Pascal

1) Elementos básicos del lenguaje:

- Identificadores: Se generan como en C o en el pseudocódigo.
- Cadenas: Las cadenas van entre el símbolo ', ejemplo: 'Hola mundo!!!'.
- Comentarios: { Comentarios… } (* Comentarios… *)
- Estructura:

```
Program NombePrograma;
Uses Unidad1, … , UnidadN;

Const
      Constante1 = Valor1;
      …
      Constante1 = Valor1;

Type
      Tipo1 = …;
      …
      TipoN = …;

Var
      Var1, … : Tipo;
      …
      VarN : Tipo;

{ Procedimientos y funciones }
…

Begin
      Sentencias;
End.
```

- Procedimientos y funciones:

```
Procedure identificador [ (parametros) ];
[ Var
      { Declaración de variables } ]
Begin
      { Cuerpo de la función }
End;

Function identificador [ (parametros) ]: Tipo;
[ Var
      { Declaración de variables } ]
Begin
      { Cuerpo de la función }
End;
```

2) Unidades:

- Unidades estándar:

Crt	Soporte de pantalla y teclado.
Dos, WinDos	Funciones de propósito general del DOS.
Graph, Graph3	Rutinas gráficas.
Overlay	Implementa el administrador de solapamientos.
Printer	Acceso a la impresora.
Strings	Tratamiento de cadenas terminadas en nulo.
System	Rutinas de la biblioteca en tiempo de ejecución.
Turbo3	Mantener compatibilidad con Turbo Pascal 3.0.

- Sintáxis de la unidad:

```
Unit NombreUnidad;

Interface { Parte pública }

      { Uses unidades }
      { Declaraciones de variables globales, constantes,
        funciones y subrutinas }

Implementation { Parte privada }

      { Uses unidades }
      { Declaraciones de variables globales, constantes }
      { Implementación de funciones y subrutinas }

Begin
      { Código de inicialización }
End.
```

3) Dispositivos:

CON	Terminal.
PRN	Impresora.
AUX	Dispositivo auxiliar.
LPT1	Impresora.
NUL	Dispositivo nulo.

4) Tipos de datos:

- Tipos simples: (Sección Var) Identificador1, … , IdentificadorN: Tipo;
  + Lógicos: Boolean. (*)
  + Enteros: Byte, Word, ShortInt, Integer, LongInt. (*)
  + Reales: Real, Single, Double, Extended, Comp.
  + Carácter: Char. (*)
  + Cadenas: String, String[tamaño].
  + Punteros: Pointer.

  (*) Nota: Todos esos tipos son ordinales.

- Tipos compuestos o complejos: (Sección Type)
  + Subrango: Tipo = ValorInicial..ValorFinal;
  + Enumerados: Tipo = (identificador1, … , identificadorN);

+ Array: Tipo = array[ValIni..ValFin, … , Tamaño] of Tipo;
{ Tamaño = 0..(Tamaño - 1) }
+ Registro: Tipo = record
Var1, … : Tipo;
…
end;
+ Objeto: Tipo = object
Var1, … : Tipo;
…
end;
+ Conjunto: Tipo = set of Tipo;
+ Ficheros:
- De texto: Tipo = Text; { = file of char }
- Con tipo: Tipo = file of Tipo;

5) Datos predefinidos en Pascal:

- Constantes: true, false, maxint.
- Variables: Input, Output, ExitCode, FileMode, InOutRes, RandSeed, StackLimit.

6) Operadores:

- Aritméticos: + - * / (real) div (entera) mod (resto de la división)
- Lógicos: not and or xor
- Relacionales: = <> < <= > >=
- De conjuntos: = <> < <= > >= in
- Dirección: @ (Dirección de memoria de una variable, función o subrutina)
- Concatenación: +
- Prioridad: Nivel 1 @ not + (unario) - (unario)
Nivel 2 * / div mod and shl shr
Nivel 3 + - or xor
Nivel 4 = <> < <= > >= in
- Asignación: :=

7) Sentencias:

- Asignación: identificador := expresión;

- Bloque:
```
begin
    sentencias;
end;
```

- Condicional if: (Nota: Delante de un else no se puede poner un ;)
```
if condición then
    bloque o sentencia
else
    bloque o sentencia;
```
- Condicional case:

```
case VariableOrdinal of
        lista de valores: { Val1, Val2..Val3, … , ValN }
                bloque o sentencia;
        …
        [ else
                bloque o sentencia; ]
        end;
```

- Bucle for:

```
for VarOrdinal := ValIni { to | downto } ValFin do
        bloque o sentencia;
```

- Bucle while:

```
while condición do
        bloque o sentencia;
```

- Bucle repeat:

```
repeat
        sentencias;
until condición;
```

8) Ámbito y localidad:

Son globales las variables que se encuentran en el bloque Var general, y locales las de los bloques Var de las funciones y subrutinas.

