Introducción a la Algorítmica y Programación (3300)

Prof. Ariel Ferreira Szpiniak - aferreira@exa.unrc.edu.ar
Departamento de Computación
Facultad de Cs. Exactas, Fco-Qcas y Naturales
Universidad Nacional de Río Cuarto

Teoría 8

Abstracción, Modularización, Units

Abstracción

Como hemos visto hasta ahora a los largo del curso, una de las herramienta fundamentales para construir programas es la **Abstracción**, que permite dividir un problema grande en pequeños problemas de más fácil solución. La solución del problema original es la composición de las pequeñas soluciones:

- **Abstraer**: separar las cualidades de un objeto para considerarlas aisladamente o para considerar el mismo objeto en su pura esencia o noción.
- **Abstracto**: Que significa alguna cualidad con exclusión del sujeto.

Las abstracciones que nosotros hemos analizado son:
Acciones y Funciones. Más adelante analizaremos TAD´s.

Abstracción

En los lenguajes de programación, el concepto fundamental que se usa para lograr la abstracción es el de **Módulo**, que es una sección de un programa bien construida, con un fin específico, y que puede ser reutilizado.

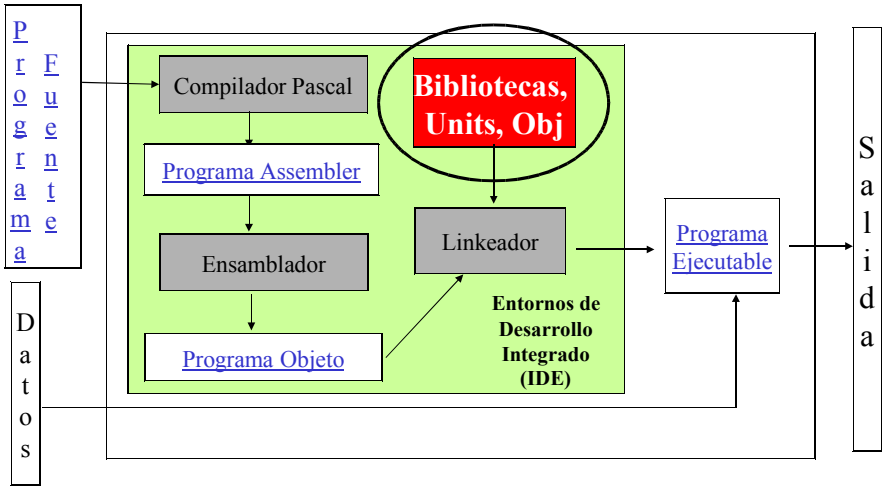
Una forma de profundizar el concepto de abstracción es mediante la construcción de módulos que sean independientes de un algoritmo o programa en particular, es decir, que no formen parte indisoluble del mismo.

Abstracción

Tradicionalmente, este tipo de módulos se han utilizado para lograr la **Compilación separada** de un programa, la que permite recompilar únicamente las secciones del programa que han sufrido cambios. Esta técnica hace posible crear bibliotecas de programas en las que el código fuente de dichos programas no tiene porqué ser accesible por el programador que hace uso de esa biblioteca.

En Pascal existe una manera de generar módulos, independientes de cualquier programa, mediante el uso de Unidades o **UNITS**. Por ejemplo **CTR** es una **UNIT**, la cual usamos en nuestro programas pero no tenemos acceso a su implementación.

Abstracción



Unidad - Concepto

- Una unidad es un conjunto de constantes, tipos de datos variables, procedimientos y funciones.
- Cada unidad es como un algoritmo independiente o bien una librería de declaraciones que permite que se pueda y compilar independientemente.
- Una unidad puede utilizar otras unidades.
- Tienen una estructura similar a los algoritmos
- El concepto de unidad acentúa el carácter estructurado y modular
- Además, las unidades son reusables, con lo cual se puede escribir una librería ahora y usarla en cuantos algoritmos sea necesario.

Unidad - Estructura

A continuación analizaremos las unidades como las provee Pascal. En notación algorítmica usaremos una sintaxis similar y las presentaremos de manera conjunta.

Una unidad está constituida por las siguientes secciones:

– Cabecera de la unidad

– Sección **Interface / Interface** (o sección pública)

– Sección **Implementación / Implementation** (o sección privada)

– Sección de inicialización

Unidad - Estructura

```
Unidad <nombre>
Interfaz
Usa <lista de unidades> {opcional}
{declaraciones públicas}
Implementación
{declaraciones privadas}
{definición de procedimientos y funciones}
Inicio
 <secuencia de acciones> {opcional}
Fin
```

Units de Pascal - Concepto

Es un conjunto de constantes, tipos de datos, variables, procedimientos y funciones encapsuladas bajo un mismo identificador.

