Introdução à Computação

(Civil)

Luís Vieira Lobo

ISEP, 2005

Enunciados

1 - Suponha que pretende pavimentar com tijoleiras um terraço rectangular. Faça uma aplicação com um botão que calcule o seguinte:

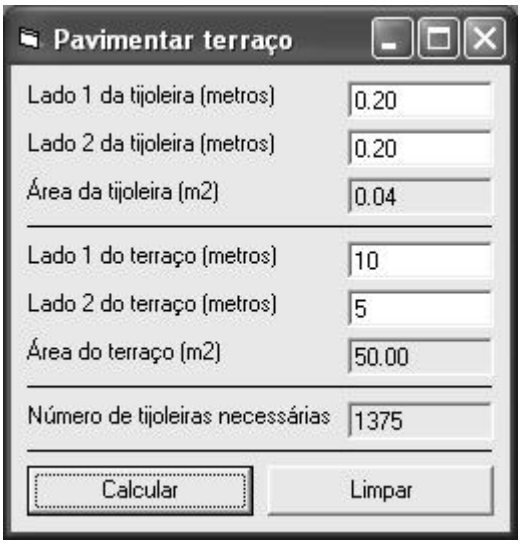
A área duma tijoleira, sendo dadas as medidas dos lados da tijoleira (metros);

A área do terraço, sendo dadas as medidas dos lados do terraço (metros);

O número de tijoleiras necessárias para pavimentar o terraço. Considere mais 10% de tijoleiras do que as estritamente necessárias, tendo em conta as perdas criadas nos ajustes dos lados do terraço.

Outro botão da aplicação deve poder limpar todas as caixas de texto de entrada e de saída.

Exemplo:



Resolução

2 - Suponha que pretende pintar interiormente uma habitação. Faça uma aplicação com um botão que calcule o seguinte:

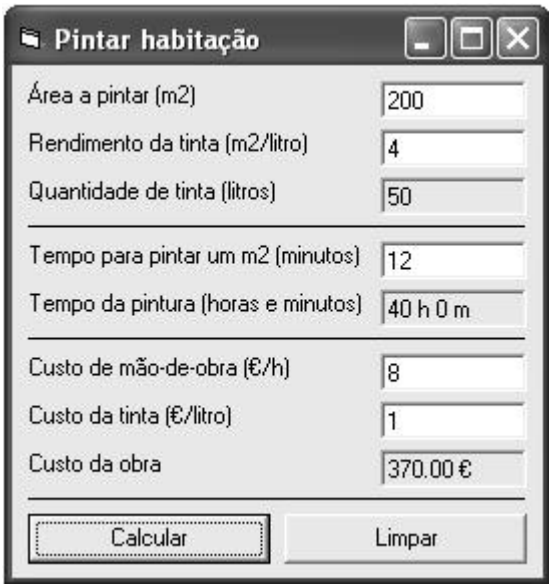
A quantidade de tinta necessária para pintar a habitação, sendo dada a respectiva área (m^2) e quantos metros quadrados é possível pintar com um litro de tinta;

O tempo (horas e minutos) que demora a execução da pintura, sendo dado o tempo (minutos) que demora a pintar um metro quadrado;

O custo total da obra, sendo dado o custo de mão-de-obra (euros/hora) e o preço da tinta (euros/litro).

Outro botão da aplicação deve poder limpar todas as caixas de texto de entrada e de saída.

Exemplo:



Resolução

3 - Suponha que, após ter construído a estrutura de um edifício (pilares, vigas e lajes) de *n* andares, pretende construir as paredes envolventes, as quais são constituídas por uma parede exterior, um isolamento e uma parede interior. Faça uma aplicação com um botão que calcule o seguinte:

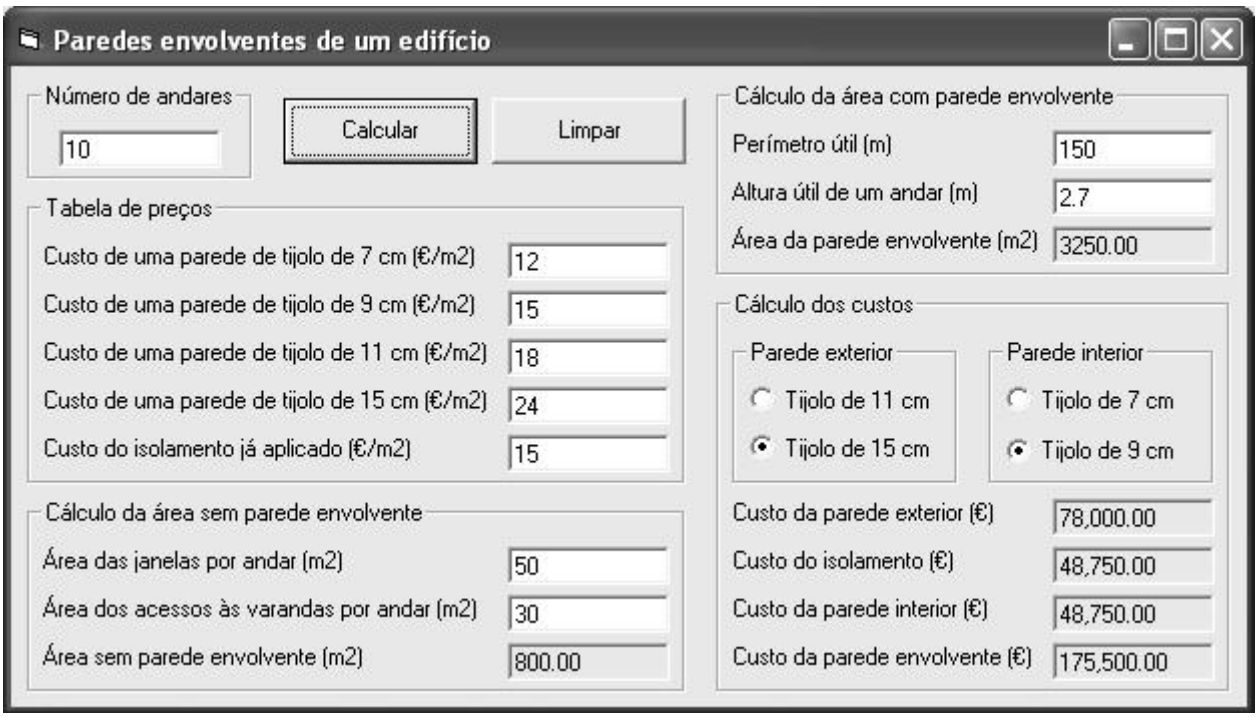
A área sem parede envolvente, ou seja, a área ocupada pelas janelas e pelo acesso às varandas em todo o edifício, sendo dadas as somas das áreas (em m^2) das janelas e dos acessos às das varandas por andar;

A área da parede envolvente, sendo dado, em metros, o perímetro útil (perímetro do edifício sem os pilares exteriores) e a altura útil de um andar (altura entre lajes). Repare que deve descontar a área calculada no parágrafo anterior;

O custo da parede envolvente. Para isso, deve calcular os custos da parede exterior (que pode ser em tijolo de 11cm ou de 15cm), da parede interior (que pode ser em tijolo de 7cm ou de 9cm) e do isolamento. Visualize também estes custos. Para efectuar estes últimos cálculos, atenda a uma tabela de preços que é introduzida pelo utilizador (ver exemplo).

Outro botão da aplicação deve poder limpar todas as caixas de texto de entrada e de saída, exceptuando as da tabela de preços. Desta forma, o utilizador pode carregar novamente no primeiro botão para outro edifício sem ter necessidade de voltar a introduzir a tabela de preços.

Exemplo:



Resolução

4 - Suponha que pretende pavimentar um passeio rectangular. Faça uma aplicação com um botão que calcule o seguinte:

A área do passeio, sendo dadas as respectivas dimensões (metros);

O número de trabalhadores necessários para a execução da obra de acordo com o critério expresso na seguinte tabela:

Critério	N. trabalhadores
$0 < \text{Área do passeio} < 140\ m^2$	1
$140\ m^2 \leq \text{Área do passeio} < 300\ m^2$	2
$300\ m^2 \leq \text{Área do passeio} < 500\ m^2$	3
$\text{Área do passeio} \geq 500\ m^2$	4

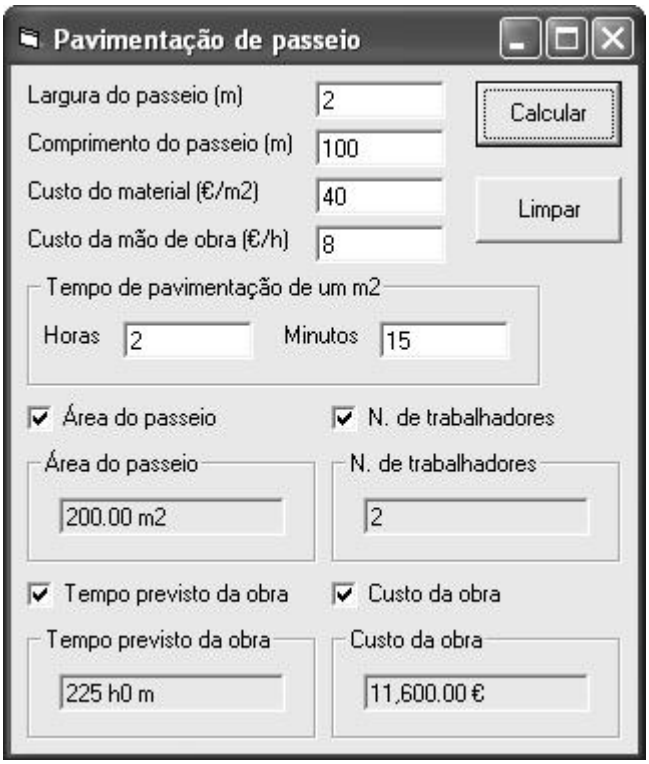
O tempo previsto da obra, sendo dado o tempo (horas) que cada trabalhador demora, em média, a pavimentar um metro quadrado de passeio;

O custo da obra, sendo dados os custos da mão-de-obra (euros/hora) e do material (euros/m^2).

Os valores calculados anteriormente devem ser visualizados ou escondidos consoante a selecção ou não de quatro caixas de verificação (uma para cada valor).

Outro botão da aplicação deve poder limpar todas as caixas de texto de entrada e de saída.

