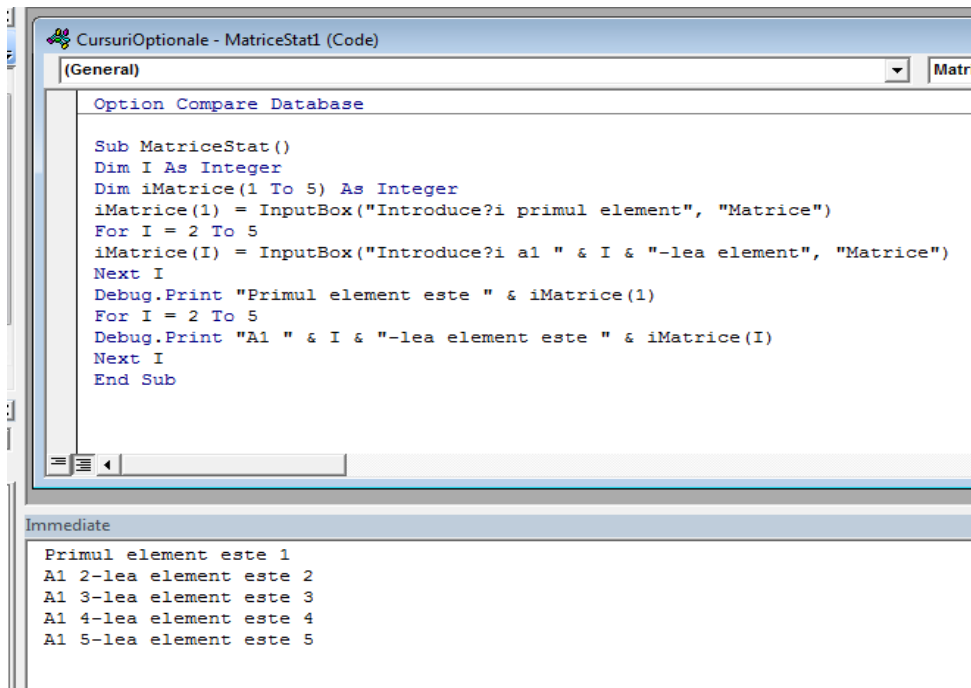


LUCRARE DE LABORATOR 9 *Birotică și baze de date*, L. Mocean

Partea 1.

1. Să se citească de la tastatură elementele unei matrice de 1*5 elemente și să se afișeze valorile citite. Procedura se execută cu *Run*



The screenshot shows the Visual Basic IDE with the code editor for 'CursuriOptionale - MatriceStat1 (Code)'. The code is as follows:

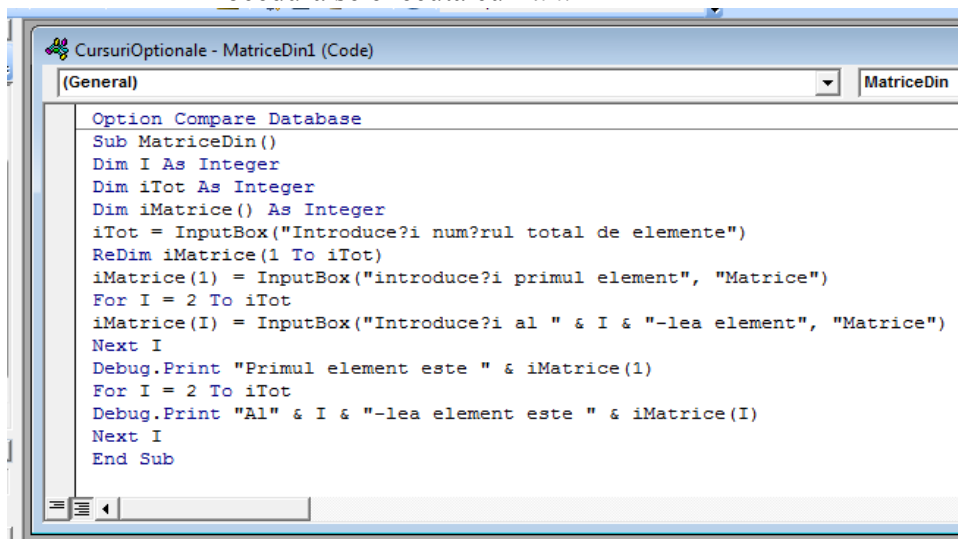
```
Option Compare Database

Sub MatriceStat()
    Dim I As Integer
    Dim iMatrice(1 To 5) As Integer
    iMatrice(1) = InputBox("Introduceți primul element", "Matrice")
    For I = 2 To 5
        iMatrice(I) = InputBox("Introduceți a1 " & I & "-lea element", "Matrice")
    Next I
    Debug.Print "Primul element este " & iMatrice(1)
    For I = 2 To 5
        Debug.Print "A1 " & I & "-lea element este " & iMatrice(I)
    Next I
End Sub
```