9) Funciones estandar: (Mirar también en la ayuda)

+ Aritméticas:

Abs(x);	|x|
Arctan(x: real): real;	ArcTan x
Cos(x: real): real;	Cos x
Exp(x: real): real;	e^x
Frac(x: real): real;	Parte decimal del número.
Int(x: real): real;	Parte entera del número.
Ln(x: real): real;	Ln x
Pi: real;	Pi = 3.14159
Sin(x: real): real;	Sen x
Sqr(x);	x^2
Sqrt(x: real): real;	$x^{1/2}$ = Raiz cuadrada de X.

+ De transformación:

Chr(x: byte): char;	Da el carácter correspondiente al código ascii.
Ord(x): longint;	Número ordinal de un tipo ordinal.
Round(x: real): longint;	Redondea un real a entero largo.
Trunc(x: real): longint;	Trunca un real a entero largo.

+ De flujo de control:

Break;	Sale del bucle en el que está.
Continue;	Salta a la siguiente iteración del bucle.

Exit;	Sale de la subrutina o función en la que está. (En la función "main", saldría del programa).
Halt **[**(error: word)**]**;	Sale del programa.
RunError **[**(error: byte)**]**;	Detiene el programa y genera un error.

+ Ordinales:

Dec(var x; **[**n: longint**]**);	Equivale a x-- o x -= n.
Inc(var x; **[**n: longint**]**);	Equivale a x++ o x += n.
High(x);	Da el último elemento de un array o un string.
Low(x);	Da el primer elemento de un array o un string.
Odd(x: longint): boolean;	Devuelve true si x es impar.
Pred(x);	Equivale a x - 1.
Succ(x);	Equivale a x + 1.

+ Manejo de cadenas:

Delete(var s: string; pos, len: integer);
Borra un chacho de la cadena.

Insert(var s: string; d: string; pos: integer);
Inserta una cadena en otra.

Str(i: integer, var s: string);
Str(r: real, var s: string);
Convierte un número a cadena.

Val(s: string; var i: integer; p: integer);
Val(s: string; var r: real; p: integer);
Convierte una cadena a número.
Si p = 0 entonces la transformación ha tenido éxito
sino da la posición del carácter en la cadena que lo impide.

Concat(s1, … , sN: string): string;
Concatena las cadenas indicadas.

Copy(s: string; pos, long: integer): string;
Da el fragmento indicado de la cadena.

Lenght(s: string): integer;
Da el tamaño de la cadena.

Pos(cad, patron: string; posini: integer): integer;
Posición de la 1ª ocurrencia en la cadena.

+ Punteros y direcciones:

Addr(x): pointer;	Sptr: word;	Assigned(var p): boolean;
CSeg: word;	DSeg: word;	SSeg: word;
Ofs(x): word;	Seg(x): word;	Ptr(seg, ofs: word): pointer;

+ Asignación dinámica:

Dispose(var p: pointer **[**, destructor**]**);	Elimina la memoria reservada.

FreeMem(var p: pointer; tamaño: word);Elimina la memoria reservada.
GetMem(var p: pointer; tamaño: word); Reserva una región de memoria.
New(var p: pointer; **[**, constructor**]**); Reserva una región de memoria.

+ Otras:

MaxAvail: longint; Tamaño del mayor bloque disponible en el "heap".
MemAvail: longint; Cantidad de memoria libre en el "heap".

BlockRead(var f: file: var buf; cuenta: word **[**; var resultado: word**]**);
BlockWrite(var f: file: var buf; cuenta: word **[**; var resultado: word**]**);

Close(var f: file); Cierra un archivo abierto.
Erase(var f: file); Borra un archivo externo.
FileSize(var f): longint; Tamaño actual de un archivo.
Flush(var f: text); Limpia el buffer de salida de un archivo.
Read(var f: file; v1, …, vN); Lee uno o más valores de un archivo.
ReadLn(var f: text; v1, …, vN);
Como Read, pero al finalizar se posiciona en el principio de la linea siguiente.

