



## Resolución de Problemas y Algoritmos

### Clase 11: Definición y compatibilidad de tipos de datos. Sentencia condicional CASE.



**Dr. Diego R. García**

Departamento de Ciencias e Ingeniería de la Computación  
Universidad Nacional del Sur  
Bahía Blanca - Argentina



### Observaciones importantes

- Los temas sobre los cuales trabajaremos a partir de esta clase:
  - serán evaluados en el segundo parcial,
  - y dado que no tendrán la práctica suficiente, no se espera que los usen para resolver problemas del primer parcial.
- Todos los temas que ya hemos trabajado y que se evalúan en el primer parcial se seguirán usando como base para los temas que veremos a partir de hoy.

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Conceptos: Tipos de datos en Pascal

Se pueden dividir en:

Tipos de datos

- Predefinidos
- Definidos por el programador

Ejemplos de tipos que se usan en RPA:

- **BOOLEAN** (ordinal)
- **CHAR** (ordinal)
- **INTEGER** (ordinal)
- **REAL**
- **TEXT** (estructurado)
- **FILE OF...** (estructurado)
- **Subrangos** (ordinal)

Definir nuevos tipos de datos permite claridad y abstracción. Dos conceptos fundamentales en el desarrollo de Software

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Tipos definidos por el programador

- Poder definir y usar tipos de datos fue un muy importante avance en la evolución de los lenguajes de programación.
- Permiten dar **claridad** al código fuente. Esto ayuda al programador al leer el código y entonces **prevenir errores** de programación.
- También dan **información** al **compilador**, que puede ser usada para **prevenir errores**, y además generar un **mejor código** ejecutable.
- Hay compiladores que realizan un chequeo de tipos al compilar, otros al ejecutar, y otros en ambos momentos (en algunos casos se puede configurar mediante opciones al compilar).

**¡IMPORTANTE!**

- En esta materia vamos a usar solamente algunas de las ventajas de poder definir nuevos tipos.
- Más adelante, en otras materias de su carrera descubrirá muchas más ventajas del uso de tipos definidos por el programador.

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Tipos definidos por el programador: subrangos

**TYPE** ← Palabra reservada que indica la sección de declaración de nuevos tipos.

**LetrasMayusculas = 'A' .. 'Z';** ← Subrango de CHAR

**Numeros\_de\_Mes = 1..12;** ← Subrangos de INTEGER

**Digitos = 0..9;** ← Subrango de CHAR

**Nombre de un nuevo tipo (identificador)**

- Se puede definir **tipos subrangos**, de cualquier otro **tipo ordinal**.
- Se indica: un identificador como nombre del nuevo tipo, luego el símbolo “=”, y finalmente, separados por un par de puntos consecutivos “..” un valor inicial y un valor final de algún tipo ordinal.
- Estos valores definen el “RANGO” de todos los valores posibles para los elementos de este nuevo tipo de dato. Las operaciones son las mismas que las del tipo ordinal del cual se hace el subrango.

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Nuevos tipos de datos definidos por el programador

```

PROGRAM Ejemplo;
CONST meses=12; } Declaración de constantes
TYPE
Tipo_Digito = 0..9;
Tipo_char_digito = '0'..'9';
TNumMes = 1..meses;
TNumDeCarta = 1..12;
LetrasMayusculas = 'A' .. 'Z';
LetrasMinusculas = 'a' .. 'z';
VAR
digito: tipo_digito; carta: TnumDeCarta; } Declaración de tipos
Inicial: LetrasMayusculas; num: integer; } Declaración de variables
BEGIN
digito:= 3; carta:=12; Inicial:='A'; num:=digito;
    
```

Resolución de Problemas y Algoritmos Dr. Diego R. García

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:  
 “Resolución de Problemas y Algoritmos. Notas de Clase”. Diego R. García. Universidad Nacional del Sur. (c) 02/10/2019