Exemplo:



Resolução

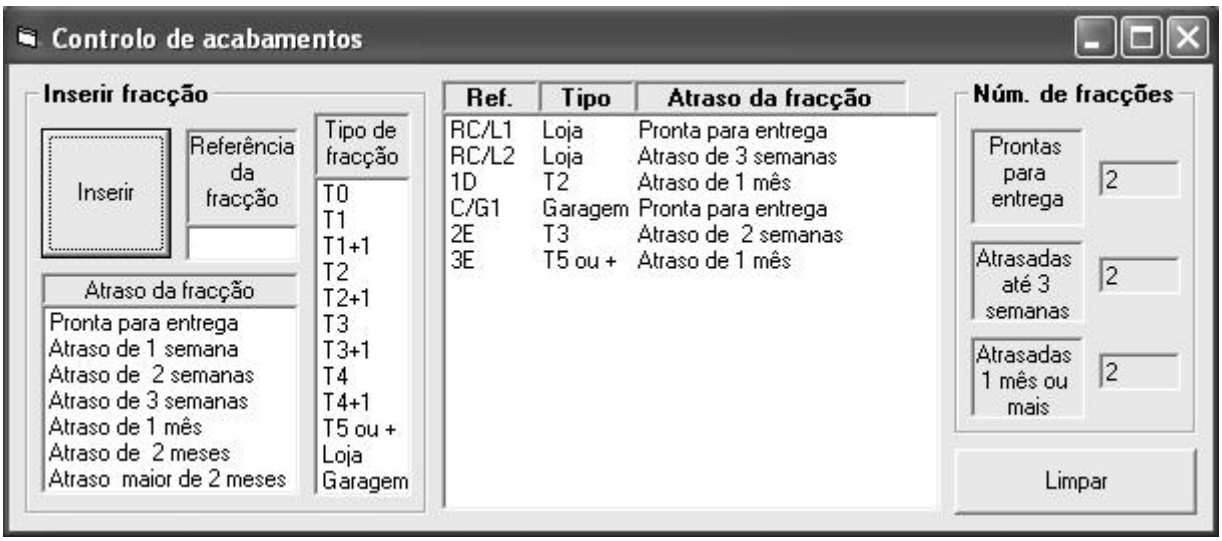
5 - Considere um edifício que se encontra já em fase de acabamentos. Elabore uma aplicação com um botão que execute o seguinte:

Inserir, numa caixa de listagem, uma fracção de cada vez, sendo dados a referência (através de uma caixa de texto), o tipo e o atraso no acabamento (seleccionados a partir de duas caixas de listagem). Os tipos e os atrasos no acabamento das fracções devem ser os existentes no exemplo abaixo. Se a referência, o tipo ou o atraso não forem especificados, devem surgir mensagens adequadas.

Calcular o número de fracções prontas para entrega, atrasadas até três semanas e atrasadas um mês ou mais.

Outro botão da aplicação deve poder limpar todas as caixas de texto de entrada e de saída e a caixa de listagem das fracções.

Exemplo:



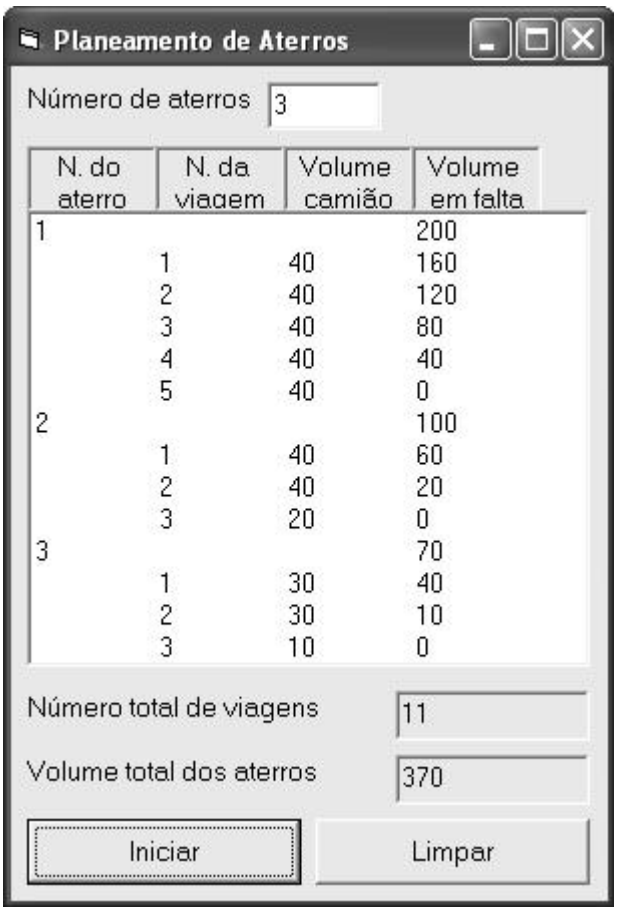
Resolução

6 - Uma empresa que faz aterros pretende uma aplicação para gerir a informação referente a um dado número de aterros. Um botão da aplicação deve executar o seguinte:

> Ler e tratar a informação referente a um dado número de aterros. Para cada aterro, deve ser visualizada, numa caixa de listagem, uma linha com o número e o volume do aterro, seguida de outras linhas onde constem, em cada uma delas, o número da viagem de transporte de terra para esse aterro, o volume do respectivo camião e o volume que ainda falta aterrar. A numeração dos aterros e das viagens segue a ordem dos números naturais e a numeração das viagens volta novamente a um sempre que se inicia um novo aterro.
>
> Após serem tratados todos os aterros, também deve ser indicado o número total de viagens efectuadas no transporte de terra para esses aterros e o volume total dos referidos aterros.

Outro botão da aplicação deve poder limpar todas as caixas de texto de entrada e de saída e a caixa de listagem.

Exemplo:



Resolução

7 - Uma empresa faz terraplanagens e pretende uma aplicação com um botão que faça o seguinte:

Apresentar, numa caixa de listagem, uma descrição dos serviços efectuados a vários clientes, onde deve constar, para cada cliente, uma linha com o respectivo nome, seguida das descrições das *n* terraplanagens realizadas a esse cliente. Nestas descrições, deve constar o seguinte:

Referência do terreno a terraplanar;

Área do terreno (m^2);

Custo da terraplanagem de um metro quadrado desse terreno (€/m^2);

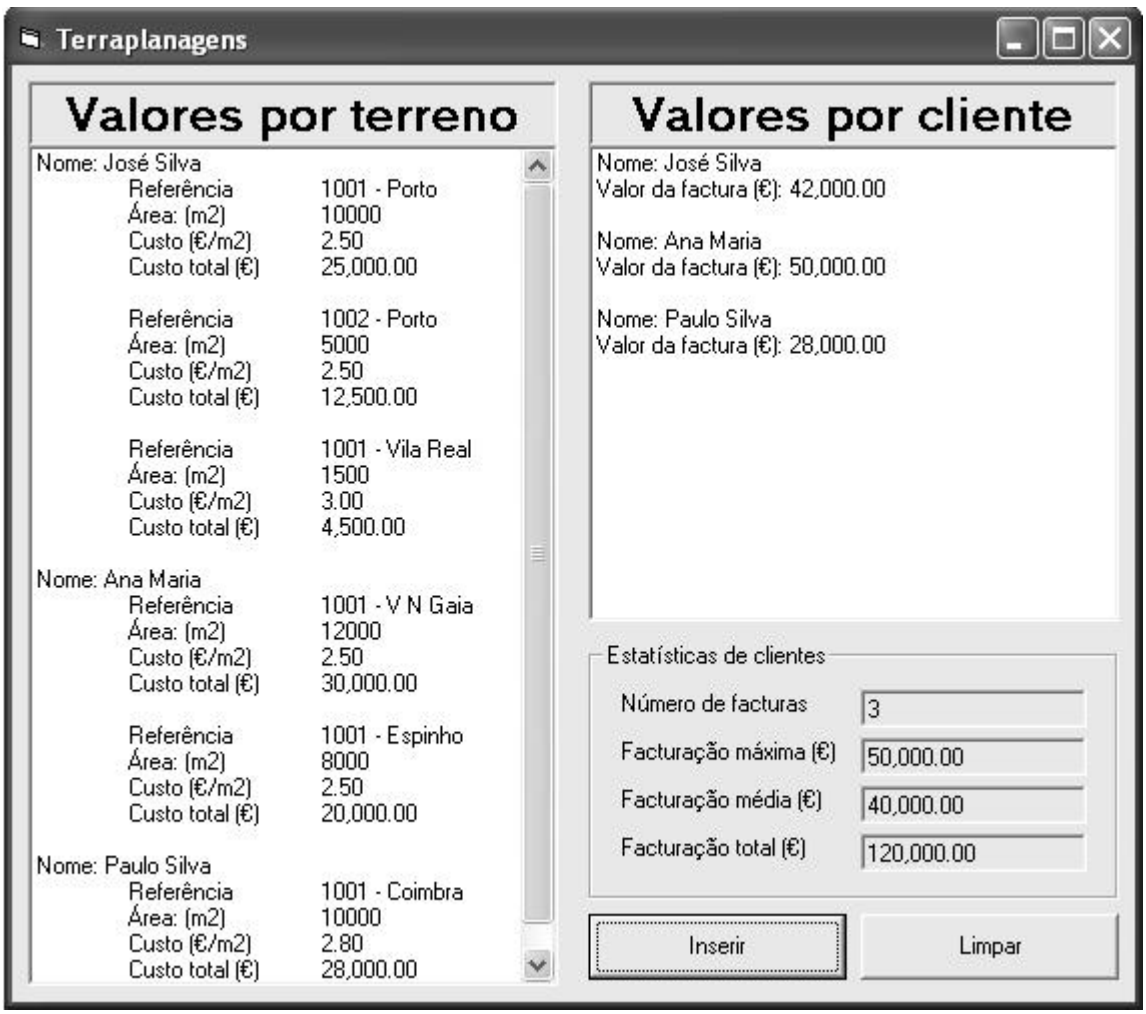
Custo da terraplanagem do terreno todo (€).

Apresentar, noutra caixa de listagem, o valor da factura por cliente.

Calcular o número de facturas e as facturações máxima, média e total relativamente ao conjunto dos clientes.

Outro botão da aplicação deve poder limpar as caixas de listagem e o resto das saídas. Neste caso, devem ser reinicializadas as variáveis (globais) necessárias ao cálculo dos valores pedidos no parágrafo anterior.

Exemplo:

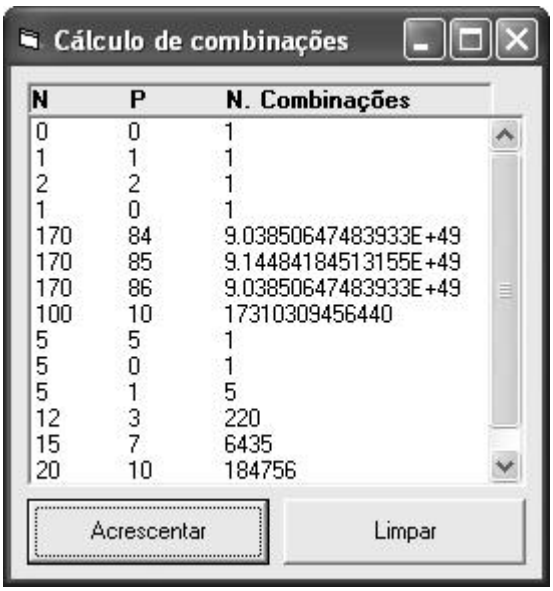


Resolução

8 - Faça uma aplicação com um botão que leia e acrescente a uma caixa de listagem uma sequência de pares de números (*n* e *p*) e os respectivos cálculos do número de combinações de *n*, *p* a *p*. A entrada de dados termina quando for feito "cancelar" na *inputbox* de leitura de *n* e, neste caso, o valor atribuído a *n* deve ser -1 (valor sentinela). Verifique se os valores introduzidos de *n* e *p* são numéricos, inteiros e não negativos. Verifique ainda se o valor de *n* não ultrapassa 170 e se o de *p* não ultrapassa o de *n*. Calcule os factoriais que necessitar em precisão dupla.

