Auxiliar 4
Funciones de Primer Orden y Scope

P1. ***Preguntas teóricas***

(a) ¿Es posible implementar funciones recursivas en el lenguaje visto en clases hasta ahora? ¿Por qué?

(b) Explique por qué la gran mayoría de los lenguajes adoptan el scope estático por defecto.

(c) Indique un escenario donde es útil tener scope dinámico.

P2. ***¿¿¿Primer que???***

Consideremos el lenguage *WAE* extendido con funciones que toman un argumento y la sentencia if0 que tiene el mismo funcionamiento que if, solo que la condición que verifica es si el primer argumento es o no 0 (*ExWAE*):

```
<ExWAE> ::= <num>
          | (add <ExWAE> <ExWAE>)
          | (sub <ExWAE> <ExWAE>)
          | with (<id> <ExWAE>) <ExWAE>
          | <id>
          | (app <id> <ExWAE>)
          | (if0  <ExWAE> <ExWAE> <ExWAE>)

<FunDef> ::= (fundef <id> <id> <expr>)
```

(a) Defina funciones *sum* y *fib* en *ExWAE* que calculan la suma de los números hasta el argumento y los números de fibonacci respectivamente.

(b) Defina las funciones mutuamente recursivas *even?* y *odd?* en *ExWAE*.

P3. ***¿Dinamico o Léxico?***

Para los siguientes programas indique sus resultados considerando que se presenta Scope Dinámico y Léxico.

(a)
```
(define my-funcs (list (fundef 'f 'y (parse '{+ y x}))))
  (run '{with {x 5}
          {+ {with {x 8} x}
             {f 3}}} my-funcs)
```

(b)
```
(define my-funcs (list (fundef 'f 'y (parse '{+ x n}))
                       (fundef 'g 'x (parse '{+ x {f x}}))))
  (run '{with {n 5} {g n}} my-funcs)
```