SENIOR CERTIFICATE EXAMINATION
SENIORSERTIFIKAAT-EKSAMEN

FEBRUARY / MARCH
FEBRUARIE / MAART

2007

COMPUTER STUDIES
REKENAARSTUDIE

HG

724-1/1

First Paper : Practical
Eerste Vraestel: Praktiese

COMPUTER STUDIES/REKENAARSTUDIE HG : P1/V1
Practical/Praktiese

724 1 1 HG

18 Pages

18 b*ladsye*

X05

GAUTENGSE DEPARTEMENT VAN ONDERWYS

SENIORSERTIFIKAAT-EKSAMEN

REKENAARSTUDIE HG
(Eerste Vraestel: Prakties)

TYD: 3 uur

PUNTE: 100

INSTRUKSIES:

- Hierdie vraestel bestaan uit TWEE afdelings. Beantwoord slegs die afdeling wat op jou programmeertaal van toepassing is.
- Indien jy **Delphi** behandel het, moet jy **Afdeling A** beantwoord. Indien jy **Pascal** behandel het, moet jy **Afdeling B** beantwoord.
- Geen komponente mag bygevoeg of weggelaat word op die gegewe Delphi-vorms nie, tensy anders vermeld.
- Swak programmeringstegnieke sal gepenaliseer word.
- Die hulpfunksie (F1) van jou programmeertaal mag enige tyd gedurende hierdie eksamen geraadpleeg word.
- Stoor jou werk met gereelde tussenposes.
- Jou volledige eksamennommer moet op elke bladsy wat ingehandig word, verskyn.

AFDELING A
DELPHI
VRAAG 1

'n Klerewinkel wil hê dat jy vir hulle 'n program moet skryf wat aan die einde van elke maand 'n tekslêer sal skep wat alle kliënte met uitstaande saldo's sal bevat.

TWEE datalêers genaamd **DCresta.dat** en **DMenlyn.dat**, is reeds op jou eksamendisket gestoor en bevat die inligting van die kliënte van TWEE takke van die klerewinkel.

Elke rekord in beide die bogenoemde datalêers bevat die volgende velde:

```
Titel  -> 3 karakters
Voorl  -> 4 karakters
Van    -> 30 karakters
Saldo  -> reële waarde
```

Maak die lêer '`Vr1P.dpr`' in Delphi oop, gaan na File|Save As... en stoor die *unit* as '`Vr1U_XXXX.pas`' (`XXXX` stel die laaste vier syfers van jou eksamennommer voor). Gaan nou na File|Save Project As... en stoor die *projek* as '`Vr1P_XXXX.dpr`'.

GAUTENG DEPARTMENT OF EDUCATION
SENIOR CERTIFICATE EXAMINATION

COMPUTER STUDIES HG
(First Paper: Practical)

TIME : 3 hours

MARKS : 100

INSTRUCTIONS:

- This paper consists of TWO sections. Only answer the section applicable to your programming language.
- If you studied **Delphi**, you must answer **Section A**. If you studied **Pascal**, you must answer **Section B**.
- No components may be added to or removed from the given Delphi forms, unless otherwise stated.
- Poor programming techniques will be penalised.
- The help function (F1) of your programming language may be consulted anytime during this examination.
- Save your work at regular intervals.
- Your complete examination number must appear on every page that you hand in.

SECTION A
DELPHI
QUESTION 1

A clothing store wants you to write them a program that will, at the end of each month, create a text file containing all clients who have outstanding balances.

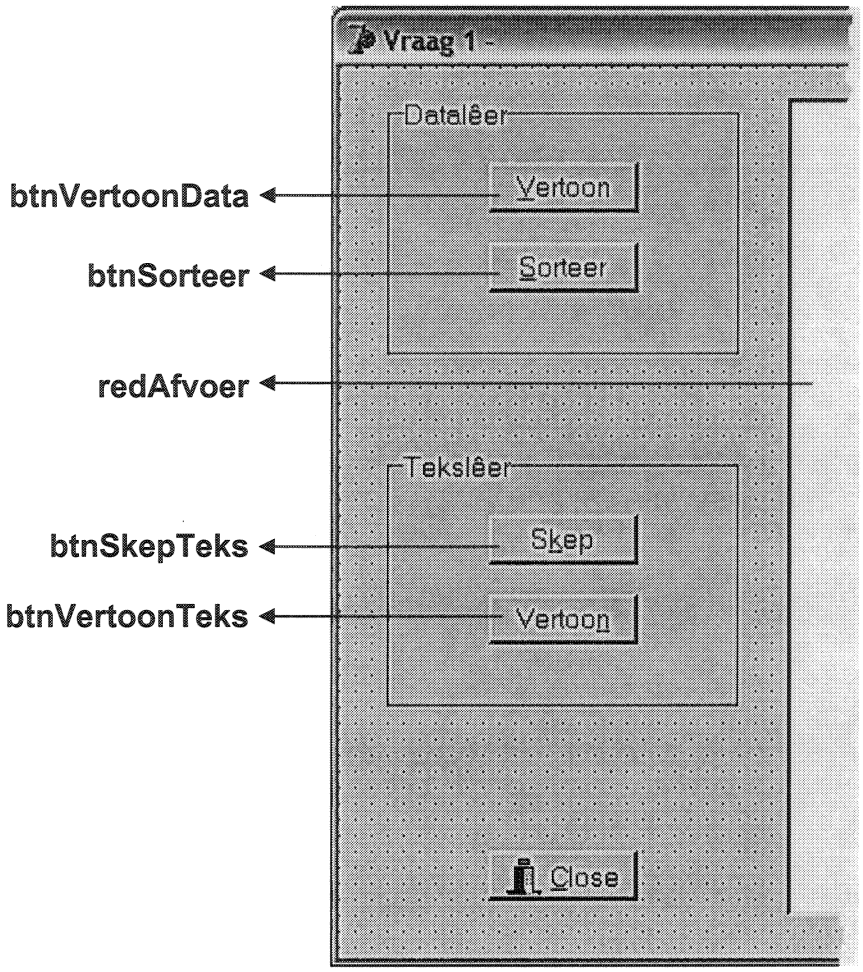
TWO data files named **DCresta.dat** and **DMenlyn.dat,** are already stored on your examination disk and contain the information of the clients of TWO branches of the clothing shop.

Each record in both the above-mentioned files contains the following fields:

```
Title     -> 3 characters
Initials  -> 4 characters
Surname   -> 30 characters
Balance   -> real value
```

Open the file '`Q1P.dpr`' in Delphi, go to File|Save As... and save the *unit* as '`Q1U_XXXX.pas`' (`XXXX` represents the last four digits of your examination number). Now go to File|Save Project As... and save the *project* as '`Q1P_XXXX.dpr`'.

1.1 Verander **slegs** die *Caption*- en *Name*-eienskappe van die onderskeie komponente op die vorm sodat dit ooreenstem met die onderstaande figuur. Voeg ook jou eksamennommer by die *Caption* van die vorm langs "Vraag 1" by. (2)

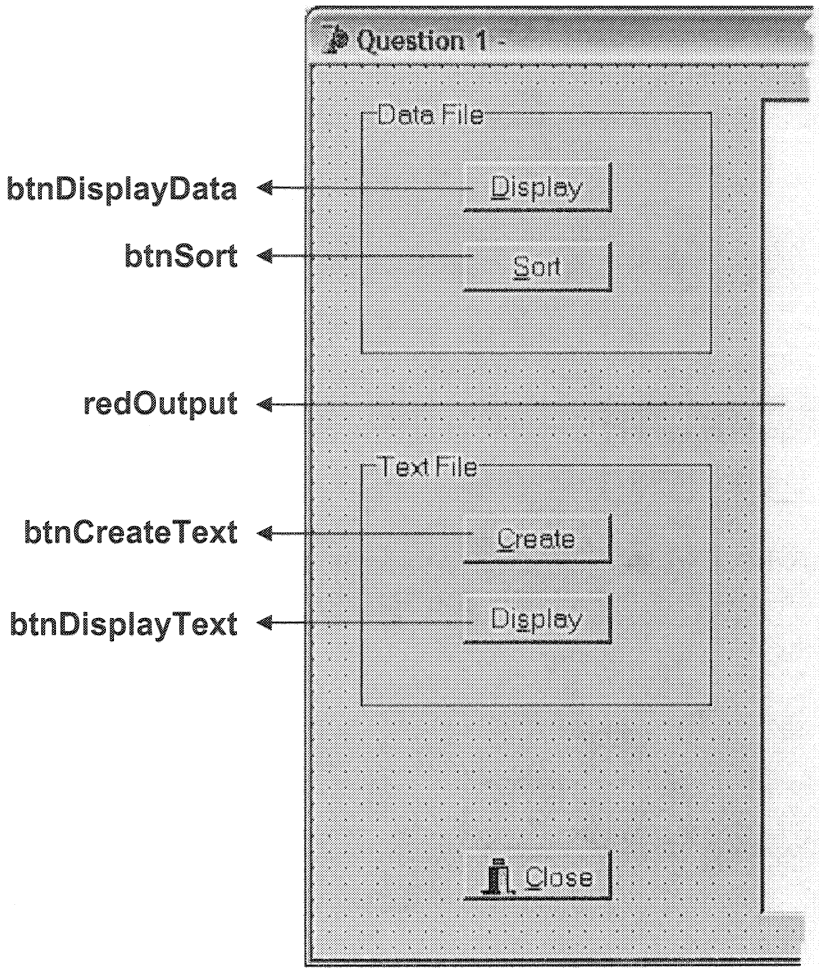


1.2 VERTOON INHOUD VAN DATALÊER OP RICHEDIT

Skryf 'n gebeurtenishanteerder (*Event Handler*) vir **btnVertoonData** wat aan die volgende sal voldoen:

- Wanneer die gebruiker op **btnVertoonData** klik, moet die inhoud van **DCresta.dat** in kolomme in **redAfvoer** vertoon word.
- Elke kolom moet oor 'n gepaste opskrif beskik.
- Alle saldo's moet na regs gespasieer wees en moet 'n geldeenheid simbool vooraan die bedrag vertoon, byvoorbeeld **R**. (13)

1.1 Change **only** the *Caption* and *Name* properties of the different components on the form so that each corresponds with the figure below. Also add your examination number to the *Caption* of the form next to "Question 1". (2)

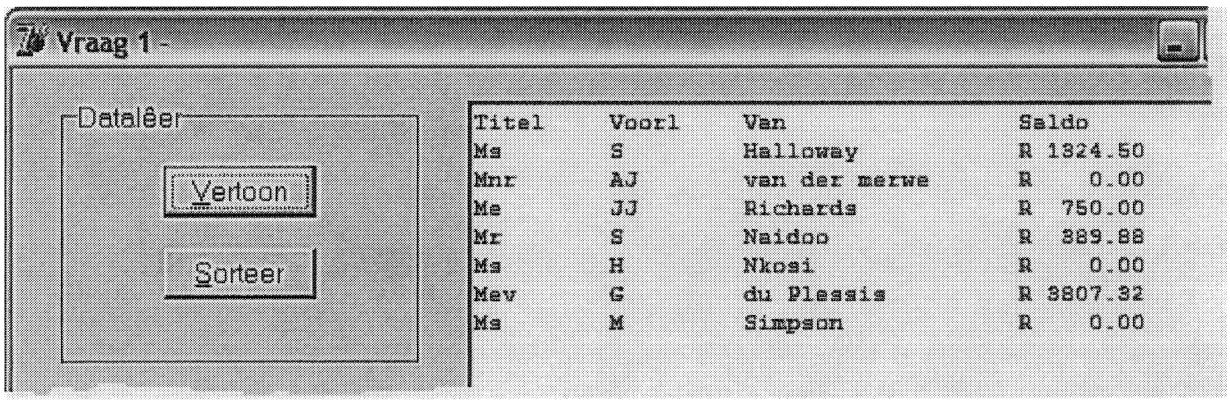


1.2 **DISPLAY CONTENTS OF DATA FILE ON RICHEDIT**

Write an e*vent handler* for **btnDisplayData** that will adhere to the following:

- Whenever the user clicks **btnDisplayData**, the contents of **DCresta.dat** must be displayed on **redOutput** in a columnar fashion.
- Each column must have an appropriate heading.
- All balances must be justified to the right and must start with a currency symbol, e.g. **R**. (13)

Nadat die gebruiker op **btnVertoonData** geklik het, behoort die vorm soos volg te vertoon:

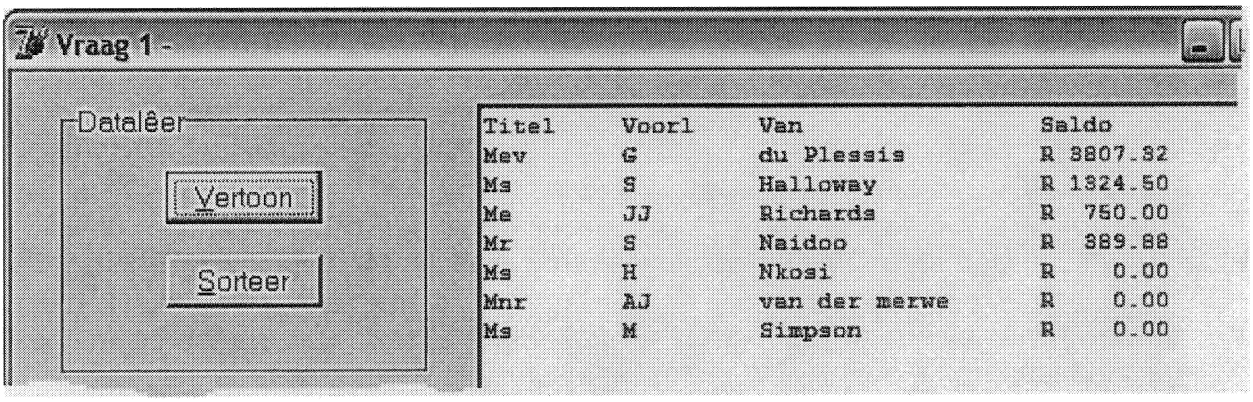


1.3 SORTEER INHOUD VAN DATALÊER

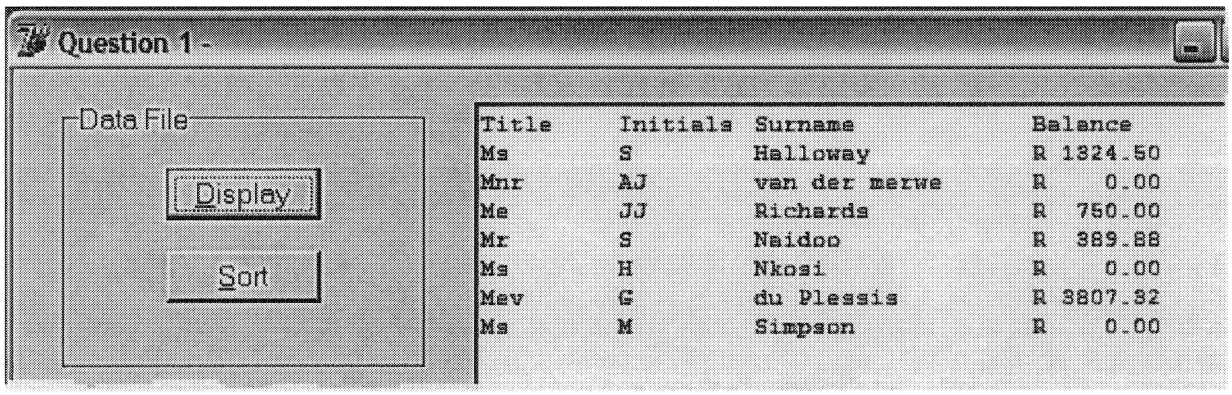
Skryf 'n gebeurtenishanteerder (*Event Handler*) vir **btnSorteer** wat aan die volgende sal voldoen: (Maak eers 'n rugsteunkopie van **DCresta.dat)**.

- Wanneer die gebruiker op **btnSorteer** klik, moet die inhoud van **DCresta.dat** in dalende orde volgens die saldo's gesorteer word. Neem aan daar sal nooit meer as 20 kliënte in die lêer gestoor wees nie.
- Nadat die inligting gesorteer is, moet dit oor die vorige inligting in **DCresta.dat** geskryf word. (16)

Nadat **DCresta.dat** gesorteer is en die gebruiker dan weer op **btnVertoonData** klik, behoort die afvoer soos volg te vertoon:



After the user has clicked on **btnDisplayData**, the form should display as follows:

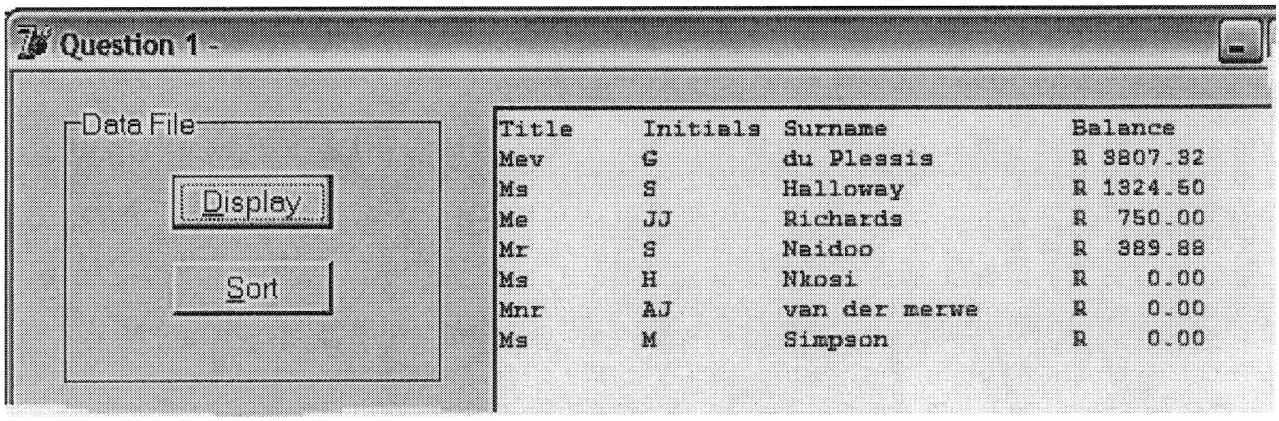


1.3 SORT CONTENTS OF DATA FILE

Write an e*vent handler* for **btnSort** that will comply with the following: (First make a backup copy of **DCresta.dat.)**

- When the user clicks **btnSort**, the contents of **DCresta.dat** must be sorted in descending order according to the balances. Assume that there will never be more than 20 clients stored within the file.
- After the information has been sorted, it must be written over the previous information in **DCresta.dat**. (16)

After **DCresta.dat** has been sorted and the user again clicks **btnDisplayData**, the output should display as follows:



1.4 SKEP VAN TEKSLÊER

Skryf 'n gebeurtenishanteerder (*Event Handler*) vir **btnSkepTeks** wat aan die volgende sal voldoen:

- Wanneer die gebruiker op **btnSkepTeks** klik, moet die lêer **DMenlyn.dat** gebruik word om 'n tekslêer, genaamd **skuld.txt,** te skep. Die tekslêer moet alle kliënte in **DMenlyn.dat** bevat wat uitstaande saldo's het (saldo's groter as nul).
- Elke reël in die tekslêer moet die kliënt se titel, voorletters, van en saldo bevat (elk geskei deur kommas).