Outro botão da aplicação deve poder limpar a caixa de listagem.

Exemplo:



Resolução

9 - Faça uma aplicação com um botão que execute o seguinte:

> Ler e acrescentar a uma caixa de listagem uma sequência de números inteiros longos não negativos e as respectivas somas e produtos dos algarismos. A entrada de dados termina quando for feito "cancelar" na respectiva *inputbox* e, neste caso, o valor a atribuir como resultado da leitura deve ser -1 (valor sentinela). Verifique se cada valor introduzido é numérico, inteiro, não negativo e não superior ao limite máximo dos inteiros longos (2147483647);

Outro botão da aplicação deve poder limpar a caixa de listagem.

Exemplo:

Resolução

10 - Considere as notas a uma determinada disciplina dos n alunos $(1 \leq n \leq 50)$ de uma turma. Considere também que os alunos estão numerados 1 a n. Faça uma aplicação com um botão que execute o seguinte:

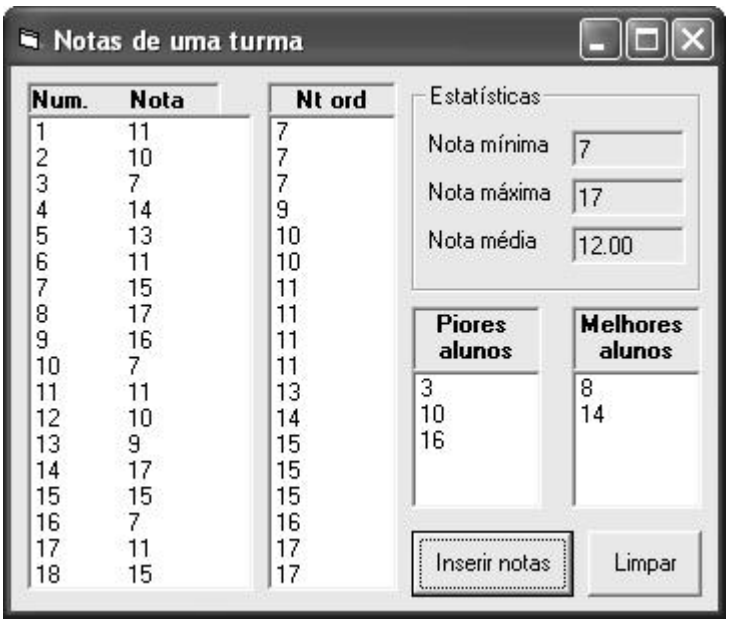
Ler para um vector as notas (inteiros de 0 a 20) dos alunos da turma numa disciplina. Visualizar, numa caixa de listagem, os números dos alunos e as respectivas notas. Fazer protecção de entrada de dados;

Determinar a nota mínima, a nota máxima, a nota média e os números do(s) prior(es) e do(s) melhor(es) alunos;

Visualizar, numa caixa de listagem, as notas ordenadas por ordem crescente (só as notas sem os números dos alunos).

Outro botão da aplicação deve poder limpar todos os dados de entrada e saída.

Exemplo:



Resolução

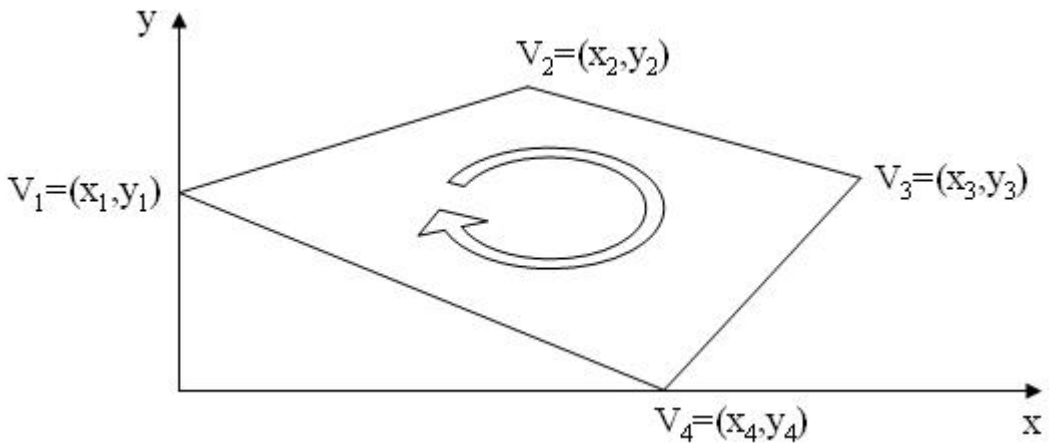
11 - Suponha que pretende medir a área de vários terrenos, sendo conhecidas as coordenadas dos n $(n \leq 20)$ vértices de cada um deles num determinado sistema de eixos. Faça uma aplicação com um botão que, para cada terreno, adicione a uma caixa de listagem o seguinte:

Uma referência para identificar cada terreno;

As coordenadas dos vértices do terreno;

A área do terreno.

Para calcular a área de um terreno, deverá atender à figura e à fórmula abaixo indicadas.



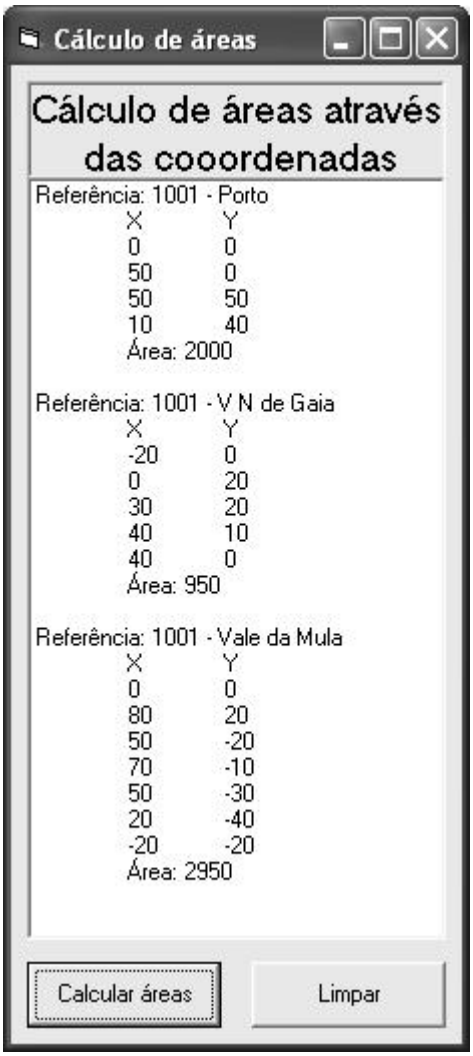
$$A_i = (x_i - x_j)\cdot(y_i + y_j) \quad \text{se } (i = 1 \Rightarrow j = n) \quad \text{e se } (i \in \{2,...,n\} \Rightarrow j = i - 1)$$

$$\text{Área do terreno} = \frac{1}{2}\left|\sum A_i\right|$$

Os vértices do terreno devem ser numerados segundo uma disposição circular, quer esta se efectue no sentido horário ou anti-horário. Use precisão dupla para os valores reais.

Outro botão da aplicação deverá poder limpar a referida caixa de listagem.

Exemplo:



Resolução

12 - Suponha que se pretende projectar uma nova rede de abastecimento de água para as n $(n \le 10)$ freguesias de um determinado concelho. Para isso, durante m $(m \le 50)$ dias registou-se o consumo diário de água de cada freguesia. Faça uma aplicação com um botão que execute o seguinte:

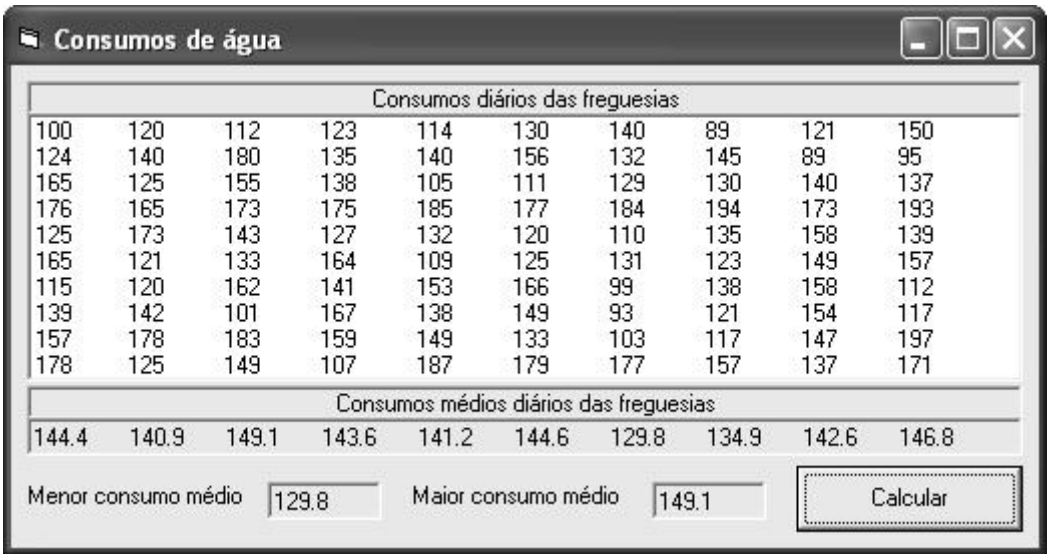
Ler para uma matriz os consumos diários de água (valores inteiros) de cada freguesia. Cada linha da matriz deverá conter os consumos diários de água de todas as freguesias;

Calcular os consumos médios diários de água (valores reais) de todas as freguesias;

Calcular o maior e o menor consumo médio diário de água de todas freguesias.

O botão deve ainda ter a funcionalidade de poder sair sem executar nada, se o utilizador assim o desejar.

