



### Cuadernillo de examen

Asignatura:	<b>Interacción Hombre-Máquina / Interacción Persona-Computadora</b>	Código:	<b>208</b>
Titulación:	<b>Ingeniero en Informática / Ingeniero Técnico en Informática</b>	Curso:	<b>2º</b>
Especialidad:	<b>Común</b>	Plan de estudios:	<b>2000/2002</b>
Carácter:	<b>Optativa /Obligatoria</b>	Curso académico:	<b>2006/2007</b>
Convocatoria:	<b>Parcial Junio 2007</b>	Turno:	<b>Tarde</b>
Duración:	<b>Aproximadamente 2 horas y media</b>		

### Soluciones propuestas

## PRIMERA PARTE: INTERACCIÓN HOMBRE-MÁQUINA

Seleccione la opción correcta en el cuadernillo de examen. Sólo existe una opción verdadera, si considera que hay más de una, justifíquelo junto a la pregunta. Cada pregunta bien contestada valdrá 0,2 puntos. Cada opción mal contestada restará 0,1 puntos. Las preguntas no contestadas no restan puntos.

- La affordance...**
  - Es uno de los aspectos clave en los que debe sustentarse una interfaz de usuario.
  - Indica que los objetos deben sugerir por si mismos su utilidad y su forma de utilización.
  - Las dos respuestas son correctas.
- Un sistema usable...**
  - Es un sistema fácil de utilizar.
  - Es un sistema fácil de aprender.
  - Es un sistema fácil de utilizar y fácil de aprender.
- Los paradigmas de interacción...**
  - Son los determinados por las guías de estilo.
  - Hacen referencia a los modelos o hitos de los que se derivan todos los sistemas interactivos.
  - Agrupan las diferentes maneras en que los usuarios se comunican con el ordenador.
- Un agente...**
  - Es una de las formas de computación colaborativa.
  - Forma parte de otra aplicación y ayuda al usuario a realizar diversas tareas de la aplicación bajo petición.
  - Realiza las tareas por iniciativa propia, a partir de las características y preferencias de un usuario.
- Los objetivos de las “Reglas de oro de Mandel” son...**
  - Visibilidad del sistema, retroalimentación y un sistema de ayuda adecuado a las tareas del usuario.
  - Dar a los usuarios el control de la interfaz, reducir la carga de la memoria y realizar una interfaz de usuario consistente.
  - Facilidad de aprendizaje, flexibilidad, y solidez.
- Manteniendo las mismas técnicas de interacción a lo largo de la aplicación...**
  - Se suelen utilizar elementos del estilo de interacción *point-and-click*.
  - Se favorece el uso adecuado de los modos.
  - Se consigue aumentar la consistencia del sistema.
- La evaluación de un sistema en la fase final de desarrollo...**
  - Supone una dependencia demasiado grande a los algoritmos y las estructuras de datos utilizadas.
  - Es la mejor opción para conseguir un sistema centrado en el usuario.
  - Debe seguir siempre un método de evaluación por inspección.
- Frente a otros métodos de evaluación la evaluación heurística...**
  - Es más cara y precisa más tiempo de realización.
  - Puede utilizarse en cualquier fase de desarrollo.
  - No es adecuada para las fases finales de desarrollo.



9. El *cardsorting* ...

- Es un método de evaluación por inspección.
- Es un método de evaluación por indagación.
- Es un método de evaluación por test.

10. El diseño de sistemas interactivos supone añadir a la ingeniería de software clásica...

- La fase de evaluación y lanzamiento.
- La fase de prototipado y lanzamiento.
- Las fases de prototipado y evaluación.

Conteste brevemente las siguientes preguntas. Cada pregunta tendrá una puntuación máxima de 1 PUNTO

1. Características de la evaluación heurística. ¿A qué tipo de método de evaluación pertenece? ¿Por qué? ¿Quiénes intervienen en la evaluación? ¿Cómo y cuando se realiza este tipo de evaluación? ¿En qué aspectos de la interfaz deberán fijarse los evaluadores?

*Tema 5 de la documentación aportada por el profesor. Páginas 31-38*

2. Enumere y describa los distintos **estilos de interacción** que conozca. Si tomamos como ejemplo una aplicación ofimática de tratamiento de texto como Word... ¿qué estilos de interacción se observan en ella?

*Tema 2 de la documentación aportada por el profesor. Páginas 2-20.*

## SEGUNDA PARTE: LENGUAJE DE PROGRAMACIÓN VISUAL

### Parte teórica

1. Tipos de datos en VB.NET. Explique las diferencias entre los tipos de datos valor y los tipos de datos de referencia en VB.NET. Enumere y explique las características de los tipos de datos valor en VB.NET

*Tema 8 de la documentación aportada por el profesor. Páginas 7-19.*

2. El modelo de objetos de ADO.NET. ¿Qué es un proveedor de datos? Explique las características de los principales tipos de objetos que intervienen en el acceso a datos en .NET Framework.

