PROGRAMACIÓN I 4-septiembre-2003

1. *(1.5 puntos)* Escribir las sentencias necesarias para calcular el siguiente producto:

$$1 \times 2 \times 4 \times 8 \times 16 \times \ldots \times 2^{20}$$

a. Mediante un bucle for.
b. Mediante un bucle while.

Solución:

a. Bucle for

```
double termino, producto = 1.0;
for (int i=0; i<=20; i++)
{
      termino = Math.pow(2,i);
      producto *= termino;
}
```

b. Bucle while

```
double termino, producto = 1.0;
int i=0;
while (i<=20)
{
      termino = Math.pow(2,i);
      producto *= termino;
      i++;
}
```

2. *(3.5 puntos)* El Ayuntamiento de Vitoria-Gasteiz encarga a una empresa de servicios informáticos que desarrolle una aplicación para gestionar las multas de tráfico sólo de los coches matriculados en Vitoria. Entre los datos de una multa que se deben grabar en una base de datos se encuentra la matrícula del vehículo. Pero antes de almacenarla se debe controlar que sea válida. Por ello se pide realizar un programa en Java que compruebe si la matrícula es o no correcta. El programa se debe codificar utilizando las clases del paquete javabook y teniendo en cuenta las siguientes condiciones:

- La matrícula tecleada debe ser de una longitud igual a 7 caracteres.
- Asimismo, puede corresponder bien al formato antiguo (p. e., VI2003P) o bien al formato europeo (p. e., 1234BCP). Vamos a suponer que los caracteres alfabéticos son introducidos en mayúsculas.
    - Una matrícula según el formato antiguo se considera válida si las dos primeras posiciones son "VI", las cuatro siguientes son numéricas y la última es un carácter alfabético distinto de vocal y de las letras 'Ñ'y 'Q'.
    - Una matrícula según el formato europeo se considera válida si las cuatro primeras posiciones son numéricas y las tres últimas son alfabéticas, pero que no contengan la letra 'Ñ', ni la 'Q' ni las vocales.
- Cuando la matrícula introducida no sea correcta se tiene que visualizar un mensaje de error y repetir su entrada.
- Hay que implementar dos clases denominadas MatriMain y Matri. El diseño de la clase Matri se deja al arbitrio del programador, pero debe ser coherente. Aún así, se valorará que la clase Matri tenga el menor número posible de atributos.
- Como mínimo se deben escribir tres métodos, uno para controlar la longitud y los otros dos para validar la matrícula en función del tipo de formato.
- El programa finaliza cuando la matrícula introducida ya no posea errores.

Solución:

```
class MatriMain
{
    public static void main (String[] args)
    {
        Matri mat = new Matri( );
        mat.empezar();
    }
}
import javabook.*;

class Matri
{
   /**********************
      Datos Miembro
   **********************/

   private MainWindow  ventanaPrincipal;
   private InputBox    cuadroEntrada;
   private OutputBox   cuadroSalida;
   private MessageBox  cuadroMensaje;
   private ResponseBox cuadroRespuesta;

   private String mat;
```

```
   /***********************
      Constructor
   ***********************/

   public Matri()
   {
      ventanaPrincipal       =  new  MainWindow("Validación  de
Matrículas");
      cuadroSalida       = new OutputBox(ventanaPrincipal);
      cuadroMensaje      = new MessageBox(ventanaPrincipal);
      cuadroRespuesta    = new ResponseBox(ventanaPrincipal);
      cuadroEntrada      = new InputBox(ventanaPrincipal);

      ventanaPrincipal.show();
      cuadroSalida.show();
   }

   /***********************
      Métodos Públicos
   ***********************/

   public void empezar ()
   {
      boolean valida = true;

      do
      {
         leerMatricula();
         valida = validarLongitud();
         if (valida)
            if (mat.substring(0,2).equals("VI"))
               valida = validarMatAntigua();
            else
               valida = validarMatEuropea();

      } while (valida == false);
   }

   /***********************
      Métodos Privados
   ***********************/

   private void leerMatricula()
   {
      mat = cuadroEntrada.getString("Introduzca la matrícula del
vehículo: ");
   }

   private boolean validarLongitud()
   {
      if (mat.length() != 7)
         cuadroMensaje.show ("La longitud de la matrícula debe
ser 7. Repetir ");
      return (mat.length() == 7);
   }
```

```
   private boolean validarMatAntigua()
   {
      boolean correcta1 = true, correcta2 = true;

      String numericaAnt = mat.substring(2, 6);
      correcta1 = validarNumerica(numericaAnt);
      if (!correcta1)
         cuadroMensaje.show("Las posiciones 3, 4, 5 y 6 deben
ser números. Repetir ");

      char ultPos = mat.charAt(6);
      correcta2 = validarLetra(ultPos);
      if (!correcta2)
      cuadroMensaje.show("La última letra no puede ser vocal ni
'Ñ' ni 'Q'. Repetir ");

      return (correcta1 && correcta2);
   }

   private boolean validarMatEuropea()
   {
      boolean correcta1 = true, correcta2 = true;

      String numericaEurop = mat.substring(0, 4);
      correcta1 = validarNumerica(numericaEurop);
      if (!correcta1)
         cuadroMensaje.show("Las cuatro primeras posiciones
deben ser numéricas. Repetir ");

       correcta2      =      validarLetra(mat.charAt(4))      &&
validarLetra(mat.charAt(5)) && validarLetra(mat.charAt(6));
       if (!correcta2)
          cuadroMensaje.show("Las tres últimas posiciones
alfabéticas no pueden contener vocales ni 'Ñ' ni 'Q'. Repetir
");

       return (correcta1 && correcta2);
   }

   private boolean validarNumerica(String num)
   {
      boolean correcta = true;
      int longitud = num.length();
      int i =0;

      while( i <= longitud-1 && correcta == true)
      {
         char aux = num.charAt(i);
         if ( aux < '0' || aux > '9')
             correcta = false;
         i++;
      }

      return correcta;
   }
```

```
    private boolean validarLetra (char letra)
    {
       return (letra != 'A' && letra != 'E' &&
               letra != 'I' && letra != 'O' && letra != 'U' &&
               letra != 'Ñ' && letra != 'Q' &&
               letra >= 'A' && letra <='Z');
    }

}
```