Exemplo:



Resolução

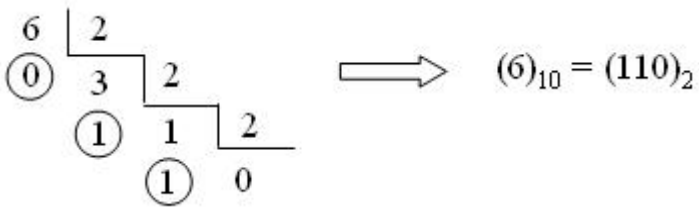
13 - Para bombear água de uma captação para um depósito a cota superior, dispõe-se de n bombas $(n \leq 8)$. Faça uma aplicação que contenha um botão que execute o seguinte:

Ler o volume do depósito (v) e os caudais debitados pelas bombas (vector c);

Gerar uma matriz h em que cada linha corresponde a uma das m hipóteses possíveis de funcionamento das bombas $(m = 2^n - 1)$ e cada coluna corresponde a uma bomba. Cada elemento da matriz será nulo se a respectiva bomba estiver desligada ou unitário se estiver em funcionamento. Então, para gerar a hipótese da linha i da matriz h, obtenha os algarismos de i na base 2 e coloque-os nessa linha, começando da direita para a esquerda. Por exemplo, se $n = 3$, então $m = 7$, sendo h a seguinte matriz:

$i=(1)_{10}$	$\Rightarrow$	$i=(001)_2$	$\Rightarrow$	0	0	1
$i=(2)_{10}$	$\Rightarrow$	$i=(010)_2$	$\Rightarrow$	0	1	0
$i=(3)_{10}$	$\Rightarrow$	$i=(011)_2$	$\Rightarrow$	0	1	1
$i=(4)_{10}$	$\Rightarrow$	$i=(100)_2$	$\Rightarrow$	1	0	0
$i=(5)_{10}$	$\Rightarrow$	$i=(101)_2$	$\Rightarrow$	1	0	1
$i=(6)_{10}$	$\Rightarrow$	$i=(110)_2$	$\Rightarrow$	1	1	0
$i=(7)_{10}$	$\Rightarrow$	$i=(111)_2$	$\Rightarrow$	1	1	1

Nota: Para passar os algarismos de um número para a base 2, atenda ao seguinte exemplo:



Calcular e visualizar o tempo de enchimento do depósito (em horas e minutos), para cada uma das hipóteses de funcionamento das bombas, utilizando a seguinte fórmula:

$$tempo = \frac{v}{\sum c_i \times h_{ij}} \quad (1 \le i \le m \quad e \quad 1 \le j \le n)$$

Exemplo:

Resolução

14 - Uma empresa que faz colocação de soalhos pretende uma aplicação com os seguintes botões:

Um botão que permita inserir os m $(m \leq 10)$ tipos de soalho com que a empresa trabalha numa caixa de listagem e noutra os respectivos preços (€/m^2). Guarde estas informações em vectores globais. A caixa de listagem dos tipos de soalho deve permitir uma selecção múltipla simples ou estendida. A inserção de informações de novos tipos de soalho só deve ser possível se forem removidas previamente as informações dos anteriores tipos de soalho. Este botão deve ainda colocar a zero, noutra caixa de listagem, as áreas totais gastas em todas as obras para cada tipo de soalho. Guarde também estas últimas informações num vector global.

Um botão que permita acrescentar a outra caixa de listagem um número identificativo, as áreas dos diversos tipos de soalho gasto e a área total de cada obra. As obras são numeradas a partir do número 1 e nunca ultrapassam as 50. Os tipos de soalho não utilizados na obra devem ter área igual a zero. Esta caixa de listagem deve estar programada para oito colunas visíveis e deve ser inserida uma obra em cada coluna. Para isso, deve dar à caixa de listagem uma altura apropriada. Para letra de tamanho 8, cada linha corresponde a uma altura de 195 e 60 é a altura mínima obrigatória. A barra de deslocamento horizontal, que surge quando o número de obras for superior a oito, vai exigir um acréscimo de 390 de altura. Antes de carregar neste botão, os tipos de soalhos devem estar previamente seleccionados na caixa de listagem correspondente. Se estes ainda não tiverem sido introduzidos ou se não tiver sido seleccionado nenhum deles, então deve ser visualizada uma mensagem adequada. Antes terminar o programa, deve desactivar a selecção referida anteriormente. Guarde as áreas dos diversos tipos de soalho gasto nas obras numa matriz global, na qual cada coluna corresponde a uma obra. Este botão deve ainda actualizar a caixa de listagem e o vector das áreas totais gastas em todas as obras para cada tipo de soalho.

Um botão que permita visualizar uma factura duma dada obra. Na factura, para cada soalho utilizado, deve constar o seguinte:

O nome do tipo de soalho;

O custo por metro quadrado;

A área de soalho gasto;

O custo total do soalho;

Para isso, construa previamente uma matriz de *strings* com a informação da factura e depois coloque-a numa caixa de listagem programada para quatro colunas (uma para cada um dos *itens* anteriores). Para programar a altura certa da caixa de listagem, atenda ao que foi referido na descrição do botão anterior;

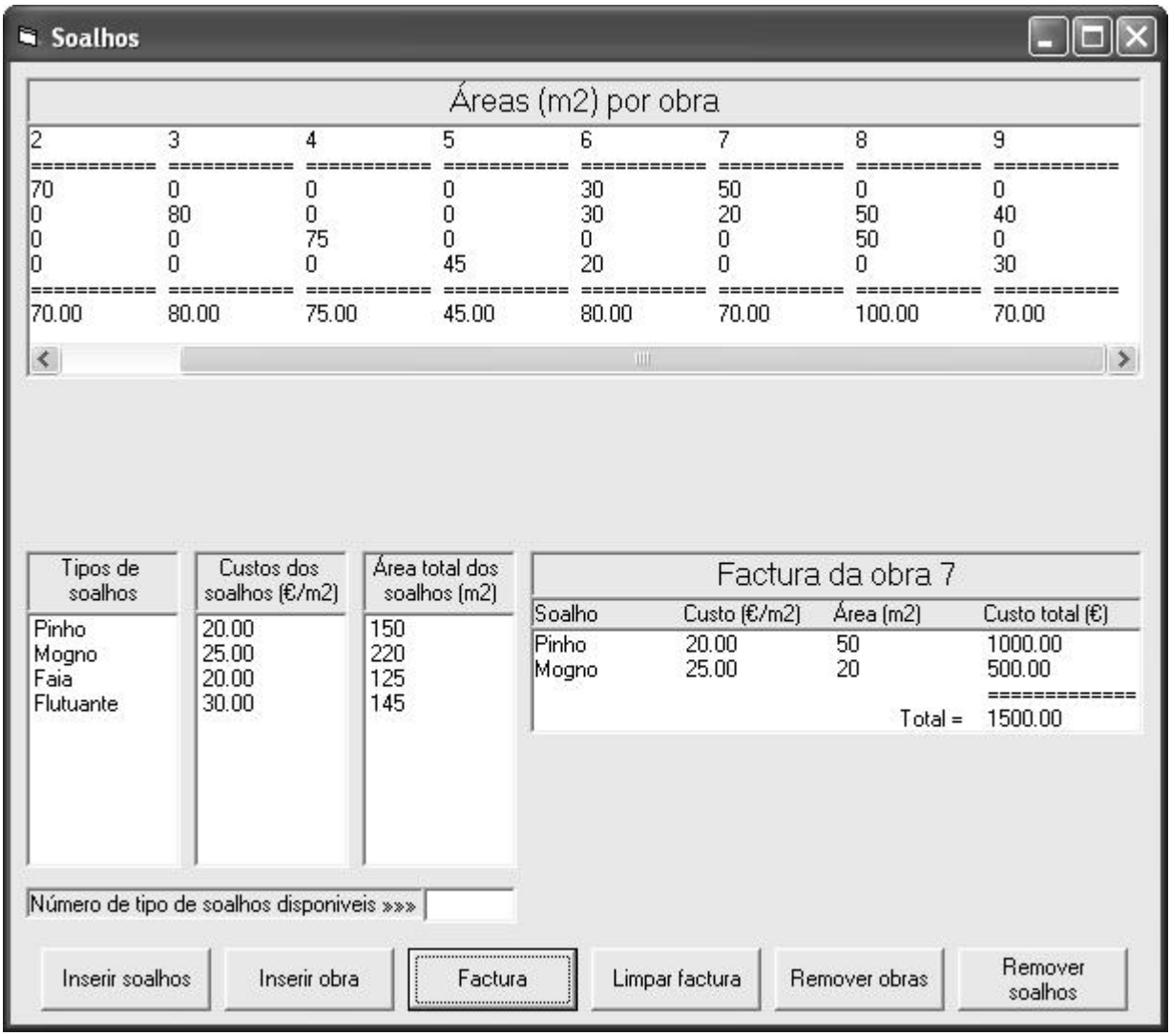
Um botão para limpar uma factura;

Um botão para remover as informações referentes às obras mediante uma confirmação do utilizador.

Um botão para remover as informações referentes aos tipos de soalhos, mediante uma confirmação do utilizador. Esta operação só pode ser efectuada se não existir nenhuma obra inserida.

Nota: Use precisão dupla para os valores reais.

Exemplo:



Resolução

Resoluções

1 - Resolução - Pavimentar terraço.

Enunciado

```
Option Explicit

Private Sub CmdCalc_Click()
    Dim ltj1 As Single, ltj2 As Single, ltr1 As Single, ltr2 As Single, _
    atj As Single, atr As Single
    atj = Val(TxtLTj1) * Val(TxtLTj2)
    atr = Val(TxtLTr1) * Val(TxtLTr2)
    TxtATj = Format(atj, "0.00"): TxtATr = Format(atr, "0.00")
    TxtNTj = Format(atr / atj * 1.1, "0")
End Sub

Private Sub CmdLimp_Click()
    TxtLTj1 = "": TxtLTj2 = "": TxtLTr1 = "": TxtLTr2 = ""
    TxtATj = "": TxtATr = "": TxtNTj = ""
End Sub
```

2 - Resolução - Pintar habitação.

Enunciado

```
Option Explicit

Private Sub CmdCalc_Click()
    Dim ap As Single, qt As Single, tm2 As Integer, tp As Long
    ap = Val(TxtAP): qt = Format(ap / Val(TxtRT), "0.00"): TxtQT = qt
    tm2 = Val(TxtTM2): tp = Round(ap * tm2)
    TxtTP = tp \ 60 & " h " & tp Mod 60 & " m"
    TxtCO = Format(qt * Val(TxtCT) + ap * tm2 / 60 * Val(TxtCMO), "0.00 €")
End Sub

Private Sub CmdLimp_Click()
    TxtAP = "": TxtRT = "": TxtQT = "": TxtTM2 = ""
    TxtTP = "": TxtCO = "": TxtCT = "": TxtCMO = ""
End Sub
```

3 - Resolução - Paredes envolventes dum edifício.

