SEGUNDO EXAMEN FINAL DE ESTRCTURA DE LOS LENGUAJES 21/12/2016

TEMA 1 (10 puntos)

a) Escriba una gramática para un lenguaje que consiste en cadenas que tienen n copias de la letra a seguidas de un número igual de copias de la letra b, donde n> 0. Por ejemplo, las cadenas ab, aabb, aaaabbbb y aaaaaaaabbbbbbbb son parte del lenguaje pero a, abb, ba y aaabb no lo son. (6 puntos)
b) Escriba *parse trees* para las sentencias ab y aabb, que son derivadas de la gramática del inciso a. (4 puntos)

TEMA 2 (5 puntos)

Probar que la siguiente gramática es ambigua:

<S> → <A>
<A> → <A> + <A> | <id>
<id> → a | b | c

TEMA 3 (10 puntos)

a) Defina *static scoping* y *dynamic scoping*. Cuáles son las reglas que se aplican en cada caso (5 puntos)
b) Había 3 ejercicios y tenías que elegir uno: (5 puntos)

```
#1)

var x;
function sub1() {
   var x;
function sub2() {
    . . .
}
}
function sub3() {
    . . .
}
```

Asuma que la ejecución de este programa se da en el siguiente orden
`main` llama a `sub1`
`sub1` llama a `sub2`
`sub2` llama a `sub3`
a. Asumiendo *static scoping*, ¿cuál es la declaración correcta para hacer referencia a x en las siguientes funciones?
i. `sub1`
ii. `sub2`
iii. `sub3`

b. Repita el inciso a, pero asumiendo *dynamic scoping*.

#2) Asuma que el siguiente programa fue interpretado usando *static-scoping*. ¿ Cuál valor de `x` es mostrado en la función `sub1`?

Bajo reglas de *dynamic-scoping*. ¿ Cuál valor de `x` es mostrado en la función `sub1`?

```
var x;
function sub1() {
       document.write("x = " + x + "");
}
function sub2() {
       var x;
       x = 10;
       sub1();
}
x = 5;
sub2();
```

#3) Considere el siguiente programa en JavaScript:

```
var x, y, z;
function sub1() {

       var a, y, z;

       function sub2() {

              var a, b, z;

              . . .

       }
       . . .
}

function sub3() {
              var a, x, w;
              . . .
}
```

Enumere todas las variables, junto con las unidades de programa en las que se declaran, que son visibles en los cuerpos de sub1, sub2 y sub3, suponiendo que se utiliza *static-scoping.*

TEMA 4 (5 puntos)

Un array se puede almacenar en *row-major-order* como se hace en C++ o en *column-major-order* como ocurre en FORTRAN.

Escriba una función de acceso para *row-major-order*

TEMA 5 (18 puntos) 3 puntos cada uno

a) Defina *named type equivalence* y *structure type equivalence*

b) Cite las cuestiones de diseño de los selectores de 2 vías

c) ¿Qué es un *heap-dynamic array*?. ¿Cuál es su ventaja (dar un ejemplo)?. ¿Cuál es su desventaja?

d) ¿Cuál es la diferencia entre un *static array* y un *heap dynamic array*?

e) Cite las 6 cuestiones de diseño de las expresiones aritméticas según Sebesta

f) Cite las cuestiones de diseño de los registros.

TEMA 6 (8 PUNTOS)

Considere el siguiente programa escrito en C:

```
int fun(int *i) {
        *i += 5;
        return 4;
}

void main() {
        int x = 3;
        x = x + fun(&x);
}
```

a) Cite cuáles son los tipos de orden de evaluación posibles de los operadores. (2 puntos)
b) ¿Cuál es el valor de x después de la asignación en el main en cada uno de los tipos? (6 puntos)

TEMA 7 (15 PUNTOS)

a) Dibuje la organización del almacenamiento en tiempo de ejecución. Describa brevemente sus partes (2 puntos)
b) Dibuje un registro de activación. Describa brevemente sus partes (2 puntos)
c) Completa:
   Los pasos que se siguen al ser llamado un procedimiento (3 puntos)
   1) ……
   2) ……
   3) ……
   4) ……
   5) ……
   Los pasos que se siguen cuando el procedimiento sale de escena (3 puntos)
   1) ……
   2) ……
   3) ……
   4) ……

d) Describa un ambiente de ejecución basado en pila sin procedimientos locales (5 puntos)