DATA BASE APPLICATION IN VISUAL FOXPRO

by
Valentin Casavela

Abstract: Visual studio 6.0, witch comprises Visual FoxPro6.0, must be installed.In order to build a screen menu, either we entirely create that, or we call the Wizard from **Run menu**, but ***.scx** and ***.sct** files will be built up. So, the file s**tudents. scx** was built up by us and belongs to screen menu designing , but this can not be used in a program and must be transformed in a ***.spr** file, which belongs to that design too.
Our screen menu was designed with the Wizard hereby: From **RUN\Witzard** from **FoxProW** menu , the **Screen** option was chosen. A window was opening , where the database could be chosen and we might input values. With **NEXT** option we may select fields from the database, in which the values were inputting. In order to talk about a Database, the terms FIELD and RECORD must be understood.
Keywords : data base, field, record, prompt, bar, popup

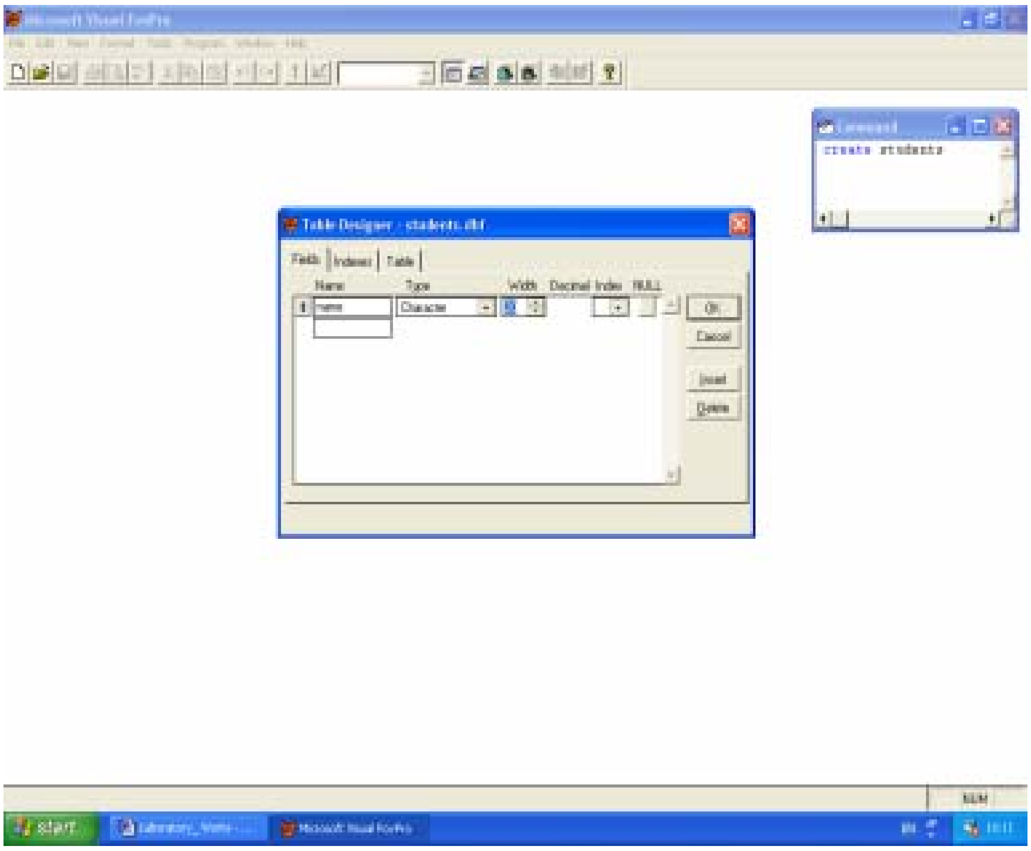
Data base

Database may be seen as a table, where a field means all the values or strings in a column and a record means the same, but in a row, as in the example bellow:

	field1	field2	field3
	Name	birthday	Tell
record	Brand Johns	04/03/1981	0256-285643
record	Osborn Monica	01/09/1980	0257-214532

For a ***new*** Database, in the **Command Window** , write: CREATE < file_name >

In the new window, introduce the name, the type and the field length, as in figure:



```
create students
```

Name	Type	Width
name	Character	

If the example above is used, then complete this:

Name	**Type**	**width (nr. of chars)**
name	Character	20
data	Date	8
tel	Character	11

Press **Enter** and a window with the question : **Input data record now ?** is opening . Answer **NO** and **Close** it. The new created file is **students.dbf**. We will choose earlier another way for inputting dates: **form**.