Enunciado

```
Option Explicit

Private Sub CmdCalc_Click()
    Dim na As Integer, asp As Single, ap As Single, _
    cpe As Single, ci As Single, cpi As Single
    na = Val(TxtNA)
    asp = na * (Val(TxtAJ) + Val(TxtAV)): TxtASP = Format(asp, "0.00")
    ap = na * Val(TxtPer) * Val(TxtA) - asp: TxtAP = Format(ap, "0.00")
    If OptT11 Then
        cpe = ap * Val(TxtT11)
    Else
        cpe = ap * Val(TxtT15)
    End If
    ci = ap * Val(TxtI)
    If OptT7 Then
        cpi = ap * Val(TxtT7)
    Else
        cpi = ap * Val(TxtT9)
    End If
    TxtCPE = Format(cpe, "#,##0.00"): TxtCI = Format(ci, "#,##0.00")
    TxtCPI = Format(cpi, "#,##0.00"): TxtCPEnv = Format(cpe + ci + cpi, "#,##0.00")
End Sub

Private Sub CmdLimp_Click()
    TxtNA = ""
    TxtAJ = "": TxtAV = "": TxtASP = ""
    TxtPer = "": TxtA = "":  TxtAP = ""
    TxtCPE = "": TxtCI = "": TxtCPI = "": TxtCPEnv = ""
End Sub
```

4 - Resolução - Pavimentação de passeio.

Enunciado

```
Option Explicit

Private Sub CmdCalc_Click()
    Dim ap As Single, nt As Integer, tt As Single, tpo As Single, _
    co As Single
    ap = Val(TxtCP) * Val(TxtLP)
    If ap <= 0 Then
        MsgBox "Medidas do passeio incorrectas", vbCritical, "Erro nos dados"
        Exit Sub
    ElseIf ap < 140 Then
        nt = 1
    ElseIf ap < 300 Then
        nt = 2
    ElseIf ap < 500 Then
        nt = 3
    Else
        nt = 4
    End If
    tt = ap * (Val(TxtTPavH) + Val(TxtTPavM) / 60): tpo = tt / nt
    co = ap * Val(TxtCM) + tt * Val(TxtCMO)
    TxtAP = Format(ap, "0.00 m2"): TxtNT = nt
    TxtTPO = Format(Int(tpo), "0 h") & Format((tpo - Int(tpo)) * 60, "0 m")
    TxtCO = Format(co, "#,##0.00 €")
End Sub

Private Sub ChkAP_Click()
    FraAP.Visible = ChkAP = 1
End Sub

Private Sub ChkNT_Click()
    FraNT.Visible = ChkNT = 1
End Sub

Private Sub ChkTPO_Click()
    FraTPO.Visible = ChkTPO = 1
End Sub

Private Sub ChkCO_Click()
    FraCO.Visible = ChkCO = 1
End Sub

Private Sub CmdLimp_Click()
    TxtCP = "": TxtLP = "": TxtCM = "": TxtCMO = "":
    TxtTPavH = "": TxtTPavM = "":
    TxtAP = "": TxtNT = "": TxtTPO = "": TxtCO = ""
End Sub
```

5 - **Resolução -** Controlo de acabamentos.

Enunciado

```
Option Explicit

Private Sub CmdInser_Click()
    Dim i As Integer, j As Integer, n As Integer
    If TxtRef = "" Then
        MsgBox "Referência vazia", vbCritical, "Falta de dados"
        Exit Sub
    End If
    n = LstTF.ListCount: i = 0
    Do Until LstTF.Selected(i)
        i = i + 1
        If i = n Then
            MsgBox "Não seleccionou o tipo de fracção", vbCritical, _
            "Falta de informação"
            Exit Sub
        End If
    Loop
    n = LstAF.ListCount: j = 0
    Do Until LstAF.Selected(j)
        j = j + 1
        If j = n Then
            MsgBox "Não seleccionou o atraso da fracção", vbCritical, _
            "Falta de informação"
            Exit Sub
        End If
    Loop
    LstF.AddItem TxtRef & vbTab & LstTF.List(i) & vbTab & LstAF.List(j)
    TxtRef = "": LstTF.Selected(i) = False: LstAF.Selected(j) = False
    If j = 0 Then
        TxtFP = Val(TxtFP) + 1
    ElseIf j < 4 Then
        TxtF3S = Val(TxtF3S) + 1
    Else
        TxtF1M = Val(TxtF1M) + 1
    End If
End Sub

Private Sub CmdLimp_Click()
    LstF.Clear
    TxtRef = "": TxtFP = "": TxtF3S = "": TxtF1M = ""
End Sub
```

6 - Resolução - Aterros.

Enunciado

```
Option Explicit

Private Sub CmdInic_Click()
    Dim na As Integer, a As Integer, va As Single, vta As Single, _
    vg As Integer, ntvg As Integer, vc As Single
    LstA.Clear: ntvg = 0: vta = 0
    na = Val(TxtNA)
    For a = 1 To na
        va = Val(InputBox("Volume do aterro " & a)): vta = vta + va
        LstA.AddItem a & vbTab & vbTab & vbTab & va
        vg = 0
        Do
            vg = vg + 1
            vc = Val(InputBox("Volume do camião (aterro " & a & ", viagem " & vg & ")"))
            If vc > va Then
                MsgBox "O camião só necessita de " & va & " m3": vc = va
            End If
            va = va - vc
            LstA.AddItem vbTab & vg & vbTab & vc & vbTab & va
        Loop Until va = 0
        ntvg = ntvg + vg
    Next
    TxtNTVg = ntvg: TxtVTA = vta
End Sub

Private Sub CmdLimp_Click()
    LstA.Clear: TxtNA = "": TxtNTVg = "": TxtVTA = ""
End Sub
```

7 - Resolução - Terraplanagens.

Enunciado

```
Option Explicit
Dim NFact As Integer, FactMax As Single, FactTot As Single

Private Sub Form_Load()
    NFact = 0: FactMax = 0: FactTot = 0
    ' Esta sub-rotina é activada quando se carrega no start e o formulário é
    ' carregado, mas é dispensável pois as variaveis NFact, FactMax e FactTot
    ' já estavam inicializadas a zero, mesmo antes da execução deste programa.
End Sub
```

```
Private Sub CmdInser_Click()
    Dim nome As String, FactClt As Single, n As Integer, i As Integer, _
    ref As String, a As Single, cm2 As Single, ct As Single
    nome = InputBox("Nome do cliente ou cancel para terminar", "Terraplanagens")
    Do While nome <> ""
        LstT.AddItem "Nome: " & nome
        FactClt = 0
        n = Val(InputBox("Número de terraplanagens", nome))
        For i = 1 To n
            ref = InputBox("Referência do terreno " & i, nome)
            a = Val(InputBox("Área (m2) do terreno " & ref, nome))
            cm2 = Val(InputBox("Custo da terraplanagem (€/m2) do terreno " & ref, nome))
            ct = a * cm2: FactClt = FactClt + ct
            LstT.AddItem vbTab & "Referência" & vbTab & ref
            LstT.AddItem vbTab & "Área: (m2)" & vbTab & a
            LstT.AddItem vbTab & "Custo (€/m2)" & vbTab & Format(cm2, "0.00")
            LstT.AddItem vbTab & "Custo total (€)" & vbTab & Format(ct, "#,##0.00")
            LstT.AddItem ""
        Next
        LstC.AddItem "Nome: " & nome
        LstC.AddItem "Valor da factura (€): " & Format(FactClt, "#,##0.00")
        LstC.AddItem ""
        NFact = NFact + 1
        If FactClt > FactMax Then FactMax = FactClt
        FactTot = FactTot + FactClt
        nome = InputBox("Nome do cliente ou cancel para terminar", "Terraplanagens")
    Loop
    If NFact > 0 Then
        TxtNFact = NFact
        TxtFactMax = Format(FactMax, "#,##0.00")
        TxtFactMed = Format(FactTot / NFact, "#,##0.00")
        TxtFactTot = Format(FactTot, "#,##0.00")
    End If
End Sub

Private Sub CmdLimp_Click()
    LstT.Clear: LstC.Clear
    TxtNFact = "": TxtFactMax = "": TxtFactMed = "": TxtFactTot = ""
    NFact = 0: FactMax = 0: FactTot = 0
    ' Neste caso, já não são dispensávies as inicializações anteriores.
End Sub
```

8 - Resolução - Combinações.

Enunciado

```
Option Explicit

Private Sub CmdAcresc_Click()
    Dim n As Integer, p As Integer
    n = LerNum("Valor de N ou cancelar para terminar", 170, True)
    Do While n > -1
        p = LerNum("Valor de P", n, False)
        LstResult.AddItem n & vbTab & p & vbTab & Fact(n) / Fact(p) / Fact(n - p)
        n = LerNum("Valor de N ou cancelar para terminar", 170, True)
    Loop
End Sub

Function LerNum(ByVal coment As String, ByVal max As Integer, _
ByVal CancelActiv As Boolean) As Integer
    Dim sair As Boolean, s As String, num As Double
    sair = False
    ' Inicialmente a variável sair contém o valor booleano False. Logo a instrução
    ' anterior podia ser dispensada.
    Do
        s = InputBox(coment, "Entrada de dados")
        If s = "" And CancelActiv Then
            LerNum = -1
            sair = True
        ElseIf Not IsNumeric(s) Then
            MsgBox "Deve introduzir um valor numérico", vbCritical, _
            "Erro nos dados"
        Else
            num = Val(s)
            If num <> Fix(num) Then
                MsgBox "Deve introduzir um valor inteiro", vbCritical, _
                "Erro nos dados"
            ElseIf num < 0 Then
                MsgBox "Deve introduzir um valor não negativo", vbCritical, _
                "Erro nos dados"
            ElseIf num > max Then
                MsgBox "Deve introduzir um valor não superior a " & max, _
                vbCritical, "Erro nos dados"
            Else
                LerNum = num
                sair = True
            End If
        End If
    Loop Until sair
End Function
```

```
Function Fact(ByVal n As Integer) As Double
    Dim f As Double, i As Integer
    f = 1
    For i = 2 To n
        f = f * i
    Next
    Fact = f
End Function

Private Sub CmdLimp_Click()
    LstResult.Clear
End Sub
```

9 - Resolução - Soma e produto de algarismos.

