Investigación de Lenguajes - LIPS

Rodrigo Castro
Jorge Vergara
Oswaldo Bayona

23 de octubre de 2013

1. Introducción

2. Características

Es una familia de lenguajes que soporta la aplicación de varios estilos de programación (multiparadigma) aunque prevalece la programación funcional completamente declarativa.

El término programación declarativa se refiere al uso de sentencias con las cuáles se modela el problema pero no se especifica cómo resolverlo, el computador es el encargado de encontrar la solución utilizando inferencias matemáticas y lógicas, en contraste con la programación imperativa que requiere que el programador implemente la solución del problema mediante algoritmos.

Se lo utiliza principalmente para estudios en el área de la Inteligencia Artificial (AI)

Es muy sencillo y elegante, la implementación de programas recursivos es mucho más fácil que en el lenguaje C.

Las expresiones matemáticas o en general utilizan notación pre-fix por ejemplo:
(4 + (5 / 7)) se expresa + 4 / (5 7)
Se pueden declarar funciones y organizarlas en módulos, siguiendo el siguiente esquema:
(defun nombreDeLaFuncion (arg1, arg2, . . . ,argn)"Comentarios" (Cuerpo))

Todas las variables son punteros, una característica representativa de LISP es que las variables no existen como en otros lenguajes, aquí se llaman valores, representa los valores ubicándolos en zonas específicas de la memoria (dependiendo de la ubicación se simbolizan los valores de cadenas y números), conceptualmente se considera que existen todas las representaciones numéricas y de cadenas en las zonas de la memoria.

La memoria se constituye de dos partes: el diccionario de símbolos y la memoria de datos.
Es interpretado, aunque en algunas versiones también es compilado.
Sus tipos de datos átomos son: símbolos, strings, enteros, reales, racionales, par; y los tipos de datos compuestos son las Listas; los símbolos se refieren a nombres; en LISP las cadenas de caracteres son tipos de dato primitivos; los pares o conses (cons cell) es un tipo de dato atómico que se usa para estructurar otros datos primitivos en forma de pares, incluso se pueden agrupar otros pares y estructurar arboles binarios, su esquema es el siguiente (car . cdr), donde "car" es el primer átomo del par y "cdr" el segundo átomo; por otra parte una Lista es una estructura que se basa también en pares siendo más completa pero es más sencilla de entender.

La memoria utiliza el mecanismo de recolección de basura, si un dato ya no es referenciado es llevado al recolector.
Algunas versiones de LISP son: Emacs Lisp, Common Lisp, Scheme.

3. Historia

La historia del lenguaje LISP es muy extensa, he aquí un resumen; fue concebido originalmente en 1958 por John McCarthy y sus colaboradores en el Instituto Tecnológico de Massachusetts como parte de su labor científica, la inteligencia artificial, es el segundo lenguaje de alto nivel seguido de FORTRAN, ambos siguen usándose.

LISP (LISt Processing) fue el primer lenguaje desarrollado para estudiar la inteligencia artificial, lo nuevo que introdujo este lenguaje como aporte a la historia fueron las estructuras de datos de árbol, el compilador auto-contenido, tipos dinámicos etc.

En los años 70 la capacidad tecnológica en cuanto a hardware no estaba tan avanzada razón por la cual no era viable programar con LISP, haciendo de este lenguaje no tan popular y limitando su uso solo para inteligencia artificial en máquinas especialmente diseñadas llamadas maquinas Lisp, posteriormente quedaron obsoletas y la popularidad de Lisp tuvo un incremento.

A lo largo de los años fueron apareciendo nuevos dialectos de LISP, entre los cuales podemos mencionar a InterLISP, Franz Lisp, ZetaLisp; debido a esta gran variedad, entre los años 1980 y 1990 se creó un nuevo dialecto compatible con los anteriores llamado Common Lisp, otro dialecto muy importante es Scheme escrito por G. Steele y G. Sussman en 1970.

4. Tutorial de Instalación

Un interprete que podemos utilizar para programar en Lisp es DrRacket. Su instalacion es muy sencilla y rapida. El entorno de trabajo para programar en este lenguaje se puede apreciar en la Figura 5

Figura 1

5. Hola Mundo y otros Programas Introductorios

5.1. Hola Mundo

```
(defun HolaMundo()
        (print 'Hola Mundo')]
```

5.2. Factorial

```
;Calculo del factorial de un numero
(defun Factorial(n)
        (cond( ( lessp n 2) 1)
              (t  (* n  (Factorial(sub 1 n))]
```

Como se puede ver la funcion factorial tiene una estructura recursiva, adiferencia de los lenguajes de pogramacion tradificionales que tienen sentencias especializadas para realizar iteraciones en LIP no contamos con esas herramientas.
Descripcion de la funcion: 1.- como paso base vemos si el numero ingresado es menor

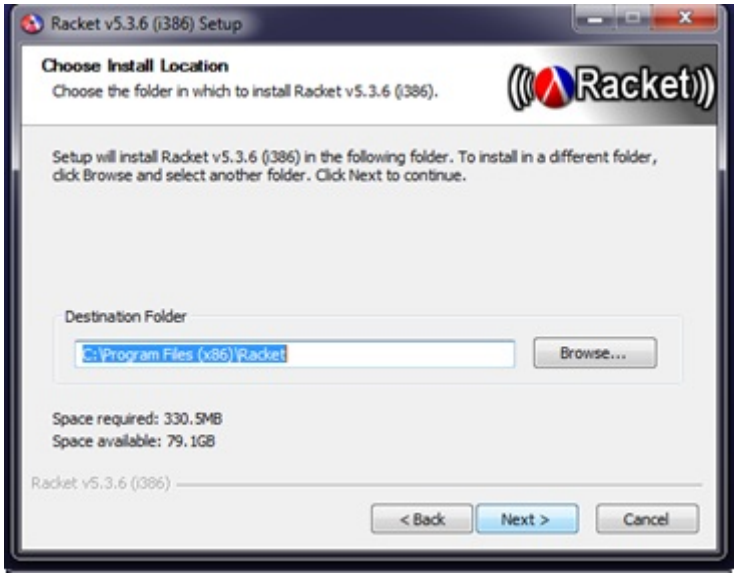


Figura 2

que 2 en este caso devolvemos 1 2.-si el numero no es menor que 2 este numero lo multipilcamos por el factorial de n-1