- Poseen una estructura bien definida.
- Las unidades pueden ser predefinidas (estándar) o definidas por el usuario.
- Turbo Pascal varias unidades estándar:
  - SYSTEM
  - GRAPH
  - DOS
  - CRT
  - PRINTER
  - GRAPH3

Units de Pascal - Estructura

UNIT <identificador>;

INTERFACE

USES <lista de unidades>; {opcional}

{declaraciones públicas}

IMPLEMENTATION

{declaraciones privadas}

{definición de procedimientos y funciones}

BEGIN

{código de inicialización} {opcional}

END.

Units de Pascal - Estructura

CABECERA DE LA UNIDAD

- Comienza con la palabra reservada UNIT seguido del identificador y finalizado con un punto y coma.
- Un identificador válido está formado por una cadena de 1 a 8 caracteres.
- El nombre de la unidad puede ser arbitrario pero debe coincidir con el nombre del archivo que lo contiene.
- Ejemplo:

– La “UNIT test” debe almacenarse en un fichero denominado test.pas. Una vez compilado, la extensión del fichero será TPU (turbo pascal unit), es decir test.tpu

Units de Pascal - Estructura

SECCIÓN INTERFACE

- Esta parte es la que permite conectar esta unidad con otras unidades y programas.
- También es conocida como la sección pública ya que todo lo en esta sección es visto desde el exterior de la unidad.
- Se pueden declarar constantes, tipos de datos, variables, funciones y procedimientos.
- Sólo se declara la cabecera de las funciones y procedimientos. Su implementación se encuentra en la sección “Implementation”.

Units de Pascal - Estructura

SECCIÓN IMPLEMENTATION

• Esta sección es estrictamente privada y por tanto su contenido no puede ser visto desde el exterior de la unidad.
• Esta sección contiene los cuerpos de los procedimientos y funciones declarados en la sección “Interface”.
• Las variables declaradas dentro de esta sección serán de uso exclusivo para los procedimientos y funciones de dicha unidad.

Units de Pascal - Estructura

SECCIÓN DE INICIALIZACIÓN

• Esta sección opcional puede contener, por ejemplo, instrucciones que sirvan para inicializar variables.
• La ejecución de estas instrucciones se efectúa antes de ejecutar la primera instrucción del programa que usa dicha unidad.
• En esta sección también se pueden inicializar cualquier estructura de datos que emplee la unidad.

Units de Pascal

CREACIÓN DE UNIDADES

• Una vez que se dispone del código fuente la unidad se compila de la misma forma que un programa.
• El archivo obtenido posee la extensión TPU y es un archivo no ejecutable.
• Para poder utilizar una unidad se debe declarar su uso en la sección **USES** del programa que hará uso de la misma

• Ejemplo:

```
PROGRAM Prueba;
USES Utilidad;
```


Units de Pascal

UNIDADES ESTANDAR

• Estas unidades se sitúan en un archivo denominado **turbo.tpl** y son bibliotecas que enriquecen el lenguaje y facilitan la realización de algunas tareas. Describiremos sintéticamente cada una de las siguientes unidades:

- SYSTEM
- CRT
- DOS
- PRINTER

Units de Pascal

UNIDAD SYSTEM

• Contiene los procedimientos y funciones estándar de Turbo Pascal relativas a la entrada/salida, cadenas de caracteres, calculo en coma flotante, gestión de memoria, etc.

• Esta unidad no necesita ser referenciada ya que se carga automáticamente en la compilación de cada programa.

Units de Pascal

UNIDAD CRT

• Esta unidad proporciona un conjunto de declaraciones que permiten el acceso al control de los modos de pantalla, de teclado, posicionamiento del cursor, etc.

• Algunos de sus procedimientos son:
– Clrscr: borra la pantalla.
– Keypressed: detecta la pulsación de una tecla.
– Sound: hace sonar el altavoz interno (se agradece no usarla)
– Window: define una ventana de texto en la pantalla.

Units de Pascal

EJEMPLO UNIDAD CRT

```
PROGRAM CalcularAreaTriangulo;
USES CRT;
VAR
  base, altura: Real;
  area: Real;
  PROCEDURE LeerDatos(VAR b:Real; VAR h: Real);
  ...
  PROCEDURE CalcArea(b:Real; h: Real; VAR a:Real);
  ...
  PROCEDURE MostrarArea(a:Real);
  ...
BEGIN
  LeerDatos(base, altura);
  CalcArea(base, altura, area);
  Clrscr;
  MostrarArea(area);
  Readkey
END.
```


Units de Pascal

UNIDAD DOS

• Esta unidad contiene declaraciones, constantes, tipos variables, procedimientos y funciones relacionadas con el sistema operativo DOS y la gestión de archivos.

• Algunos de sus procedimientos son:
- Gettime: proporciona la hora a través del reloj interno
- Getdate: proporciona fecha registrada en la computadora.
- Disksize: proporciona el tamaño de la unidad de almacenamiento.
- Getdir: obtiene la unidad y el camino del directorio actual de una unidad.
- Erase: borra un archivo.
- Rename: cambia el nombre de un archivo.
- Chdir: cambia de directorio.
- Mkdir: crea un directorio.
- Rmdir: borra un directorio.