Enunciado

```
Option Explicit

Private Sub CmdAcrescNums_Click()
    Dim n As Long, soma As Integer, prod As Long, s As String
    n = LerNum()
    Do While n > -1
        Call SomaProdAlg(n, soma, prod)
        s = n & vbTab
        ' Na concatenação anterior (operados &) o valor de n é transformado em string
        ' antes de ser efectuada essa operação. No entanto a variável n continua a
        ' conter um valor inteiro longo.
        If n < 10000000# Then s = s & vbTab
        ' Se o número contém 7 algarismos ou menos então inserir mais um vbTab para
        ' acertar o alinhamento das colunas.
        LstNums.AddItem s & soma & vbTab & prod
        n = LerNum()
    Loop
End Sub
```

```
Function LerNum() As Long
    Dim sair As Boolean, s As String, num As Double
    sair = False
    ' Inicialmente a variável sair contém o valor booleano False. Logo a instrução
    ' anterior podia ser dispensada.
    Do
        s = InputBox("Introduza um inteiro não negativo ou cancel para sair", _
        "Entrada de dados")
        If s = "" Then
            LerNum = -1
            sair = True
        ElseIf Not IsNumeric(s) Then
            MsgBox "Deve introduzir um valor numérico", vbCritical, _
            "Erro nos dados"
        Else
            num = Val(s)
            If num <> Fix(num) Then
                MsgBox "Deve introduzir um valor inteiro", vbCritical, _
                "Erro nos dados"
            ElseIf num < 0 Then
                MsgBox "Deve introduzir um valor não negativo", vbCritical, _
                "Erro nos dados"
            ElseIf num > 2147483647 Then
                MsgBox "Deve introduzir um valor não superior a 2147483647", _
                vbCritical, "Erro nos dados"
            Else
                LerNum = num
                sair = True
            End If
        End If
    Loop Until sair
End Function

Sub SomaProdAlg(ByVal n As Long, ByRef soma As Integer, ByRef prod As Long)
    Dim alg As Integer
    soma = 0: prod = 1
    Do
        alg = n Mod 10
        soma = soma + alg
        prod = prod * alg
        n = n \ 10
    Loop Until n = 0
End Sub

Private Sub CmdLimp_Click()
    LstNums.Clear
End Sub
```

10 - Resolução - Notas de uma turma.
Enunciado

Versão 1 (Sem possibilidade de desistir no botão de inserção de notas)

```
Option Explicit
Option Base 1

Private Sub CmdInser_Click()
    Dim nt(50) As Integer, n As Integer
    Call Limpar
    n = LerInt("Número de notas", 1, 50)
    Call LerVisNt(nt, n)
    Call NtMinMaxMedPioresMelhores(nt, n)
    Call OrdenarVisNt(nt, n)
End Sub

Sub Limpar()
    LstN.Clear: LstNO.Clear: LstP.Clear: LstM.Clear
    TxtNMin = "": TxtNMax = "": TxtNMed = ""
End Sub

Function LerInt(ByVal coment As String, ByVal min As Integer, ByVal max As Integer) _
As Integer
    Dim sair As Boolean, s As String, num As Double
    sair = False
    ' Inicialmente a variável sair contém o valor booleano False. Logo a instrução
    ' anterior podia ser dispensada.
    Do
        s = InputBox(coment, "Entrada de dados")
        If Not IsNumeric(s) Then
            MsgBox "Não introduziu um número", vbCritical, "Erro nos dados"
        Else
            num = Val(s)
            If num <> Fix(num) Then
                MsgBox "Não introduziu um inteiro", vbCritical, "Erro nos dados"
            ElseIf num < min Then
                MsgBox "O valor não pode ser inferior a " & min, vbCritical, _
                "Erro nos dados"
            ElseIf num > max Then
                MsgBox "O valor não pode ser superior a " & max, vbCritical, _
                "Erro nos dados"
            Else
                LerInt = num
                sair = True
            End If
        End If
    Loop Until sair
End Function
```

```
Sub LerVisNt(ByRef nt() As Integer, ByVal n As Integer)
	Dim i As Integer
	For i = 1 To n
		nt(i) = LerInt("Nota do aluno " & i, 0, 20)
		LstN.AddItem i & vbTab & nt(i)
	Next
End Sub

Sub NtMinMaxMedPioresMelhores(ByRef nt() As Integer, ByVal n As Integer)
	Dim NtMin As Integer, NtMax As Integer, soma As Integer, i As Integer
	NtMin = nt(1): NtMax = nt(1): soma = nt(1)
	For i = 2 To n
		If nt(i) < NtMin Then
			NtMin = nt(i)
		ElseIf nt(i) > NtMax Then
			NtMax = nt(i)
		End If
		soma = soma + nt(i)
	Next
	TxtNMin = NtMin
	TxtNMax = NtMax
	TxtNMed = Format(soma / n, "0.00")
	' Repare que o valor da variável n nunca é zero.
	Call VisNumAlun(NtMin, nt, n, LstP)
	Call VisNumAlun(NtMax, nt, n, LstM)
End Sub

Sub VisNumAlun(ByVal nota As Integer, ByRef nt() As Integer, ByVal n As Integer, _
ByRef Lst As ListBox)
	Dim i As Integer
	For i = 1 To n
		If nt(i) = nota Then Lst.AddItem i
	Next
End Sub

Sub OrdenarVisNt(ByRef nt() As Integer, ByVal n As Integer)
	Dim i As Integer, j As Integer, k As Integer, aux As Integer
	For i = 1 To n - 1
		k = i
		For j = i + 1 To n
			If nt(j) < nt(k) Then k = j
		Next
		If k <> i Then aux = nt(i): nt(i) = nt(k): nt(k) = aux
		LstNO.AddItem nt(i)
	Next
	LstNO.AddItem nt(n)
End Sub

Private Sub CmdLimp_Click()
	Call Limpar
End Sub
```

Versão 2 (Com possibilidade de desistir no botão de inserção de notas)

```
Option Explicit
Option Base 1

Private Sub CmdInser_Click()
    Dim nt(50) As Integer, n As Integer
    n = LerInt("Número de notas", 1, 50, True)
    If n > 0 Then
        Call Limpar
        Call LerVisNt(nt, n)
        Call NtMinMaxMedPioresMelhores(nt, n)
        Call OrdenarVisNt(nt, n)
    End If
End Sub

Sub Limpar()
    LstN.Clear: LstNO.Clear: LstP.Clear: LstM.Clear
    TxtNMin = "": TxtNMax = "": TxtNMed = ""
End Sub

Function LerInt(ByVal coment As String, ByVal min As Integer, ByVal max As Integer, _
ByVal CancelActiv As Boolean) As Integer
    Dim sair As Boolean, s As String, num As Double
    sair = False
    ' Inicialmente a variável sair contém o valor booleano False. Logo a instrução
    ' anterior podia ser dispensada.
    Do
        s = InputBox(coment, "Entrada de dados")
        If s = "" And CancelActiv Then
            LerInt = 0
            sair = True
        ElseIf Not IsNumeric(s) Then
            MsgBox "Não introduziu um número", vbCritical, "Erro nos dados"
        Else
            num = Val(s)
            If num <> Fix(num) Then
                MsgBox "Não introduziu um inteiro", vbCritical, "Erro nos dados"
            ElseIf num < min Then
                MsgBox "O valor não pode ser inferior a " & min, vbCritical, _
                "Erro nos dados"
            ElseIf num > max Then
                MsgBox "O valor não pode ser superior a " & max, vbCritical, _
                "Erro nos dados"
            Else
                LerInt = num
                sair = True
            End If
        End If
    Loop Until sair
End Function
```

```
Sub LerVisNt(ByRef nt() As Integer, ByVal n As Integer)
	Dim i As Integer
	For i = 1 To n
		nt(i) = LerInt("Nota do aluno " & i, 0, 20, False)
		LstN.AddItem i & vbTab & nt(i)
	Next
End Sub

Sub NtMinMaxMedPioresMelhores(ByRef nt() As Integer, ByVal n As Integer)
	Dim NtMin As Integer, NtMax As Integer, soma As Integer, i As Integer
	NtMin = nt(1): NtMax = nt(1): soma = nt(1)
	For i = 2 To n
		If nt(i) < NtMin Then
			NtMin = nt(i)
		ElseIf nt(i) > NtMax Then
			NtMax = nt(i)
		End If
		soma = soma + nt(i)
	Next
	TxtNMin = NtMin
	TxtNMax = NtMax
	TxtNMed = Format(soma / n, "0.00")
	' Repare que o valor da variável n nunca é zero.
	Call VisNumAlun(NtMin, nt, n, LstP)
	Call VisNumAlun(NtMax, nt, n, LstM)
End Sub

Sub VisNumAlun(ByVal nota As Integer, ByRef nt() As Integer, ByVal n As Integer, _
ByRef Lst As ListBox)
	Dim i As Integer
	For i = 1 To n
		If nt(i) = nota Then Lst.AddItem i
	Next
End Sub

Sub OrdenarVisNt(ByRef nt() As Integer, ByVal n As Integer)
	Dim i As Integer, j As Integer, k As Integer, aux As Integer
	For i = 1 To n - 1
		k = i
		For j = i + 1 To n
			If nt(j) < nt(k) Then k = j
		Next
		If k <> i Then aux = nt(i): nt(i) = nt(k): nt(k) = aux
		LstNO.AddItem nt(i)
	Next
	LstNO.AddItem nt(n)
End Sub

Private Sub CmdLimp_Click()
	Call Limpar
End Sub
```

11 - Resolução - Área de terrenos

Enunciado

```
Option Explicit
Option Base 1
Const MaxV As Integer = 20

Private Sub CmdCalcAreas_Click()
    Dim ref As String, x(MaxV) As Double, y(MaxV) As Double, n As Integer
    ref = InputBox("Referência do terreno ou cancel para terminar", _
    "Novo terreno")
    Do While ref <> ""
        n = LerNumVertices(ref)
        Call LerVisDadosTerreno(ref, x, y, n)
        Call CalcArea(x, y, n)
        ref = InputBox("Referência do terreno ou cancel para terminar", _
        "Novo terreno")
    Loop
End Sub

Function LerNumVertices(ByVal titulo As String) As Integer
    Dim sair As Boolean, s As String, num As Double
    sair = False
    ' Inicialmente a variável sair contém o valor booleano False. Logo a instrução
    ' anterior podia ser dispensada.
    Do
        s = InputBox("Número de vértices do terreno", titulo)
        If Not IsNumeric(s) Then
            MsgBox "Não introduziu um número", vbCritical, "Erro nos dados"
        Else
            num = Val(s)
            If num <> Fix(num) Then
                MsgBox "Não introduziu um inteiro", vbCritical, "Erro nos dados"
            ElseIf num < 3 Then
                MsgBox "O número de vértices não pode ser inferior a 3", _
                vbCritical, "Erro nos dados"
            ElseIf num > MaxV Then
                MsgBox "O número de vértices não pode ser superior a " & MaxV, _
                vbCritical, "Erro nos dados"
            Else
                LerNumVertices = num
                sair = True
            End If
        End If
    Loop Until sair
End Function
```

```
Sub LerVisDadosTerreno(ByVal ref As String, ByRef x() As Double, _
ByRef y() As Double, ByVal n As Integer)
    Dim i As Integer
    LstT.AddItem "Referência: " & ref
    LstT.AddItem vbTab & "X" & vbTab & "Y"
    For i = 1 To n
        x(i) = LerOrdenada("Abcissa do vértice " & i, ref)
        y(i) = LerOrdenada("Ordenada do vértice " & i, ref)
        LstT.AddItem vbTab & x(i) & vbTab & y(i)
    Next
End Sub

Function LerOrdenada(ByVal coment As String, ByVal titulo As String) As Double
    Dim s As String
    s = InputBox(coment, titulo)
    Do Until IsNumeric(s)
        MsgBox "Deve Introduzir um valor numérico", vbCritical, "Erro nos dados"
        s = InputBox(coment, titulo)
    Loop
    LerOrdenada = Val(s)
End Function

Sub CalcArea(ByRef x() As Double, ByRef y() As Double, ByVal n As Integer)
    Dim i As Integer, soma As Double
    soma = Area(1, n, x, y)
    For i = 2 To n
        soma = soma + Area(i, i - 1, x, y)
    Next
    LstT.AddItem vbTab & "Área: " & Abs(soma) / 2
    LstT.AddItem ""
End Sub

Function Area(ByVal i As Integer, ByVal j As Integer, _
ByRef x() As Double, ByRef y() As Double) As Double
    Area = (x(i) - x(j)) * (y(i) + y(j))
End Function

Private Sub CmdLimp_Click()
    LstT.Clear
End Sub
```

