



PROGRAMACIÓN FUNCIONAL

Conceptos Preliminares



Conceptos Preliminares

- ◆ Conceptos de programación y programa
- ◆ Propiedades de los programas
- ◆ Valores y expresiones.
- ◆ Transparencia referencial.
- ◆ Funciones
 - ◆ Visión denotacional
 - ◆ Visión operacional
- ◆ Ecuaciones Orientadas. Reducción.
- ◆ Funciones como valores.

Definiciones

◆ Programación:

- ◆ es una tarea que comprende los siguientes puntos
 - ◆ analizar problemas
 - ◆ diseñar soluciones para ellos que puedan ejecutarse
 - ◆ codificar dichas soluciones
 - ◆ verificar propiedades deseadas de las mismas

◆ Programa:

- ◆ descripción de una solución a un problema, que puede ejecutarse de alguna manera para obtener una instancia particular de dicha solución.

Programación

- ◆ ¿Cuáles son los dos aspectos fundamentales?
 - ◆ transformación de información
 - ◆ interacción con el medio
- ◆ Ejemplos:
 - ◆ calcular el promedio de notas de examen
 - ◆ cargar datos de un paciente en su historia clínica
- ◆ Este curso se concentrará en el primero de estos aspectos.

Preguntas

◆ ¿Cuáles propiedades de un programa son importantes?

- ◆ eficiencia
- ◆ corrección
- ◆ claridad
- ◆ modificabilidad
- ◆ terminación
- ◆ equivalencia
- ◆ generalidad
- ◆ simplicidad

◆ ¿En cuáles debería focalizarse un programador?
¿Por qué?

Propiedades

- ◆ Si podemos probar fácilmente equivalencia de programas, podemos
 - ◆ reemplazar un programa por otro más eficiente
 - ◆ usar un programa correcto para ver que otro lo es
 - ◆ ver que no alteramos el significado al modificarlo
- ◆ ¿Qué necesitamos para poder probar equivalencia de programas con sencillez?

Preguntas

- ◆ ¿Cómo saber cuándo dos programas son iguales?
- ◆ Ejemplo:
 - ◆ ¿Son equivalentes ' $f(3)+f(3)$ ' y ' $2*f(3)$ '?
 - ◆ ¿Siempre?
 - ◆ ¿Sería deseable que siempre lo fueran?
¿Por qué?

Ejemplo

- ◆ ¿Qué imprime este programa?

```
Program test;
```

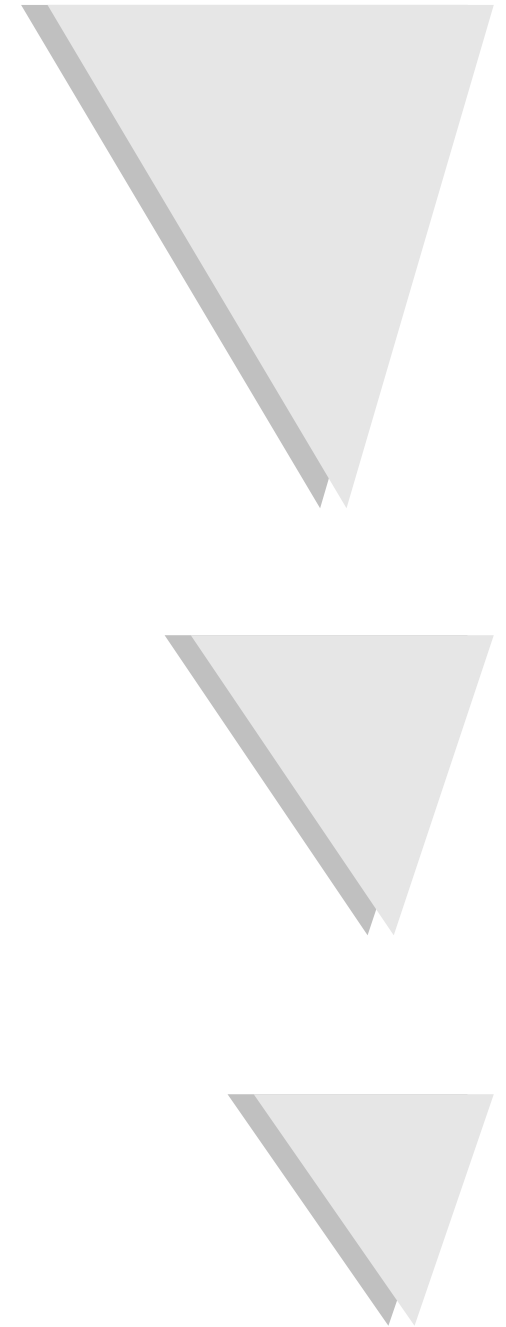
```
var x : integer;
```

```
function f(y:integer):integer;
```

```
begin x := x+1; f :=x+y; end;
```

```
begin x := 0; writeln(2*f(3)); end;
```

