Facultad de Ciencias, UNAM
Lenguajes de Programación
Tarea 1

Rubí Rojas Tania Michelle

09 de octubre de 2020

1. Explica brevemente qué tipos de problemas puedes resolver con cada uno de los siguientes paradigmas y nombre un lenguaje perteneciente a cada uno.
   a) Paradigma Estructurado.
   b) Paradigma Orientado a Objetos.
   c) Paradigma Funcional.
   d) Paradigma Lógico.
2. Usando el lenguaje de programación Prolog, dar un ejemplo de cada uno de los siguientes conceptos y justificar. No es necesario que sean ejemplos demasiado elaborados.
   a) Sintaxis
   b) Semántica
   c) Convenciones de programación (Idioms)
   d) Bibliotecas
3. Dada la siguiente función, da una forma para la misma indicando el tipo de entrada de los parámetros, el tipo de la salida y asígnale un nombre mnemotécnico. Justifica tu respuesta.

```
(define (foo n l)
   (cond
      [(zero? n) empty]
      [else (cons (car l) (foo (sub1 n) (cdr l)))]))
```

4. Para los siguientes incisos, calcular el resultado de aplicar la función al parámetro recibido. Mostrar cada paso realizado hasta obtener el resultado final. Da tu propia implementación para ambas funciones.
   a) `(reverse '(1 7 2 9))`
   b) `(append '(m a n) (z a n a))`
   c) `(reverse (append '(m a n) (z a n a)))`

   Solución: Las implementaciones propuestas para las funciones `reverse` y `append` son las siguientes

```
(define (append xs ys)
  (if (empty? xs)
      ys
      (cons (car xs) (append (cdr xs) ys))))
```

```
(define (reverse xs)
  (if (empty? xs)
      '()
      (append (reverse (cdr xs)) (list (car xs)))))
```

Por lo tanto,

a) `(reverse '(1 7 2 9))`

```
= (append (reverse (cdr '(1 7 2 9))) (list (car '(1 7 2 9))))
= (append (reverse '(7 2 9)) (list 1))
= (append (append (reverse (cdr '(7 2 9))) (list (car '(7 2 9)))) '(1))
= (append (append (reverse '(2 9)) (list 7)) '(1))
= (append (append (append (reverse (cdr '(2 9)))
                          (list (car '(2 9)))) '(7)) '(1))
= (append (append (append (reverse '(9)) (list 2)) '(7)) '(1))
= (append (append (append (append (reverse (cdr '(9)))
                                  (list (car '(9)))) '(2)) '(7)) '(1))
= (append (append (append (append (reverse '()) (list 9)) '(2)) '(7)) '(1))
= (append (append (append (append '() '(9)) '(2)) '(7)) '(1))
= (append (append (append '(9) '(2)) '(7)) '(1))
= (append (append (cons (car '(9)) (append (cdr '(9)) '(2))) '(7)) '(1))
= (append (append (cons 9 (append '() '(2))) '(7)) '(1))
= (append (append (cons 9 '(2)) '(7)) '(1))
= (append (append '(9 2) '(7)) '(1))
= (append (cons (car '(9 2)) (append (cdr '(9 2)) '(7))) '(1))
= (append (cons 9 (append '(2) '(7))) '(1))
= (append (cons 9 (cons (car '(2)) (append (cdr '(2)) '(7)))) '(1))
= (append (cons 9 (cons 2 (append '() '(7)))) '(1))
= (append (cons 9 (cons 2 '(7))) '(1))
= (append (cons 9 '(2 7)) '(1))
= (append '(9 2 7) '(1))
= (cons (car '(9 2 7)) (append (cdr '(9 2 7)) '(1)))
= (cons 9 (append '(2 7) '(1)))
= (cons 9 (cons (car '(2 7)) (append (cdr '(2 7)) '(1))))
= (cons 9 (cons 2 (append '(7) '(1))))
= (cons 9 (cons 2 (cons (car '(7)) (append (cdr '(7)) '(1)))))
= (cons 9 (cons 2 (cons 7 (append '() '(1)))))
= (cons 9 (cons 2 (cons 7 '(1))))
= (cons 9 (cons 2 '(7 1)))
= (cons 9 '(2 7 1))
= '(9 2 7 1)
```

b) `(append '(m a n) (z a n a))`

```
= (cons (car '(m a n)) (append (cdr '(m a n)) '(z a n a)))
= (cons m (append '(a n) '(z a n a)))
= (cons m (cons (car '(a n)) (append (cdr '(a n)) '(z a n a))))
= (cons m (cons a (append '(n) '(z a n a))))
= (cons m (cons a (cons (car '(n)) (append (cdr '(n)) '(z a n a)))))
= (cons m (cons a (cons n (append '() '(z a n a)))))
= (cons m (cons a (cons n '(z a n a))))
= (cons m (cons a '(n z a n a)))
= (cons m '(a n z a n a))
= '(m a n z a n a)
```

