

Lenguajes de Programación
Facultad de ciencias
Universidad Nacional Autónoma de México
Notas para el segundo examen parcial

Montoya Montes Pedro

2 de mayo de 2019

Índice

1. Definiciones dadas por Ricardo (No formales)	2
2. Ejercicio de Memoization	3
3. Recolector de basura	3
3.1. Tipos de recolectores de basura	3
3.2. Pros y contras de cada tipo de recolector de basura	4
3.3. Forma de recolección	4
3.4. Técnicas de recolección	4

1. Definiciones dadas por Ricardo (No formales)

Definición 1 *Tipo: Abstracción de un conjunto de datos.*

Definición 2 *Tipo básico: Son las primitivas del lenguaje.*

Definición 3 *Tipo abstracto: Son aquellas construidas a partir de las primitivas.*

Definición 4 *Juicio de tipo: Es un conjunto de reglas que determina la correctud de tipos de una expresión.*

Definición 5 *Inferencia: Determina los valores de los tipos de una expresión.*

Definición 6 *Lenguaje sólido: Son aquellos que se sabe el comportamiento de sus operaciones*

Definición 7 *Lenguaje seguro: Son aquellos que las operaciones se aplican con el tipo correcto.*

Definición 8 *Lenguaje envolvente: Es aquel que alberga código ajeno al lenguaje.*

Definición 9 *Lenguaje incrustado: Es el código escrito dentro de otro lenguaje.*

Definición 10 *Variable de tipo: Determina el tipo de cosas polimorficas.*

2. Ejercicio de Memoization

El siguiente es un ejemplo de memoization con la función "lucas".

```
(define lucas (lambda (n))
  (cond
    [(= n 0) 1]
    [(= n 1) 2]
    [else (+ (lucas (- n 1)) (lucas (- n 2)))])
```

Creacion del diccionario:

```
(define dic (make_hash '()))
(hash_set dic 0 1)           //Caso base 1 de lucas.
(hash_set dic 2 1)           //Caso base 1 de lucas.
```

Creación de la memoization:

```
(define (lucas n)           //Conservamos el nombre original.
  (cond
    [(hash_ref dic n #f) (hash_ref dic n #f)]
    [else (let m (+ (lucas (- n 1)) (lucas (- n 2)))
      (begin (hash_set dic n m) m))])
```

3. Recolector de basura

3.1. Tipos de recolectores de basura

1. Marcado y compactación.

0x05	
0x04	
0x03	
0x02	0x05
0x01	0x02

2. Marcado y barrido.

0x05	
0x04	0x04
0x03	
0x02	0x02
0x01	

3. Copia y pega usando semiespacios.

0x05	0x4				
0x04		0x3			
0x03			0x2		
0x02	0x5	0x1		0x05	
0x01				0x04	
				0x02	

Heap

Heap

3.2. Pros y contras de cada tipo de recolector de basura

1. Marcado y compactación:
 - No hay fragmentación de memoria.
 - Eficiente en memoria.
2. Marcado y barrido:
 - Deja fragmentación en memoria.
 - Eficiente en tiempo.
3. Copia y pega usando semiespacios:
 - Pierde la mitad del heap.
 - Pierde tiempo en acomodo, pero no tanto como en marcado y compactación.

3.3. Forma de recolección

Definición 11 *Conjunto raíz:?*

Definición 12 *Formas de detección de basura:*

1. *Con caminos.*
2. *Con índices.*

3.4. Tecnicas de recolección

Existen dos tecnicas de recolección:

1. Tiempo: Es aquel que en un periodo de tiempo se ejecuta el recolector de basura (se puede también diseñar de tal manera en la que se ejecute en intervalos de tiempo).
2. Espacio: Es aquel en la que cada cantidad de objetos basura se ejecuta el recolector de basura.

4. RP3

1. Definiciones:
 - a) Recolector de Basura: Es un algoritmo el cual limpia la memoria de forma automatizada.
 - b) Conjunto raíz: ?