-lêer en voltooi die onderstaande program.

Skryf 'n Python-program wat as toevoer die naam en prys van 'n onbekende aantal items sal ontvang. Toevoer moet beëindig word wanneer die gebruiker 'X' as 'n item-naam insleutel.

Bereken die BTW-bedrag en die totale prys, insluitend BTW vir die items. Vertoon die item naam, BTW bedrag, en bedrag insluitend BTW.

Question 2.2 [15]

Create a **Question2.2.py** file and complete the program below.

Write a Python program that will receive as input the name and price of an unknown number of items. Input should terminate when the user enters "X" as an item name.

Calculate the VAT amount and the total price including VAT for the items. Display the item name, VAT amount, and amount including VAT.

Gebruik die volgende BTW-koerse vir u berekeninge:

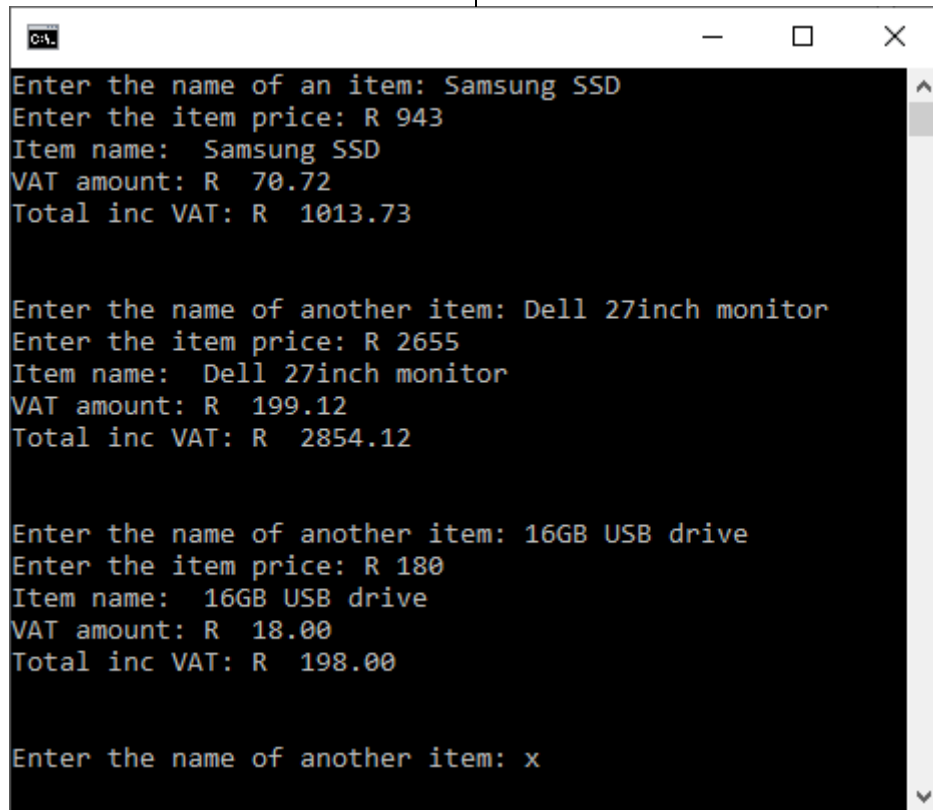
| BTW-koers | Item prys |
|-----------|----------------|
| 15% | Minder as R100 |
| 10% | R101 – R249 |
| 7.5% | Meer as R250 |

Sien Figuur 2.2 vir voorbeeldafvoer.

Use the following VAT rates for your calculations:

| VAT Rate | Item Price |
|----------|----------------|
| 15% | Less than R100 |
| 10% | R101 – R249 |
| 7.5% | More than R250 |

See Figure 2.2 for example output.



```
C:\>
Enter the name of an item: Samsung SSD
Enter the item price: R 943
Item name: Samsung SSD
VAT amount: R 70.72
Total inc VAT: R 1013.73

Enter the name of another item: Dell 27inch monitor
Enter the item price: R 2655
Item name: Dell 27inch monitor
VAT amount: R 199.12
Total inc VAT: R 2854.12

Enter the name of another item: 16GB USB drive
Enter the item price: R 180
Item name: 16GB USB drive
VAT amount: R 18.00
Total inc VAT: R 198.00

Enter the name of another item: x
```

Figuur 2.2 / Figure 2.2

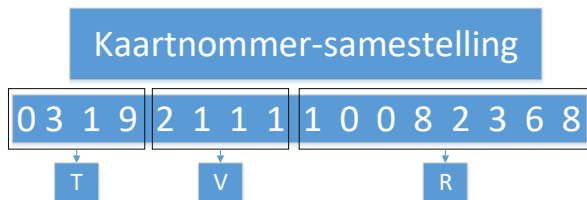
Vraag 3

[10+20+7+3=40]

Skep 'n **Vraag3.py-lêer** en voltooi die onderstaande program **as 'n geheel** deur al die sub-vrae te voltooi.