*Tema 10 de la documentación aportada por el profesor. Páginas 5-7.*

**Puntuación: 0,75 puntos cada pregunta**

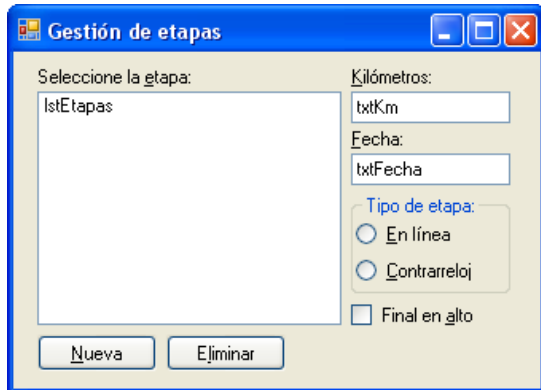
### Parte práctica

1. Se desea almacenar en un control `ListBox` una serie de etapas de una vuelta ciclista. Por cada etapa se almacenarán los siguientes datos.

- Número de etapa
- Fecha de la etapa
- Población de salida
- Población de llegada
- Kilómetros de la etapa
- Un valor lógico que indica si la etapa finaliza en lo alto de un puerto)
- Tipo de etapa (un carácter L si se trata de una etapa en línea o C si se trata de una etapa contrarreloj)

Los datos se almacenarán en memoria, sin necesidad de utilizar **ningún tipo de base de datos**.

Para gestionar el almacenamiento se utilizará un formulario con este formato:



**NOTAS IMPORTANTES:**

- Sólo existen los controles que aparecen en la imagen
- El control ListBox permite la selección múltiple

Se pide:

- a) Diseñe la estructura de datos necesaria para almacenar la información indicada en el control lstEtapas.  
**Puntuación: 0,25 puntos**

```
Structure Etapa
    Dim numEtapa As Integer
    Dim fecha As DateTime
    Dim salida As String
    Dim llegada As String
    Dim km As Integer
    Dim finalEnAlto As Boolean
    Dim tipoEtapa As String 'L, etapa en línea; C, contrarreloj

    Sub New(ByVal n As Integer, ByVal f As DateTime, ByVal s As String, _
        ByVal l As String, ByVal k As Integer, _
        ByVal final As Boolean, ByVal tipo As String)
        numEtapa = n
        fecha = f
        salida = s
        llegada = l
        km = k
        finalEnAlto = final
        tipoEtapa = tipo
    End Sub

    Public Overrides Function ToString() As String
        Return numEtapa & " - " & " de " & salida & " a " & llegada
    End Function
End Structure
```

- b) Cuando se seleccione un elemento del ListBox se cargarán en los controles de la derecha los datos correspondientes a la etapa seleccionada.  
**Puntuación: 0,5 puntos**

```
Private Sub lstEtapas_SelectedIndexChanged(
    ByVal sender As System.Object, _
    ByVal e As System.EventArgs)
    Handles lstEtapas.SelectedIndexChanged

    Dim et As Etapa = lstEtapas.SelectedItem
    txtKm.Text = et.km
    txtFecha.Text = et.fecha
    chkEnAlto.Checked = et.finalEnAlto
End Sub
```



```
    If et.tipoEtapa = "L" Then
        rbEnLinea.Checked = True
    Else
        rbContrarreloj.Checked = True
    End If
End Sub
```

- c) Al pulsar el botón Eliminar, se eliminarán de la lista **todos** los elementos seleccionados.

**Puntuación: 0,5 puntos**

```
Private Sub btnEliminar_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles btnEliminar.Click
    For i As Integer = lstEtapas.Items.Count - 1 To 0 Step -1
        If lstEtapas.GetSelected(i) Then
            lstEtapas.Items.RemoveAt(i)
        End If
    Next
End Sub
```

- d) Al pulsar el botón Nueva aparecerá el siguiente formulario modal (frmNuevaEtapa) y se añadirá una nueva etapa a partir de los datos que se introducirán en dicho formulario al pulsar el botón Aceptar. Si se pulsa el botón Cancelar no ocurrirá nada.