c) (reverse (append '(m a n) (z a n a)))

```
= (reverse (cons (car '(m a n)) (append (cdr '(m a n)) '(z a n a))))
= (reverse (cons m (append '(a n) '(z a n a))))
= (reverse (cons m (cons (car '(a n)) (append (cdr '(a n)) '(z a n a)))))
= (reverse (cons m (cons a (append '(n) '(z a n a)))))
= (reverse (cons m (cons a (cons (car '(n)) (append (cdr '(n)) '(z a n a))))))
= (reverse (cons m (cons a (cons n (append '() '(z a n a))))))
= (reverse (cons m (cons a (cons n '(z a n a)))))
= (reverse (cons m (cons a '(n z a n a))))
= (reverse (cons m '(a n z a n a)))
= (reverse '(m a n z a n a))
= (append (reverse (cdr '(m a n z a n a))) (list (car '(m a n z a n a))))
= (append (reverse '(a n z a n a)) (list m))
= (append (append (reverse (cdr '(a n z a n a))) (list (car '(a n z a n a)))) '(m))
= (append (append (reverse '(n z a n a)) (list a)) '(m))
= (append (append (append (reverse (cdr '(n z a n a))) (list (car '(n z a n a))))
                  '(a)) '(m))
= (append (append (append (reverse '(z a n a)) (list n)) '(a)) '(m))
= (append (append (append (append (reverse (cdr '(z a n a))) (list (car '(z a n a))))
                          '(n)) '(a)) '(m))
= (append (append (append (append (reverse '(a n a)) (list z)) '(n)) '(a)) '(m))
= (append (append (append (append (append (reverse (cdr '(a n a))) (list (car '(a n a))))
                                  '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (append (reverse '(n a)) (list a)) '(z)) '(n))
                  '(a)) '(m))
= (append (append (append (append (append (append (reverse (cdr '(n a)))
                                                  (list (car '(n a))))
                                          '(a)) '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (append (append (reverse '(a)) (list n))
                                          '(a)) '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (append (append (append (reverse (cdr '(a)))
                                                          (list (car '(a))))
                                                  '(n)) '(a)) '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (append (append (append (reverse '()) (list a))
                                                  '(n)) '(a)) '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (append (append (append '() '(a))
                                                  '(n)) '(a)) '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (append (append '(a) '(n)) '(a)) '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (append (cons (car '(a)) (append (cdr '(a)) '(n)))
                                          '(a)) '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (append (cons a (append '() '(n)))
                                          '(a)) '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (append (cons a '(n)) '(a)) '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (append '(a n) '(a)) '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (cons (car '(a n)) (append (cdr '(a n)) '(a)))
                                  '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (cons a (append '(n) '(a))) '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (cons a (cons (car '(n)) (append (cdr '(n)) '(a))))
                                  '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (cons a (cons n (append '() '(a))))
                                  '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (cons a (cons n '(a))) '(z)) '(n)) '(a)) '(m))
= (append (append (append (append (cons a '(n a)) '(z)) '(n)) '(a)) '(m))
= (append (append (append (append '(a n a) '(z)) '(n)) '(a)) '(m))
= (append (append (append (cons (car '(a n a)) (append (cdr '(a n a)) '(z)))
                          '(n)) '(a)) '(m))
= (append (append (append (cons a (append '(n a) '(z))) '(n)) '(a)) '(m))
= (append (append (append (cons a (cons (car '(n a)) (append (cdr '(n a)) '(z))))
                          '(n)) '(a)) '(m))
= (append (append (append (cons a (cons n (append '(a) '(z)))) '(n)) '(a)) '(m))
= (append (append (append (cons a (cons n (cons (car '(a)) (append (cdr '(a)) '(z)))))