The Immediate window at the bottom shows the following output:

```
Primul element este 1
A1 2-lea element este 2
A1 3-lea element este 3
A1 4-lea element este 4
A1 5-lea element este 5
```

2. Să se citească de la tastatură numărul de elemente ale unei matrice, elementele, și să se afișeze valorile citite. Procedura se execută cu *Run*.

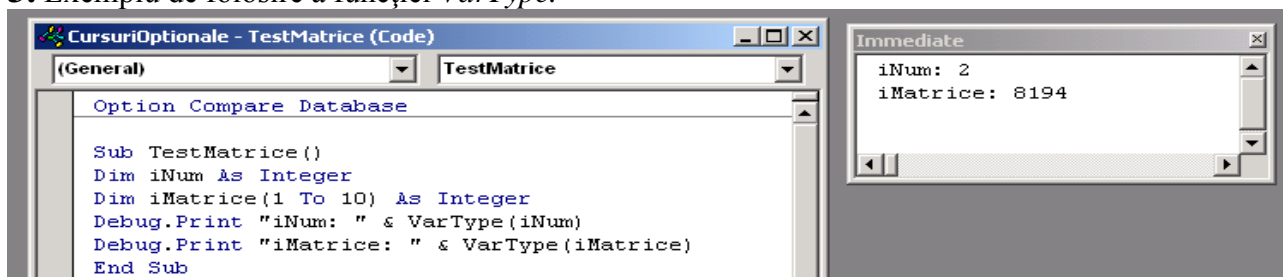


The screenshot shows the Visual Basic IDE with the code editor for 'CursuriOptionale - MatriceDin1 (Code)'. The code is as follows:

```
Option Compare Database

Sub MatriceDin()
    Dim I As Integer
    Dim iTot As Integer
    Dim iMatrice() As Integer
    iTot = InputBox("Introduceți numărul total de elemente")
    ReDim iMatrice(1 To iTot)
    iMatrice(1) = InputBox("introduceți primul element", "Matrice")
    For I = 2 To iTot
        iMatrice(I) = InputBox("Introduceți a1 " & I & "-lea element", "Matrice")
    Next I
    Debug.Print "Primul element este " & iMatrice(1)
    For I = 2 To iTot
        Debug.Print "A1 " & I & "-lea element este " & iMatrice(I)
    Next I
End Sub
```

3. Exemplu de folosire a funcției *VarType*.



The screenshot shows the Visual Basic IDE with the code editor for 'CursuriOptionale - TestMatrice (Code)'. The code is as follows:

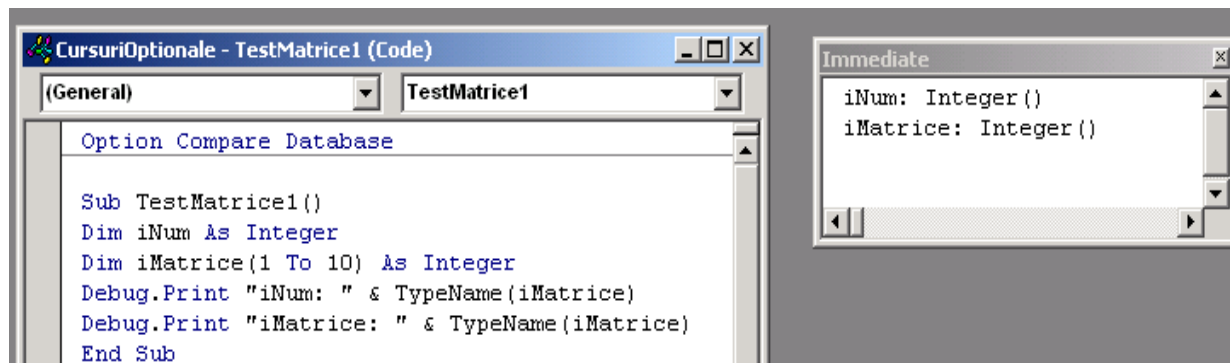
```
Option Compare Database

Sub TestMatrice()
    Dim iNum As Integer
    Dim iMatrice(1 To 10) As Integer
    Debug.Print "iNum: " & VarType(iNum)
    Debug.Print "iMatrice: " & VarType(iMatrice)
End Sub
```