12 - Resolução - Consumos de água

Enunciado

```
Option Explicit
Option Base 1
```

```
Private Sub CmdCalc_Click()
    Const MaxLin As Integer = 50, MaxCol As Integer = 10
    Dim c(MaxLin, MaxCol) As Integer, m As Integer, n As Integer, _
    cm(MaxCol) As Single, min As Single, max As Single
    m = LerInt("Número de dias ou cancel para desistir", 1, MaxLin, True)
    If m > 0 Then
        LstCons.Clear: TxtConsMed = "": TxtConsMin = "": TxtConsMax = ""
        n = LerInt("Número de freguesias", 1, MaxCol, False)
        Call LerVisCons(c, m, n)
        Call CalcVisConsMed(c, m, n, cm)
        Call CalcMenorMaiorConsMed(cm, n, min, max)
        TxtConsMin = Format(min, "0.0"): TxtConsMax = Format(max, "0.0")
    End If
End Sub

Function LerInt(ByVal msg As String, ByVal min As Integer, ByVal max As Integer, _
ByVal CancelActiv As Boolean) As Integer
    Dim sair As Boolean, s As String, num As Double
    sair = False
    ' Inicialmente a variável sair contém o valor booleano False. Logo a instrução
    ' anterior podia ser dispensada.
    Do
        s = InputBox(msg, "Entrada de dados")
        If s = "" And CancelActiv Then
            LerInt = 0
            sair = True
        ElseIf Not IsNumeric(s) Then
            MsgBox "Não introduziu um número", vbCritical, "Erro nos dados"
        Else
            num = Val(s)
            If num <> Fix(num) Then
                MsgBox "Não introduziu um inteiro", vbCritical, "Erro nos dados"
            ElseIf num < min Then
                MsgBox "O valor não pode ser inferior a " & min, vbCritical, _
                "Erro nos dados"
            ElseIf num > max Then
                MsgBox "O valor não pode ser superior a " & max, vbCritical, _
                "Erro nos dados"
            Else
                LerInt = num
                sair = True
            End If
        End If
    Loop Until sair
End Function
```

```
Sub LerVisCons(ByRef c() As Integer, ByVal m As Integer, ByVal n As Integer)
    Const MaxInt As Integer = 32767
    Dim i As Integer, j As Integer, linha As String
    For i = 1 To m
        linha = ""
        For j = 1 To n
            c(i, j) = LerInt("Consumo de água (m3) no dia " & i & _
            " na freguesia " & j, 0, MaxInt, False)
            linha = linha & c(i, j) & vbTab
        Next
        LstCons.AddItem linha
    Next
End Sub

Sub CalcVisConsMed(ByRef c() As Integer, ByVal m As Integer, ByVal n As Integer, _
ByRef cm() As Single)
    Dim i As Integer, j As Integer, s As Long
    For j = 1 To n
        s = 0
        For i = 1 To m
            s = s + c(i, j)
        Next
        cm(j) = s / m
        TxtConsMed = TxtConsMed & Format(cm(j), "0.0") & vbTab
        ' Previamente deverá ser atribuído o valor True à propriedade MultiLine
        ' da caixa de texto TxtConsMed para que os vbTab funcionem correctamente.
    Next
End Sub

Sub CalcMenorMaiorConsMed(ByRef cm() As Single, ByVal n As Integer, _
ByRef min As Single, ByRef max As Single)
    Dim i As Integer
    min = cm(1): max = cm(1)
    For i = 2 To n
        If cm(i) < min Then
            min = cm(i)
        ElseIf cm(i) > max Then
            max = cm(i)
        End If
    Next
End Sub
```

13 - Resolução - Depósito de água

Enunciado

```
Option Explicit
Option Base 1
Const MaxSingle As Single = 3.402823E+38
```

```
Private Sub CmdCalc_Click()
    Dim h(255, 8) As Integer, m As Integer, n As Integer, _
    c(8) As Single, v As Single, aux As Double
    ' Para o máximo de 8 bombas de água há 255 hipóteses de funcionamento.
    aux = Val(TxtNBomb)
    If aux < 1 Or aux > 8 Or aux <> Fix(aux) Then
        MsgBox "Número de bombas é inteiro e varia de 1 a 8", vbCritical, _
        "Erro nos dados"
    Else
        n = aux
        LstVolCaud.Clear: LstTemp.Clear: TxtCabec = ""
        Call LerVisVolCaud(v, c, n)
        m = 2 ^ n - 1
        Call GerarHipóteses(h, m, n)
        Call CalcVisTemp(h, m, n, c, v)
        TxtNBomb = ""
        ' A instrução anterior faz com que este programa não corra acidentalmente,
        ' pois obriga o utilizador a introduzir novamente o número de bombas para
        ' que isso seja possível.
    End If
End Sub

Sub LerVisVolCaud(ByRef v As Single, ByRef c() As Single, ByVal n As Integer)
    Dim i As Integer
    v = LerSinglePos("Volume (m3) do depósito de água")
    LstVolCaud.AddItem "Volume do depósito (m3)"
    LstVolCaud.AddItem v
    LstVolCaud.AddItem "Bomba" & vbTab & "Caudal (m3/h)"
    For i = 1 To n
        c(i) = LerSinglePos("Caudal (m3/h) da bomba " & i)
        LstVolCaud.AddItem i & vbTab & c(i)
    Next
End Sub
```

```
Function LerSinglePos(ByVal msg As String) As Single
	Dim sair As Boolean, s As String, num As Double
	sair = False
	' Inicialmente a variável sair contém o valor booleano False. Logo a instrução
	' anterior podia ser dispensada.
	Do
		s = InputBox(msg, "Entrada de dados")
		If Not IsNumeric(s) Then
			MsgBox "Não introduziu um número", vbCritical, "Erro nos dados"
		Else
			num = Val(s)
			If num <= 0 Then
				MsgBox "O valor deve ser positivo", vbCritical, "Erro nos dados"
			ElseIf num > MaxSingle Then
				MsgBox "O valor demasiado elevado", vbCritical, "Erro nos dados"
			Else
				LerSinglePos = num
				sair = True
			End If
		End If
	Loop Until sair
End Function

Sub GerarHipóteses(ByRef h() As Integer, ByVal m As Integer, ByVal n As Integer)
	Dim i As Integer, j As Integer, k As Integer
	For i = 1 To m
		k = i: j = n
		Do
			h(i, j) = k Mod 2
			j = j - 1
			k = k \ 2
		Loop Until k = 0
	Next
End Sub
```

```
Sub CalcVisTemp(ByRef h() As Integer, ByVal m As Integer, ByVal n As Integer, _
ByRef c() As Single, ByVal v As Single)
    Dim i As Integer, j As Integer, linha As String, ct As Single, t As Single
    linha = ""
    For j = 1 To n
        linha = linha & "B" & j & vbTab
    Next
    TxtCabec = linha & "Horas" & vbTab & "Min."
    For i = 1 To m
        linha = "": ct = 0
        For j = 1 To n
            linha = linha & h(i, j) & vbTab
            ct = ct + c(j) * h(i, j)
        Next
        t = v / ct
        LstTemp.AddItem linha & Format(Int(t), "0") & vbTab & _
        Format((t - Int(t)) * 60, "0")
    Next
End Sub
```

14 - Resolução - Soalhos.