But, for modifying this database, type in **Command Window: Close all** and from **File**, **open** in **Microsoft Visual Studio\Vfp** the new created file, **students.dbf**. In **Command Window :USE "c:\ program files\microsoft visual studio\vfp98\stu-dents.dbf" EXCLUSIVE** appears.

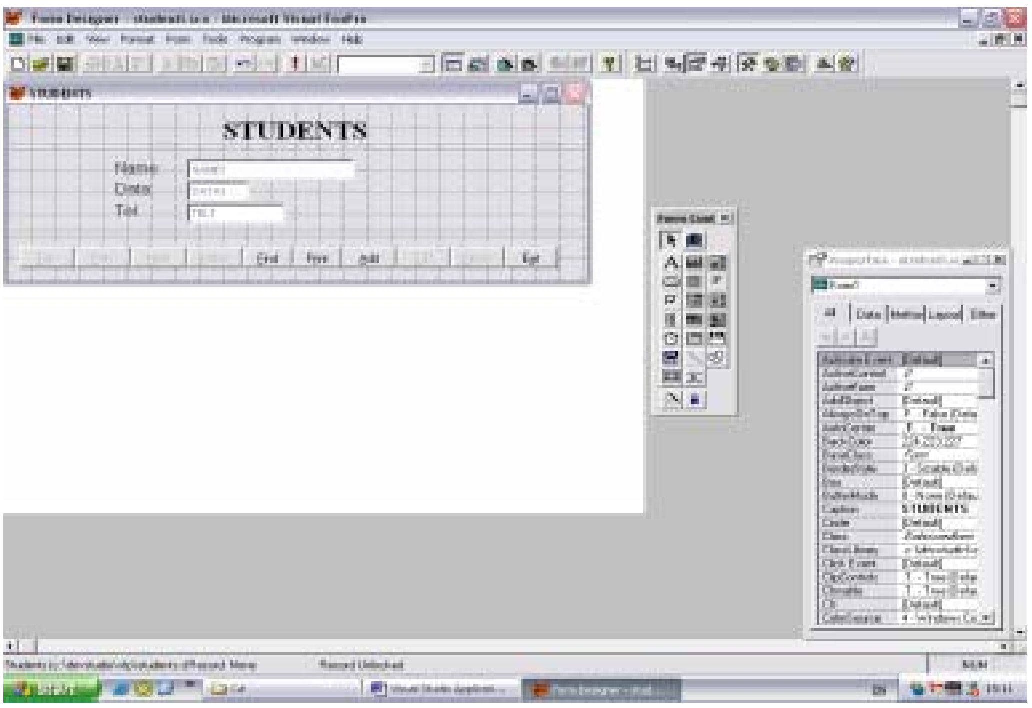
Enter and type in **Command Window**: **modi stru. Enter** and the figure above is shown again. You may modify it and, with **YES,** save the changes.

But, let’s come back to **form**. Our screen menu was designed with the Wizard hereby: from **FoxPro** menu select **Tools\Witzard\Form** and a window, **Wizard Selection** appears. Select **Form Wizard** and press **O.K**. In the window **Form Wizard\ Databases and tables** select **Students** and in **Available fields** select ‘ >>’ (all). In **Selected fields** appear all students’ fields. Click on **Next.** In the next window, **Step2**, select **Style: Embossed** and **Button Type: Text Buttons .** Many choices above and which follow are default, but you may try others.