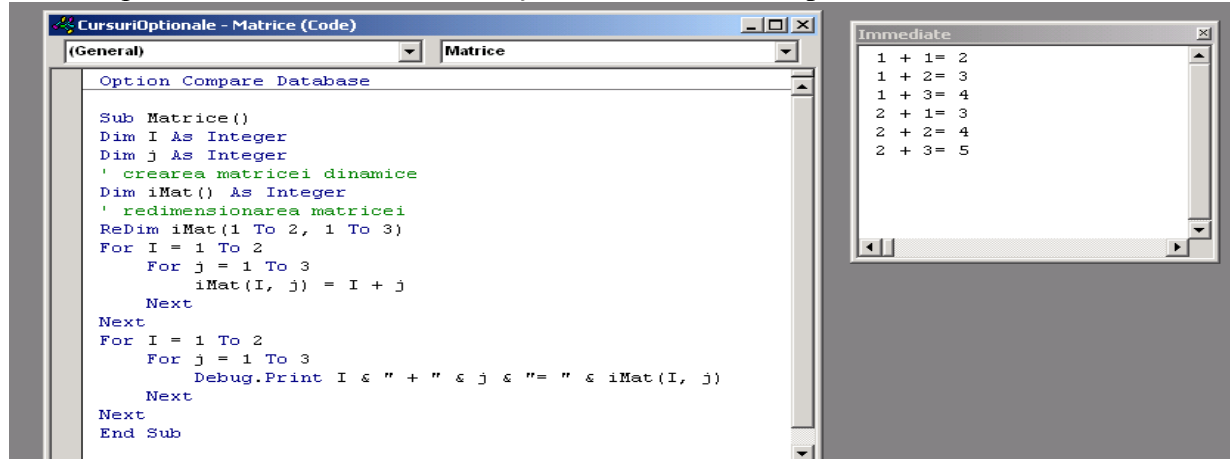
The Immediate window at the bottom right shows the following output:

```
iNum: 2
iMatrice: 8194
```

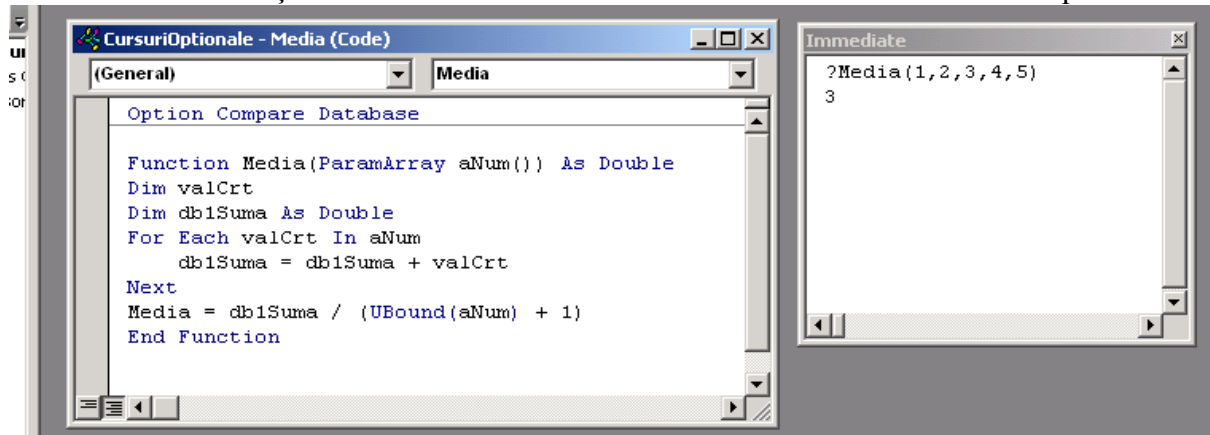
4. Exemplu de folosire a funcției *TypeName*.