                          '(n)) '(a)) '(m))
= (append (append (append (cons a (cons n (cons a (append '() '(z)))))
                          '(n)) '(a)) '(m))
= (append (append (append (cons a (cons n (cons a '(z)))) '(n)) '(a)) '(m))
= (append (append (append (cons a (cons n '(a z))) '(n)) '(a)) '(m))
= (append (append (append (cons a '(n a z)) '(n)) '(a)) '(m))
= (append (append (append '(a n a z) '(n)) '(a)) '(m))
= (append (append (cons (car '(a n a z)) (append (cdr '(a n a z)) '(n))) '(a)) '(m))
= (append (append (cons a (append '(n a z) '(n))) '(a)) '(m))
= (append (append (cons a (cons (car '(n a z)) (append (cdr '(n a z)) '(n))))
                  '(a)) '(m))
= (append (append (cons a (cons n (append '(a z) '(n)))) '(a)) '(m))
= (append (append (cons a (cons n (cons (car '(a z)) (append (cdr '(a z)) '(n)))))
                  '(a)) '(m))
= (append (append (cons a (cons n (cons a (append '(z) '(n))))) '(a)) '(m))
= (append (append (cons a (cons n (cons a (cons (car '(z)) (append (cdr '(z)) '(n))))))
                  '(a)) '(m))
= (append (append (cons a (cons n (cons a (cons z (append '() '(n)))))) '(a)) '(m))
= (append (append (cons a (cons n (cons a (cons z '(n))))) '(a)) '(m))
= (append (append (cons a (cons n (cons a '(z n)))) '(a)) '(m))
= (append (append (cons a (cons n '(a z n))) '(a)) '(m))
= (append (append (cons a '(n a z n)) '(a)) '(m))
= (append (append '(a n a z n) '(a)) '(m))
= (append (cons (car '(a n a z n)) (append (cdr '(a n a z n)) '(a))) '(m))
= (append (cons a (append '(n a z n) '(a))) '(m))
= (append (cons a (cons (car '(n a z n)) (append (cdr '(n a z n)) '(a)))) '(m))
= (append (cons a (cons n (append '(a z n) '(a)))) '(m))
= (append (cons a (cons n (cons (car '(a z n)) (append (cdr '(a z n)) '(a))))) '(m))
= (append (cons a (cons n (cons a (append '(z n) '(a))))) '(m))
= (append (cons a (cons n (cons a (cons (car '(z n)) (append (cdr '(z n)) '(a)))))) '(m))
= (append (cons a (cons n (cons a (cons z (append '(n) '(a)))))) '(m))
= (append (cons a (cons n (cons a (cons z (cons (car '(n)) (append (cdr '(n)) '(a))))))) (m))
= (append (cons a (cons n (cons a (cons z (cons n (append '() '(a))))))) '(m))
= (append (cons a (cons n (cons a (cons z (cons n '(a)))))) '(m))
= (append (cons a (cons n (cons a (cons z '(n a))))) '(m))
= (append (cons a (cons n (cons a '(z n a)))) '(m))
= (append (cons a (cons n '(a z n a))) '(m))
= (append (cons a '(n a z n a)) '(m))
= (append '(a n a z n a) '(m))
= (cons (car '(a n a z n a)) (append (cdr '(a n a z n a)) '(m)))
= (cons a (append '(n a z n a) '(m)))
= (cons a (cons (car '(n a z n a)) (append (cdr '(n a z n a)) '(m))))
= (cons a (cons n (append '(a z n a) '(m))))
= (cons a (cons n (cons (car '(a z n a)) (append (cdr '(a z n a)) '(m)))))
= (cons a (cons n (cons a (append '(z n a) '(m)))))
= (cons a (cons n (cons a (cons (car '(z n a)) (append (cdr '(z n a)) '(m))))))
= (cons a (cons n (cons a (cons z (append '(n a) '(m))))))
= (cons a (cons n (cons a (cons z (cons (car '(n a)) (append (cdr '(n a)) '(m)))))))
= (cons a (cons n (cons a (cons z (cons n (append '(a) '(m)))))))
= (cons a (cons n (cons a (cons z (cons n (cons (car '(a)) (append (cdr '(a)) '(m))))))))
= (cons a (cons n (cons a (cons z (cons n (cons a (append '() '(m))))))))
= (cons a (cons n (cons a (cons z (cons n (cons a '(m)))))))
= (cons a (cons n (cons a (cons z (cons n '(a m))))))
= (cons a (cons n (cons a (cons z '(n a m)))))
= (cons a (cons n (cons a '(z n a m))))
= (cons a (cons n '(a z n a m)))
= (cons a '(n a z n a m))
= '(a n a z n a m)
```

5. Da una tabla donde expliques las principales diferencias entre un compilador y un intérprete.

6. Dibuja un mapa mental que muestre las fases de generación de código ejecutable, sus principales características y elementos involucrados.

7. Dadas las siguientes expresiones de AE en sintaxis concreta, da su respectiva representación en sintaxis abstracta por medio de los Árboles de Sintaxos Abstracta correspondientes. En caso de no poder generar el árbol, justificar.

a) `{+ 18 {- 15 {+ 40 5}}}`

b) `{+ {- 15 {+ 40}}}`