Lenguajes de Programación

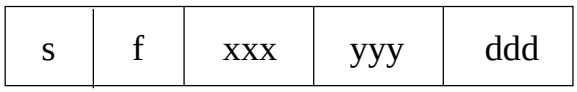
Dr. Carlos Artemio Coello Coello
Tarea No. 1
12 de enero de 2015

Tomaremos el pseudo-código proporcionado en el libro de texto, cuyas instrucciones guardan el formato ilustrado en la figura 1. Las operaciones disponibles para dicho pseudo-código, son las ilustradas en la tabla 1.

1. (**5 puntos**) Denominemos a las posiciones de memoria 402 y 761 como x & y, respectivamente. Codifique instrucciones para calcular $(x + y)^2$ y almacenar el resultado de la operación en la posición de memoria 100. Suponga que las primeras diez posiciones de la memoria de datos están disponibles para almacenamiento temporal.

2. (**5 puntos**) ¿Cómo pueden alterarse las operaciones +3 y -3 para que sean más regulares (es decir, para que se parezcan más a las otras operaciones), sin dejar de efectuar las mismas funciones que ya tienen asignadas (o sea, elevar al cuadrado y sacar raíz cuadrada)?

3. (**5 puntos**) Codifique instrucciones para calcular el valor absoluto del contenido de la posición 231 y almacenar el resultado en la posición 505. Suponga que las instrucciones que escribirá estarán en las posiciones de programa 102 y subsecuentes. Mencione cualquier otra suposición que realice.



s = signo
f = funcion
xxx = operando1
yyy = operando2
ddd = destino

Figure 1: Formato de las instrucciones usadas por nuestro pseudo-código

Operación #	+	-
0	mover	(sin utilizar)
1	+	-
2	×	÷
3	elevar al cuadrado	raíz cuadrada
4	if = goto	if ≠ goto
5	if ≥ goto	if < goto
6	x(y) → z	x → y(z)
7	incrementar y evaluar	(sin utilizar)
8	leer	imprimir
9	parar	(sin utilizar)

Table 1: Pseudo-código a utilizarse.

4. (**5 puntos**) Este pseudo-código no incluye un salto incondicional (goto). ¿Cómo implementaría un salto incondicional usando únicamente los comandos proporcionados?

5. (**5 puntos**) Codifique una instrucción que mueva el contenido de la posición de datos 100 a la posición de datos 101.

6. (**10 puntos**) ¿Cuál debería ser la función de la operación -0? Nuestro diseño simétrico nos lleva a esperar que esté relacionada con la operación **mover**. Sin embargo, debiera tratarse de una operación útil que no sea efectuada fácil o eficientemente con las otras operaciones definidas previamente.

7. (**5 puntos**) Suponga que un arreglo empieza en la posición 100 de la memoria de datos y que la posición 030 contiene el número de elementos del arreglo. Codifique instrucciones que sumen los elementos del arreglo y coloquen el resultado en la posición 005. Describa cualquier presuposición que realice.

8. (**5 puntos**) Suponga que un arreglo empieza en la posición 650 en la memoria de datos, y que la posición 907 contiene el número de elementos del arreglo. Codifique instrucciones que impriman todos los elementos del arreglo. Describa cualquier presuposición que realice.

9. (**5 puntos**) Suponga que un arreglo empieza en la posición 100 en la memoria de datos. Codifique instrucciones para leer números en elementos consecutivos del arreglo, hasta que se lea una tarjeta que contenga +9 999 999 999. Describa cualquier presuposición que realice.

10. (**5 puntos**) Escriba un programa completo (en pseudo-código) el cual lea tarjetas (hasta alcanzar una marcada con +9 999 999 999), sume los números que éstas contienen, e imprima la suma total.

11. (**5 puntos**) Escriba un programa completo (en pseudo-código) que imprima los cuadrados de los enteros del 1 al 100.

12. (**10 puntos**) Escriba un programa completo que lea los coeficientes de una ecuación cuadrática e imprima ambas raíces (en caso de que éstas existan). Puede suponerse que sólo nos interesan las raíces reales. Al resolver este ejercicio, probablemente le resulte de utilidad construir un *mapa de variables* que muestre las posiciones de memoria donde se encuentra almacenada cada variable. También le puede resultar útil usar etiquetas simbólicas hasta haber escrito suficiente del programa, a fin de facilitar la localización en memoria de las instrucciones.

Programación en Scheme

A fin de que vaya desarrollando destreza en el lenguaje Scheme, se irán asignando ejercicios con grados de dificultad incrementales. Los problemas que deberá resolver en esta ocasión son los siguientes:

1. (**5 puntos**) Considere el siguiente procedimiento:

```
(define misterio
  (lambda (ls)
    (if (null? (cddr ls))
        (cons (car ls) '())
        (cons (car ls) (misterio (cdr ls))))))
```

   ¿Qué cosa devuelve (**misterio** '(1 2 3 4 5))? Describa el comportamiento general de **misterio**. Sugiera un buen nombre para el procedimiento **misterio**.