Een reël in die tekslêer kan dus as volg lyk:

```
Me,JJ,Richards,750.00
.
.
```

- Nadat al die nodige kliënte na die tekslêer geskryf is, moet 'n boodskap soortgelyk aan die een hieronder verskyn, wat sal aandui hoeveel kliënte na die tekslêer geskryf is: (15)

1.5 VERTOON INHOUD VAN TEKSLÊER OP RICHEDIT

Skryf 'n gebeurtenishanteerder (*Event Handler*) vir **btnVertoonTeks** wat aan die volgende sal voldoen:

- Wanneer die gebruiker op **btnVertoonTeks** klik, moet die inhoud van **skuld.txt** op **redAfvoer** verskyn.
- Indien die lêer nie bestaan nie, moet die program 'n geskikte boodskap vertoon en nie probeer om die inhoud van die tekslêer te vertoon nie. (7)

1.4 CREATE TEXT FILE

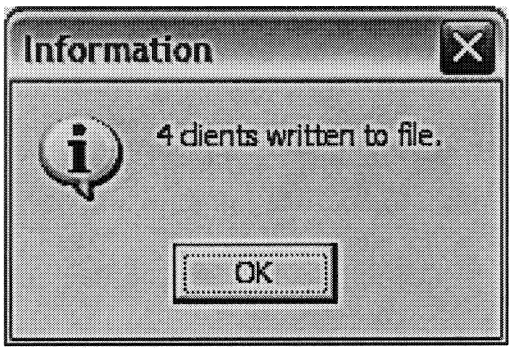
Write an e*vent handler* for **btnCreateText** that will comply with the following:

- When the user clicks **btnCreateText**, the file **DMenlyn.dat** must be used to create a text file named **debt.txt**. The text file must contain all clients from **DMenlyn.dat** with outstanding balances (balances greater than zero).
- Each line within the text file must consist of the client's title, initials, surname and balance (each separated by commas).

One line from the text file may look like this:

```
Me,JJ,Richards,750.00
.
.
```

- After all necessary clients have been written to the text file, a message similar to the one below, must be displayed, indicating the number of clients that have been written to the text file: (15)

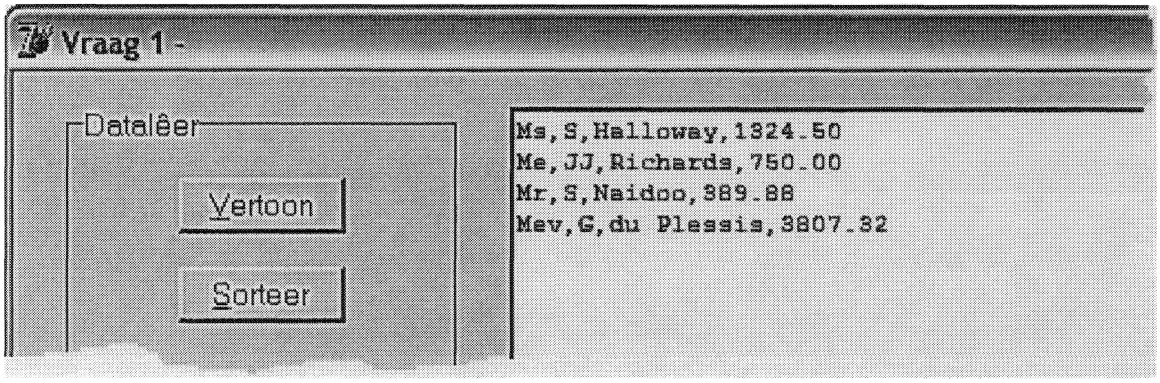


1.5 DISPLAY CONTENTS OF TEXT FILE ON RICHEDIT

Write an e*vent handler* for **btnDisplayText** that will comply with the following:

- When the user clicks **btnDisplayText**, the contents of **debt.txt** must be displayed on **redOutput**.
- If the file does not exist, an appropriate message must be displayed and the program must not attempt to display the contents of the file. (7)

Nadat die gebruiker op **btnVertoonTeks** geklik het, behoort die afvoer soos volg te vertoon:



Die volgende moet ingehandig word vir Vraag 1 (Delphi):
- 'n Drukstuk van `Vr1U_XXXX.pas`.
- 'n "Alt | Print Scrn" van die vorm *terwyl die afvoer van **Vraag 1.2** vertoon word.* **[53]**

VRAAG 2

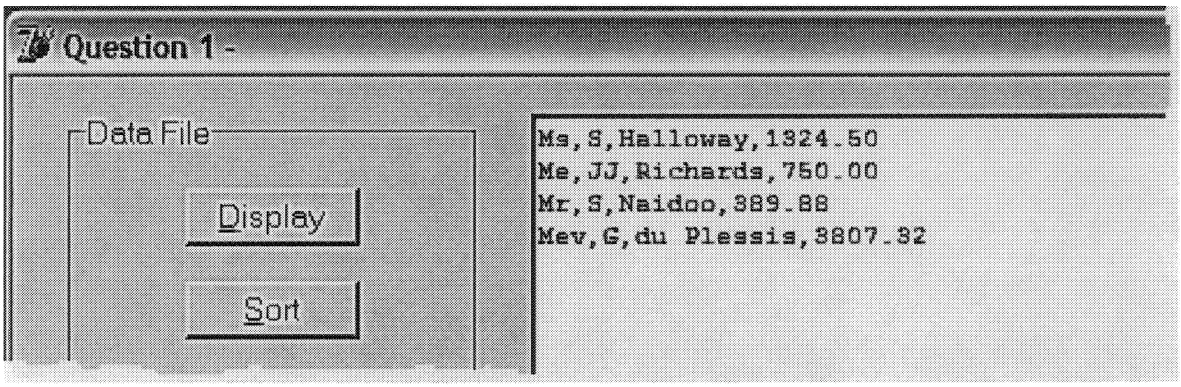
Om 'n geldige wagwoord vir die installering van 'n sekere program te genereer, word die volgende algoritme gebruik:

- Vra die persoon se naam **in kleinletters**. (bv. john g smith)
- Skakel die naam om na titelkas. (john g smith → John G Smith)
  (Titelkas beteken dat die eerste letter van elke woord 'n hoofletter is.)
- Las die naam aanhoudend aan homself totdat dit 'n string van meer as 19 karakters vorm. ('John G SmithJohn G Smith')
- Kry die ASCII-waardes van die 3^e, 13^e en 19^e karakters in bostaande string.
  (h=104 ; J=74 ; spasie=32)
- Skakel bostaande ASCII-waardes om na stringe en las dit dan aanmekaar om EEN string te vorm. ('1047432')
- Gaan bostaande string karakter vir karakter deur en kry telkens die ooreenstemmende letter van die alfabet. Ignoreer alle nulle.
  (1='A' ; 4='D' ; 7='G' ; 4='D' ; 3='C' ; 2='B') **L.W. 'A' se ASCII-waarde is 65.**
- Die wagwoord vir *john g smith* is dus 'ADGDCB'.