Units de Pascal

UNIDAD PRINTER

• Esta unidad facilita la tarea del programador cuando utiliza la impresora como dispositivo de salida.

• Permite enviar la salida estándar de Pascal a la impresora utilizando para ello los procedimientos: “Write” y “Writeln”.

Units de Pascal

EJEMPLO UNIDAD PRINTER

```
PROGRAM Impresora;
USES Printer;
VAR
  a, b, c: Integer;
BEGIN
  Writeln(lst, ‘este texto aparece en la impresora’);
  Read(a,b);
  c:=a+b;
  Writeln(lst,c:6);
END.
```


Units de Pascal

EJEMPLO UNIDAD Bibcar

```
PROGRAM mayusVocal;
USES CRT, Bibcar;
VAR
letra: Char;
BEGIN
  WRITELN('Ingrese caracter: ');
  READLN(letra);
  IF (EsMayuscula(letra)) THEN BEGIN
     WRITELN('El caracter ingresado ', letra,' es una mayuscula');
  END
  ELSE BEGIN
     WRITELN('El caracter ingresado ', letra,' es una minuscula');
  END;
  IF (EsVocal(letra)) THEN BEGIN
     WRITELN('El caracter ingresado ', letra,' es una vocal');
  END
  ELSE BEGIN
     WRITELN('El caracter ingresado', letra,' NO es una vocal');
  END;
  READKEY
END.
```


Units de Pascal

EJEMPLO UNIDAD Bibcar

```
UNIT Bibcar;
INTERFACE
FUNCTION AMinuscula (c: Char): Char;
FUNCTION EsMayuscula (c: Char): Boolean;
FUNCTION EsVocal(c: Char): Boolean;

IMPLEMENTATION
FUNCTION AMinuscula (c: Char): Char;
BEGIN
  AMinuscula := Chr(Ord(c)+(Ord('a')-Ord('A')))
END;

{sigue...}
```

Units de Pascal

EJEMPLO UNIDAD Bibcar

```
FUNCTION EsMayuscula (c: Char): Boolean;
BEGIN
  EsMayuscula :=  ((c >= 'A') AND (c <= 'Z'))
END;
FUNCTION EsVocal(c: Char): Boolean;
  VAR minus: Char;
BEGIN
  IF (EsMayuscula(c)) THEN BEGIN
    minus := AMinuscula(c);
  END
  ELSE BEGIN
    minus := c;
  END;
  EsVocal := ((minus = 'a') OR (minus = 'e') OR (minus = 'i') OR (minus =
'o') OR (minus = 'u'))
END;
```


Units de Pascal

EJEMPLO UNIDAD Numeros

```
PROGRAM Sumatoria;
USES Crt, Numeros;
VAR
  acum: Integer;
BEGIN {programa principal}
  CargarNumeros;
  Iniciar;
  IF numActual= fin
  THEN BEGIN
    Writeln('No hay numeros ingresados');
    Readkey
  END
{sigue…}
```


Units de Pascal

```
  ELSE BEGIN
    acum:=0;
    WHILE numActual<>fin DO
    BEGIN
      acum:=acum+numActual;
      ProximoNumero
    END;
    Writeln('La sumatoria de los numeros es: ', acum);
    Readkey
  END;
  Readkey
END. {Fin del programa}
```


Units de Pascal

```
{Unidad Numeros}
UNIT Numeros;
INTERFACE
CONST
  fin= -32768;
VAR
  numActual: Integer;   {variable que contiene el numero
actualmente en uso}
PROCEDURE CargarNumeros; {procedimiento que guarda los
numeros ingresados por el usuario}
PROCEDURE Iniciar; {procedimiento que deja todo listo para
comenzar a usar los numeros ingresados por el usuario}
PROCEDURE ProximoNumero;  {procedimiento que avanza al
proximo numero ingresado por el usuario}
{sigue…}
```

Units de Pascal

```
IMPLEMENTATION
CONST
  marca=chr(254);
VAR
  i: Integer;
  cintaNum: ARRAY[1..255] OF Integer;
  cintaCar: String[255];
  numS: String;
  numI: Integer;
PROCEDURE StringToArrayInt; .....
PROCEDURE CargarNumeros; …..
PROCEDURE Iniciar; …..
PROCEDURE ProximoNumero; …..
END {Fin Unidad Numeros}.
```


Bibliografía

– Pascal
- introturbopascal.pdf (aula virtual)
- laprogramacionenlenguajepascal.pdf (aula virtual)
- pascalyturbopascal.pdf (aula virtual)
- Joyanes Aguilar, L., “Programación en Turbo Pascal”. Mc Graw Hill, 1993.
- Wirth, N. and K. Jensen, “Pascal: User Manual and Report”, 4°ed., New York, Springer-Verlag, 1991 (traducción de la primera edición “Pascal: Manual del Usuario e Informe”, Buenos Aires, El Ateneo, 1984).