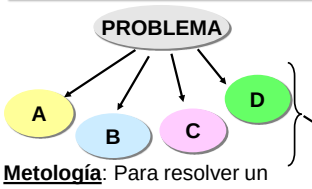
Resolución de Problemas y Algoritmos

Clase 14
Lenguaje Pascal:
estructura de bloques, entornos de referencia, visibilidad de identificadores.

Dr. Alejandro J. García
http:// cs.uns.edu.ar /~ajg

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

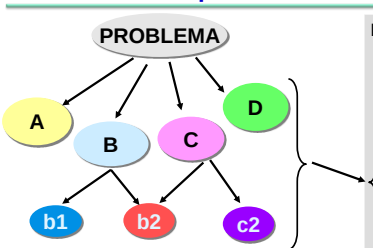
División del problema en subproblemas



Metología: Para resolver un problema complejo se propone:
1) **dividir** en subproblemas,
2) **resolver** cada parte y luego
3) **para cada parte implementar primitivas** en Pascal: como funciones o procedimientos

```
Program SOLUCIÓN;
  Function A
  Procedure B
  Function C
  Procedure D
Begin
  …
End.
```


División del problema en subproblemas



El programa puede reflejar la división del problema realizada en el diseño.
En Pascal no hay límite en cantidad o anidamiento de bloques.

```
Program SOLUCIÓN;
  Function A
  Function b2
  Procedure B
    Procedure b1
  Function C
    Procedure c2
  Procedure D
Begin … End.
```

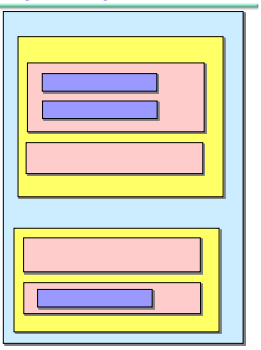

"El límite está dado por su imaginación"

```
PROGRAM PROGRAMA1;
  PROCEDURE…
  FUNCTION …
  PROCEDURE…
  PROCEDURE…
  FUNCTION …
  PROCEDURE…
  {…puede incluir todos los procedimientos o funciones que quiera…}
BEGIN ... END.
```

```
PROGRAM PROGRAMA2;
  PROCEDURE …
    FUNCTION …
      PROCEDURE …
      BEGIN ..... END;
  BEGIN ..... END;
  {..y en c/bloque, todo el "anidamiento" que quiera}
BEGIN... END.
```


Pascal: estructurado por bloques

- **En un programa pueden incluirse tantos procedimientos y funciones como se desee.**
- **Cada uno de ellos puede a su vez tener sus bloques internos y así siguiendo.**
- **Esto permite implementar cualquier división del problema en subproblema que se diseñe.**



Pascal es estructurado por bloques

```
PROGRAM MIPROGRAMA;
CONST ..... TYPE.... VAR .....
FUNCTION F(X:real):real;
CONST ..... TYPE....
VAR .....
  PROCEDURE …
  FUNCTION …
BEGIN ...sentencias...END;

PROCEDURE P(Var X: char);
CONST..... TYPE....
VAR .....
  PROCEDURE …
  FUNCTION …
BEGIN...sentencias...END;

BEGIN
  ...sentencias...
END.
```

- En Pascal, un programa constituye un bloque compuesto por:
  - Constantes, tipos, Variables,
  - funciones, procedimientos,
  - y sentencias.
- Cada procedimiento o función también constituye un bloque con:
  - parámetros
  - constantes, tipos, variables,
  - procedimientos, funciones,
  - y sentencias.

Bloques e identificadores

```
PROGRAM simple; {para entender los conceptos}
Const Pi= 3.14; type Tdig=0..9; var  A, B, C:CHAR;
   PROCEDURE P1  (A:REAL);
   var B: REAL; F2: Tdig;
   begin  B:= A; WRITE(B)  end; {P1}
   PROCEDURE P2  (A:REAL);
   var B, MIA: real;
      FUNCTION  F2 (A:REAL):REAL;
      var B, DE_F2: REAL;
      begin  B:= A; F2:= B + Pi; end; {F2}
   begin  B:= A; WRITE(F2(A)); P1(B)  end; {P2}
BEGIN
   P2(5); P1(10);
END.
```


Bloques (demarcados)

```
PROGRAM simple; {para entender los conceptos}
Const Pi= 3.14; type Tdig=0..9; var  A, B, C:CHAR;
   PROCEDURE P1  (A:REAL);
   var B: REAL; F2: Tdig;
   begin  B:= A; WRITE(B)  end; {P1}
   PROCEDURE P2  (A:REAL);
   var B, MIA: real;
      FUNCTION  F2 (A:REAL):REAL;
      var B, DE_F2: REAL;
      begin  B:= A; F2:= B + Pi; end; {F2}
   begin  B:= A; WRITE(F2(A)); P1(B)  end; {P2}
BEGIN
   P2(5); P1(10);
END.
```