5. Exemplu de declarare a unei matrici și de alocare de valori pentru elemente.



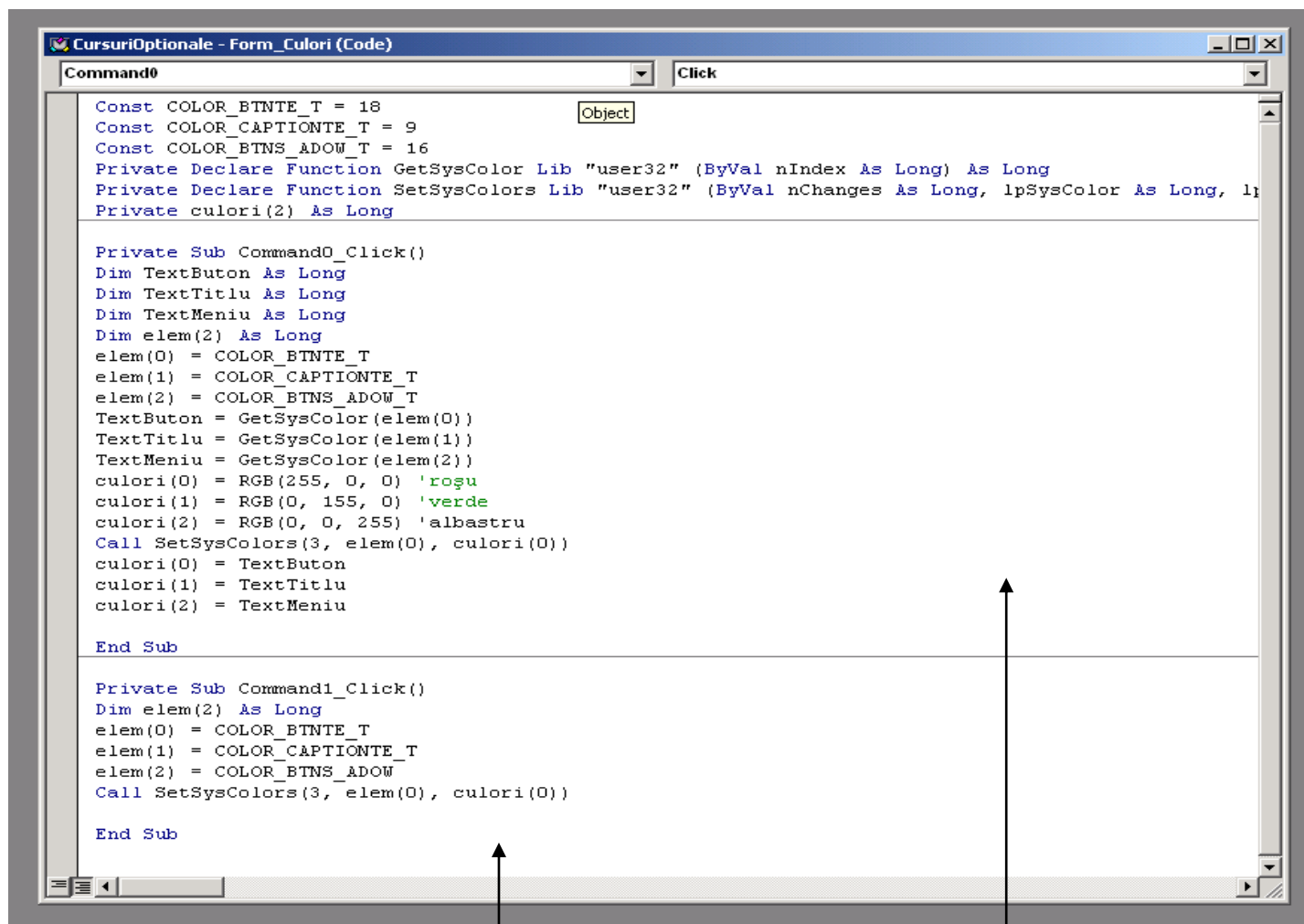
6. Următoarea funcție calculează media aritmetică a numerelor ce îi sunt date ca parametri:



7. Exemplul 1 din curs9.

Dorim să schimbăm câteva dintre culorile pe care Windows le folosește pentru orice aplicație: culoarea cu care apare scrisă denumirea aplicației în bara de titlu, culoarea folosită pentru textul butoanelor și culoarea folosită pentru a desena umbra butoanelor.