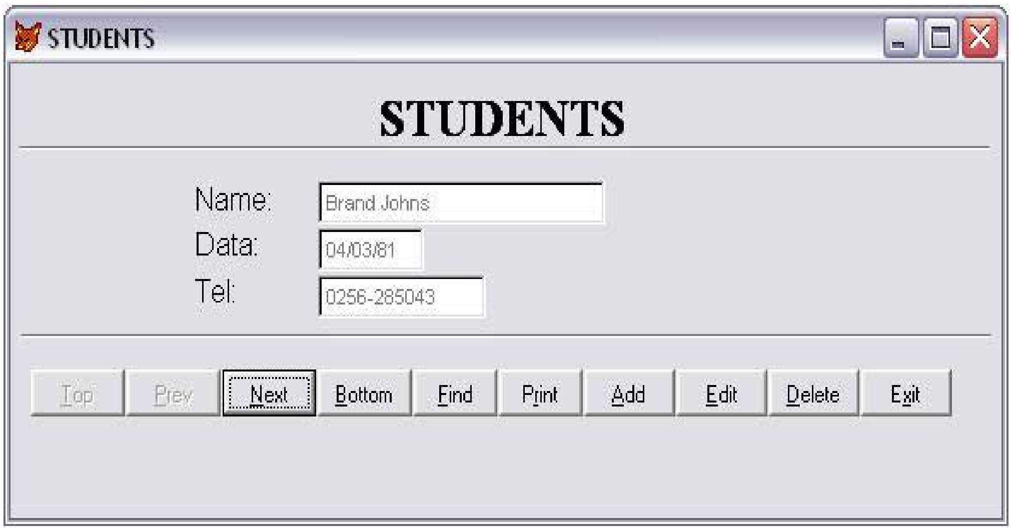
Click **Next.: Step3.** In **Available fields or index tag,** we order names by alphabetic: **Ascending** and, with **Add>** select all fields for **Selected files,** by repeated action. In other cases, you may omit some fields. In **Step4- Finish: Type a title for your form** write **Students**, select **Save form and modify in the Form Designer** and click on **Finish.** A window **Save as** appears and click on **Save.** The new file **Sudents.scx** may be found in **Program Files\Microsoft Visual Studio\Vfp98\.** Verify that !

The follow window is opening and….

.



….from **View** select **Proprieties;** so you may change the aspect of the window **STUDENTS.** For instance, click on '**Name**' and in **Proprieties** select **FontSize** and type 12 upside. After, select **FontBold** and select **True** upside, as follows:


STUDENTS

Name: Brand Johns

Data: 04/03/81

Tel: 0256-285043

Close it and operate in **Command Window.** Type **do FORM students.scx** (or select **Program\Do** and a **Do** window appears, in which select **Form** in **Files of type.** Open **STUDENTS.SCX,** a new window, **STUDENTS** is opening and here you may input dates. Click on **Add** and type dates from the first table of this paragraph: Bend Johns…i.e. After every record click **Save** and **Add.** Click **Exit** when you finished.

This database is called "students" and it should be used in the application which we will design in the next step. To modify this, type in **Command Window: DO FORM students.scx** or select in **Program\Do\All files: Students (**above **STUDENTS .CDX).**

Besides that, it'll be another database, "**medium**" (**medium.dbf**, created in the same way), with structure:

name	**type**	**width**	**decimals**
Name	Char	20	
Statistics	Numeric	3	0
Modeling	Numeric	3	0
C_sharp	Numeric	3	0
C_plusplus	Numeric	3	0

Database	Numeric	3	0
Lisp	Numeric	3	0
Medium	Numeric	5	2

To use this database, a Visual FoxPro program must be written. Further, it is shown this program, which holds a situation of several students' names, exams results and mediums.A program is opened by writing in the Command Window:

close all

MODI COMM < program name>

and is running with

DO <program name >.

In this case, the program is named "application". So we write:

MODI COMM application

Then, an **edit window** is opening. You may type in the following instructions, or you may copy the entire program **application.prg**. For that, from **CD**, in **aplicatii\ DATABASE-VS-FOXPRO-20** click right on **application.prg** and open it with **WordPad**. Press **Ctrl-A** for **Select All** and **Ctrl-C** for copy. Now, return in **edit Window** and press **Ctrl-V** for past. You may see the program.

But, to write your own application, type the next statements, without comments (*…):

set talk off ***** stop printing on screen the commands results
clear * clean screen
@ 5,10,9,65 box '*' * it draws a rectangle with characters ' * ', at coordinates 5,10 *(left-top corner) and 9,65 (bottom-right corner), where the first value is for row and *the second is for column.
* for printing something, somewhere you would desire on screen, use the command:
****@ <line >,<column> say "text"*** * where "text" is what it will appear printed, as in *program:
@ 7,12 say '   Students' situation and exam results'

@ 23,10 say ' '***

@ 37,10 say ' VALENTIN CASAVELA ‘

@ 10,25 to 20,56 PANEL ***** it draws a rectangle, with left top corner coordinates: *10,25 and right bottom coordinates : 20,56.