### Otra forma de definir nuevos tipos de datos

```

TYPE
Archivo_Letras = FILE OF char;
Archi_temperaturas = FILE OF real;
componentes = integer;
Productos = FILE OF componentes;
    
```

Esto será necesario para conceptos de se verán en las próximas clases.

```

VAR
Letras: Archivo_letas;
Elem: componente;
Stock: Productos;
    
```

- En Pascal también se pueden definir **nuevos tipos** (o **nuevos nombres para un tipo**) en función de tipos ya existentes.
- Se indica: un identificador como nombre para el nuevo tipo, luego el símbolo "=", y luego algún tipo predefinido, tipo estructurado, o tipo definido por el usuario antes.
- Los valores posibles y las operaciones disponibles este nuevo tipo de dato, son las mismas del tipo usado a la derecha del "=".

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Tipos definidos por el programador

```

PROGRAM Ejemplo;
TYPE Entero = integer; Logico = Boolean;
      NroReal = real; Letra = char;
      Telemento = Entero;
      TipoArchivo = FILE OF Telemento;
VAR Inicial: Letra; Es_Par: Logico;
      Num: Telemento;
      Archivo1, Archivo2: TipoArchivo;
BEGIN
Inicial := 'A'; Num:= 4;
Es_Par := (Num MOD 2) = 0
Assign(Archivo1,'num.dat');
rewrite(archivo1);
write(Archivo1,num);
    
```

Declaración de tipos

Declaración de variables

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Relaciones entre tipos de datos

**En Pascal existen tres relaciones entre tipos de datos:**

1. Igualdad o Identidad.
2. Compatibilidad.
3. Compatibilidad de asignación.

Resolución de Problemas y Algoritmos Dr. Diego R. García

### (1) Tipos idénticos en Pascal

Dos elementos tienen **tipos idénticos** si se cumple una de las siguientes opciones:

- Están declarados con el mismo identificador de tipo.
- Los identificadores de tipo son diferentes ( ej: **T1** y **T2**) pero han sido definidos como equivalentes por una declaración de la forma **T1 = T2**.

**Ejemplo: ¿Cuáles variables tienen tipos idénticos?**

```

TYPE T = INTEGER;
      T1 = T;
VAR A, A1: T; A2: REAL;
      B: INTEGER;
      C: T1;
      D: -32768 .. 32767;
    
```

Resolución de Problemas y Algoritmos Dr. Diego R. García

### (2) Tipos compatibles en Pascal

Dos tipos son **compatibles** si al menos una de las siguientes opciones es verdadera:

- Ellos son idénticos.
- Uno es subrango del otro.
- Ambos son subrangos del mismo tipo.

**Ejemplo: ¿Cuáles son compatibles?**

```

TYPE T = Integer; Sub = 1..1000; Sub2 = Sub;
      Sub1 = 100..200; Sub3 = 0..99; TipoNum = Real;
VAR A: Sub; B: INTEGER; C: Sub1;
      D: Sub2; E: Sub3; F: TipoNum;
    
```

Resolución de Problemas y Algoritmos Dr. Diego R. García

### (3) Compatibilidad de asignación

Una expresión **E** de tipo **T2** es **asignación-compatibile** con el identificador **V** de tipo **T1** si al menos una de las siguientes declaraciones es verdadera:

- 1) **T1** y **T2** son idénticos.
- 2) **T1** es real y **T2** es entero o subrango de entero.
- 3) **T1** y **T2** son subrangos o enteros, y el valor de **E** es un valor permitido del tipo **T1**.