- ◆ ¿Y con 'f(3)+f(3)' en lugar de '2*f(3)'?



Valores y Expresiones

◆ Valores

- ◆ entidades (matemáticas) abstractas con ciertas propiedades
 - ◆ **Ejs:** el número dos, el valor de verdad falso.

◆ Expresiones

- ◆ cadenas de símbolos utilizadas para denotar (escribir, nombrar, referenciar) valores
 - ◆ **Ejs:** 2, (1+1), False, (True && False)

Transparencia Referencial

- ◆ “El valor de una expresión depende sólo de los elementos que la constituyen.”
- ◆ Implica:
 - ◆ consideración sólo del comportamiento externo de un programa (abstracción de detalles de ejecución).
 - ◆ posibilidad de demostrar propiedades usando las propiedades de las subexpresiones y métodos de deducción lógica.

Expresiones

- ◆ Expresiones atómicas
 - ◆ son las expresiones más simples
 - ◆ llamadas también formas normales
 - ◆ por abuso de lenguaje, les decimos *valores*
 - ◆ Ejs: 2, False, (3,True)
- ◆ Expresiones compuestas
 - ◆ se 'arman' combinando subexpresiones
 - ◆ por abuso de lenguaje, les decimos *expresiones*
 - ◆ Ejs: (1+1), (2==1), (4 - 1, True || False)

Expresiones

- ◆ Puede haber expresiones incorrectas (“mal formadas”)

- ◆ por errores sintácticos

*12 (True ('a',)

- ◆ por errores de tipo

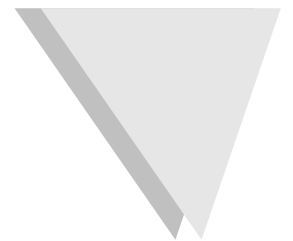
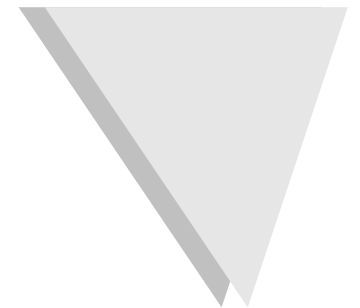
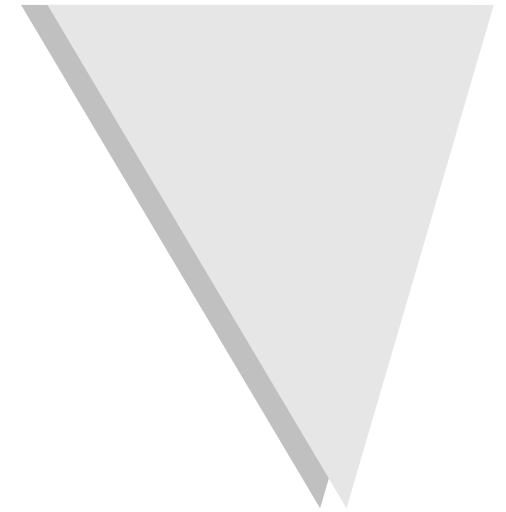
(2+False) (2||'a') 4 'b'

- ◆ ¿Cómo saber si una expresión está “bien formada”?

- ◆ Reglas sintácticas
- ◆ Reglas de asignación de tipo

Funciones

- ◆ Valores especiales, que representan “transformación de datos”
- ◆ Dos formas de entender las funciones
 - ◆ VISION DENOTACIONAL
 - ◆ una función es un valor matemático que relaciona cada elemento de un conjunto (de partida) con un único elemento de otro conjunto (de llegada).
 - ◆ VISION OPERACIONAL
 - ◆ una función es un mecanismo (método, procedimiento, algoritmo, programa) que dado un elemento del conjunto de partida, calcula (devuelve, retorna) el elemento correspondiente del conjunto de llegada.