@ 12,29 say '- Western University -'

@ 13,33 say ‘“Vasile Goldis"’ ***** the Informatics Faculty identifications are *displayed inside the…

@ 16,30 say ’Informatics Faculty' * …rectangle

@ 17,30 say 'Tel.: 0257-214505 '

define menu meniu

“bare” option will be named **< bare1_option >** and will belong to the menu bare **<menu_ name>,** and the text will be **<expC1>,** displayed at a certain position, with the specification **AT <line>,< column>**. To set a certain order for bare options, we use the clauses **BEFORE** or **AFTER**. For more details, in **Command Window** type **help DEFINE PAD** and **Enter.**

define pad opt1 of meniu prompt 'Overview'
define pad opt2 of meniu prompt 'Display Printing'
define pad opt4 of meniu prompt 'Searching'
define pad opt3 of meniu prompt 'Exit'

The submenu bare options may also contain a database structure, opened previously. For more details, in **Command Window** type **help DEFINE POPUP** and **Enter.**

define popup rez
*”***rez”*** from ..result.

With this command, the bar option numbered <expN1> is defined from the sub-menu <popup name> and is called string <expC1>. But if we use <system option name> in FoxPro for Windows or VisualFoxPro, system menu options will be added. For more details, in **Command Window** type **help DEFINE BAR** and **Enter.**

define bar 1 of rez prompt 'Students'
define bar 2 of rez prompt 'Results'

define popup search
define bar 1 of search prompt 'Student'

define popup rat *******rat is a character string, given by us to this submenu
define bar 1 of rat prompt 'Students'
define bar 2 of rat prompt 'Results'
define bar 3 of rat prompt 'Results Selection'

define popup rest
define bar 1 of rest prompt 'Failed to Exams'
define bar 2 of rest prompt 'To Which Exam ?'
define bar 3 of rest prompt 'First'

define popup ex
define bar 1 of ex prompt 'Windows'
define bar 2 of ex prompt 'Fox Pro'

The option <pad name> of the menu bare <menu name1> activates the sub-menu <popup name>, or the menu <menu name2>.

on pad opt1 of meniu activate popup rez
on pad opt2 of meniu activate popup rat * we launch the submenu rat
pad opt3 of meniu activate popup ex
on pad opt4 of meniu activate popup search
on bar 3 of rat activate popup rest

Besides this, the command ON SELECT would be employed in order to choose a bare option or a submenu, going to the execution of anything else then the activation of a submenu or a menu. So, procedures from the same program, other programs from disk, or forms i.e., may be called and run .

on select bar 2 of rat do afis * call and run the 'afis' procedure
on select bar 2 of rez do intr * call and run the 'intr' procedure
on selection bar 2 of ex return * call and run the 'return' function, to exit the program *and to return to Visual FoxPro
on selection bar 1 of ex quit * call and run the 'return' function to exit the program *and to return to Windows
on selection bar 1 of rez do form students.scx *call and run the form "studenti.scx," *previously created. The passway is: ***define pad opt1 of meniu prompt 'Overview'***
*… ***on pad opt1 of meniu activate popup rez… define bar 1 of rez prompt 'Students'***
on selection bar 1 of rat do afisst * call and run the "afisst" procedure.

on selection bar 1 of rest do rest * call and run the “ rest” procedure
on selection bar 1 of search do search* call and run the “search” procedure
on select bar 2 of rest do searching* call and run the “searching” procedure
on select bar 3 of rest do goods* call and run the “goods” procedure

The deactivation of a menu bare is made with:

DEACTIVATE MENU <menu name1>
[, <menu name2> ...] | ALL

deactivate menu meniu

Our menu will look like:

Overview	**Display Printing**	**Search**	**Exit**
Students	Students	Students	Windows
Results	Results		FoxPro
	Faild to exams	All	
		Exam	
		First	

Next, we will define the procedures just called above.After we selected the “**Students**” option, from the submenu “**Overview**”, the **form** (screen) **students.scx** run and appears and we may select, from the same submenu, the “**Results**” option and then the procedure “**intr**”, which is shown bellow, is launched. The passway is… ***define pad opt1 of meniu prompt 'Overview'****….* ***on pad opt1 of meniu activate popup rez*** *…* ***define bar 2 of rez prompt 'Results'****….* ***on select bar 2 of rez do intr***

procedure intr * Procedure defining is made with PROCEDURE <name *procedure>.
@ 4,0 clear to 32,200 * clean the screen between points 4,0 and 32,200