2. (**5 puntos**) Defina un procedimiento **lista-de-primeros-elementos** que tome como argumento una lista compuesta de listas (no vacías) de elementos. Su valor es una lista compuesta de los primeros elementos de alto nivel de cada una de las sublistas. Pruebe su procedimiento con los siguientes ejemplos:

   (**lista-de-primeros-elementos** '((a) (b c d) (e f))) $\Longrightarrow$ (a b e)
   (**lista-de-primeros-elementos** '((1 2 3) (4 5 6))) $\Longrightarrow$ (1 4)
   (**lista-de-primeros-elementos** '((one))) $\Longrightarrow$ (one)
   (**lista-de-primeros-elementos** '()) $\Longrightarrow$ ()

3. (**5 puntos**) Defina un procedimiento **todos-iguales?** que tome una lista **ls** como su argumento y evalúe si todos los elementos de nivel superior de **ls** son iguales. Pruebe su procedimiento con los siguientes ejemplos:

(**todos-iguales?** '(a a a a a)) $\Longrightarrow$ #t
(**todos-iguales?** '(a b a b a b)) $\Longrightarrow$ ()
(**todos-iguales?** '((a b) (a b) (a b))) $\Longrightarrow$ #t
(**todos-iguales?** '(a)) $\Longrightarrow$ #t
(**todos-iguales?** '()) $\Longrightarrow$ #t

4. (**5 puntos**) Defina un procedimiento **remueve-todo-menos-el-ultimo**, el cual deberá borrar todas las instancias de alto nivel de una lista, excepto por la última. Pruebe su procedimiento con los siguientes ejemplos:

   (**remueve-todo-menos-el-ultimo** 'a '(b a n a n a s)) $\Longrightarrow$ (b n n a s)
   (**remueve-todo-menos-el-ultimo** 'a '(b a n a n a)) $\Longrightarrow$ (b n n a)
   (**remueve-todo-menos-el-ultimo** 'a '(c a r l o s)) $\Longrightarrow$ (c a r l o s)
   (**remueve-todo-menos-el-ultimo** 'a '(r o b e r t)) $\Longrightarrow$ (r o b e r t)
   (**remueve-todo-menos-el-ultimo** 'a '()) $\Longrightarrow$ ()

5. (**10 puntos**) Defina un procedimiento **plural-ingles** que tome como argumento una lista **ls** y cambie sus últimos elementos de acuerdo a las reglas siguientes:

   - Si el último elemento de alto nivel de la lista es **f**, entonces reemplazarla por una **v** y agregar **e s** al final de la lista.
   - Si los últimos dos elementos de alto nivel de la lista son **f e**, entonces reemplazar la **f** por una **v**, dejar la **e** intacta y agregar una **s** al final de la lista.
   - Si ninguna de las dos reglas anteriores aplica, entonces dejar la lista intacta.

   Pruebe su procedimiento con los siguientes ejemplos:

   (**plural-ingles** '(s h e l f)) $\Longrightarrow$ (s h e l v e s)
   (**plural-ingles** '(w i f e)) $\Longrightarrow$ (w i v e s)
   (**plural-ingles** '(p e o p l e)) $\Longrightarrow$ (p e o p l e)
   (**plural-ingles** '(f e e l)) $\Longrightarrow$ (f e e l)
   (**plural-ingles** '(r e f e r e e)) $\Longrightarrow$ (r e f e r e e)
   (**plural-ingles** '(c o f f e e)) $\Longrightarrow$ (c o f f e e)
   (**plural-ingles** '(g i f t)) $\Longrightarrow$ (g i f t)
   (**plural-ingles** '(f e f e f e)) $\Longrightarrow$ (f e f e v e s)
   (**plural-ingles** '()) $\Longrightarrow$ ()

Le resultará útil leer los capítulos 1 y 2 de los apuntes de Scheme que están disponibles en la página web del curso.

La solución a TODOS estos problemas deberá incluirse en un solo archivo ASCII (llamado **sol-tarea1.scm**) en un CD que se entregará junto con el resto

de su tarea. Es necesario que también proporcione una versión impresa de sus programas y de sus respuestas a los problemas de Scheme en el reporte que entregue. Las soluciones deben implementarse con los nombres que se indican en cada problema de los antes enumerados.

(**Bonificación: 10 puntos**) ¿Cree que nuestras capacidades de pensamiento están influenciadas por nuestro lenguaje (hipótesis de Sapir-Whorf)? Sustente adecuadamente su opinión. Traslade ahora la pregunta al contexto de los lenguajes de programación. Discuta.

Fecha de entrega: *Jueves 21 de enero a las 12:00hrs.* Toda tarea entregada tarde será penalizada con 10% (sobre la calificación obtenida) por cada periodo de 24 horas que se retrase su entrega.