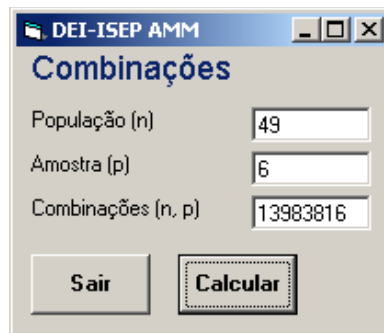


Ficha nº 3 – Subrotinas e Funções

1. Elabore uma função que calcule o factorial de um número inteiro. Utilizando a função, elabore um programa para cálculo de combinações, apresentando o respectivo formulário e identificando todos os objectos. $[C(n,p) = n!/((n-p)!p!)]$



Option Explicit

```
Private Sub cmdCalc_Click()  
    Dim n As Integer, p As Integer  
    n = Abs(Val(txtN.Text))  
    p = Abs(Val(txtP.Text))  
    If n > 0 And p > 0 And n >= p Then  
        txtComb.Text = fact(n) / fact(n - p) / fact(p)  
    Else  
        txtComb.Text = ""  
        MsgBox "Erro: dados inválidos!", vbOKOnly  
    End If  
End Sub
```

```
Private Sub cmdSair_Click()  
End  
End Sub
```

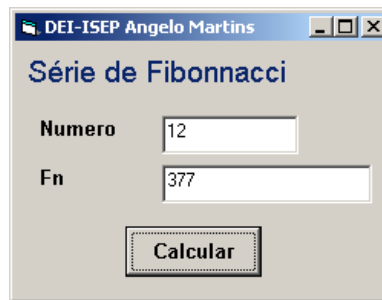
```
Function fact(ByVal n As Integer) As Double
```

```
    Dim x As Integer, f As Double  
    f = 1  
    For x = 2 To n  
        f = f * x  
    Next  
    fact = f  
End Function
```

Apesar de o factorial ser um número inteiro, o seu valor cresce de tal forma que só consegue ser guardado num Double. Não se consegue calcular o factorial de números superiores a 170 (VB 6).

2. Elabore uma função que calcule o elemento de ordem n da sequência de Fibonacci. Utilizando a função, elabore um programa que leia um número inteiro e apresente o elemento correspondente da série.

$$f_n = f_{n-1} + f_{n-2}, f_1 = 1 \text{ e } f_0 = 1$$



Option Explicit

```
Private Sub cmdCalc_Click()  
    Dim num As Integer
```

```
    num = Val(txtN.Text)  
    If num >= 0 Then  
        txtFn.Text = fibo(num)  
    Else  
        MsgBox "Número inválido!"  
        txtFn.Text = ""  
    End If  
End Sub
```

```
Function fibo(ByVal n As Integer) As Double  
    Dim fn As Double, fn1 As Double, fn2 As Double  
    Dim x As Integer
```

```
    fn1 = 1  
    fn2 = 1  
    For x = 1 To n  
        fn = fn1 + fn2  
        fn2 = fn1  
        fn1 = fn  
    Next
```

```
    fibo = fn
```

Este é o mecanismo pelo qual a função retorna o seu resultado. A variável `fibo` também poderia ter sido usada nos cálculos.

```
End Function
```

3. Elabore um programa que permita ler um conjunto de notas e apresente a média das notas, o número e a percentagens de aprovações. O número de alunos é dado pelo utilizador e a leitura das notas deve ser efectuada por uma subrotina.

```
Option Explicit
Private Sub cmdLer_Click()
    Dim na As Integer, med As Single, nap As Integer

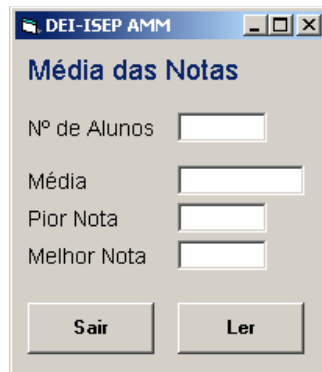
    na = Val(txtNA.Text)
    If na > 0 Then
        LerNotas na, med, nap
        txtMed.Text = med
        txtNap.Text = nap
        txtPAp.Text = nap / na * 100 & " %"
    Else
        MsgBox "Número de Alunos inválido!"
    End If
End Sub

Private Sub cmdSair_Click()
    End
End Sub

Sub LerNotas(ByVal n As Integer, med As Single, nap As Integer)
    Dim x As Integer, s As Long, nota As Integer

    For x = 1 To n
        Do
            nota = Val(InputBox("Nota n° " & x))
        Loop Until nota >= 0 And nota <= 20
        s = s + nota
        If nota >= 10 Then      ' conta o n° de aprovações
            nap = nap + 1
        End If
    Next
    med = s / n
End Sub
```

4. Elabore um programa que permita ler um conjunto de notas (entre 0 e 20) de alunos e apresente a média, a pior e a melhor nota. A leitura das notas deve ser efectuada por uma subrotina.