Bankkaarte word aan kliente van ABC-bank uitgereik om transaksies op hulle rekeninge aan te gaan. 'n Nuwe soort kaart wat hulle beskikbaar stel het kaartnommer wat bestaan uit sestien syfers wat uniek aan elke kaart is wat uitgereik word. Die kaartnommer word só saamgestel dat sekere basiese inligting daarin geënkodeer word deur sekere syfers saam te groepeer.

Die samestelling van 'n kaartnommer lyk soos volg:



Waar:

- T – Tipe kaart
- V – Vervaldatum
- R – Rekeningnommer

Die eerste vier syfers (T) stel die kaart se tipe en uitreikingsjaar voor in die formaat

TTJJ

waar T die tipe (sien Tabel 3.1) en J die jaartal is, dus in die voorbeeld is 0319 = *Diner's club, uitgereik in 2019*.

Die volgende vier syfers (V) dui die kaart se vervaldatum aan in die formaat

JJMM

waar J die jaartal en M die Maand van die jaar is, dus in die voorbeeld is 2111 = 2021, 11e maand.

Die laaste agt syfers (R) dui die kaart se rekeningnommer aan. Die reeks wat die rekeningnommer voorstel begin met 1 of 0:

0XXXXXXX – Debietrekening
1XXXXXXX – Kredietrekening

dus in die voorbeeld is
10082368 = *Kredietrekening*.

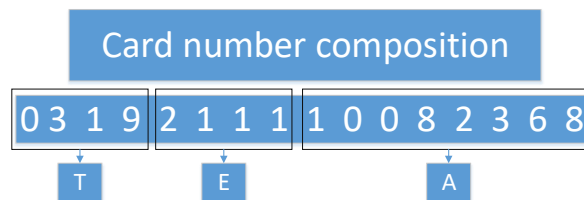
Question 3

[10+20+7+3=40]

Create a **Question3.py** file and complete the program below **as a whole** by completing all of the sub questions.

Bank cards are issued to clients of ABC Bank to enter into transactions on their accounts. A new type of card that they issue has a card number which consists of sixteen digits that are unique to each card issued. The card number is compiled in such a way that certain basic information is encoded therein by grouping certain digits.

The composition of a card number is as follows:



Where:

- T – Type
- E – Expiry date
- A – Account number

The first four digits (T) represent the card type and issue year in the format

TTYJ

where T is the type (see Table 3.1) and Y is the year, so in the example is 0319 = *Diner's club, issued in 2019*.

The following four digits (E) indicate the card's expiry date in the format

YYMM

Where Y is the year and M is the Month of the year, so in the example 2111 = 2021, 11th month.

The last eight digits (A) indicate the card's account number. The series representing the account number starts with 1 or 0:

0XXXXXXX – Debit account
1XXXXXXX – Credit account

so, in the example 10082368 = *Credit account*.

Skryf nou 'n Python-program wat 'n lys van kaartnommers uit 'n teks lêer kan lees en die kaartnommers kan analiseer om die geënkodeerde gegewens soos hierbo vir elke nommer te onttrek. Die gedekodeerde gegewens moet na 'n teks lêer geskryf word.

Sien Figure 3.1 en 3.2 vir voorbeeldafvoer.

Vrae 3.1-3.4 moet, in volgorde, voltooi word **in een Python-lêer** ten einde 'n volledige, werkende program daar te stel.

Vraag 3.1 [10]

Skryf 'n funksie, **ReadFile()**, wat die inhoud van 'n gegewe teks lêer lees en in 'n lys stoor. Die teks lêer, Data.txt, word voorsien en die inhoud daarvan word in Tabel 3.2 voorgehou.

Die Data.txt-lêer word gestoor onder "K:\Data.txt". Maak 'n rugsteun van hierdie lêer voordat die lêer deur die program gebruik word.

Elke reël in die teks lêer bestaan uit een kaartnommer.