Estructurado por Bloques

simple

- **Elementos que puede tener un BLOQUE:**
  1. **identificadores de constantes**
  2. **identificadores de tipos**
  3. **identificadores de variables**
  4. **identificadores parámetros formales (en proc. y fn.)**
  5. **identificadores de procedimientos**
  6. **identificadores de funciones**
  7. **sentencias**
- **Dentro de un mismo bloque no puede haber dos identificadores iguales para distintos elementos.**
- **Dos elementos pueden tener el mismo identificador si pertenecen a diferentes bloques.**

Preguntas sobre el programa "simple"

simple

- **¿puedo llamar a P1 desde las sentencias de P2?**
- **¿puedo llamar a F2 desde las sentencias de P2?**
- **¿puedo llamar a F2 desde las sentencias de P1?**
- **¿puedo llamar a P1 desde las sentencias de F2?**
- **¡ HAGA AHORA SUS PREGUNTAS ! (y copie las de sus compañeros)**
- Pregunta más general: **¿desde qué lugar del programa puedo llamar a una función o proced.?**
- **¿en qué bloques puedo usar la variable "MIA"?**
- **¿y la variable DE_F2?**
- **¿en qué bloques puedo usar una variable?**
- **Todas las respuestas en la teoría que sigue a continuación…**

Vocabulario: declaración vs. referencia

simple

Es importante distinguir entre:

1. **La declaración de un identificador** de constante, tipo, variable, parámetro, función, o procedimiento**. Ejemplos:**
   ```
   CONST pi=3.14;  TYPE archi: FILE OF integer;
   VAR precio: real;      a_pagar: integer;
   PROCEDURE recargo(precio, rec:real; var monto:real);
   FUNCTION intereses(monto:integer):real
   ```
2. **La referencia (uso) de un identificador. Ejemplos:**
   ```
   recargo(24,incremento,precio);
   a_pagar := precio+intereses(round(precio));
   ```

En cada bloque, se declaran identificadores; y además, se hace referencia (usan) identificadores.

Concepto: Entorno de referencia para un bloque B

El entorno de referencia de un bloque B está formado por:

simple

- **El entorno local: parámetros formales, constantes, tipos y variables declarados dentro de B y el nombre de los procedimientos y funciones declarados dentro del bloque B.**
- **El entorno global: conjunto de identificadores declarados en el bloque del programa principal**
- **El entorno no-local: conjunto de identificadores declarados en los bloques que contienen al bloque B, exceptuando al global**
- **El entorno predefinido: conjunto de identificadores ya declarados por el compilador de Pascal y disponible para todo programa (Ej: maxint, char, write, eof).**

Ejemplos de entornos de referencia

- El programa "simple" tiene 4 bloques:
  P1,
  P2,
  F2
  y el bloque del programa "simple".
- A continuación se muestran los entornos de referencia para cada uno de estos bloques.
- Observe que el entorno predefinido y el entorno global es siempre el mismo para todos.

Ejemplos: bloques del programa "simple"

Entorno de referencia para el Bloque "F2 "
- Entorno local: A, B, DE_F2
- Entorno no-local: A, B, MIA, F2 (declarados en P2)
- Entorno global: Pi, Tdig, A, B, C, P1, P2
- Entorno predifinido: maxint, char, write, …

Entorno de referencia para el Bloque "P2"
- Entorno local: A, B, MIA, F2
- Entorno no-local: (vacío, no tiene)
- Entorno global: Pi, Tdig, A, B, C, P1, P2
- Entorno predifinido: maxint, char, write, …

Ejemplos: bloques del programa "simple"

Entorno de referencia para el Bloque "P1"
- Entorno local: A, B, F2
- Entorno no-local: (vacío, no tiene)
- Entorno global: Pi, Tdig, A, B, C, P1, P2
- Entorno predifinido: (el mismo siempre para todos)

Entorno de referencia para el Bloque "simple"
- Entorno no-local: (vacío, no tiene)
- Entorno global (y también local): Pi, Tdig, A, B, C, P1, P2
- Entorno predifinido: (el mismo siempre para todos)

Conceptos: identificadores ocultos

simple

Cuando se hace referencia a un identificador:
1. primero se busca en su entorno de referencia local,
2. luego en su entorno de referencia no local,
3. luego en su entorno de referencia global,
4. y finalmente en el entorno de referencia predefinido

Por lo anterior, si hay identificadores iguales en diferentes entornos uno oculta al otro.

1. Un identificador de nombre N en un entorno local oculta a todo identificador del mismo nombre N en otro entorno (no-local, global, predefinido)
2. Uno no-local N oculta a otro global de nombre N,
3. Un identificador global N oculta a uno predefinido N

Conceptos y vocabulario

simple

- Un identificador es referenciable en un bloque, si es parte de su entorno de referencia y no está oculto.
- Un identificador es visible, si es referenciable.
- El alcance de un identificador D, son aquellas sentencias (o bloques) del programa donde el identificador D es visible.

Ejercicios propuestos:
- Para cada bloque del programa simple, encuentre los identificadores visibles (referenciables).
- Indique el alcance del identificador P1 y el alcance de la variable global A.

Ejemplos

- La constante Pi es visible (referenciable) en todos los bloques (ya que está en todos los entornos por ser parte del entorno global). Lo mismo ocurre con el procedimiento P1 y el tipo Tdig.
- La función F2 es visible en P2 y en F2.
- La variable "de_f2" solo es visible en F2.