The clauses SHARED and EXCLUSIVE are employed in networking, and NOUPDATE refers a READ ONLY database. For more details, in **Command Window** type **help Use** and **Enter.**

use medium in 1 * open the “**medium database**”, created above, in work area 1.
r='Y' * initialize r variable with character Y
@ 4,18 say *'--'* *display a string of chars ‘-’, *beginning with position 4,18
@ 24,18 say '--' *display a string of chars ‘-’, *beginning with position 24,18

do while upper(r)='Y' *”upper” changes the character ‘r’ in upper case. So, *while r=‘y’, or r = ’Y’, the program repeats the next instructions, till *enddo.* Other-*wise, it does’nt. ‘r’ is the character inputed somewhere bellow, but above the *ENDDO instruction, the ‘Y’ or ‘y’ inputs being the condition for *“while” cycle. The *upper instruction leaves ‘r’ unchanged if r=‘Y’.
@3,0 clear to 32, 200
new=replicate(' ',26)
@ 6,20 say 'Name :' get new* introduces in “new” field the string inputted from the *keyboard.No more then 26 digits. Press **Enter.**
read
locate for upper(alltrim(name))=upper(alltrim(new)))*turn all characters, either *upper- case or lower- case, in upper-case. While the end of the database isn’t touched…
if found()* …test if in “name” field there is the same value as in “new” variable, for *medium **database only! For avoiding the situations as “John ”** <> **“John”,** use ***alltrim** to clear spaces (ex. alltrim (“John “)=“John”). Also exist **ltrim –** clean all *left spaces. **rtrim –** clean right spaces.

@8,20 say 'statistics='
??statistics display statistics field
*For displaying , we may type a field name prefixed by ? or ??, the difference being *that ? causes passing to the next row.
?? ' '
@10,20 say 'modeling='*likewise
??modeling
?? ' '
@12,20 say 'database='
??database
?? ' '
@14,20 say 'C#='
??C_sharp
?? ' '
@16,20 say 'C++='
??c_plusplus
?? ' '
@18,20 say 'lisp='
??lisp
??' '

@20,20 say 'medium='
??medium
else
appe blank * appends a void record. The ***new*** is a new name.
go bottom * positions the cursor on the last record, which just was appended .
@7,20 say "This is a new name!! Please enter marks !!"
replace name with new
@ 8,20 say 'Statistics : ' get statistics * introduces in "Statistics" field, from *****medium database** the mark inputed from the keyboard: one or two digits. Press ***Enter.**

@ 10,20 say 'Modeling : ' get modeling* …..similarly
@ 12,20 say 'Database: ' get data_base
@ 14,20 say 'C # : ' get c_sharp
@ 16,20 say 'C++ : ' get c_plusplus
@ 18,20 say 'Lisp : ' get lisp
read* …and read them.

med=modeling+statistics+C_sharp+Lisp+C_plusplus+data_base * med variable takes *the sum of the 6 field contents

medd=med/6 * becomes med/6
replace medium with medd * replaces the contents of medium field with the *medd variable value
endif
@ 20,20 say 'Input again (Y/N) ?' get r **Enter** or **Esc** has no effect.
read
enddo * closes the DO WHILE body
@2,0 clear to 32,200
close all * close all databases, opened with this procedure.

Now, the "afis" procedure follows, which executes when the option "Results", from the submenu DISPLAY PRINTING, is selected. The passway is :… ***define pad opt2 of meniu prompt 'Display Printing'***… ***on pad opt2 of meniu activate popup rat … define bar 2 of rat prompt 'Results'***…. ***on select bar 2 of rat do afis***
procedure afis
@ 4,0 clear to 32,200* A window, but not entire screen is cleaned.
use medium in 1
i=5* The first row number for displaying names and exam results. It'll be incremented.
do while (!EOF()) * EOF() returns true only if the program arrived to the end of the ***database medium**.

j=22* The first column number for displaying exam results. It’ll be incremented
@i,1 say name*display **name** field
@i,j say 'statistics='
??statistics * display statistics field
*For displaying , we may type a field name prefixed by ? or ??, the difference being *that ? causes passing to the next row.
?? ' '
@i,j+15 say 'modeling='
??modeling
?? ' '
@i,j+30 say 'database='
??database
?? ' '
@i,j+44 say 'C#='
??C_sharp
?? ' '
@i,j+52 say 'C++='
??c_plusplus
?? ' '
@i,j+62 say 'lisp='
??lisp
??' '
@i,j+71 say 'medium='
??medium*last column or field
i=i+1*
***@i,0,i,110 box ****display a line to separate records
i=i+1 *pass to next row or record
if i>=32* does a page finish ?
p=replicate(' ',12)* Yes.. Specify the character expression that is replicated, that is *
*space.