Nog 'n voorbeeld:

- mia cox
- Mia Cox
- Mia CoxMia CoxMia Cox
- a=97 ; o=111 ; C=67 → '9711167'
- 'IGAAAFG'

After the user has clicked **btnDisplayText**, the output should appear as follows:



The following must be handed in for Question 1 (Delphi):

- A printout of **`Q1U_XXXX.pas`**.
- An "Alt | Print Scrn" of the form *while the output of* ***Question 1.2*** *is being displayed.*

[53]

QUESTION 2

To generate a valid password for the installation of a certain program, the following algorithm is used:

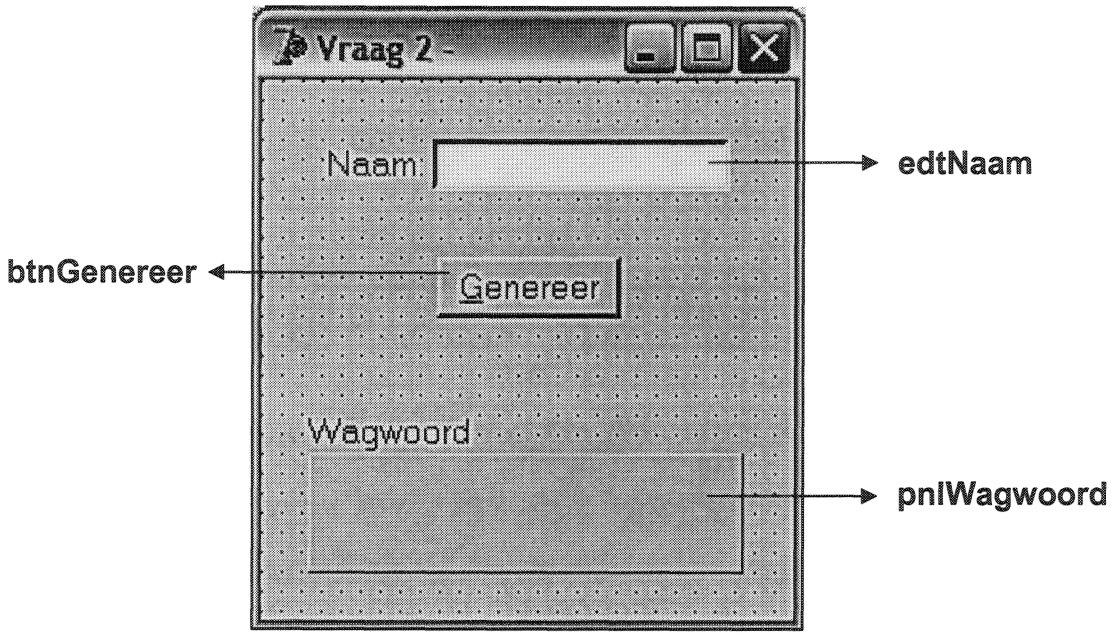
- Ask for the person's name **in lower case**. (e.g. john g smith)
- Convert the name to title case. (john g smith → John G Smith)
  (Title case means that each word starts with a capital letter)
- Repeatedly add the name to itself until it forms a string with more than 19 characters.
  ('John G SmithJohn G Smith')
- Find the ASCII-values of the 3rd, 13th and 19th characters in the above-mentioned string. (h=104 ; J=74 ; space=32)
- Convert the ASCII-values mentioned above to strings and then add them together to form ONE string. ('1047432')
- Go through the string mentioned above, one character at a time, and in each case find the corresponding letter of the alphabet. Ignore all zeros.
  (1='A' ; 4='D' ; 7='G' ; 4='D' ; 3='C' ; 2='B') **NB. The ASCII value of 'A' is 65.**
- Consequently the password for john g smith is 'ADGDCB'.

Another example:

- mia cox
- Mia Cox
- Mia CoxMia CoxMia Cox
- a=97 ; o=111 ; C=67 → '9711167'
- 'IGAAAFG'

Begin 'n nuwe toepassing in Delphi en stoor die *unit* en *project* onderskeidelik as **`Vraag2U_XXXX.pas`** en **`Vraag2P_XXXX.dpr`** (**`XXXX`** stel die laaste vier syfers van jou eksamennommer voor).

2.1 Ontwerp nou 'n vorm soos die onderstaande een en voeg jou eksamennommer langs "Vraag 2" by die *Caption* van die vorm by. (1)



2.2 Skryf 'n funksiesubprogram genaamd **TitelKas**, wat 'n string sal ontvang (in kleinletters) en dit na titelkas sal omskakel.

Voorbeeld: As '*mia cox*' na die funksie gestuur word, moet '*Mia Cox*' teruggestuur word. (8)

2.3 Skryf 'n prosedure, **KryWagwoord**, wat 'n persoon se naam sal ontvang (in titelkas) en 'n wagwoord vir daardie naam sal genereer volgens genoemde algoritme op die vorige bladsy. Slegs die wagwoord moet teruggestuur word na die roepstelling.

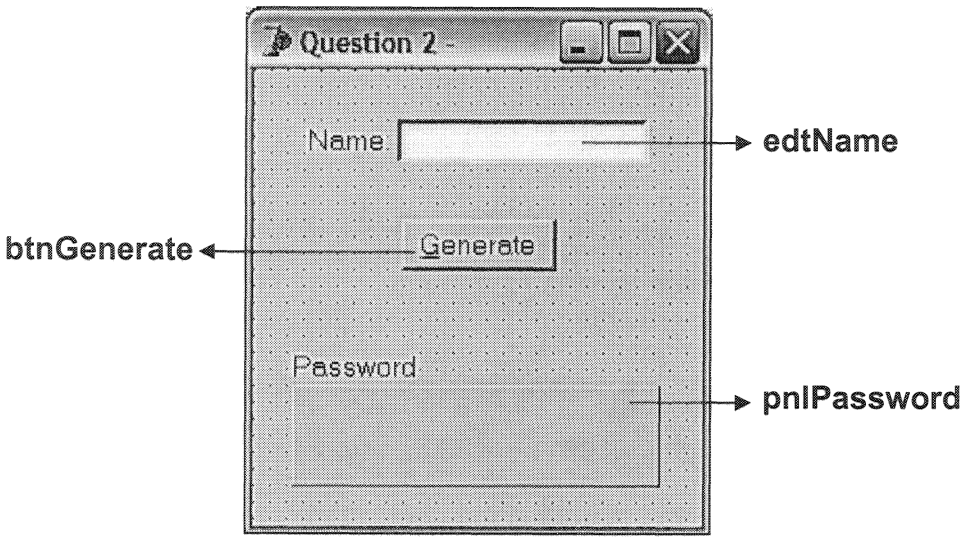
Voorbeeld: As '*Mia Cox*' na die prosedure gestuur word, moet '*IGAAAFG*' teruggestuur word. (16)

2.4 Skryf die *OnClick Event Handler* vir **btnGenereer** wat die TWEE subprogramme in Vraag 2.2 en 2.3 sal roep om 'n wagwoord te genereer vir die naam wat in **edtNaam** ingetik is. Die gegenereerde wagwoord moet op **pnlWagwoord** vertoon word. (3)

Die volgende moet ingehandig word vir Vraag 2 (Delphi):
- 'n Drukstuk van **`Vraag2U_XXXX.pas`**.
- 'n "Alt | Print Scrn" van die vorm *terwyl mia cox se wagwoord vertoon word.* **[28]**

Start a new application in Delphi and save the *unit* as **`Question2U_XXXX.pas`** and the *project* as **`Question2P_XXXX.dpr`** (**`XXXX`** represents the last four digits of your examination number).

2.1 Design a form like the one below and add your examination number next to "Question 2", to the *Caption* of the form. (1)



2.2 Write a function subprogram named **TitleCase**, that will receive a string (in lower case) and then convert it to title case.

Example: If '*mia cox*' was sent to the function, then '*Mia Cox*' must be returned. (8)

2.3 Write a procedure, **GetPassword**, that will receive a person's name (in title case) and then generate a password for that name, according to the algorithm mentioned on the previous page. Only the password must be returned to the calling statement.

Example: If '*Mia Cox*' was sent to the procedure, then '*IGAAAFG*' must be returned. (16)

2.4 Write the *OnClick Event Handler* for **btnGenerate** that will call the TWO subprograms from Question 2.2 and 2.3 to generate a password for the name typed into **edtName.** The generated password must be displayed on **pnlPassword**. (3)

The following must be handed in for Question 2 (Delphi):

- A printout of **`Question2U_XXXX.pas`**.
- An "Alt | Print Scrn" of the form *while the password of mia cox is displayed.*

[28]

VRAAG 3

Die speletjie ***Battleships*** maak gebruik van 'n matriks van rye en kolomme waarop 'n aantal skepe weggesteek word. Die speler se opponent moet dan probeer om al die skepe raak te skiet.

Elke skip het 'n sekere lengte en kan horisontaal of vertikaal op die matriks geplaas word. 'n Skip waarvan die lengte VIER is, sal VIER plekke in die matriks beset.

LET WEL: In hierdie program gaan die skepe slegs horisontaal geplaas word.

'n Voorbeeld waar DRIE skepe horisontaal op die matriks geplaas is, kan soos volg lyk:

Die eerste skip lê in **ry 1; kolomme 6 tot 9**. (Lengte = 4)

Die tweede skip lê in **ry 2; kolomme 12 tot 15**. (Lengte = 4)

Die derde skip lê in **ry 7; kolomme 4 tot 7**. (Lengte = 4)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1						#	#	#	#						
2												#	#	#	#
3															
4															
5															
6															
7				#	#	#	#								
8															
9															
10															

Maak die lêer **'Vr3P.dpr'** in Delphi oop, gaan na File|Save As... en stoor die *unit* as **'XXXX_Vr3U.pas'** (**XXXX** stel die laaste vier syfers van jou eksamennommer voor). Gaan nou na File|Save Project As... en stoor die *projek* as **'XXXX_Vr3P.dpr'**.

QUESTION 3

The game of ***Battleships*** uses a matrix of rows and columns on which a number of ships are hidden. The player's opponent must then try to hit all the ships.

Every ship has a certain length and can be placed either horizontally or vertically on the matrix. A ship of length FOUR will occupy FOUR places in the matrix.

PLEASE NOTE: In this program the ships will only be placed horizontally.

