
Cuadernillo de prácticas

Laboratorio de Programación 1



Profesores:

Carlos Cervigón

Marco Antonio Gómez Martín

Borja Manero

Curso:

2008/2009

Departamento de Ingeniería del Software e Inteligencia
Artificial

Facultad de Informática

Universidad Complutense de Madrid

Este documento está preparado para ser imprimido a doble cara.

Normas de entrega

A continuación se detallan una serie de normas para la entrega de las prácticas.

1. Entrega de prácticas

- Las prácticas tendrán una *fecha límite* de entrega, indicada en el enunciado. No se admitirán prácticas entregadas fuera de dicho plazo.
- Las prácticas se entregarán a través del campus virtual, utilizando el mecanismo de entrega de prácticas. Únicamente un miembro del grupo de prácticas deberá realizar el envío. En caso de que ambos componentes efectúen el envío de la práctica, el profesor corregirá únicamente una de ellas.
- Se enviará un único fichero comprimido¹, llamado **grupoNN**, siendo NN el número del grupo al que pertenece (dos dígitos). El fichero contendrá una carpeta con nombre **grupoNN** en la que se incluirá código completo de la práctica y otros ficheros que fueran necesarios para ejecutar la práctica. Adicionalmente deberá incluir un fichero **alumnos.txt** donde se indique el nombre de los componentes del grupo.
- El primer día de clase de laboratorio después del final del plazo, se entregará una memoria que deberá incluir al menos:
 - Una portada que indique el número de la práctica y los componentes del grupo.
 - El código fuente completo de la práctica. Todas las líneas del código deberán ser legibles; se deberá evitar en lo posible las líneas partidas.

¹Para comprimir el fichero puede utilizarse cualquiera de los compresores disponibles en el laboratorio.

- Casos de prueba utilizados para probar que el programa o programas creados son correctos.
- Otras secciones concretas que se indiquen en el enunciado.
- El día de la entrega de la memoria se *defenderá* la práctica en el laboratorio con el profesor de prácticas. El profesor de prácticas puede decidir no corregir o penalizar en los términos que estime oportunos a los alumnos que no sigan las indicaciones del formato para la entrega.

2. Normas de corrección

- Un ejercicio que no compile contará como *No Apto*. El compilador utilizado será Turbo Pascal 7.0, si bien puede utilizarse Free Pascal utilizando la opción de compilar de acuerdo al lenguaje utilizado en Turbo Pascal.
- Un ejercicio que no funcione contará como *No Apto*.
- Se evaluará positivamente entre otras cosas:
 - Comentarios en el código, claridad e indentación.
 - Uso de estructuras de control y de datos adecuadas al problema.
 - Nombres de variables y constantes acordes a la función que desempeñan en el programa.
 - Eficiencia de los algoritmos utilizados.
- Se penalizará:
 - Mala estructuración de los programas: incorrecta descomposición en subprogramas, falta de anidamiento, etc.
 - Uso de variables globales o externas en funciones y procedimientos.
 - Paso de parámetros innecesarios a funciones y procedimientos.
 - Uso de estructuras de control o de datos no adecuadas al problema.

Nota: recuerda que para poderte presentar al examen es obligatorio tener todas las prácticas calificadas como *apta*².

²Consideramos una práctica calificada como *apta* si ha sido puntuada con 4 o más.

3. Comentarios en el programa

Los programas deben estar comentados y los identificadores utilizados deben ser descriptivos. Se puede ver un ejemplo del tipo de identificadores y comentarios que se esperan en el programa de la Práctica 1.

En particular todos los subprogramas contendrán una descripción sobre los parámetros y sobre las operaciones que realiza, mediante la descripción sobre lo que acepta (antes de ejecutar el subprograma) y lo que devuelve, modifica o muestra al ejecutar el subprograma.

Se seguirán las normas de estilo del programa de la Práctica 1: mayúsculas y minúsculas, espacios desde márgenes izquierdos (sangría), etc.

Práctica 1: Uso del IDE

Fecha de entrega: No debe entregarse

OBJETIVO: Toma de contacto con el entorno Turbo-Pascal

En esta primera práctica vamos a tomar contacto con el entorno que usaremos para realizar las prácticas de la asignatura Laboratorio de Programación 1.

Para el desarrollo de la práctica se puede utilizar cualquier entorno integrado de desarrollo (IDE), aunque el código fuente generado debe poder ser compilado con Turbo Pascal¹.

1. Primera parte

Crea los programas listados a continuación. Determina los errores sintácticos que se hayan cometido y corregirlos. Ejecuta los programas y comprueba que el funcionamiento es correcto. Desarrolla los siguientes pasos con los tres programas:

1. Introduce el programa en el editor.
2. Graba cada uno de los programas con los nombres `UnoA`, `UnoB` y `UnoC`.
3. Compila el programa (en Turbo Pascal, menú *Compile* - *Compile*). Corrige los errores.