*12 specifies the number of times the character expression is replicated.
@31,7 say 'Do you want to see the next page ? y and Enter/ n and Enter' get p
read* ask user
if upper(p)='Y'*go to next page
@3,0 clear to 32, 200 * …but, first, clean the window

i=5* Again, the first row number for displaying names and exam results
else
go bottom*If 'No', positions on the last appended record
@3,0 clear to 32,200*clean the window
endif
endif
skip *causes passing to the next database
enddo
@3,0 clear to 32,200*clean the window
close all

If we select the option Students, from the Display Printing submenu, this launches the afisst procedure.. The passway is.. ***define pad opt2 of meniu prompt 'Display Printing'… on pad opt2 of meniu activate popup rat … define bar 1 of rat prompt 'Students' …on selection bar 1 of rat do afisst***
procedure afisst
@ 4,0 clear to 32,200 * clears the screen
use students * opens the "students" database
i=7 * "i" is initialized by 7 and so it will be used bellow
do while (!EOF()) * while the database end isn't touched, execute:
@ i+1,2 say name * displays in the row i+1 and column 2 the "name" field contents
@ i+1,30 say 'birthday: '
@ i+1,40 say data* displays in the row i+1 and column 40the "data" field
 *contents
@ i+1,50 say 'tel: '
@ i+1,56 say tel * displays in the row i+1 and column 42 the "tel" field contents
i=i+2*pass to the next line
@i,0,i,70 box*display a line

***if i>=32 ****likewise above procedure
p=replicate(' ',12)
@31,7 say 'Do you want to see the next page ? y and Enter/ n and Enter' get p
read
if upper(p)='Y'
@3,0 clear to 32, 200
i=5

else
go bottom
@3,0 clear to 32,200
endif
endif

skip *passes to the next record
enddo * closes the do cycle
close all * closes the database

Afterwards , only students fallen to exams and to which exams, will be dis-played. For that, we use database named **medium** in **procedure rest,** activated on the way: ***define pad opt2 of meniu prompt 'Display Printing',*****…..** ***on pad opt2 of meniu activate popup rat,*****….** ***define bar 3 of rat prompt 'Results Selection'*****… ….** ***on bar 3 of rat activate popup rest*****….*****define bar 1 of rest prompt 'Failed to exam'*** **…** ***on selection bar 1 of rest do rest.*** and we may see what student has marks less then 5 and to which discipline.

procedure rest
@ 4,0 clear to 32,200 * cleans screen
use medium * open the "medium" database. This case, if you eventually open another *atabase, this **medium** database will be closed.
i=5 * variable takes value 7
do while (!EOF()) * while we are not touching the end of the database, …
j=30* variable "j" takes value 30
if statistics <5* if statistics mark is less then 5,…
   @ i+1,2 say name* display name,
   @ i+1,j+1 say 'Statistics'*display ***Statistics***
   j=j+10* and increase column with 10, to repeat displaying for other failed exam
endif
if modeling<5
   @ i+1,2 say name* if a student failed more exams, we keep the same row,…
      @ i+1,j+1 say 'Modeling'* but jump some columns
      j=j+10
endif
if C_sharp<5
      @ i+1,2 say name* likewise
      @ i+1,j+1 say 'C#'

```
        j=j+6
endif
if database<5
        @ i+1,2 say name
        @ i+1,j+1 say 'DataBase'
        j=j+11
endif
if C_plusplus<5
        @ i+1,2 say name
        @ i+1,j+1 say 'C++'
        j=j+6
endif
if lisp<5
  @ i+1,2 say name
  @ i+1,j+1 say 'Lisp'
  j=j+1
endif
i=i+1*pass to the next line for the next  record  (name)

if i>=32 *Ask the user for next page displaying.Likewise the other procedures above.
p=replicate(' ',12)
@35,7 say 'Do you want to see the next page ? y and Enter/ n and Enter' get p
read
if upper(p)='Y'
@3,0 clear to 36, 200
i=5
else
go bottom
@3,0 clear to 36,200
endif
endif

skip * pass to the next record and repeat conditions
enddo *closes “do while”cycle
close all * closes database, previously opened.
```

If somebody wants to search in database for information about a certain stuent, he will have to input student name in a variable and run through the entire database till a record, which has in name field a value equal with that of his variable. So, the student will be found. Then the information will be displayed. To set the passway, we wrote above: ***define pad opt4 of meniu prompt 'Searching'…. on pad opt4 of meniu activate popup search …define bar 1 of search prompt 'Student',…..on selection bar 1 of search do search procedure search***

@ 4,0 clear to 32,200 * clear screen

name =replicate(' ',30) * initialize the "name" variable with a string of 30 characters.