An example of where THREE ships are placed horizontally on the matrix, may look as follows:

The first ship occupies **row 1; columns 6 to 9**. (Length = 4)

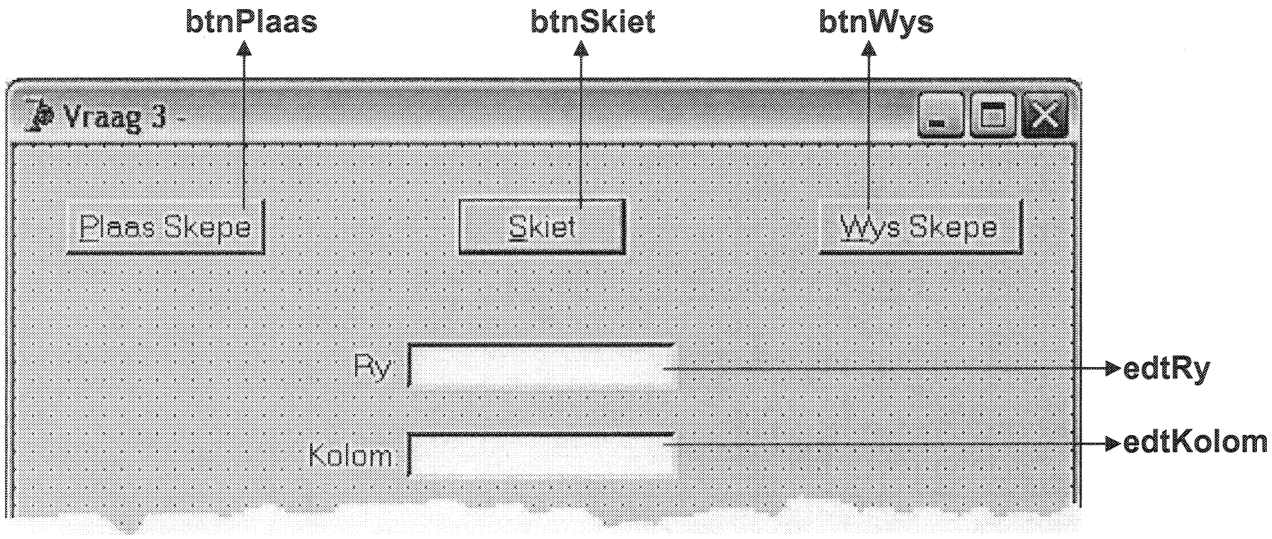
The second ship occupies **row 2; columns 12 to 15**. (Length = 4)

The third ship occupies **row 7; columns 4 to 7**. (Length = 4)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1						#	#	#	#						
2												#	#	#	#
3															
4															
5															
6															
7				#	#	#	#								
8															
9															
10															

Open the file '`Q3P.dpr`' in Delphi, go to File|Save As... and save the *unit* as '`XXXX_Q3U.pas`' (`XXXX` represents the last four digits of your examination number). Now go to File|Save Project As... and save the *project* as '`XXXX_Q3P.dpr`'.

3.1 Verander **slegs** die *Caption*- en *Name*-eienskappe van die onderskeie komponente op die vorm sodat elkeen met die onderstaande figuur ooreenstem. Voeg ook jou eksamennommer langs die *Caption* van die vorm "Vraag 3" by.

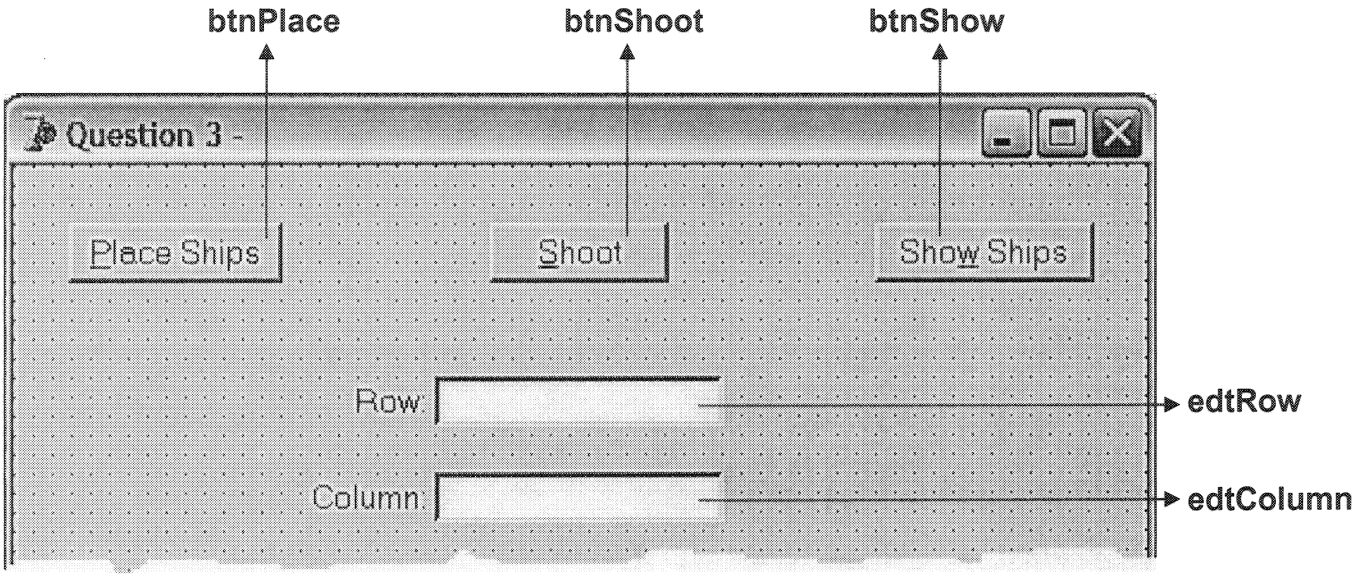


Skryf die toepaslike kode vir elkeen van die volgende gebeurtenishanteerders (*Event Handlers*):

3.2 Wanneer die speler op **btnPlaas** klik, moet die program die volgende doen:

- Skep 'n matriks met TIEN rye (1 tot 10) en VYFTIEN kolomme (1 tot 15).
- DRIE skepe, elkeen met 'n lengte van VIER, moet ***horisontaal*** in 'n matriks geplaas word.
- Die beginposisies van die skepe moet ewekansig (random) gekies word.
- Waar die skepe op die matriks lê, moet 'n waarde van *TRUE* gestoor word en op al die ander plekke, 'n waarde van *FALSE* gestoor word.
- Geen gedeelte van 'n skip mag die matriks verbysteek nie.
- Die skepe mag oorvleuel.
- Eers nadat die skepe geplaas is, moet **btnSkiet** en **btnWys** geaktiveer word. (10)

3.1 Change **only** the *Caption* and *Name* properties of the different components on the form so that each corresponds with the figure below. Also add your examination number next to the *Caption* of the form next to "Question 3".