1. Creăm formularul Culori, ce conține două butoane: "Schimbă culori" și "Refă culori". În pagina de proprietăți a formularului efectuăm modificările (apar și la problema următoare în fereastra *Properties*).
2. În pagina Event (evenimente) a ferestrei de proprietăți a butonului "Schimbă culori" apăsați butonul (...) din dreptul câmpului OnClick; alegeți opțiunea Code Builder și apăsați OK, pentru a deschide modulul formularului și a edita codul pentru tratarea evenimentului InClick al acestui buton.
3. La secțiunea (General) (Declarations) a modulului adăugați următoarele linii de cod:

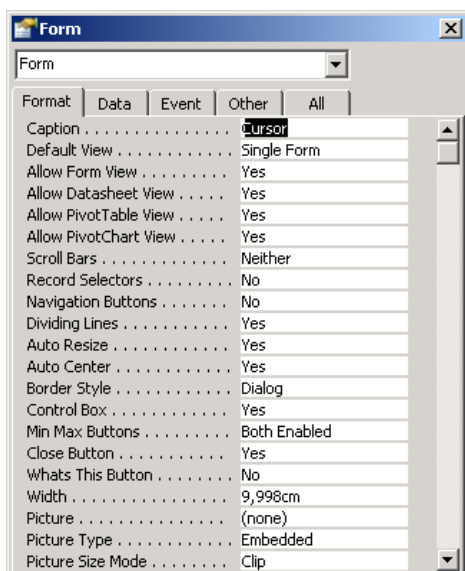


Am declarat trei constante, două funcții API și o matrice. Funcția `GetSysColor` returnează culoarea folosită pentru a desena un anumit element al interfeței cu utilizatorul. Ea primește ca argument o valoare ce definește culoarea elementului respectiv. Valoarea `COLOR_BTNTTE_T` reprezintă culoarea textului butoanelor, `COLOR_CAPTIONTE_T` reprezintă culoarea denumirii aplicațiilor, iar `COLOR_BTNS_ADOW_T` reprezintă culoarea cu care se desenează umbrele butoanelor. Funcția `SetSysColors` modifică culorile anumitor elemente ale interfeței cu utilizatorul. Ea va primi ca argument numărul elementelor ce urmează să-și modifice culoarea, o matrice ce conține aceste elemente și o matrice cu noile culori.

4. Observăm că în modul există deja o definiție vidă a procedurii pentru tratarea evenimentului Click al butonului “Schimbă culori”. În corpul acestei proceduri introducem următoarele linii de cod:

Această procedură obține întâi culorile inițiale ale elementelor specificate anterior. Funcția `RGB` creează o culoare, date fiind cantitățile de roșu, verde și albastru (între 0 și 255) ce o compun. Matricea `culori`, care este publică în cadrul modulului, primește inițial ca valori ale elementelor culorile roșu, verde și albastru pe care le vom folosi ca noi culori pentru elementele grafice specificate. Apelând funcția `SetSysColors`, facem efectiv modificările, după care, în matricea `culori` păstrăm vechile culori obținute cu ajutorul funcției `GetSysColor` (pentru a le putea reface ulterior).

5. Pentru a scrie o procedură pentru tratarea evenimentului OnClick al butonului “Refă culori”, deschideți pagina de proprietăți a acestuia și procedați ca la pasul 2. În modulul formularului va apărea o procedură cu corpul vid. Scrieți în corpul ei următoarele linii de cod:



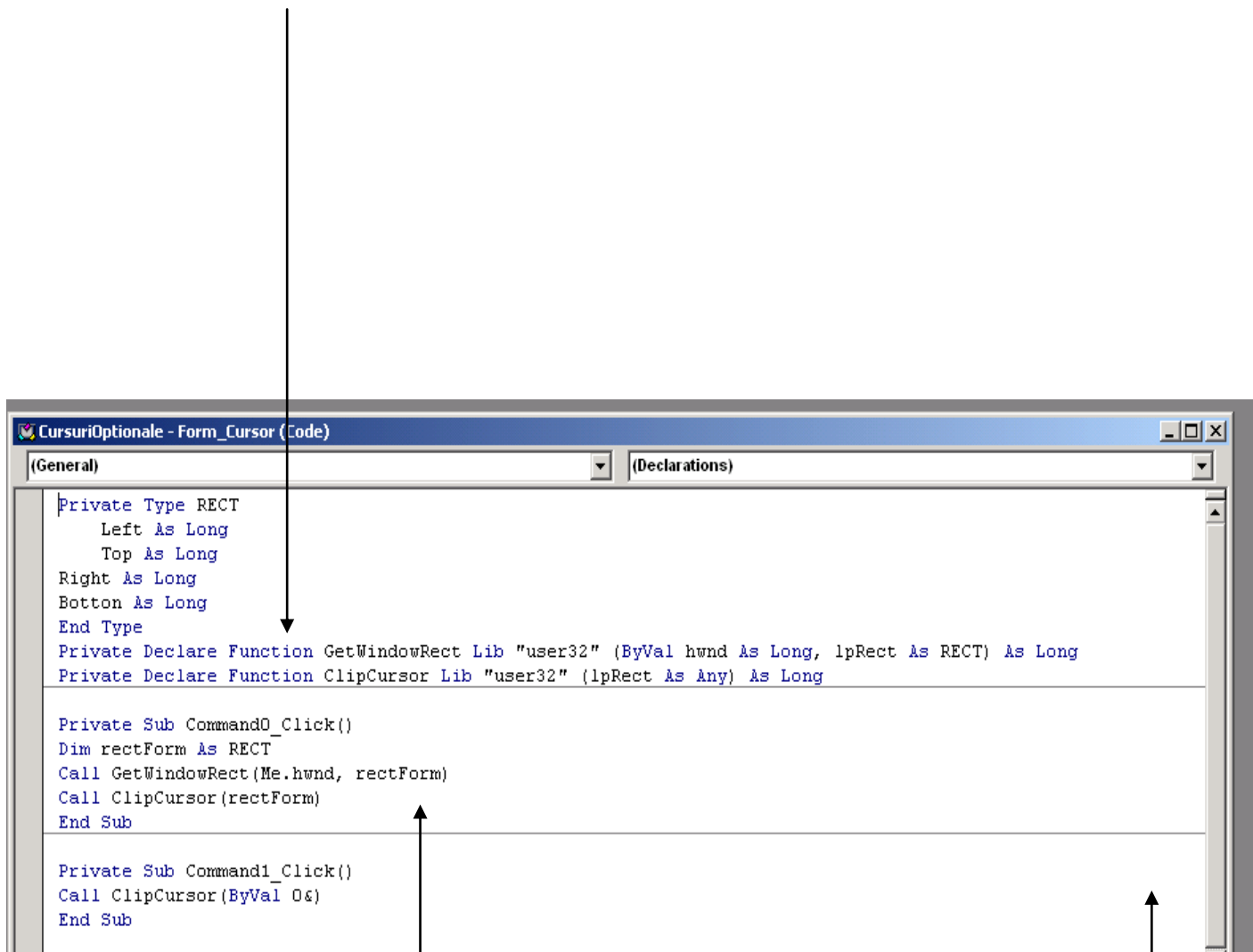
După cum v-ați putut da seama, butonul “Refă culori” va da elementelor grafice specificate vechile culori și anume cele stocate în matricea `culori` (care are ca domeniu de vizibilitate întregul modul).

6. Treceți în modul Form View și testați funcționarea celor două butoane.

8. Exemplul 2 din Curs10.

1. Creați formularul `Cursor` ce conține două butoane: `Capturează` și `Eliberează`. În pagina sa de proprietăți efectuați modificările:

2. La secțiunea (General) (Declarations) a modului formularului , adăugați următoarele declarații:



Funcția FetWindowsRect returnează dreptunghiul ocupat de o anumită fereastră.

3. În cadrul corpului procedurii pentru tratarea evenimentului OnClick al butonului Capturează, scrieți următoarele linii de cod:

Apăsând pe butonul Capturează, cursorul mouse-ului se va mai putea mișca doar în interiorul formularului Cursor. Funcția GETWindowRect primește ca argument identificarea formularului (Me.hwnd) și adresa unei variabile de tip RECT (rectForm) în care va scrie coordonatele dreptunghiului ce mărginește formularul. Apoi, dăm acest dreptunghi (adresa lui) ca argument funcției ClipCursor, care va imobiliza cursorul mouse-ului în interiorul formularului.

4. În corpul procedurii de tratare a evenimentului OnClick al butonului Eliberează, scriem:
5. Trecem în modul Form View și testăm funcționarea celor două butoane. Atunci când cursorul mouse-ului este capturat, avem acces la bara de titlu a formularului (deci și la meniul sistem și la butoanele sistem ale acestuia). Dacă nu dorim acest lucru, putem micșora corespunzător dreptunghiul în care se poate mișca mouse-ul (prin intermediul elementelor variabilei RectForm).

Partea 2.

1. Să se aplice toate operațiile anterioare pe baza de date proprie.