Rename(var f: file; nuevoNombre: string); Renombra un archivo.
Reset(var f: file **[**; tamreg: word**]**); Abre un archivo existente.
Rewrite(var f: file **[**; tamreg: word**]**); Crea y abre un archivo.
Seek(var f: file; n: longint);
Se mueve a la posición indicada en el archivo.

SetTextBuf(var f: text; var buf **[**; tamaño: word**]**);
Asigna un buffer de entrada/salida a un fichero de texto.

Truncate(var f: file); Trunca el fichero.
Write(var f: file; v1, …, vN); Escribe 1 o más valores en un archivo.
WriteLn(var f: text; v1, …, vN);
Como Write, pero al finalizar se posiciona en el principio de la linea siguiente.

Eof(var f: file): boolean; True si ha llegado al final del fichero.
Eoln(var f: text): boolean; True si ha llegado al final de la linea.
FilePos(var f: file): longint; Posición actual en el fichero.
IOResult: integer; Estado de la última operación de E/S hecha.

SeekEof**[**(var f: text)**]**: boolean;
SeekEoln**[**(var f: text)**]**: boolean;

Assing(var f: file; rutaFichero: string);
Indica cual es el fichero al que f hará referencia en futuras operaciones con ficheros.

Randomize; Inicializa el generador aleatorio de números.
Random **[**(x)**]**: integer; Genera un número aleatorio entre 0 y X-1.

10) Directivas de compilación:

{Define nombre}
{$ifdef nombre}
{$ifndef nombre}
{$ifopt conmutador}
{$else}
{$endif}
{$undef nombre}

11) Control de dispositivos:

+ Caracteres de control:

#7	BELL
#8	BS
#10	LF
#13	CR

+ Funciones y rutinas de la CRT:

KeyPressed;	Indica si se ha pulsado una tecla del teclado.
ReadKey;	Devuelve el carácter de la tecla pulsada.

WhereX;	WhereY;
InsLine;	LowVideo;
NormVideo;	NoSound;
Sound(HZ: word);	TextBackground(color: byte);
TextColor(color: byte);	TextMode(modo: word);
Window(x1, y1, x2, y2: byte);	AssignCrt(var f: text);
Delay(ms: word);	GotoXY(x, y: byte);
HighVideo;	

ClrEol;	Borra desde el cursor al final de la linea.
ClrScr;	Borra la pantalla entera.
DelLine;	Borra la linea actual del cursor.

12) Funciones exclusivas de Turbo Pascal:

Append, BlockRead, BlockWrite, ChDir, Clase, Erase, FilePos, FileSize, Flush, GetDir, MkDir, Rename, RmDir, Seek, SeekEof, SeekLn.

Addr, Concat, Copy, CSeg, DSeg, Dec, Delete, Exit, GetMem, Halt, Hi, Inc, Insert, Int, Length, Lo, Move, Ofs, ParamCount, ParamStr, Pi, Pos, Ptr, Random, Sptr, Seg, SizeOf, SSeg, Str, Swap, UpCase, Val, FillChar, Frac, Freemem, Mark, MaxAvail, MemAvail, Randomize, Release, RunError.

Apéndice de Ficheros:

Para abrir un fichero que podría existir o no, tenemos las siguientes directivas, que nos ayudarán a controlar en ejecución cualquier fallo:

```
{$I-}
Código para abrir el fichero…
{$I+}
```

Hecho eso solo tendremos que comprobar si IOResult es distinto de cero, para saber que ha habido algún error al abrir el fichero. Después las funciones que funcionan con todo tipo de ficheros son:

Assign(f, ruta);	ChDir(ruta);	Close(f); {Cierra fichero}
Erase(f);	GetDir(d, ruta);	MkDir(ruta);
Rename(f, nombre);	Reset(f); {Lectura}	ReWrite(f); {Escritura}
RmDir(ruta);	Read(f, v1, …);	Write(f, v1, …);
Eof(f); {¿Fin de fichero?}		

Luego tenemos las funciones para ficheros de texto:

Append(f);	Flush(f);	ReadLn(f, v1, …);
WriteLn(f, v1, …);	Eoln(f);	SetTextBuf(f, buff, tam);
SeekEof(f);	SeekEoln(f);	

Por último tenemos algunas funciones varias:

FilePos(f);	FileSize(f);
Seek(f, numReg);	Truncate(f);
BlockRead(f, buf, tam**[**, result**]**);	BlockWrite(f, buf, tam**[**, result**]**);

(Nota: "tam" equivale a un sizeof(buffer)).