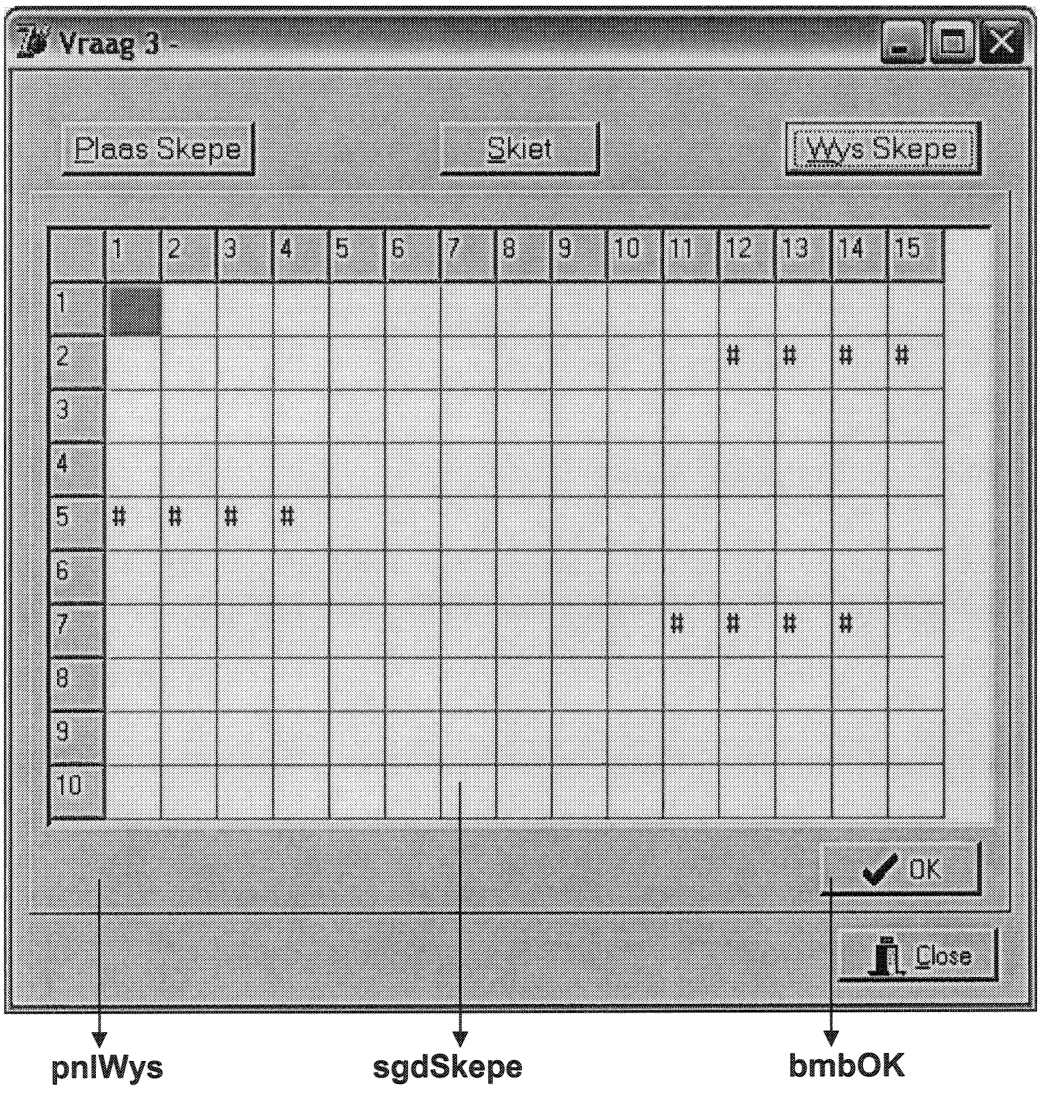


Write the appropriate code for each of the following *Event Handlers*:

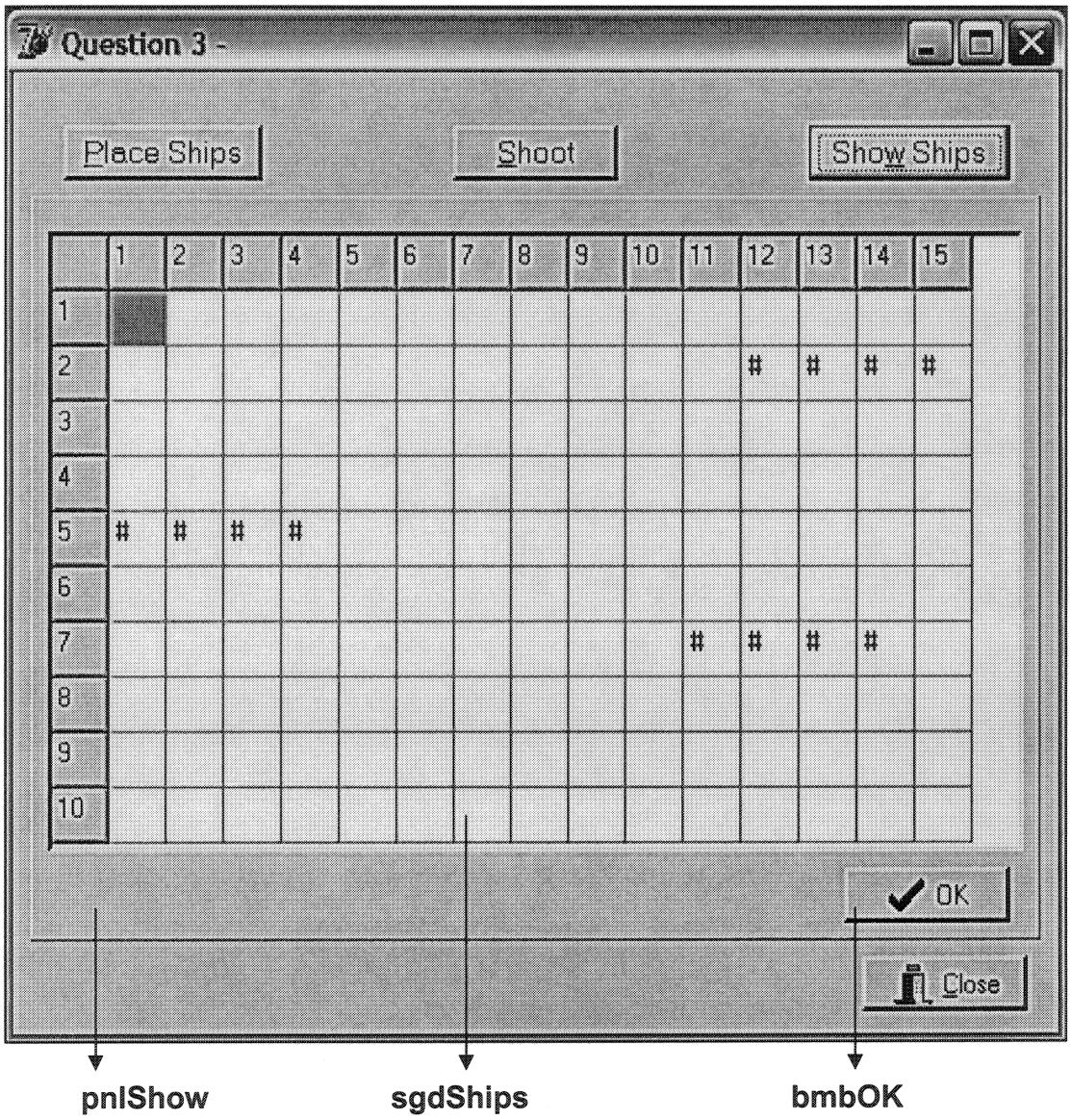
3.2 Whenever the player clicks on **btnPlace**, the program must do the following:

- Create a matrix that consists of TEN rows (1 to 10) and FIFTEEN columns (1 to 15).
- THREE ships, each with a length of FOUR, must be placed ***horizontally*** in the matrix.
- The starting positions of the ships must be chosen randomly.
- Where a ship occupies the matrix, a value of TRUE must be stored and in all other places a value of FALSE must be stored.
- No part of a ship may go outside the matrix.
- The ships may overlap.
- Only after the ships have been placed, must **btnShoot** and **btnShow** be activated. (10)

3.3 Wanneer die gebruiker op **btnWys** klik, moet 'n *Panel* verskyn met 'n *StringGrid* en 'n *OK-Button* daarop (voeg self by op die vorm). Die skepe wat geplaas is, moet dan vertoon word. Sodra die gebruiker op **bmbOK** klik, moet die *Panel* weer verdwyn: (6)



3.3 Whenever the user clicks **btnShow**, a *Panel* must appear with a *StringGrid* and an *OK-Button* on (add it to the form yourself). The ships that have been placed must then be shown. As soon as the user clicks on **bmbOK**, the panel must disappear again. (6)



3.4 Wanneer die speler op **btnSkiet** klik, moet die volgende gebeur:

- Die waardes in **edtRy** en **edtKolom** moet gebruik word om te bepaal of die speler 'n skip raakgeskiet het of nie.
- 'n Toepaslike boodskap moet verskyn om aan te dui of die speler 'n skip mis of raak geskiet het.
- Die program hoef **NIE** die aantal raakskote te tel of die inhoud van die matriks te verander nie. (3)

Voorbeeld:

Indien 'n skip in **ry 2; kolomme 1 tot 4** lê en die speler skiet op **ry 2; kolom 2**, moet die volgende boodskap verskyn:

Die volgende moet ingehandig word vir Vraag 3 (Delphi):
- 'n Drukstuk van **`XXXX_Vr3U.pas`**. **[19]**

TOTAAL VIR AFDELING A: [100]

3.4 Whenever the player clicks **btnShoot**, the following must happen:

- The values in **edtRow** and **edtColumn** must be used to determine whether the player has hit or missed a ship.
- An appropriate message must be displayed to indicate whether the player has hit or missed a ship.
- The program does **NOT** have to count the number of hits or change the content of the matrix. (3)

Example:

If a ship occupies **row 2; columns 1 to 4** and the player fires at **row 2; column 2**, the following message must appear:

The following must be handed in for Question 3 (Delphi):

- A printout of **`XXXX_Q3U.pas`**.

[19]

TOTAL FOR SECTION A: [100]

AFDELING B
PASCAL
VRAAG 4

'n Klerewinkel wil hê dat jy vir hulle 'n program moet skryf wat aan die einde van elke maand 'n tekslêer sal skep wat alle kliënte met uitstaande saldo's sal bevat.

TWEE datalêers, genaamd **PCresta.dat** en **PMenlyn.dat**, is reeds op jou eksamendisket gestoor en bevat die inligting van die kliënte van TWEE takke van die klerewinkel.

Elke rekord in beide die bogenoemde datalêers bevat die volgende velde:

```
Titel  -> 3 karakters
Voorl  -> 4 karakters
Van    -> 30 karakters
Saldo  -> reële waarde
```

Begin met 'n nuwe program in Turbo Pascal en stoor dit as '**Vr4_XXXX.pas**' (**XXXX** stel die laaste vier syfers van jou eksamennommer voor).

4.1 Die program moet die volgende opsielys *herhaaldelik* aan die gebruiker vertoon totdat **Opsie 5** gekies word: (2)

```
1. Vertoon datalêer.
2. Sorteer datalêer.
3. Skep tekslêer.
4. Vertoon tekslêer.
5. Eindig.
```

Skryf telkens by elk van die volgende vrae 'n prosedure **sonder enige parameters**.