@ 6,10 say 'Input name : ' get name'* in "name" variable input the string typed from *the keyboard.

use students in 1 * open the database "students" in the 1-st work zone.

use medium in 2 * open the database "medium " in the 2-nd work zone

sele 1 * select database, opened in the 1-st work zone.

*to search for a certain record , FoxProW has the instruction LOCATE FOR.

*locate for alltrim(name)=alltrim(name1)

locate for upper(alltrim(name))=upper(alltrim(num))*turn all characters, either upper- *case or lower- case, in upper-case. While the end of the database isn't touched…

if found()* test if in "name" field there is the same value as in "num" variable, for ***students database only! For avoiding the situations as "John " <> "John",** use ***alltrim** to clear spaces (ex. alltrim ("John ")="John"). Also exist **ltrim –** clean all *left spaces. **rtrim –** clean right spaces.

 @ 9,10 say 'Birthday : ' * display the text string "Birthday"

 @ 9,28 say birthday * display Birthday

 @ 10 ,10 say 'Tel : ' * display the text string Tel :

 @ 10,28 say tel * display the field contents " tel"

else

 @ 8,10 say 'This student dates were not introduced'* fail in searching

endif

sele 2 * select database opened in 2-nd work zone.

locate for upper(alltrim(name))=upper(alltrim(num))*Even we did'nt find **num** in ***students** database (selected from **Display Printing/Students-**only for verifying **num**), *we search for that in **medium** data base, (selected from **Display Printing/Results-** for *verifying **num).**

*Maybe here, that student name was found. To find that, FOUND() logical *instruction is used, which returns true if it could find, or false if it couldn't.

if found() * if found**..**

@ 11,10 say 'Statistics : ' * display text string " Statistics" :
@ 11,28 say statistics * display contents of statistics field
@ 12,10 say 'Modeling : '
@ 12,28 say modeling
@ 13,10 say 'DataBase : '
@ 13,28 say data_base
@ 14,10 say 'C# : '
@ 14,28 say C_sharp
@ 15,10 say 'Lisp : '
@ 15,28 say lisp
@ 16,10 say 'C++ : '
@ 16,28 say C_plusplus
@ 18,15 say 'Medium : '
@ 18,28 say medium

else @ 12,10 say 'The marks for that student where not input' * ..if the student *wasn't found neither in **medium** data base, display the text: **The marks…**

endif *close the "if conditional instruction body".
enddo* search finished
close all

Suppose we want to follow up the exam situation for a group of students, especially for students follen to a certain exam. As we have seen in the former presenting procedure, in a variable we must input the exam name and then we shell see which student follen that exam. The passway is… ***define pad opt2 of meniu prompt 'Display Printing'*** ….***on pad opt2 of meniu activate popup rat… define bar 3 of rat prompt 'Results Selection'…on bar 3 of rat activate popup rest*** ….***define bar 2 of rest prompt 'To Which Exam ?'***….***on select bar 2 of rest do searching.***

procedure searching
@ 4,0 clear to 32,200
use medium * open the database named medium
ex=replicate(' ',12) * the ex variable is 12 characters long
@ 10,10 say 'Type the exam name: ' get ex * Input in ex variable the value typed from *the keyboard.
read
i=10

if upper(alltrim(ex'))= upper('statistics') * if "statistics" was typed, then look for a *student, which has a value less then 5, in statistics field.
do while (!EOF()) *that is, run through the entire students database
    if Statistics<5 * if a student has the statistics mark less then 5, then his name will be *displayed.
     @ i+1,10 say name
     i=i+1
  do nextpage* if the table of names exceeds the current window, call the bellow ****nextpage*** procedure

```
   endif
   skip
   enddo
endif
```