Funciones

- ◆ Ejemplo: $\text{doble } x = x + x$
- ◆ Visión denotacional
 - ◆ a cada número x , doble le hace corresponder otro número, cuyo valor es la suma de x más x
 $\{ (0,0), (1,2), (2,4), (3,6), \dots \}$
- ◆ Visión operacional
 - ◆ dado un número x , doble retorna ese número sumado consigo mismo

$\text{doble } 0 \rightarrow 0$	$\text{doble } 1 \rightarrow 2$	
$\text{doble } 2 \rightarrow 4$	$\text{doble } 3 \rightarrow 6$...

Funciones

- ◆ ¿Cuál es la operación básica de una función?
 - ◆ la APLICACIÓN a un elemento de su partida
- ◆ Regla sintáctica:
 - ◆ la aplicación se escribe por yuxtaposición
 - ◆ $(f\ x)$ denota al elemento que se corresponde con x por medio de la función f .
 - ◆ Ej: $(\text{doble } 2)$ denota al número 4

Funciones

- ◆ ¿Qué expresiones denotan funciones?
 - ◆ Nombres (variables) definidos como funciones
 - ◆ Ej: doble
 - ◆ Funciones anónimas (lambda abstracciones)
 - ◆ Ej: $\lambda x \rightarrow x+x$
 - ◆ Resultado de usar otras funciones
 - ◆ Ej: `doble . doble`

Ecuaciones Orientadas

- ◆ Dada una expresión bien formada, ¿cómo determinamos el valor que denota?
 - ◆ Mediante ECUACIONES que establezcan su valor
- ◆ ¿Y cómo calculamos el valor de la misma?
 - ◆ Reemplazando subexpresiones, de acuerdo con las reglas dadas por las ecuaciones (REDUCCIÓN)
 - ◆ Por ello usamos ECUACIONES ORIENTADAS

Ecuaciones Orientadas

- ◆ Expresión-a-definir = expresión-definida
$$e1 = e2$$
- ◆ Visión denotacional
 - ◆ se define que el valor denotado por $e1$ (su significado) es el mismo que el valor denotado por la expresión $e2$
- ◆ Visión operacional
 - ◆ para calcular el valor de una expresión que contiene a $e1$, se puede reemplazar $e1$ por $e2$

Programas Funcionales

- ◆ Definición de programa funcional (*script*):
 - ◆ Conjunto de ecuaciones que definen una o más funciones (valores).
- ◆ Uso de un programa funcional
 - ◆ Reducción de la aplicación de una función a sus datos (reducción de una expresión).

Funciones como valores

- ◆ Las funciones son valores, al igual que los números, las tuplas, etc.
 - ◆ pueden ser argumento de otras funciones
 - ◆ pueden ser resultado de otras funciones
 - ◆ pueden almacenarse en estructuras de datos
 - ◆ pueden ser estructuras de datos
- ◆ (ABUSANDO DEL LENGUAJE) Función de alto orden:
 - ◆ una función que recibe otra función como argumento, o la retorna como resultado

Funciones como valores

◆ Ejemplo

$\text{compose } (f,g) = h \text{ where } h \ x = f \ (g \ x)$

$\text{sqr } x = x * x$

$\text{twice } f = g \text{ where } g \ x = f \ (f \ x)$

$\text{aLaCuarta} = \text{compose } (\text{sqr}, \text{sqr})$

$\text{aLaOctava} = \text{compose } (\text{sqr}, \text{aLaCuarta})$

$\text{fs} = [\text{sqr}, \text{aLaCuarta}, \text{aLaOctava}, \text{twice sqr}]$

$\text{aLaCuarta } 2 \rightarrow ?$

◆ ¿Será cierto que $\text{aLaCuarta} = \text{twice sqr}$?

Lenguaje Funcional Puro

- ◆ Definición de lenguaje funcional puro:

“lenguaje de expresiones con transparencia referencial y funciones de alto orden, cuyo modelo de cómputo es la reducción realizada mediante el reemplazo de iguales por iguales”

Resumen

- ◆ Conceptos Preliminares
- ◆ Valores y expresiones. Transparencia referencial.
- ◆ Funciones: visión denotacional y operacional.
- ◆ Ecuaciones orientadas. Reducción.
- ◆ Funciones como valores.