SECTION B
PASCAL
QUESTION 4

A clothing store wants you to write them a program that will, at the end of each month, create a text file containing all clients who have outstanding balances.

TWO data files named **PCresta.dat** and **PMenlyn.dat,** are already stored on your examination disk and contain the information of the clients of TWO branches of the clothing store.

Each record in both the above-mentioned files contains the following fields:

```
Title     -> 3 characters
Initials  -> 4 characters
Surname   -> 30 characters
Balance   -> real value
```

Start with a new program in Turbo Pascal and save it as '**Q4_XXXX.pas**' (where **XXXX** represents the last FOUR digits of your examination number).

4.1 The program must *continuously* display the following menu until the user chooses **Option 5**. (2)

```
1. Display data file.
2. Sort data file.
3. Create text file.
4. Display text file.
5. End.
```

For each of the following questions, write a procedure **without using parameters**.

4.2 VERTOON INHOUD VAN DATALÊER

Skryf 'n prosedure, **VertoonData**, vir **Opsie 1** wat die volgende sal doen:

- Wanneer die gebruiker **Opsie 1** kies, moet die inhoud van **PCresta.dat** in kolomme op die skerm vertoon word.
- Elke kolom moet 'n gepaste opskrif hê.
- Alle saldo's moet na regs gespasieer wees en moet 'n geldeenheid simbool vooraan die bedrag vertoon, byvoorbeeld **R**. (13)

Nadat die gebruiker **Opsie 1** gekies het, behoort die afvoer soos volg te vertoon:

```
Titel  Voorl  Van                Saldo
Ms     S      Halloway           R 1324.50
Mnr    AJ     van der Merwe      R    0.00
Me     JJ     Richards           R  750.00
Mr     S      Naidoo             R  389.88
Ms     H      Nkosi              R    0.00
Mev    G      du Plessis         R 3807.32
Ms     M      Simpson            R    0.00
```

4.3 SORTEER INHOUD VAN DATALÊER

Skryf 'n prosedure, **Sorteer**, vir **Opsie 2** wat die volgende sal doen: (Maak eers 'n rugsteunkopie van **PCresta.dat**).

- Wanneer die gebruiker **Opsie 2** kies, moet die inhoud van **PCresta.dat** in dalende orde gesorteer word volgens die saldo's. Neem aan dan daar nooit meer as 20 kliënte in die lêer gestoor sal wees nie.
- Nadat die inligting gesorteer is, moet dit oor die vorige inligting in **PCresta.dat** geskryf word. (16)

Nadat **PCresta.dat** gesorteer is en die gebruiker dan weer **Opsie 1** kies, behoort die afvoer soos volg te vertoon:

```
Titel  Voorl  Van            Saldo
Mev    G      du Plessis     R 3807.32
Ms     S      Halloway       R 1324.50
Me     JJ     Richards       R  750.00
Mr     S      Naidoo         R  389.88
Ms     H      Nkosi          R    0.00
Mnr    AJ     van der Merwe  R    0.00
Ms     M      Simpson        R    0.00
```

4.2 DISPLAY CONTENTS OF DATA FILE

For **Option 1**, write a procedure **DisplayData**, that will do the following:

- Whenever the user chooses **Option 1**, the contents of **PCresta.dat** must be displayed on screen in a columnar fashion.
- Each column must have an appropriate heading.
- All balances must be justified to the right and must start with a currency symbol, e.g. **R**. (13)

After the user has chosen **Option 1**, the output should display as follows:

```
Title  Initials  Surname          Balance
Ms     S         Halloway         R 1324.50
Mnr    AJ        van der Merwe    R    0.00
Me     JJ        Richards         R  750.00
Mr     S         Naidoo           R  389.88
Ms     H         Nkosi            R    0.00
Mev    G         du Plessis       R 3807.32
Ms     M         Simpson          R    0.00
```

4.3 SORT CONTENTS OF DATA FILE

For **Option 2**, write a procedure **SortData**, that will do the following: (First make a backup copy of **PCresta.dat**.)

- When the user chooses **Option 2**, the contents of **PCresta.dat** must be sorted in descending order according to the balances. Assume that there will never be more than 20 clients stored within the file.
- After the information has been sorted, it must be written over the previous information in **PCresta.dat**. (16)

After **PCresta.dat** has been sorted and the user again chooses **Option 1**, the output should display as follows:

```
Title  Initials  Surname          Balance
Mev    G         du Plessis       R 3807.32
Ms     S         Halloway         R 1324.50
Me     JJ        Richards         R  750.00
Mr     S         Naidoo           R  389.88
Ms     H         Nkosi            R    0.00
Mnr    AJ        van der Merwe    R    0.00
Ms     M         Simpson          R    0.00
```

4.4 SKEP VAN TEKSLÊER

Skryf 'n prosedure, **SkepTeks**, vir **Opsie 3** wat die volgende sal doen:

- Wanneer die gebruiker **Opsie 3** kies, moet die lêer **PMenlyn.dat** gebruik word om 'n tekslêer genaamd **skuld.txt** te skep. Die tekslêer moet alle kliënte in **PMenlyn.dat** bevat wat uitstaande saldo's het (saldo groter as nul).
- Elke reël in die tekslêer moet die kliënt se titel, voorletters, van en saldo (geskei deur kommas), bevat.

Een reël in die tekslêer kan dus soos volg lyk:

```
Me,JJ,Richards,750.00
.
.
.
```

- Nadat al die nodige kliënte na die tekslêer geskryf is, moet 'n boodskap soortgelyk aan die onderstaande boodskap verskyn wat sal aandui hoeveel kliënte na die tekslêer geskryf is. (15)

```
4 kliënte na leer geskryf.
```

4.5 VERTOON INHOUD VAN TEKSLÊER

Skryf 'n prosedure, **VertoonTeks,** vir **Opsie 4** wat die volgende sal doen:

- Wanneer die gebruiker **Opsie 4** kies, moet die inhoud van **skuld.txt** op die skerm verskyn.
- Indien die lêer nie bestaan nie, moet 'n geskikte boodskap vertoon word en die program moenie probeer om die inhoud van die tekslêer te vertoon nie. (7)

Nadat die gebruiker **Opsie 4** gekies het, behoort die afvoer soos volg te vertoon:

```
Ms,S,Halloway,1324.50
Me,JJ,Richards,750.00
Mr,S,Naidoo,389.88
Mev,G,du Plessis,3807.32
```

Die volgende moet ingehandig word vir Vraag 4 (Pascal):

- 'n Drukstuk van `Vr4_XXXX.pas`.

[53]

4.4 CREATE TEXT FILE

For **Option 3**, write a procedure **CreateText**, that will do the following:

- When the user chooses **Option 3**, the file **PMenlyn.dat** must be used to create a text file named **debt.txt**. The text file must contain all clients from **PMenlyn.dat** with outstanding balances (balances greater than zero).
- Each line within the text file must consist of the client's title, initials, surname and balance (each separated by commas).

One line from the text file may look like this:

```
Me,JJ,Richards,750.00
.
.
.
```

- After all necessary clients have been written to the text file, a message similar to the one below, must be displayed, indicating the number of clients that have been written to the text file. (15)

```
4 clients written to file.
```

4.5 DISPLAY CONTENTS OF TEXT FILE

For **Option 4**, write a procedure **DisplayText**, that will do the following:

- When the user chooses **Option 4**, the contents of **debt.txt** must be displayed on screen.
- If the file does not exist, an appropriate message must be displayed and the program must not attempt to display the contents of the file. (7)

After the user has chosen **Option 4**, the output should appear as follows:

```
Ms,S,Halloway,1324.50
Me,JJ,Richards,750.00
Mr,S,Naidoo,389.88
Mev,G,du Plessis,3807.32
```

The following must be handed in for Question 4 (Pascal):

- A printout of `Q4_XXXX.pas`.

[53]

VRAAG 5

Om 'n geldige wagwoord vir die installering van 'n sekere program te genereer, word die volgende algoritme gebruik:

- Vra die persoon se naam **in kleinletters**. (bv. john g smith)
- Skakel die naam om na titelkas. (john g smith → John G Smith)
  (Titelkas beteken dat die eerste letter van elke woord 'n hoofletter is.)
- Las die naam aanhoudend aan homself totdat dit 'n string van meer as 19 karakters vorm. ('John G SmithJohn G Smith')
- Kry die ASCII-waardes van die 3^e, 13^e en 19^e karakters in bostaande string.
  (h=104 ; J=74 ; spasie=32)
- Skakel bostaande ASCII-waardes om na stringe en las dit dan aanmekaar om EEN string te vorm. ('1047432')
- Gaan bostaande string karakter vir karakter deur en kry telkens die ooreenstemmende letter van die alfabet. Ignoreer alle nulle.
  (1='A' ; 4='D' ; 7='G' ; 4='D' ; 3='C' ; 2='B') **L.W. 'A' se ASCII-waarde is 65.**
- Die wagwoord vir *john g smith* is dus 'ADGDCB'.

Nog 'n voorbeeld:

- mia cox
- Mia Cox
- Mia CoxMia CoxMia Cox
- a=97 ; o=111 ; C=67 → '9711167'
- 'IGAAAFG'

Begin met 'n nuwe program in Turbo Pascal en stoor dit as '`XXXX_Vr5.pas`' (`XXXX` stel die laaste vier syfers van jou eksamennommer voor).

