Resumen PdP

Lautaro Bachmann

Contents

Qué es y qué puede hacer un lenguaje de programación

Un lenguaje de programación es un lenguaje formal dise˜nado para realizar procesos que pueden ser llevados a cabo por máquinas Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina o para expresar algoritmos con precisión

Sintaxis y semántica

Los lenguajes

son sistemas que se sirven de una **forma** para comunicar un **significado.** Lo que tiene que ver con la forma recibe el nombre de

sintaxis

lo que tiene que ver con el significado recibe el nombre de

semántica.

la forma

son los programas

el significado

es lo que los programas hacen,

Un lenguaje de programación

se describe con su **sintaxis** (qué es lo que se puede escribir legalmente en ese lenguaje) y su **semántica** (qué efectos tiene en la máquina lo que se escribe en ese lenguaje).

Alcance de los lenguajes de programación

algoritmos,

conjunto de instrucciones bien definidas, ordenadas y finitas que permite realizar algo de forma inambigua mediante pasos.

funciones computables

una función es computable si existe un algoritmo que puede hacer el trabajo de la función, es decir, dada una entrada del dominio de la función puede devolver la salida correspondiente.

tesis de Church-Turing1

las funciones computables son exactamente las funciones que se pueden calcular utilizando un dispositivo de cálculo mecánico dada una cantidad ilimitada de tiempo y espacio de almacenamiento. De manera equivalente, esta tesis establece que cualquier función que tiene un algoritmo es computable.

función no computable famosas

Halting problem o calcular la Complejidad de Kolmogorov.

Sintaxis a través de gramáticas

objetivo con respecto a la sintaxis

es describir de forma compacta e inambigua el conjunto de los programas válidos

gramáticas independientes de contexto.

El estándar para gramáticas independientes de contexto de lenguajes de programación es EBNF. Sin embargo, las gramáticas independientes de contexto no son suficientemente expresivas para describir adecuadamente la mayor parte de lenguajes de programación. Por ejemplo es difícil expresar la obligación de que una variable sea declarada antes de ser usada, o bien describir la asignación múltiple de variables,

Semántica operacional vs. lambda cálculo

La semántica operacional

describe formalmente cómo se llevan a cabo cada uno de los pasos de un cálculo en un sistema computacional. Para eso se suele trabajar sobre un modelo simplificado de la máquina. Cuando describimos la semántica de un programa mediante semántica operacional, describimos cómo un programa válido se interpreta como secuencias de pasos computacionales. Estas secuencias son el significado del programa.

Cómo funcionan los lenguajes de programación

Estructura de un compilador

Mitchell 4.1.1.

Estructuras de datos de bajo nivel

Variables

Una variable está formada por una ubicación en la memoria y un identificador asociado a esa ubicación. Esa ubicación contiene un valor, El nombre de la variable es la forma usual de referirse al valor almacenado: esta separación entre nombre y contenido permite que el nombre sea usado independientemente de la información exacta que representa.

Los compiladores
reemplazan los nombres simbólicos de las variables con la real ubicación de los datos. Diferentes identificadores del código pueden referirse a una misma ubicación en memoria, lo cual se conoce como aliasing.

aliasing.
La ubicación de una variable se llama su L-valor

L-valor
el valor almacenado en esta ubicación se llama el R-valor de la variable.

R-valor

Estructura en bloques

Código estructurado vs. código spaghetti

programas que resulten incomprensibles porque al que lee le cuesta entender la estructura de control del programa. A este tipo de código se le llama código spaghetti,

código spaghetti,

bloque.

Un bloque es una región del texto del programa con inicio y fin explícitos e inambiguos. Esta región del texto permite organizar de forma explícita la lógica del programa. además, posibilita sucesivas abstracciones sobre el flujo de ejecución.

En primer lugar,
los bloques nos permiten hacer declaraciones de variables locales.

variable local

Una variable declarada dentro de un bloque se dice que es una variable local para ese bloque. Las variables que no están declaradas en un bloque, sino en algún otro bloque que lo contiene, se dice que es una variable global para el bloque más interno.

variable global

Estructura de bloque

propiedades:

Las nuevas variables se pueden declarar en varios puntos de un programa.

nuevas variables
Cada declaración es visible dentro de una determinada región de texto del programa, llamada bloque.

bloque.
si dos bloques contienen expresiones o declaraciones en común, entonces un bloque debe estar enteramente contenida dentro del otro.

Cuando un programa inicia la ejecución de las instrucciones contenidas en un bloque en tiempo de ejecución, se asigna memoria a las variables declaradas en ese bloque.

memoria
Cuando un programa sale de un bloque, parte o toda la memoria asignada a las variables declaradas en ese bloque se libera.

identificador de variable
Un identificador de variable que no está declarado en el bloque actual se considera global a ese bloque, y su referencia es a la entidad con el mismo identificador nombre que se encuentra en el bloque más cercano que contiene al bloque actual.

cómo se manejan en memoria tres clases de variables:

variables locales
se almacenan en la pila de ejecución, en el activation record asociado al bloque.

parámetros de función
también se almacenan en el activation record asociada con el bloque.

variables globales
que se declaran en algún bloque que contiene al bloque actual y por lo tanto hay que acceder a ellos desde un activation record que se colocó en la pila de ejecución antes del bloque actual.

Activation records

El contador de programa

es la dirección donde se

encuentra la instrucción de programa que se está ejecuntando actualmente.

El puntero de entorno

nos sirve para saber cuáles son los valores que se asignan a las variables que se están usando en una parte determinada del código.

El stack

cuando el programa entra en un nuevo bloque, se agrega a la pila una estructura de datos que se llama **activation record** (stack frame), que contiene el espacio para las variables locales declaradas en el bloque, Entonces, **el puntero de entorno** apunta al nuevo activation record. Cuando el programa sale del bloque, se retira el activation record de la pila y el puntero de entorno se restablece a su ubicación anterior, El **activation record** que se apila más recientemente es el primero en ser desapilado,

control link,

contiene el que será el puntero de entorno cuando se desapile el activation record actual,

dirección de retorno,

es donde se va a guardar el resultado de la ejecución de la función, si es que lo hay.

Detalle de ejecución de un activation record

bloques in line

Un bloque in line es un bloque que no es el cuerpo de una función o procedimiento.

un activation record también puede contener espacio para resultados intermedios, Estos son valores que no reciben un identificador de variable explícito en el código, pero que se guardan temporalmente para facilitar algún cálculo.

los valores de estas subexpresiones pueden tener que ser evaluados y almacenados en algún lugar antes de multiplicarse.