**Puntuación: 0,5 puntos**

```
Private Sub btnNueva_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles btnNueva.Click
    Dim frm As New frmNuevaEtapa
    If frm.ShowDialog() = Windows.Forms.DialogResult.OK Then
        lstEtapas.Items.Add(New Etapa(frm.txtNumEtapa.Text, _
            frm.txtFecha.Text, _
            frm.txtSalida.Text, _
            frm.txtLlegada.Text, _
            frm.txtKm.Text, _
            frm.chkEnAlto.Checked, _
            IIf(frm.rbEnLinea.Checked, "L", "C")))
    End If
End Sub
```

**En el formulario frmNuevaEtapa**

```
Private Sub btnCancelar_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles btnCancelar.Click
    Me.Close()
End Sub
Private Sub btnAceptar_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles btnAceptar.Click
    Me.DialogResult = Windows.Forms.DialogResult.OK
    Me.Close()
End Sub
```



- 2. Se está celebrando la XXIII Vuelta Ciclista a León 2007. La prueba consta de 5 etapas en línea. La información sobre la vuelta se almacena en un archivo de bases de datos Access (vuelta.mdb) que, entre otras, tiene las siguientes tablas:

Tabla Etapas (Guarda información de las distintas etapas de la vuelta)	
Nombre columna	Descripción del campo
numEtapa	Clave primaria. Un valor numérico entre 1 y 5
Salida	Población donde está ubicada la salida
Llegada	Población donde está ubicada la meta
Fecha	Dato de tipo fecha con la fecha de la etapa
Km	Kilómetros de la etapa

Tabla Corredores (Guarda información de los corredores que participan en la vuelta)	
Nombre columna	Descripción del campo
Dorsal	Clave primaria. Un dato numérico con el dorsal del corredor
Apellidos	Apellidos del corredor
Nombre	Nombre del corredor
Equipo	Equipo del corredor. Es una cadena de tres caracteres

Tabla Llegadas (Guarda una fila con los datos de cada corredor que ha llegado a la meta en cada etapa)	
Nombre columna	Descripción del campo
IdLlegada	Clave primaria. Valor autonumérico
Etapa	Número de la etapa de la que se trata
Dorsal	Dorsal del corredor que ha llegado
Tiempo	Tiempo efectuado por el corredor en recorrer la etapa

Tabla Clasificación (Guarda información de la clasificación general, sólo aparecen los corredores que no se han retirado)	
Nombre columna	Descripción del campo
Dorsal	Dorsal del corredor que ha llegado
Tiempo	Tiempo total del corredor en las etapas que se llevan disputadas

Se dispone también de un formulario con este formato:

chkRetirado

**NOTA IMPORTANTE: Sólo existen los controles que aparecen en la imagen**

Se pide:

- a) Cree todos los objetos y variables y realice todas las operaciones necesarias para conectarse a la base de datos.

**Puntuación: 0,5 puntos**



### Declaraciones de variables y objetos

```
Private cn As New SqlConnection
Private daEtapas As SqlDataAdapter
Private daCorredores As SqlDataAdapter
Private daLlegadas As SqlDataAdapter
Private daClasificación As SqlDataAdapter
Public ds As New DataSet
Private nombreBBDD = Application.StartupPath & "\\Vuelta.mdb"
```

### Instrucciones para conectarse a la base de datos

```
cn.ConnectionString = "PROVIDER=Microsoft.jet.oledb.4.0;" & _
    "Data Source=" & nombreBBDD
daEtapas = New SqlDataAdapter("SELECT * FROM Etapas", cn)
daCorredores = New SqlDataAdapter("SELECT * FROM Corredores", cn)
daLlegadas = New SqlDataAdapter("SELECT * FROM Llegadas", cn)
daClasificación = New SqlDataAdapter("SELECT * FROM Clasificacion", cn)
cn.Open()
```

```
'Rellenar el DataSet
daEtapas.Fill(ds, "Etapas")
daCorredores.Fill(ds, "Corredores")
daLlegadas.Fill(ds, "Llegadas")
daClasificación.Fill(ds, "Clasificacion")
```

```
cn.Close()
```

```
'Crear claves primarias
Dim pk(0) As DataColumn
pk(0) = New DataColumn()
pk(0) = ds.Tables("Etapas").Columns("NumEtapa")
ds.Tables("Etapas").PrimaryKey = pk
pk(0) = New DataColumn()
pk(0) = ds.Tables("Corredores").Columns("Dorsal")
ds.Tables("Corredores").PrimaryKey = pk
pk(0) = New DataColumn()
pk(0) = ds.Tables("Llegadas").Columns("IdLlegada")
ds.Tables("Llegadas").PrimaryKey = pk
pk(0) = New DataColumn()
pk(0) = ds.Tables("Clasificacion").Columns("Dorsal")
ds.Tables("Clasificacion").PrimaryKey = pk
```

```
'Crear órdenes de actualización
Dim cbLlegadas As SqlCommandBuilder = New SqlCommandBuilder(daLlegadas)
Dim cbClasificación As SqlCommandBuilder = _
    New SqlCommandBuilder(daClasificación)
```

- b) Al pulsar sobre el botón Buscar, se buscará la etapa en la tabla Etapas a partir del número de etapa introducido en el control txtEtapa. Si se encuentra, los datos de la misma aparecerán en los controles.