Resolución de Problemas y Algoritmos Dr. Diego R. García

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:  
 "Resolución de Problemas y Algoritmos. Notas de Clase". Diego R. García. Universidad Nacional del Sur. (c) 02/10/2019

## Sentencia condicional: CASE

### Sentencias condicionales en Pascal

**Problema propuesto:** Escriba un programa en Pascal que lea un valor char, y si es "@" indique en pantalla "arroba"; si es un dígito '0' a '9' indique "dígito"; si es un operador de suma, resta, multiplicación o división, indique "operador"; si es una letra mayúscula o minúscula, indique "letra".

```

IF valor = '@'
THEN write(' arroba ')
ELSE IF (valor >= '0') and (valor <= '9')
THEN writeln(' dígito')
ELSE IF (valor = '+') or (valor = '-') or (valor = '*') or (valor = '/')
THEN writeln(' operador ')
ELSE IF (valor >= 'A') and (valor <= 'Z')
or (valor >= 'a') and (valor <= 'z')
THEN writeln(' letra');
    
```

← Parte de una posible solución

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Sentencia condicional CASE en Pascal

**CASE** es un sentencia condicional que permite discriminar para distintos "casos" (valores) que sentencia debe ejecutarse.

Aquí se especifica un solo caso.

Aquí se especifican 10 casos

Aquí se especifican 4 casos.

Aquí se especifican 52 casos (2x26)

```

program opciones; //reconoce simbolos
var valor: char;
begin
write('ingrese valor ASCII');
readln(valor);
case valor of
'@' : write('arroba ');
'0'..'9' : writeln(' digito ');
'+', '-', '*', '/' : writeln(' operador ');
'A'..'Z', 'a'..'z': writeln(' letra');
end;
write('fin del programa'); readln;
end.
    
```

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Sentencias condicionales en Pascal

- Una sentencia **CASE** puede considerarse como una "abreviatura" de varios **IF-THEN-ELSE** anidados.
- Todo **CASE** puede reescribirse con **IF-THEN-ELSE** anidados.
- Por ejemplo, el case anterior puede reescribirse

```

IF valor = '@'
THEN write(' arroba ')
ELSE IF (valor >= '0') and (valor <= '9')
THEN writeln(' dígito')
ELSE IF (valor = '+') or (valor = '-') or (valor = '*') or (valor = '/')
THEN writeln(' operador ')
ELSE IF (valor >= 'A') and (valor <= 'Z')
or (valor >= 'a') and (valor <= 'z')
THEN writeln(' letra');
    
```

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Sentencia CASE (sintaxis)

```

CASE <expresion> OF
<lista_opciones>: <una sentencia simple o compuesta>;
<lista_opciones>: <una sentencia simple o compuesta>;
...
END; {sugerencia: vea el diagrama sintáctico de CASE}
    
```

- <expresion>** cualquier expresión que sea de tipo ordinal
- <lista\_opciones>** puede ser:
  - un valor individual. Ej: 2
  - valores individuales separados por coma. Ej: 2,5,7
  - Rangos de valores. Ej: 1..100
  - una combinación de (b) y (c) ej: 1..10, 13, 15..20
- Las listas de opciones **deben ser disjuntas**. No puede haber opciones repetidas, es un error de compilación

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Sentencia CASE (semántica)

```

CASE <expresion> OF
<lista_opciones>: <una sentencia simple o compuesta>;
<lista_opciones>: <una sentencia simple o compuesta>;
...
END;
    
```

**<expresion>** y **<lista\_opciones>** deben ser del mismo tipo

- Se evalúa **<expresion>** y se obtiene un **valor**
- Se busca (de arriba hacia abajo) **valor** está en una de las **<lista\_opciones>**
- Si se encuentra el **valor** se ejecuta la sentencia siguiente al "." y luego sigue en el **END;**
- Si **valor no pertenece** a ninguna de las **<lista\_opciones>** no se ejecuta ninguna sentencia

Resolución de Problemas y Algoritmos Dr. Diego R. García

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:  
 "Resolución de Problemas y Algoritmos. Notas de Clase". Diego R. García. Universidad Nacional del Sur. (c) 02/10/2019