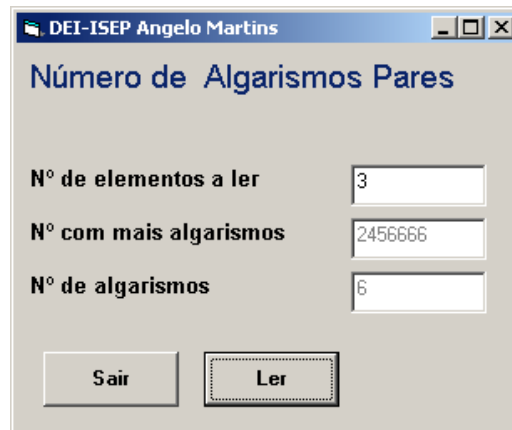
```
Private Sub cmdLer_Click()
    Dim na As Integer, med As Single, max As Integer, min As Integer

    na = Val(txtNA.Text)
    If na < 1 Then
        MsgBox "Nº de alunos inválido!"
        Exit Sub
    End If
    LerNotas na, med, max, min
    txtMed.Text = med
    txtMin.Text = min
    txtMax.Text = max
End Sub

Sub LerNotas(ByVal n As Integer, med As Single, melhor As Integer, _
    pior As Integer)
    Dim x As Integer, soma As Integer, nota As Integer

    pior = 21      ' valores iniciais
    melhor = -1
    For x = 1 To n
        nota = Val(InputBox("Insira a nota nº " & x))
        Do While nota < 0 Or nota > 20
            nota = Val(InputBox("Valor errado! Insira novamente a nota nº " & x))
        Loop
        soma = soma + nota
        If nota > melhor Then
            melhor = nota
        End If
        If nota < pior Then
            pior = nota
        End If
    Next
    med = soma / n
End Sub
```

5. Elabore uma subrotina que leia um conjunto de números inteiros (o número de elementos a ler dado como parâmetro da subrotina) e retorne o número com maior número de algarismos pares, bem como o respectivo número de algarismos. Elabore um programa para testar a subrotina e desenhe o respectivo formulário, identificando todos os objectos.



Option Explicit

```
Private Sub cmdLer_Click()
    Dim ne As Integer, nalg As Integer, max As Long

    ne = Abs(Val(txtNE.Text))
    If ne <= 0 Then
        MsgBox "Não há números a ler!"
        Exit Sub
    End If

    MaisPares ne, max, nalg

    txtMax.Text = max
    txtNumAlg.Text = nalg
End Sub
```

Chamada da subrotina. Note-se a ausência de parêntesis.

A subrotina tem três argumentos, o primeiro passado por valor e os restantes dois por referência. Só as alterações nestes dois parâmetros são visíveis no exterior da subrotina.

```
Sub MaisPares(ByVal n As Integer, maisP As Long, nalg As Integer)
    Dim x As Integer, num As Long, na As Integer

    nalg = 0
    maisP = 0
    For x = 1 To n
        num = Val(TextBox("Elemento nº " & x))

        na = nPares(num)

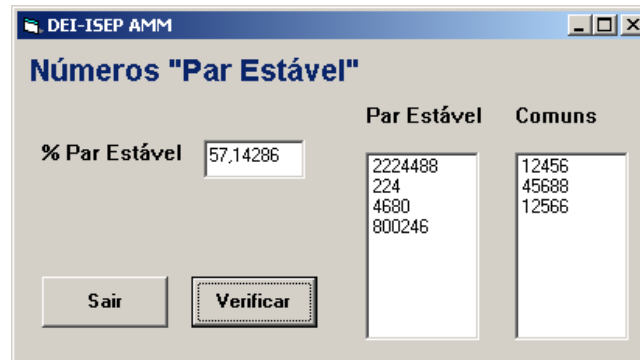
        If na > nalg Then
            nalg = na
            maisP = num
        End If
    Next
End Sub
```

Criou-se uma função que retorna o número de algarismos pares de um número

```
Function nPares(ByVal n As Long) As Integer
    Dim alg As Integer, np As Integer
    Do Until n = 0
        alg = n Mod 10
        n = n \ 10
        If alg Mod 2 = 0 Then
            np = np + 1
        End If
    Loop
    nPares = np
End Function

Private Sub cmdSair_Click()
    End
End Sub
```

6. Diz-se que o equivalente par de um número corresponde ao número que resulta da substituição de cada um dos algarismos ímpares do número pelo algarismo para imediatamente superior. Elabore uma função em Visual Basic que retorne o equivalente par de um número inteiro longo passado como parâmetro. Elabore ainda uma função que verifique se um número é “par estável”, isto é, se o equivalente par é o próprio número. Elabore um programa que leia uma sequência de valores, terminada por -99, e separe em duas ListBox os pares estáveis dos números comuns.



```
Option Explicit
Const Parar = -99 ' indica paragem na leitura

Private Sub cmdSair_Click()
    End
End Sub

Private Sub cmdVer_Click()
    Dim num As Long, c As Integer, npe As Integer
    lstPE.Clear
    lstC.Clear
    txtPPE.Text = ""
    Do
        Do
            num = Val(InputBox("Elemento n° " & (c + 1)))
        Loop Until num >= 0 Or num = Parar
        If num <> Parar Then
            If ParEstavel(num) Then
                npe = npe + 1
                lstPE.AddItem num
            Else
                lstC.AddItem num
            End If
            c = c + 1
        End If
    Loop Until num = Parar

    If c > 0 Then
        txtPPE.Text = npe / c * 100
    Else
        txtPPE.Text = "Não há valores"
    End If
End Sub

Function ParEstavel(ByVal n As Long) As Boolean
    Dim aux As Long

    aux = EqPar(n)
    If aux = n Then
        ParEstavel = True
    Else
        ParEstavel = False
    End If
End Function
```

```
Function EqPar(ByVal n As Long) As Long
    Dim alg As Integer, x As Integer, ep As Long

    x = 0
    Do While n <> 0
        alg = n Mod 10
        n = n \ 10
        If alg Mod 2 <> 0 Then
            alg = (alg + 1) Mod 10
        End If
        ep = ep + alg * 10 ^ x      ' constrói o número
        x = x + 1
    Loop
    EqPar = ep
End Function
```