5.1 Skryf 'n funksiesubprogram genaamd **TitelKas**, wat 'n string sal ontvang (in kleinletters) en dit dan na titelkas sal omskakel.

Voorbeeld: As '*mia cox*' na die funksie gestuur word, moet '*Mia Cox*' teruggestuur word. (8)

5.2 Skryf 'n prosedure **KryWagwoord**, wat 'n persoon se naam sal ontvang (in titelkas) en 'n wagwoord daarvoor sal genereer volgens bogenoemde algoritme. Slegs die wagwoord moet teruggestuur word na die roepstelling. (16)

Voorbeeld: As '*Mia Cox*' na die prosedure gestuur word, moet '*IGAAAFG*' teruggestuur word.

QUESTION 5

To generate a valid password for the installation of a certain program, the following algorithm is used:

- Ask for the person's name **in lower case**. (e.g. john g smith)
- Convert the name to title case. (john g smith → John G Smith)
  (Title case means that each word starts with a capital letter)
- Repeatedly add the name to it self until it forms a string with more than 19 characters.
  ('John G SmithJohn G Smith')
- Find the ASCII-values of the 3rd, 13th and 19th characters in the above-mentioned string. (h=104 ; J=74 ; space=32)
- Convert the ASCII-values mentioned above to strings and then add them together to form ONE string. ('1047432')
- Go through the string mentioned above, one character at a time, and in each case find the corresponding letter of the alphabet. Ignore all zeros.
  (1='A' ; 4='D' ; 7='G' ; 4='D' ; 3='C' ; 2='B') **NB. The ASCII value of 'A' is 65.**
- Consequently the password for john g smith is 'ADGDCB'.

Another example:

- mia cox
- Mia Cox
- Mia CoxMia CoxMia Cox
- a=97 ; o=111 ; C=67 → '9711167'
- 'IGAAAFG'

Start with a new program in Turbo Pascal and save it as '**XXXX_Q5.pas**' (**XXXX** represents the last four digits of your examination number).

5.1 Write a function subprogram named **TitleCase**, that will receive a string (in lower case) and then convert it to title case.

Example: If '*mia cox*' was sent to the function, then '*Mia Cox*' must be returned. (8)

5.2 Write a procedure **GetPassword**, that will receive a person's name (in title case) and then generate a password for that name, according to the algorithm mentioned above. Only the password must be returned to the calling statement. (16)

Example: If '*Mia Cox*' was sent to the procedure, then '*IGAAAFG*' must be returned.

5.3 Wanneer die program uitgevoer word, moet die gebruiker geleentheid gegee word om 'n naam in te tik. Die program moet dan die TWEE subprogramme in Vraag 5.1 en 5.2 roep om 'n wagwoord te kry vir die naam wat ingetik is. Die gegenereerde wagwoord moet as afvoer op die skerm verskyn. (4)

Die volgende moet ingehandig word vir Vraag 5 (Pascal):
- 'n Drukstuk van **`XXXX_Vr5.pas`**.

[28]

VRAAG 6

Die speletjie ***Battleships*** maak gebruik van 'n matriks van rye en kolomme waarop 'n aantal skepe weggesteek word. Die speler se opponent moet dan probeer om al die skepe raak te skiet.

Elke skip het 'n sekere lengte en kan horisontaal of vertikaal op die matriks geplaas word. 'n Skip waarvan die lengte VIER is, sal VIER plekke in die matriks beset.

LET WEL: In hierdie program gaan die skepe slegs horisontaal geplaas word.

'n Voorbeeld waar DRIE skepe horisontaal op die matriks geplaas is, kan soos volg lyk:

Die eerste skip lê in **ry 1; kolomme 6 tot 9**. (Lengte = 4)

Die tweede skip lê in **ry 2; kolomme 12 tot 15**. (Lengte = 4)

Die derde skip lê in **ry 7; kolomme 4 tot 7**. (Lengte = 4)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1						#	#	#	#						
2												#	#	#	#
3															
4															
5															
6															
7				#	#	#	#								
8															
9															
10															

Begin met 'n nuwe program in Turbo Pascal en stoor dit as '**`Vr6_XXXX.pas`**' (**`XXXX`** stel die laaste vier syfers van jou eksamennommer voor).

5.3 When the program is executed, the user must be prompted to enter a name. The program must then call the TWO subprograms from Questions 5.1 and 5.2 to get a password for the name entered by the user. The generated password must be displayed on screen. (4)

The following must be handed in for Question 5 (Pascal): • A printout of **XXXX_Q5.pas**.	**[28]**

QUESTION 6

The game of ***Battleships*** uses a matrix of rows and columns on which a number of ships are hidden. The player's opponent must then try to hit all the ships.

Every ship has a certain length and can be placed either horizontally or vertically on the matrix. A ship of length FOUR will occupy FOUR places in the matrix.

PLEASE NOTE: In this program the ships will only be placed horizontally.

An example of where THREE ships are placed horizontally on the matrix, may look as follows:

The first ship occupies **row 1; columns 6 to 9**. (Length = 4)

The second ship occupies **row 2; columns 12 to 15**. (Length = 4)

The third ship occupies **row 7; columns 4 to 7**. (Length = 4)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1						#	#	#	#						
2												#	#	#	#
3															
4															
5															
6															
7				#	#	#	#								
8															
9															
10															

Start with a new program in Turbo Pascal and save it as '**Q6_XXXX.pas**' (where **XXXX** represents the last FOUR digits of your examination number).

Die program moet aan die volgende voldoen:

6.1 Wanneer die program uitgevoer word, moet die program dadelik die volgende doen:

- Skep 'n matriks met TIEN rye (1 tot 10) en VYFTIEN kolomme (1 tot 15).
- DRIE skepe, elkeen met 'n lengte van VIER, moet ***horisontaal*** in 'n matriks geplaas word.
- Die beginposisies van die skepe moet ewekansig (random) gekies word.
- Waar die skepe op die matriks lê, moet 'n waarde van *TRUE* gestoor word en op al die ander plekke, 'n waarde van *FALSE* gestoor word.
- Geen gedeelte van 'n skip mag die matriks verbysteek nie.
- Die skepe mag oorvleuel.
- Sodra die skepe geplaas is, moet 'n keuselys op die skerm verskyn. (10)

Voorbeeld:

KEUSELYS
1. Wys plasings
2. Skiet skepe
3. Verlaat program

Keuse:_

6.2 As die gebruiker **Opsie 1** kies, moet die inhoud van die tweedimensionele skikking vertoon word. Sodra die gebruiker die Enter-sleutel druk, moet die matriks verdwyn en die opsielys moet weer verskyn: (6)

Voorbeeld:

```
    1  2  3  4  5  6  7  8  9  10 12 13 14 15
 1
 2                                  #  #  #  #
 3
 4
 5  #  #  #  #
 6
 7                              #  #  #  #
 8
 9
10
```

The program must comply with the following:

6.1 When the program is run, it must immediately do the following:

- Create a matrix that consists of TEN rows (1 to 10) and FIFTEEN columns (1 to 15).
- THREE ships, each with a length of FOUR, must be placed ***horizontally*** in the matrix.
- The starting positions of the ships must be chosen randomly.
- Where a ship occupies the matrix, a value of TRUE must be stored and in all other places a value of FALSE must be stored.
- No part of a ship may go outside the matrix.
- The ships may overlap.
- As soon as the ships have been placed, a menu must appear on screen. (10)

Example:

MENU
1. Show placings
2. Shoot at ships
3. Exit program

Choice:_

6.2 When the user chooses **Option 1**, the contents of the two-dimensional array must be displayed. As soon as the user presses the Enter key, the matrix must disappear and the menu must be displayed again. (6)

Example:

```
    1  2  3  4  5  6  7  8  9  10 12 13 14 15
 1
 2                                  #  #  #  #
 3
 4
 5  #  #  #  #
 6
 7                                  #  #  #  #
 8
 9
10
```

6.3 As die gebruiker **Opsie 2** kies, moet die program aan die gebruiker die geleentheid gee om 'n ry en 'n kolom in te tik (elkeen afsonderlik).

'n Gepaste boodskap moet vertoon word om aan te toon of die speler 'n skip mis of raakgeskiet het. (3)

Die program hoef **NIE** die aantal raakskote te tel of die inhoud van die matriks te verander nie.

Voorbeeld:

Indien 'n skip in **ry 2; kolomme 1 tot 4** lê en die speler skiet na **ry 2; kolom 2**, moet die volgende boodskap verskyn: **Dit was raak!**

Die volgende moet ingehandig word vir Vraag 6 (Pascal):
• 'n Drukstuk van `Vr6_XXXX.pas`.

[19]

TOTAAL: 100

EINDE

6.3 When the user chooses **Option 2**, the user must be prompted to enter a row and a column (each separately).

An appropriate message must then be displayed to indicate whether the player has hit or missed a ship. (3)

The program does **NOT** have to count the number of hits or change the content of the matrix.

Example:

If a ship occupies **row 2; columns 1 to 4** and the player fires at **row 2;column 2**, the following message must appear: **That was a hit!**

The following must be handed in for Question 6(Pascal): • A printout of `Q6_XXXX.pas`.

[19]

TOTAL: 100

END