Die funksie ontvang een parameter: 'n string-lys waarin die individuele inskrywings uit die teks lêer gestoor moet word. Die lys word as verwysing aan die funksie gestuur.

Die funksie stuur die hoeveelheid reëls wat uit die teks lêer gelees is terug na die roepende stelling. Indien die lêer nie gelees kan word nie, moet 'n foutboodskap vertoon word en die program moet toemaak sonder om verder uit te voer.

NB! – Indien die funksie in Vraag 3.1 nie reg funksioneer nie mag die lys eksplisiet geskep word in die hoofprogram asof die funksie die inhoud uit die lêer verkry het. Hierdie lys is nodig vir die funksies wat geskep word in die volgende vrae. Sien Tabel 3.1.

Now write a Python program that can read a list of card numbers from a text file and analyse the card numbers to extract the encoded data as shown above for each number. The decoded data must be written to a text file.

See Figures 3.1 and 3.2 for example output.

Questions 3.1-3.4 must be completed, in order, **in one Python file** to create a complete, working program.

Question 3.1 [10]

Write a function, **ReadFile()**, that reads the contents of a text file and stores it in a list. The text file, Data.txt, is provided and its content is presented in Table 3.2.

The Data.txt file is saved under "K:\Data.txt". Make a backup of the file before the file is accessed by the program.

Each line in the text file consists of one card number.

The function receives one parameter: A string list for where each individual entry from the text file must be stored. The list is passed by reference to the function.

The function returns the number of lines that were read from the text file to the calling statement. If the file cannot be read, an error message should be displayed, and the program should close without any further execution.

NB! – If the function in Question 3.1 does not function correctly, a list may be explicitly created in the main program as if the function has read the contents from the file. This list is to be used in the following questions. See Table 3.1.

Vraag 3.2

[20]

Skryf 'n funksie, **Decode()** wat 'n lys van kaartnommers kan analiseer. Die funksie moet soos volg saamgestel word:

Die funksie ontvang die string-lys (soos in Vraag 3.1) as parameter en stuur geen waarde aan die roepende stelling terug nie.

Wanneer hierdie funksie uitvoer moet die volgende gebeur:

Elke kaartnommer in die lys moet om die beurt geanaliseer word om die geënkodeerde inligting te onttrek in die volgende formaat:

```
0318241215528881: Was issued by Diner's club in
2018. The card expires on 24/12. The card is
linked to a Credit account with account number:
15528881.
```

Daar mag aangeneem word dat enige jaartal (JJ) na die jare 2000+ verwys.

Die inligting vir elke kaartnommer moet in een string-veranderlike gestoor word. Die veranderlike moet dan aan die WriteFile()-funksie (sien Vraag 3.3 hieronder) gestuur word sodat die resultaat gestoor kan word. Die proses herhaal totdat elke kaartnommer in die lys na die WriteFile()-funksie gestuur is.

Wenke: Maak van stringhantering gebruik om die kaartnommer se verskillende groeperings te onttrek.
Skryf 'n ekstra funksie om die syfers wat die tipe kaart voorstel om te skakel na 'n teksvoorstelling, bv. 03 → Diner's club

Vraag 3.3

[7]

Skryf 'n funksie, **WriteFile()**, wat inligting in 'n tekslêer stoor. Die funksie ontvang 'n string-waarde (verkry uit Vraag 3.2) wat na die tekslêer geskryf moet word. Die funksie stuur nie enige waarde aan die roepende stelling terug nie.

Stoor die afvoertekslêer as Output.txt.

Die inligting moet telkens op 'n nuwe reël in die tekslêer bygevoeg word sonder om die bestaande inligting uit te vee.

Question 3.2

[20]

Write a function, **Decode()** that can analyse a list of card numbers. The function is constructed as follows:

The function receives the string list (as in Question 3.1) as parameter and returns no value to the calling statement.

When this function executes the following should happen:

Each card number in the list should be analysed in turn to extract the encoded information in the following format:

```
0318241215528881: Was issued by Diner's club in
2018. The card expires on 24/12. The card is
linked to a Credit account with account number:
15528881.
```

It may be assumed that any year (YY) refers to the years 2000+.

The information for each card number must be stored in one string variable. The variable must then be sent to the WriteFile() function (see Question 3.3 below) so that the result can be saved. The process repeats until each card number in the list is sent to the WriteFile() function.