5.3. Calcular Media Aritmetica de una Lista

Para resolver este problema lo vamos a dividir en 3 funciones mas bacisas

La primera funcion que vamos a desarrolar es "Suma"que calcula la sumatoria de cada elemento de la lista

```
;Suma todos los elementos de una lista
(defun Suma(lista)
        (cond((null lista) 0 )
                ((atom lista) lista)
                (t  (+ (car lista) (suma (cdr lista))))
```

Detallando un poco la funcion:
1.- en el caso que la lista este vacia nuestra funcion devuelve 0 nuestro primer caso base
2.- si nuestra lista es un atomo es decir un simple numero devolvemos la lista este seria otro caso base
3.- en el caso que la lista aun tenga mas de 1 elemento tomamos el 1 elemento de la lista y lo sumamos al resto de elementos de la lista

Como segunda parte desarrolarremos una funcion que cuente la cantidad de elementos que se encuentran en la lista

Figura 3

```
;cuenta los elementos de la lista
(defun Contar(lista)
        (cond((null lista) 0 )
                ((atom lista) 1)
                (t (addl (contar (cdr lista)]
```

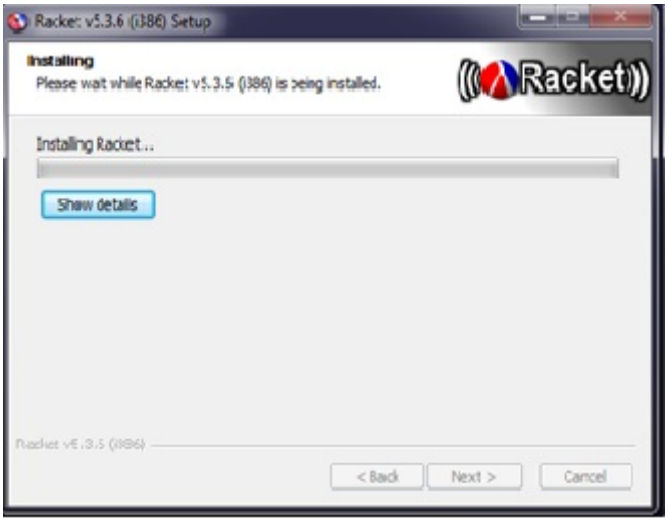
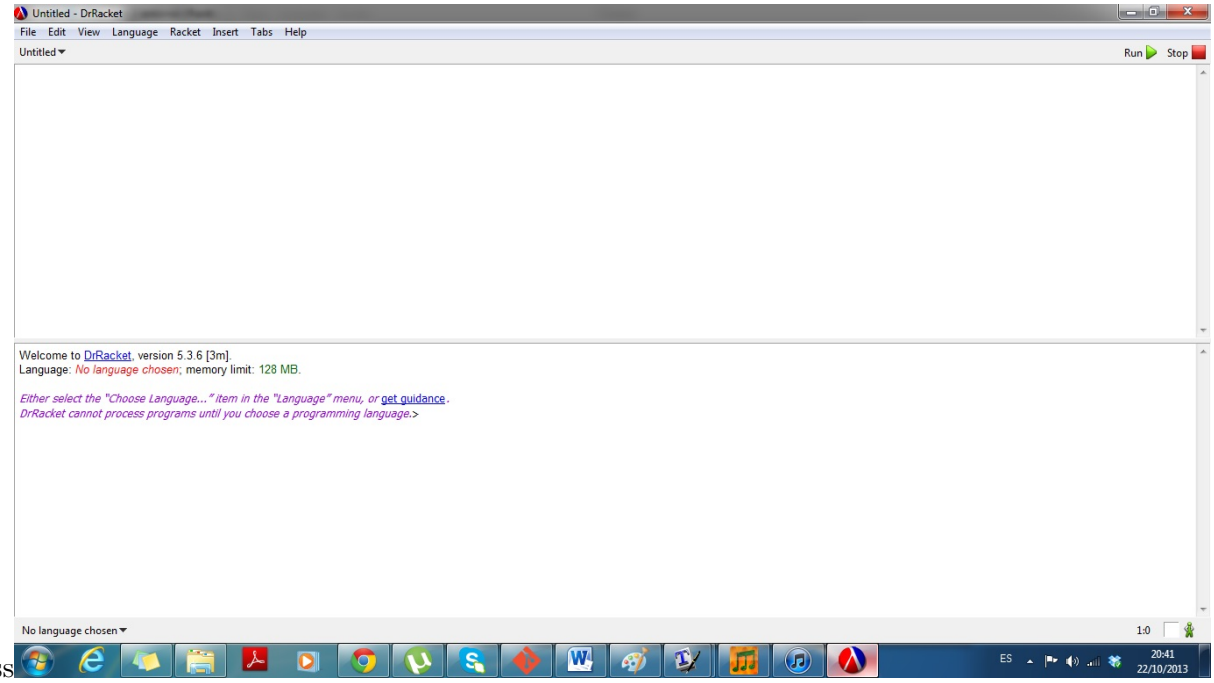


Figura 4



SS

Figura 5: //Entorno de trabajo de DrRocket