Enunciado

Versão 1 (Sem proteger alguns dados)

```
Option Explicit
Option Base 0
Dim a(9, 49) As Double, m As Integer, n As Integer, _
ts(9) As String, cs(9) As Double, ats(9) As Double
' Quando carrega no start e a aplicação arranca, estas variavies globais são
' colocadas automaticamente a zero.
```

```
Private Sub CmdInserSoalhos_Click()
    Dim i As Integer, num As Double
    If m > 0 Then
        MsgBox "Remova primeiro os tipos de soalhos existentes", vbCritical, _
        "Acção não autorizada"
    Else
        num = Val(TxtNTS)
        If num <> Fix(num) Or num < 1 Or num > 10 Then
            MsgBox "O número de tipos de soalhos disponiveis deve ser um " & _
            "inteiro de 1 a 10", vbCritical, "Erro nos dados"
        Else
            m = num
            LstA.Height = 840 + m * 195
            ' 840 = 60 + 4 * 195
            ' Mínimo obrigatório (60) mais 4 linhas (4 *195) que contêm:
            '   Números das obras;
            '   Duas linhas de traços;
            '   Áreas totais das obras.
            For i = 0 To m - 1
                ts(i) = InputBox("Tipo de soalho " & i + 1, "Soalhos disponiveis")
                LstTS.AddItem ts(i)
                cs(i) = Val(InputBox("Custo (€/m2) do soalho " & ts(i), _
                "Soalhos disponiveis"))
                LstCS.AddItem Format(cs(i), "0.00")
            Next
            Call IniciarATS
            LstTS.ListIndex = -1
            ' Como LstTS é uma lista de selecção múltipla, LstTS.ListIndex seria
            ' igual a zero, mesmo sem haver nenhuma selecção. Futuramente, o uso
            ' da condição LstTS.ListIndex = -1 para verificar se há alguma selecção
            ' na lista estaria comprometido. Daí ser necessário fazer a atribuição
            ' LstTS.ListIndex = -1.
        End If
    End If
    TxtNTS = ""
End Sub

Sub IniciarATS()
    Dim i As Integer
    For i = 0 To m - 1
        ats(i) = 0
        LstATS.AddItem 0
    Next
End Sub
```

```
Private Sub CmdInserObra_Click()
    Dim i As Integer, s As Double
    If m = 0 Then
        MsgBox "Ainda não introduziu os soalhos disponivies", vbCritical, _
        "Falta de informação"
    ElseIf n = 50 Then
        MsgBox "Não é possivel introduzir mais obras", vbCritical, "Matriz cheia"
    ElseIf LstTS.ListIndex = -1 Then
        MsgBox "Não seleccionou nenhum soalho", vbCritical, "Falta de informação"
    Else
        s = 0
        For i = 0 To m - 1
            If LstTS.Selected(i) Then
                a(i, n) = Val(InputBox("Área (m2) do soalho " & ts(i), _
                "Obra " & n + 1))
                s = s + a(i, n)
                ats(i) = ats(i) + a(i, n)
                LstATS.List(i) = ats(i)
                LstTS.Selected(i) = False
            Else
                a(i, n) = 0
            End If
        Next
        LstTS.ListIndex = -1
        ' Como LstTS é uma lista de selecção múltipla, LstTS.ListIndex seria
        ' igual à posição da última selecção, mesmo sem haver nenhuma selecção.
        ' Futuramente, o uso da condição LstTS.ListIndex = -1 para verificar
        ' se há alguma selecção na lista estaria comprometido. Daí ser
        ' necessário fazer a atribuição LstTS.ListIndex = -1.
        LstA.AddItem n + 1
        LstA.AddItem "==========="
        For i = 0 To m - 1
            LstA.AddItem a(i, n)
        Next
        LstA.AddItem "==========="
        LstA.AddItem Format(s, "0.00")
        If n = LstA.Columns Then
            LstA.Height = LstA.Height + 390
            ' Aumento da altura em 390 devido a passar a existir a barra de
            ' deslocamento horizontal.
        End If
        n = n + 1
    End If
End Sub
```

```
Private Sub CmdFact_Click()
    Dim fact(11, 3) As String, i As Integer, j As Integer, k As Integer, _
    c As Double, s As Double
    j = LstA.ListIndex
    ' Neste caso, a lista LstA não é de selecção múltipla e a propriedade
    ' ListIndex reflecte correctamente se há ou não selecção na lista.
    If j = -1 Then
        MsgBox "Não seleccionou nenhuma obra", vbCritical, "Falta de informação"
    Else
        LstA.Selected(j) = False
        j = j \ (m + 4)
        ' Repare que a lista das obras tem m+4 linhas.
        LblFact = "Factura da obra " & j + 1
        LstFact.Clear: k = 0
        For i = 0 To m - 1
            If a(i, j) > 0 Then
                c = a(i, j) * cs(i): s = s + c
                fact(k, 0) = ts(i): fact(k, 1) = Format(cs(i), "0.00")
                fact(k, 2) = a(i, j): fact(k, 3) = Format(c, "0.00")
                k = k + 1
            End If
        Next
        fact(k, 3) = "============": k = k + 1
        fact(k, 2) = "                Total =": fact(k, 3) = Format(s, "0.00")
        LstFact.Height = 255 + k * 195
        ' LstFact.Height = 60 + (k + 1) * 195 = 60 + k * 195 + 195 = 255 + k * 195
        For j = 0 To 3
            For i = 0 To k
                LstFact.AddItem fact(i, j)
            Next
        Next
    End If
End Sub

Private Sub CmdLimpFact_Click()
    LstFact.Clear: LblFact = ""
End Sub

Private Sub CmdRemovObras_Click()
    If MsgBox("Tem a certeza?", vbYesNo, "Remover obras") = vbYes Then
        LstA.Clear: LstFact.Clear: LblFact = ""
        LstATS.Clear: Call IniciarATS
        n = 0
    End If
End Sub
```

```
Private Sub CmdRemovSoalhos_Click()
        If n > 0 Then
                MsgBox "Remova primeiro as obras", vbCritical, "Acção não autorizada"
        ElseIf MsgBox("Tem a certeza?", vbYesNo, "Remover soalhos") = vbYes Then
                LstTS.Clear: LstCS.Clear:  LstATS.Clear
                m = 0
        End If
End Sub
```

Versão 2 – (Com protecção de dados)

```
Option Explicit
Option Base 0
Dim a(9, 49) As Double, m As Integer, n As Integer, _
ts(9) As String, cs(9) As Double, ats(9) As Double
' Quando carrega no start e a aplicação arranca, estas variavies globais são
' colocadas automaticamente a zero.
```

```
Private Sub CmdInserSoalhos_Click()
    Dim i As Integer, num As Double
    If m > 0 Then
        MsgBox "Remova primeiro os tipos de soalhos existentes", vbCritical, _
        "Acção não autorizada"
    Else
        num = Val(TxtNTS)
        If num <> Fix(num) Or num < 1 Or num > 10 Then
            MsgBox "O número de tipos de soalhos disponiveis deve ser um " & _
            "inteiro de 1 a 10", vbCritical, "Erro nos dados"
        Else
            m = num
            LstA.Height = 840 + m * 195
            ' 840 = 60 + 4 * 195
            ' Mínimo obrigatório (60) mais 4 linhas (4 *195) que contêm:
            '   Números das obras;
            '   Duas linhas de traços;
            '   Áreas totais das obras.
            For i = 0 To m - 1
                ts(i) = Ler("Tipo de soalho " & i + 1, "Soalhos disponiveis", True)
                LstTS.AddItem ts(i)
                cs(i) = Val(Ler("Custo (€/m2) do soalho " & ts(i), _
                "Soalhos disponiveis", False))
                LstCS.AddItem Format(cs(i), "0.00")
            Next
            Call IniciarATS
            LstTS.ListIndex = -1
            ' Como LstTS é uma lista de selecção múltipla, LstTS.ListIndex seria
            ' igual a zero, mesmo sem haver nenhuma selecção. Futuramente, o uso
            ' da condição LstTS.ListIndex = -1 para verificar se há alguma selecção
            ' na lista estaria comprometido. Daí ser necessário fazer a atribuição
            ' LstTS.ListIndex = -1.
        End If
    End If
    TxtNTS = ""
End Sub
```

```
Function Ler(ByVal coment As String, ByVal titulo As String, _
ByVal LerStr As Boolean) As String
	Dim sair As Boolean, s As String
	sair = False
	' Inicialmente a variável sair contém o valor booleano False. Logo a instrução
	' anterior podia ser dispensada.
	Do
		s = InputBox(coment, titulo)
		If s = "" Then
			MsgBox "Introduza um valor não vazio", vbCritical, "Erro nos dados"
		ElseIf LerStr Then
			sair = True
			' As condições seguintes só são avaliadas quando é efectuada a leitura
			' de um string que futuramente vai dar origem a um valor numérico.
		ElseIf Not IsNumeric(s) Then
			MsgBox "Introduza um número", vbCritical, "Erro nos dados"
		ElseIf Val(s) <= 0 Then
			' Os valores numéricos que se pretende ler são áreas e custos
			' das obras. Logo esses valores têm de ser positivos.
			MsgBox "Introduza um número positico", vbCritical, "Erro nos dados"
		Else
			sair = True
		End If
	Loop Until sair
	Ler = s
End Function

Sub IniciarATS()
	Dim i As Integer
	For i = 0 To m - 1
		ats(i) = 0
		LstATS.AddItem 0
	Next
End Sub
```

```
Private Sub CmdInserObra_Click()
    Dim i As Integer, s As Double
    If m = 0 Then
        MsgBox "Ainda não introduziu os soalhos disponivies", vbCritical, _
        "Falta de informação"
    ElseIf n = 50 Then
        MsgBox "Não é possivel introduzir mais obras", vbCritical, "Matriz cheia"
    ElseIf LstTS.ListIndex = -1 Then
        MsgBox "Não seleccionou nenhum soalho", vbCritical, "Falta de informação"
    Else
        s = 0
        For i = 0 To m - 1
            If LstTS.Selected(i) Then
                a(i, n) = Val(Ler("Área (m2) do soalho " & ts(i), _
                "Obra " & n + 1, False))
                s = s + a(i, n)
                ats(i) = ats(i) + a(i, n)
                LstATS.List(i) = ats(i)
                LstTS.Selected(i) = False
            Else
                a(i, n) = 0
            End If
        Next
        LstTS.ListIndex = -1
        ' Como LstTS é uma lista de selecção múltipla, LstTS.ListIndex seria
        ' igual à posição da última selecção, mesmo sem haver nenhuma selecção.
        ' Futuramente, o uso da condição LstTS.ListIndex = -1 para verificar
        ' se há alguma selecção na lista estaria comprometido. Daí ser
        ' necessário fazer a atribuição LstTS.ListIndex = -1.
        LstA.AddItem n + 1
        LstA.AddItem "==========="
        For i = 0 To m - 1
            LstA.AddItem a(i, n)
        Next
        LstA.AddItem "==========="
        LstA.AddItem Format(s, "0.00")
        If n = LstA.Columns Then
            LstA.Height = LstA.Height + 390
            ' Aumento da altura em 390 devido a passar a existir a barra de
            ' deslocamento horizontal.
        End If
        n = n + 1
    End If
End Sub
```

```
Private Sub CmdFact_Click()
    Dim fact(11, 3) As String, i As Integer, j As Integer, k As Integer, _
    c As Double, s As Double
    j = LstA.ListIndex
    ' Neste caso, a lista LstA não é de selecção múltipla e a propriedade
    ' ListIndex reflecte correctamente se há ou não selecção na lista.
    If j = -1 Then
        MsgBox "Não seleccionou nenhuma obra", vbCritical, "Falta de informação"
    Else
        LstA.Selected(j) = False
        j = j \ (m + 4)
        ' Repare que a lista das obras tem m+4 linhas.
        LblFact = "Factura da obra " & j + 1
        LstFact.Clear: k = 0
        For i = 0 To m - 1
            If a(i, j) > 0 Then
                c = a(i, j) * cs(i): s = s + c
                fact(k, 0) = ts(i): fact(k, 1) = Format(cs(i), "0.00")
                fact(k, 2) = a(i, j): fact(k, 3) = Format(c, "0.00")
                k = k + 1
            End If
        Next
        fact(k, 3) = "============": k = k + 1
        fact(k, 2) = "                    Total =": fact(k, 3) = Format(s, "0.00")
        LstFact.Height = 255 + k * 195
        ' LstFact.Height = 60 + (k + 1) * 195 = 60 + k * 195 + 195 = 255 + k * 195
        For j = 0 To 3
            For i = 0 To k
                LstFact.AddItem fact(i, j)
            Next
        Next
    End If
End Sub

Private Sub CmdLimpFact_Click()
    LstFact.Clear: LblFact = ""
End Sub

Private Sub CmdRemovObras_Click()
    If MsgBox("Tem a certeza?", vbYesNo, "Remover obras") = vbYes Then
        LstA.Clear: LstFact.Clear: LblFact = ""
        LstATS.Clear: Call IniciarATS
        n = 0
    End If
End Sub
```

```
Private Sub CmdRemovSoalhos_Click()
    If n > 0 Then
        MsgBox "Remova primeiro as obras", vbCritical, "Acção não autorizada"
    ElseIf MsgBox("Tem a certeza?", vbYesNo, "Remover soalhos") = vbYes Then
        LstTS.Clear: LstCS.Clear:  LstATS.Clear
        m = 0
    End If
End Sub
```