Hints:

Make use of string handling to extract the different groupings of the card number.
Write an extra function that converts the digits that represent the type of card to a text representation, e.g. 03 → Diner's club

Question 3.3

[7]

Write a function, **WriteFile()**, that saves information to a text file. The function receives a string value (obtained from Question 3.2) that is written to the text file. The function does not return any value to the calling statement.

Store the output text file as Output.txt.

The output must be added on a new line to the text file each time without erasing any existing information.

Vraag 3.4

[3]

Die bostaande funksies (**ReadFile()**, **Decode()**, en **WriteFile()**) moet nou in die hoofprogram geïmplementeer word in die volgende volgorde:

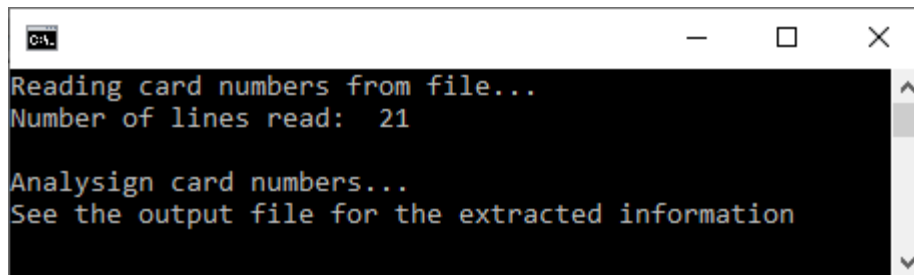
1. Skep 'n lys van tipe karakter, d.i. die datatype wat in die lys gestoor word, is teks.
2. Lees die inhoud van die lêer na die lys met **ReadFile()**. Vertoon die hoeveelheid reëls wat uit die tekslêer gelees is.
3. Dekodeer elke kaartnommer met die **Decode()**-funksie deur die lys vir die funksie te stuur.
4. Soos wat elke kaartnommer gedekodeer word, skryf die gedekodeerde inligting na 'n afvoerlêer met **WriteFile()**. Die WriteFile()-funksie moet vanuit die Decode()-Funksie geroep word.

Question 3.4

[3]

The above functions (**ReadFile()**, **Decode()**, and **WriteFile()**) must now be implemented in the main program in the following order:

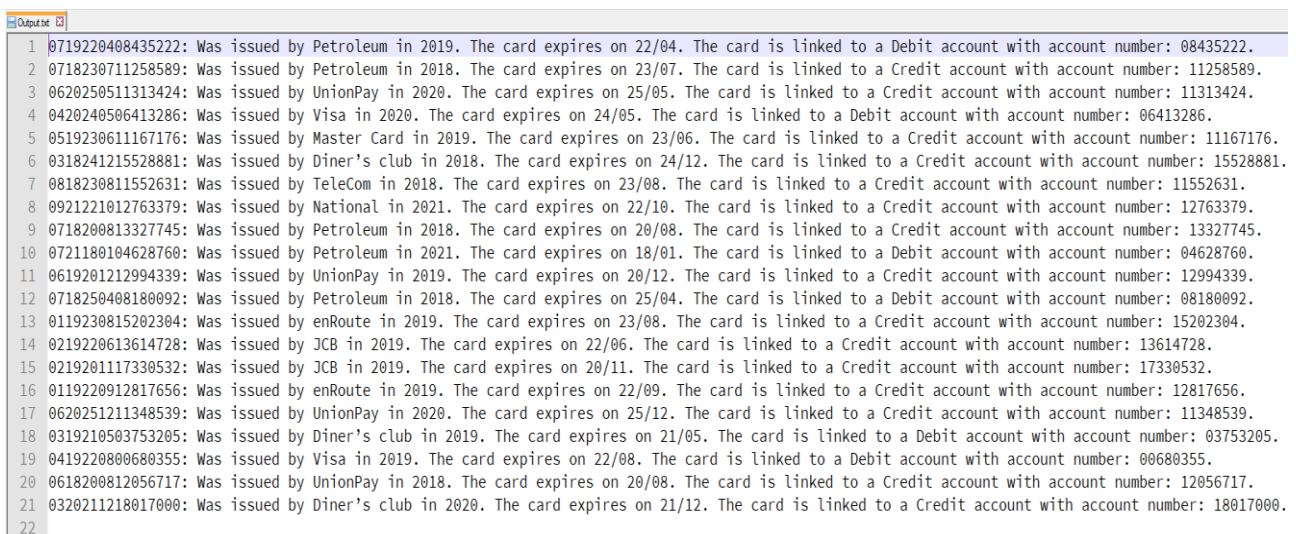
1. Create a list of type character, i.e. the data type that is stored in the list is text.
2. Read the contents of the file to the lists with **ReadFile()**. Display the number of lines that are read from the text file.
3. Decode each card number with the **Decode()** function by passing the list to the function.
4. As each card number is decoded, write the information to an output file with **WriteFile()**. The WriteFile() function should be called from within the Decode() function.



```
Reading card numbers from file...
Number of lines read: 21

Analysing card numbers...
See the output file for the extracted information
```

