



Fundamentos de lenguajes de programación

Duración 2 horas

Carlos Andres Delgado S, Msc

`carlos.andres.delgado@correounivalle.edu.co`

9 de Febrero de 2021

Importante: Debe explicar el procedimiento realizado en cada uno de los puntos, no se considera válido únicamente mostrar la respuesta.

1. Reglas

- Entregue un sólo archivo en formato PDF del examen y los archivos separados para los puntos 1, 2 y 3. Se le sancionará con 0.3 en la nota si no cumple la regla.
- No use enlaces externos para las capturas, no se valdrán.
- Las capturas de los puntos deben estar en buena calidad, si alguna no se entiende no se le valdrá el punto en cuestión.
- Sea ordenado en las capturas de sus puntos, no se valdrán puntos desorganizados y que no sea fácil entender su respuesta.
- Entregue el examen por google forms, no se aceptarán entregas por correo
- Entregue el examen antes del 9 de Febrero a las 4:20pm, de lo contrario no se recibirá.
- Puede hacer el examen en parejas, al inicio del PDF de entrega coloque los nombres y código de los estudiantes. No se aceptarán reclamos si no coloca correctamente estos datos.

2. Enunciado

1. (25 puntos) Para el caso del árbol ternario:

```
<arbol-t> ::= ' ( )
              arbol-vacio()
              ::= simbolo <arbol-t> <arbol-t> <arbol-t>
              arbol-novacio(key, hij1,hij2,hij3)
```

Diseñe las funciones **recorrido-preorden**, **recorrido-inorden**, **recorrido-posorden** las cuales reciben un arbol-t y retornan una lista de símbolos representando los respectivos recorridos. Use **datatypes**

2. (25 puntos) Haga una representación basada en procedimientos del siguiente tipo de dato:

```
<lstbool> ::= ' ( )
              lst-vacio()
              ::= <booleano> <lstbool>
              lst-novacio(elm, rest)
```

Realice la función **contar-true**, la cual retorna el número de valores booleano true (verdadero) presentes en este tipo de dato.

3. (25 puntos) Se desean agregar las listas a nuestro lenguaje bajo la siguiente gramática:

```
<expresion> ::= "[ (expresion ',')* "]"
              list-exp(elementos)
```

- Indique la producción que se agrega a la gramática
- Indique la modificación que se haría en la función evaluar-expresion

Entregue el interpretador de procedimientos recursivos con esta regla introducida, haga comentarios donde hizo los cambios para validar su trabajo.

4. (25 puntos) Dibuje los ambientes para la expresión:

```
let
  f=proc(x, y) +(* (2,x), *(x,y))
  a = 5
  b = 4
  in
  let rec
    funcion(x,y) = if >(x,0)
                    then +(y, (funcion -(x,1) +(y,1)))
                    else (proc(x) +(x,2) a)
  in
  let
```

Handwritten annotations:

- Handwritten "4" above "let" and "20" above "f=proc".
- Handwritten "28" above "a = 5".
- Handwritten "28" above "b = 4".
- Handwritten "closure env x 0" with "closure" underlined, below "let".

```

k = (f b a)
in   (function a k)

```