¹Si se utiliza un compilador distinto, como FreePascal, éste puede configurarse para que admita únicamente código *compatible* con Turbo Pascal.

4. Ejecuta el programa (*Run - Run*) introduciendo números enteros con y sin signo, y verifica que el resultado es el deseado. Prueba a introducir valores erróneos como letras.
5. Graba el programa depurado (*File-Save*).

Fichero: UnoA.pas

```
{*****}
Program UnoA
Const
    NumEjecuciones = 3;
Var
    i, numero : Integer;
    sumaDeNumeros : Integer

Begin
    sumaDeNumeros := 0;
    Write('Introduzca 3 números positivos o negativos y ');
    Write('calcularé su suma: ');
    For X:=1 To NumEjecuciones Do
        Begin
            ReadLn(numero);
            sumaDeNumeros:=SumaDeNumeros + Numero;
        End;
    Writeln;
    Writeln('La suma de los números es = ',sumaDeNumeros);
End.
```

Fichero: UnoB.pas

```
{*****}
Program UnoB
Const
    NumEjecuciones = 3;
Var
    sumaDeNumeros : Integer

    {Este procedimiento se encarga de sumar tres numeros que
    se piden al usuario}
    {suma[OUT]: parametro de salida que almacena el valor de la
    suma resultante}
Procedure SumaTresNumeros(var suma:Integer);
Var
    i, num : Integer;

    { Este procedimiento se encarga de pedir datos al usuario}
    { numero[OUT]: almacena el valor que ha introducido el
    usuario }
Procedure LeerNumero ( numero :Ineger );
```

```

Begin
  Writeln('Introduzca un número : ');
  Readln(numero);
End;

Begin
  suma:=0;
  For i:=1 To NumEjecuciones Do
    Begin
      LeerNumero(num);
      suma:=Suma + num;
    End;
End;

  {Este procedimiento se encarga de mostrar los resultados
  de las operaciones al usuario}
  {resultado[IN]: valor que sera mostrado al usuario}
  Procedure MostrarResultados(resultado: Integer);

  Begin
    Writeln("*****");
    Writeln('La suma de los números es = ', Resultado);
    Writeln("*****");
  End;

Begin
  smaDeNumeros:=0;
  SumaTresNumeros(sumaDeNumeros);
  MostrarResultados(SumaDeNumeros);
End.

```

```

Fichero: UnoC.pas

```

```

{*****}
Program UnoC;
{Dibuja escalones o bloques en pantalla dependiendo de los }
{valores introducidos por teclado}

Var
  numPeldanios, anchura, altura : Integer;

  {Este procedimiento se encarga de pintar una escalera en
  la pantalla}
  {peldanios[IN]: numero de peldanios que tiene la escalera}
  {alto[IN]: alto de cada uno de los peldanios de la escalera}
  {ancho[IN]: acho de cada uno de los peldanios de la
               escalera }
  Procedure Escalera(peldanios ,alto ,ancho: integer);
  Var
    i: integer;

```

```

    {Este procedimiento se encarga de generar un peldanio en
      la pantalla}
    {cual[IN]: numero de peldanio que se va a pintar de la
      escalera}
    {alt[IN]: altura del peldanio}
    {anch[IN]: anchura del peldanio}
Procedure ladrillo (cual , alt , anch : integer) ;
Var
    i , j : integer ;
Begin
    For i := 1 To anch * cual Do write ( '-' ) ; Writeln ;
    For j := 1 To alt Do
        begin
            write ( '|' ) ;
            writeln ( '|' : anch * cual - 1 ) ;
        end ;
    For i := 1 To anch * cual do write ( '-' ) ; Writeln ;
End ;

Begin
    For i := 1 To peldanios Do
        ladrillo ( i , alto , ancho ) ;
End ;

Begin
    Writeln ( 'Introduzca numero de peldaños : ' ) ;
    Readln ( NumPeldanios ) ;
    Writeln ( 'Introduzca altura de cada peldaño : ' ) ;
    Readln ( Altura ) ;
    Writeln ( 'Introduzca anchura de cada peldaño : ' ) ;
    Readln ( anchura ) ;
    escalera ( NumPeldanios , Altura , Anchura ) ;
End .

```

2. Segunda parte

2.1. Conversión de grados celsius a Fahrenheit

Escribe un programa (estructurado al máximo) que convierta grados Celsius a grados Fahrenheit. El programa mostrará en forma de tabla *todas* las conversiones de temperatura entre -15°C y $+15^{\circ}\text{C}$ en intervalos de 0.5° .

La fórmula de conversión de grados Celsius (C) a grados Fahrenheit (F) es:

$$F = 9/5 * C + 32$$

2.2. Modificación de la escalera

Crea el programa `UnoCMejor`, que dibuje la misma escalera que el programa `UnoC` pero como si fueran escalones de subida. Por ejemplo, la salida para 3 peldaños de ancho 7 y altura 1 sería:

```
      -----
      |      |
      -----
    -----
    |      |
    -----
-----
|      |
-----
```