Figuur 3.1 / Figure 3.1



```
1 0719220408435222: Was issued by Petroleum in 2019. The card expires on 22/04. The card is linked to a Debit account with account number: 08435222.
2 0718230711258589: Was issued by Petroleum in 2018. The card expires on 23/07. The card is linked to a Credit account with account number: 11258589.
3 0620250511313424: Was issued by UnionPay in 2020. The card expires on 25/05. The card is linked to a Credit account with account number: 11313424.
4 0420240506413286: Was issued by Visa in 2020. The card expires on 24/05. The card is linked to a Debit account with account number: 06413286.
5 0519230611167176: Was issued by Master Card in 2019. The card expires on 23/06. The card is linked to a Credit account with account number: 11167176.
6 0318241215528881: Was issued by Diner's club in 2018. The card expires on 24/12. The card is linked to a Credit account with account number: 15528881.
7 0818230811552631: Was issued by TeleCom in 2018. The card expires on 23/08. The card is linked to a Credit account with account number: 11552631.
8 0921221012763379: Was issued by National in 2021. The card expires on 22/10. The card is linked to a Credit account with account number: 12763379.
9 0718200813327745: Was issued by Petroleum in 2018. The card expires on 20/08. The card is linked to a Credit account with account number: 13327745.
10 0721180104628760: Was issued by Petroleum in 2021. The card expires on 18/01. The card is linked to a Debit account with account number: 04628760.
11 0619201212994339: Was issued by UnionPay in 2019. The card expires on 20/12. The card is linked to a Credit account with account number: 12994339.
12 0718250408180092: Was issued by Petroleum in 2018. The card expires on 25/04. The card is linked to a Debit account with account number: 08180092.
13 0119230815202304: Was issued by enRoute in 2019. The card expires on 23/08. The card is linked to a Credit account with account number: 15202304.
14 0219220613614728: Was issued by JCB in 2019. The card expires on 22/06. The card is linked to a Credit account with account number: 13614728.
15 0219201117330532: Was issued by JCB in 2019. The card expires on 20/11. The card is linked to a Credit account with account number: 17330532.
16 0119220912817656: Was issued by enRoute in 2019. The card expires on 22/09. The card is linked to a Credit account with account number: 12817656.
17 0620251211348539: Was issued by UnionPay in 2020. The card expires on 25/12. The card is linked to a Credit account with account number: 11348539.
18 0319210503753205: Was issued by Diner's club in 2019. The card expires on 21/05. The card is linked to a Debit account with account number: 03753205.
19 0419220800608355: Was issued by Visa in 2019. The card expires on 22/08. The card is linked to a Debit account with account number: 00608355.
20 0618200812056717: Was issued by UnionPay in 2018. The card expires on 20/08. The card is linked to a Credit account with account number: 12056717.
21 0320211218017000: Was issued by Diner's club in 2020. The card expires on 21/12. The card is linked to a Credit account with account number: 18017000.
22
```

Figuur 3.2 / Figure 3.2

| | |
|----|--------------|
| 01 | enRoute |
| 02 | JCB |
| 03 | Diner's club |
| 04 | Visa |
| 05 | Master Card |
| 06 | UnionPay |
| 07 | Petroleum |
| 08 | TeleCom |
| 09 | National |

Tabel 3.1 / Table 3.1 – Data.txt

```

0719220408435222
0718230711258589
0620250511313424
0420240506413286
0519230611167176
0318241215528881
0818230811552631
0921221012763379
0718200813327745
0721180104628760
0619201212994339
0718250408180092
0119230815202304
0219220613614728
0219201117330532
0119220912817656
0620251211348539
0319210503753205
0419220800680355
0618200812056717
0320211218017000

```

Tabel 3.2 / Table 3.2 – Data.txt

EINDE | END

TOTAAL/TOTAL: 100