### Opciones de una sentencia CASE

```

Una expresión (ordinal) → CASE trunc(R)-3*2 OF
4 : BEGIN
    ...sentencias
    ...
END;
Puede haber un único valor en la opción
Pueden haber varios separados por comas → 1,2,3: write(' 1 2 o 3');
Pueden haber un rango de valores → 50..100: WRITE(' 5 a 10 ');
101,201, 300..400,
501..1001, 2001: BEGIN
    ...
END;
Pueden haber una combinación de valores y rangos
END; {del case}
    
```

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Observaciones sobre sentencia CASE

Opciones repetidas

```

VAR M: INTEGER;
CASE M OF
1, 5 : <sentencia>
5, 3 : <sentencia>
4 ..10: <sentencia>
END;
    
```

**MAL**

- No puede haber opciones repetidas, es un error de compilación.
- Las listas de opciones deben ser disjuntas.

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Observaciones sobre sentencia CASE

Extensión a Pascal estándar

```

VAR M: integer;

CASE M OF
-9..9: write(' 1 dígito');
-99..-11,11..99: write(' 2 dígitos');
ELSE write(' más de 2 dígitos');
END;
    
```

- el **ELSE** se ejecuta cuando el valor no corresponde a ninguna opción

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Otra solución para "días de un mes" (usando CASE)

```

mes, anio, cant_dias: INTEGER;

CASE MES OF
11,4,6,9: cant_dias :=30;
2: IF (anio mod 4=0) and (anio mod 100<>0) or (anio mod 400=0)
THEN cant_dias := 29
ELSE cant_dias := 28;
1,3,5,7,8,10,12: cant_dias :=31;
END; {--- fin del case ---}
Writeln('Tiene', cant_dias, ' días');
END.
    
```

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Funcionamiento de CASE en Pascal

```

...
readln(mes, anio);
CASE MES OF
11,4,6,9: cant_dias :=30;
2: IF (anio mod 4=0) and (anio mod 100<>0) or (anio mod 400=0)
THEN cant_dias := 29
ELSE cant_dias := 28;
1,3,5,7,8,10,12: cant_dias :=31;
END; {--- fin del case ---}
Writeln(cant_dias);
END.
    
```

- si el valor de MES está entre estos valores
- entonces se ejecuta esta sentencia,
- una vez que se ejecuta una opción se pasa a la sentencia que sigue al case

Resolución de Problemas y Algoritmos Dr. Diego R. García

### Funcionamiento de CASE en Pascal

```

...
readln(mes, anio);
CASE MES OF
11,4,6,9: cant_dias :=30;
2: IF (anio mod 4=0) and (anio mod 100<>0) or (anio mod 400=0)
THEN cant_dias := 29
ELSE cant_dias := 28;
1,3,5,7,8,10,12: cant_dias :=31;
END; {--- fin del case ---}
Writeln(cant_dias);
END.
    
```

Si el valor de MES NO está entre estos valores, pasa a la siguiente opción, y así sucesivamente...

Si MES tiene un valor que no figura en ninguna de las opciones, entonces no se ejecuta ninguna opción del case.

Resolución de Problemas y Algoritmos Dr. Diego R. García

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente: "Resolución de Problemas y Algoritmos. Notas de Clase". Diego R. García. Universidad Nacional del Sur. (c) 02/10/2019

### Problema propuesto

Un día es muy frío si la temperatura máxima está entre -20 y 1 grado, es frío si su máxima está entre 2 y 10, es templado si está entre 11 y 20, es cálido entre 21 y 28 y muy caluroso entre 29 y 45. Considere un archivo de enteros 'temperaturas.dat' que tiene las temperaturas máximas de un mes. Escriba un programa que calcule cuantos días muy fríos, fríos, templados, cálidos, y muy calurosos ocurrieron en ese mes.

### Sentencia CASE. Ejemplo

```
CASE ... OF
-20..1: ... muy frío ....
2..10: ... frío ...
11..20: .... Templado ....
21..28: ... cálido ...
29..45: ... muy caluroso ...
END
```

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:  
"Resolución de Problemas y Algoritmos. Notas de Clase". Diego R. García. Universidad Nacional del Sur. (c) 02/10/2019