We proceed similarly for every exam, to which the student was possibly present.

```
if upper(alltrim(ex))= upper('modeling')
do while (!EOF())
    if modeling<5
     @ i+1,10 say name
     i=i+1
```

do nextpage* if the table of names exceeds the current window, call the bellow ****nextpage*** procedure

```
   endif
   skip
   enddo
   endif

if upper(alltrim(ex))= upper('database')
    do while (!EOF())
    if Data_base<5
     @ i+1,10 say name
     i=i+1
```

   do nextpage* if the table of names exceeds the current window, call the bellow ****nextpage*** procedure

```
   endif
   skip
   enddo
```

```
    endif

if upper(alltrim(ex))= upper('c#')
     do while (!EOF())
     if C_sharp<5
      @ i+1,10 say name
      i=i+1
do nextpage*if the table of names exceeds the current window, call the bellow
*nextpage procedure

    endif
    skip
    enddo
endif

if upper(alltrim(ex))= upper('c++')
     do while (!EOF())
     if C_plusplus<5
      @ i+1,10 say name
      i=i+1
do nextpage*if the table of names exceeds the current window, call the bellow
*nextpage procedure

    endif
    skip
    enddo
endif

if upper(alltrim(ex) )= upper('lisp')
     do while (!EOF())
     If Lisp<5
      @ i+1,10 say name
      i=i+1
do nextpage*if the table of names exceeds the current window, call the bellow
*nextpage procedure
    endif
    skip
    enddo
endif
```

close all

A database sorting leads to a new database, called <file>, sorted by field <field1> in ascening or in descending order. This sorting has however a problem: every time when we select the option “First”, that sorting will be made and therefore a file, with the same name, will be created. So, every time a message will be displayed, a message which tells us that this file exists and asks if we would write over that. Every time choose YES.

The passway is… ***define pad opt2 of meniu prompt 'Display Printing'***…… ***on pad opt2 of meniu activate popup rat***….. ***define bar 3 of rat prompt 'Results Selection'***….. ***on bar 3 of rat activate popup rest*** … ***define bar 3 of rest prompt 'First'***… ***on select bar 3 of rest do goods***

procedure goods
@ 4,0 clear to 24,79 * clear screen
use medium in 1 * open the database „medium“
sort to med on medium /D *sort descending database after the field “medium” and the *sorted database “med” will result .
use med in 2 * open the new database in work zone 2
sele 2 * select work zone 2
k=7

For a variable taking values from <expN1> to <expN2>, with the “ STEP <expN3>”, the <statement>.instruction group is executed.

```
for i=1 to 5
        @ k+i,5 say name
        @ k+i,30 say medium
        skip
endfor
close all
```

The last procedure, called many times above, is **nextpage**.This was discussed when it was used as a simple program sequence, in **afisst** procedure or in others.

```
procedure nextpage
if i>=28
p=replicate(' ',12)
@31,7 say 'Do you want to see the next page ? y and Enter/ n and Enter' get p
read
if upper(p)='Y'
```

```
@11,0 clear to 32, 200
i=10
else
go bottom
@
11,0 clear to 32,200
endif
endif
```

After writing all those instructions in the program editing window, save and close this window, with key combination CTRL+W. After that, write, in the command window, "DO Application". If all was correctly written, the program is launching, but if it wasn't, an error message is displaying.

There may be exist fields in a database, in which the values must not be input, because those either are co-pied or result from a calculation. Again **NEXT** and we select one of the 3 predefined display types, again **NEXT** and input the students' names. Chose **Modify Screen** with **DesignTtool** and select **FINISH**. After few moments, the window **studenti.scx** is ope-ning and we may see how our screen looks like and we would modify the design. After the modifies were finished, from **Program** submenu (FoxProW) select GENERATE option, which will build the **studenti.spr** file and, automatically, **studenti.spx** file too.

That file is launched in execution, when click on **Students** in **Input** submenu. In the same moment with the **student.scx** file building, the **studenti.sct** file was built up too.

References:

1. Julian Templeman: "Visual C++.NET", Teora,ROMANIA, 2003, ISBN 973-20-0643-9
2. Charles Petzold: "Prgramarea in Windows cu C#",Teora,ROMANIA, 2003,ISBN 973-20-0639-0
3. MSDN ACADEMIC ALLIANCE –License Number: 0702 Part No.08_80170 for Visual Studio 6.0

Author:
Valentin Casavela - University of Petrosani, Romania