**Puntuación: 0,5 puntos**

```
Private Sub btnBuscar_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles btnBuscar.Click
    'Buscar la etapa
    Dim etapaEncontrada As DataRow = _
        ds.Tables("Etapas").Rows.Find(txtEtapa.Text)
    If etapaEncontrada Is Nothing Then
        MessageBox.Show("No existe esa etapa", Me.Text, _
            MessageBoxButtons.OK, _
            MessageBoxIcon.Information)
        txtSalida.Text = String.Empty
        txtLlegada.Text = String.Empty
    End If
End Sub
```



```
txtFecha.Text = String.Empty
txtKm.Text = String.Empty
Else
txtSalida.Text = etapaEncontrada.Item("Salida")
txtLlegada.Text = etapaEncontrada.Item("Llegada")
txtFecha.Text = etapaEncontrada.Item("Fecha")
txtKm.Text = etapaEncontrada.Item("km")
End If
End Sub
```

- c) En el ComboBox cmbrredores aparecerán todos los dorsales de los corredores que no se hayan retirado (los corredores que no se han retirado permanecen en la tabla Clasificación). Al seleccionar uno de ellos. Aparecerán los datos del corredor seleccionado en los controles correspondientes.

**Puntuación: 0,5 puntos**

```
Private Sub EnlazarDatos()
cmbCorredores.DataSource = ds.Tables("Corredores")
cmbCorredores.DisplayMember = "Dorsal"
txtApellidos.DataBindings.Add(
New Binding("Text", ds.Tables("Corredores"), "Apellidos"))
txtNombre.DataBindings.Add(
New Binding("Text", ds.Tables("Corredores"), "Nombre"))
txtEquipo.DataBindings.Add(
New Binding("Text", ds.Tables("Corredores"), "Equipo"))
End Sub
```

- d) Al pulsar sobre el botón Aceptar se realizarán las siguientes operaciones:
- Si el corredor se ha retirado, se eliminará a ese dorsal de la tabla Clasificación.
  - Si el corredor no se ha retirado, se añadirá una nueva fila en la tabla Llegadas con los datos de la etapa, el corredor y el tiempo que aparezcan en el formulario.
  - Si el corredor no se ha retirado también se sumará el tiempo obtenido en esta etapa al tiempo total de la tabla Clasificación.

**Puntuación: 1,25 puntos**

```
Private Sub btnAceptar_Click(ByVal sender As System.Object, _
ByVal e As System.EventArgs) Handles btnAceptar.Click
'Primero hay que comprobar si ya se ha introducido el tiempo
'de este corredor en esta etapa
Dim dr As DataRow() = ds.Tables("Llegadas").Select(
"Etapa=" & txtEtapa.Text & _
" AND Dorsal=" & cmbCorredores.Text)
If dr.Length <> 0 Then
MessageBox.Show("El corredor ya tiene tiempo en esta etapa", _
Me.Text, _
MessageBoxButtons.OK, MessageBoxIcon.Information)
Exit Sub
End If
If Not chkRetirado.Checked Then
'Hay que comprobar si el contenido de txtTiempo
'es un dato de tipo Hora
If Not IsDate(txtTiempo.Text) Then
MessageBox.Show(
"El valor del campo tiempo no es un dato válido", _
Me.Text, _
MessageBoxButtons.OK, MessageBoxIcon.Information)
Exit Sub
End If
'Se da un alta en la tabla llegadas
Dim nuevaFila As DataRow = ds.Tables("Llegadas").NewRow
'Se da un valor falso al dato autonumérico
nuevaFila.Item("idLlegada") = Integer.MaxValue
```



```
nuevaFila.Item("Etapa") = txtEtapa.Text
nuevaFila.Item("Dorsal") = cmbCorredores.Text
nuevaFila.Item("Tiempo") = txtTiempo.Text

ds.Tables("Llegadas").Rows.Add(nuevaFila)

'Se suma el tiempo efectuado al
'tiempo total de la tabla Clasificación
'Hay que buscar al corredor en la tabla de Clasificación
Dim corredorEncontrado As DataRow =
    ds.Tables("Clasificacion").Rows.Find(cmbCorredores.Text)
corredorEncontrado.Item("Tiempo") =
    corredorEncontrado.Item("Tiempo").add(
        New TimeSpan(CDate(txtTiempo.Text).TimeOfDay.Ticks))
Else
    'Si se ha retirado
    'Se busca al corredor en la tabla de clasificación
    Dim corredorEncontrado As DataRow =
        ds.Tables("Clasificacion").Rows.Find(cmbCorredores.Text)
    '...y se le borra
    corredorEncontrado.Delete()
End If
daLlegadas.Update(ds, "Llegadas")
daClasificación.Update(ds, "Clasificacion")
End Sub
```