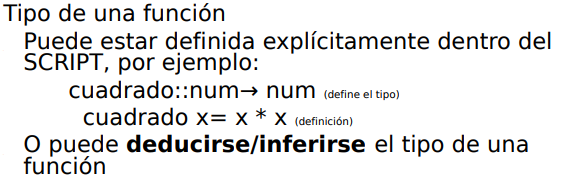
1

Un programa escrito en un lenguaje funcional es un conjunto de funciones que reciben datos de entrada, los cuales atraviesan las distintas funciones y genera un dato de salida.

El rol de la computadora es el de ejecutar las operaciones descritas en las funciones.

2



Un script es una lista de definiciones y este puede someterlas a evaluación

3

La noción de Variable es la de “variable matemática”, no la de celda de memoria.

4

Una expresión es su valor. El valor de una expresión depende únicamente de los valores de las sub expresiones que la componen. Las expresiones también pueden contener variables (valores desconocidos).

5

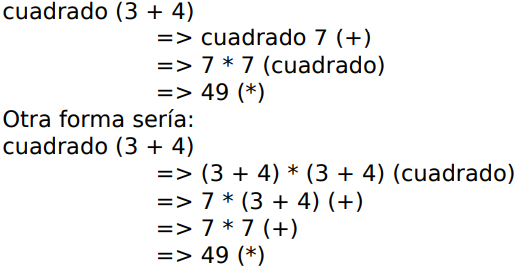
La forma de evaluar es a través de un mecanismo de reducción o simplificación. No importa la forma de evaluarla, siempre el resultado final será el mismo.

Existen 2 formas de reducción

Orden aplicativo: Aunque no lo necesite SIEMPRE evalúa los argumentos

Orden normal (lazy evaluation): No calcula más de lo necesario La expresión NO es evaluada hasta que su valor se necesite Una expresión compartida NO es evaluada más de una vez

Ejemplos de reducción o simplificación:



6

Si, un lenguaje funcional es fuertemente tipado, porque establece restricciones para la combinación de distintos tipos de datos. Por ejemplo, no se puede sumar un integer con un character.

Tipos

* NUM: INT y FLOAT (Números)
* BOOL (Valores de verdad)
* CHAR (Caracteres)
* Derivados: se construyen de otros tipos. Ejemplo:

(num,char) Tipo de pares de valores

(num→[]char) Tipo de una función

* También hay un tipo llamado polimórfico (representado con letras griegas) que se usa para representar un tipo de datos que no está definido o puede cambiar. Por ejemplo en la función id x = x, puede ser de char→ char, de num→ num, etc. Por lo tanto su tipo será de β→β.

7

Un programa escrito con una lenguaje OO es un conjunto de objetos que interactúan mandándose mensajes.

8

* Objetos: son entidades que poseen estado interno y comportamiento. Es el equivalente a un dato abstracto.
* Mensajes: es una petición de un objeto a otro para que este se comporte de una determinada manera, ejecutando uno de sus métodos. Todo el procesamiento en este modelo es activado por mensajes entre objetos.
* Métodos: es un programa que está asociado a un objeto determinado y cuya ejecución solo puede desencadenarse a través de un mensaje recibido por éste o por sus descendientes.
* Clases: es un tipo definido por el usuario que determina las estructuras de datos y las operaciones asociadas con ese tipo. Cada objeto pertenece a una clase y recibe de ella su funcionalidad. La información contenida en el objeto solo puede ser accedida por la ejecución de los métodos correspondientes
  + Instancia de clase: cada vez que se construye un objeto se está creando una instancia de esa clase. Una instancia es un objeto individualizado por los valores que tomen sus atributos.

9

El segundo nivel de abstracción consiste en agrupar las clases en jerarquías de clases (definiendo SUB y SUPER clases), de forma tal que una clase A herede todas las propiedades de su superclase B (suponiendo que tiene una).

10

Tipos de herencia:

* Simple: indica que se pueden definir nuevas clases solamente a partir de una clase inicial
* Múltiple: indica que se pueden definir nuevas clases a partir de dos o más clases iniciales.
* De implementación: la implementación de los métodos es heredada. Puede sobreescribirse en las clases derivadas.
* De interfaz: sólo se hereda la interfaz, no hay implementación a nivel de clase base (interfaces en Java, clases abstractas en C++).
* Estricta: Una sub-clase NO puede redefinir en forma contradictoria atributos definidos en la super-clase
* No estricta: Alguna sub-clase NO quiere heredar algún atributo de la super-clase

C++ usa todas y Smalltalk todas menos la múltiple

11

* Variables: Se refieren a elementos indeterminados que pueden sustituirse por cualquier otro. En “humano(X)”, la X puede ser sustituida por constantes como: juan, pepe, etc
* Constantes: A diferencia de las variables son elementos determinados. Ejemplo. “humano(juan)”. Las constantes son string de letras en minúsculas (representan objetos atómicos) o string de dígitos (representan números).

12

Se escribe una serie de aserciones lógicas. Un programa es una secuencia de “clausulas”.

El conocimiento se representa a través de reglas y hechos (que serían las clausulas).

Los objetos son representados por términos, los cuales contienen constantes y variables.

Un programa es un conjunto de reglas y hechos que proveen una especificación declarativa de que es lo que se conoce y la pregunta es el objetivo que queremos alcanzar.

13

14

Los lenguajes script asumen la existencia de componentes útiles en otros lenguajes. Su intención no es escribir aplicaciones desde el comienzo sino por combinación de componentes

Ejemplos de lenguajes basados en script: Perl, Python, PHP, Javascript.

Características más imporantes:

* Uso de scripts para “pegar” o combinar programas.

Se utilizan como lenguajes de extensión, ya que permiten al usuario adaptar o extender las funcionalidad de las herramientas script.

* Desarrollo y evolución rápida.

Por el contexto en el que se utilizan, deben ser fáciles de escribir y con una sintaxis concisa.

* Asociado a editores livianos.

Pueden ser escritos en procesadores de texto simples e incluso ejecutados en consola por su intérprete.

* Interpretados – (modestos requerimientos de eficiencia)

La velocidad de ejecución de los script no es de importancia crítica. Los gastos generales de interpretación y de comprobación dinámica se puede tolerar

* Alto nivel de funcionalidad en aplicaciones de áreas específicas.

Algunas generalidades entre lenguajes de este paradigma:

* Alto nivel de procesamiento de Strings y generación de Reportes. (Expresiones Regulares)
* Alto nivel para soporte de interfaces de usuario (GUI).
* Tipado dinámico.

15

Si, existen otros. Por ejemplo:

* Dirigido por eventos: el flujo del programa está determinado por sucesos externos (por ejemplo, una acción del usuario).
* Orientado a aspectos. apunta a dividir el programa en módulos independientes, cada uno con un comportamiento y responsabilidad bien definido.