PRÁCTICA Unidad 3 – ENTIDADES Y LIGADURAS

1. ¿Qué cuestiones deben tenerse en cuenta en el diseño de los nombres?
2. ¿Qué es un alias?
3. ¿Qué es el l-value y el r-value de una variable?
4. ¿Cuál de las siguientes formas resulta mejor para la lectura? Justifique.
   SumaDeSaldos Suma_de_saldos SUMADESALDOS
5. Defina vinculación (binding) y tiempo de vinculación (binding time).
6. Al estudiar la carga de variables a memoria se las divide en 4 categorías. ¿Cuáles son? Defina cada una. Analice sus ventajas y desventajas.
7. Defina coerción, error de tipo, chequeo de tipo y fuertemente tipado.
8. Dé ejemplos de lenguajes fuertemente tipados y débilmente tipados.
9. Defina los 3 métodos de compatibilidad de tipos. Analice los méritos de cada uno.
10. Defina alcance (scope) estático y dinámico.
11. ¿Qué es el entorno de referencia de una sentencia o instrucción?
12. Analice ventajas y desventajas del alcance dinámico.
13. Describa las vinculaciones ejecutadas por una declaración externa de C. ¿Por qué es necesario este tipo de declaración?
14. C y C++ hacen distinción entre una declaración y una definición. ¿Cuáles de las siguientes son declaraciones y cuáles son definiciones?:

```
char * name;
struct rec;
double y;
typedef int * IntPtr;
int gcd(int,int);
extern int z;
```

15. Dado el siguiente programa:

```
Program P
  var X,Y,Z,A:Integer;
  Procedure Q
    var Z,D:Boolean;
    Procedure S
      var X,B:Integer;
          . . . R . . .
    End S;
    . . . S . . .
  End Q;
  Procedure R
    var W,C:Integer;
    if Y>=1 then
      Y:=Y-1;
    . . . Q . . .
  End R;
  Y:=1;
  . . . R . . .
End P.
```

Indique qué identificadores determinan el ambiente de referencia (local y no local) de cada unidad de programa, para los casos en que en el lenguaje se definan:
- Reglas de alcance estático
- Reglas de alcance dinámico

16. Considere el siguiente programa Pascal:

```
program main;
    var x : integer;
    procedure sub3; forward;
    procedure sub1;
            var x : integer;
            procedure sub2;
                    begin  (sub2)
                    …….
                    end;  (sub2)
            begin  (sub1)
            .....
            end;  (sub1)
    procedure sub3;
            begin  (sub3)
            ......
            end;  (sub3)
    begin  (main)
    ......
    end.  (main)
```

Asuma que:
Main llama a sub1
Sub1 llama a sub2
Sub2 llama a sub3

a. Con alcance estático, cuál es la declaración de x válida en:
   i. sub1
   ii. sub2
   iii. sub3
b. Repita el punto anterior teniendo en cuenta alcance dinámico

17. Para cada caso, liste qué variables están visibles en sub1, sub2 y sub3 asumiendo alcance estático.

```
program main;
        var x,y,z : integer;
        procedure sub1;
        var a,y,z:integer
                procedure sub2;
                var a,b,z : integer;
                begin  (sub2)
                ……..
                end;  (sub2)
        begin  (sub1)
        ……..
        end;  (sub1)
        procedure sub3;
                var a,x,w:integer
                begin  (sub3)
                ……..
                end;  (sub3)
begin  (main)
……
end.  (main)
```

```
program main;
      var x,y,z : integer;
      procedure sub1;
            var a,y,z:integer
            begin  (sub1)
            ……..
            end;  (sub1)
      procedure sub2;
            var a,x,w : integer;
            procedure sub3;
                  var a,b,z:integer
                  begin  (sub3)
                  ……..
                  end;  (sub3)

            begin  (sub2)
            ……..
            end;  (sub2)
begin  (main)
……
end.  (main)
```

18. Considere la siguiente función y liste las variables visibles indicando su definición para los 4 puntos indicados:

```
void fun(void) {
int a,b,c;   /* definición 1 */
……
while (…..) {
        int b,c,d;    /* definición 2 */
        …….
        while                                   (……) {
                int c,d,e;    /* definición 3 */
                ……
                }
        …..
        }
…….
}
```

1 → 2 → 3 → 4

19. Considere el siguiente programa:

```
void fun1(void);
void fun2(void);
void fun3(void);
void main() {
int a,b,c;
……..
}
void fun1(void) {
int b,c,d;
……..
}
void fun2(void) {
int c,d,e;
……..
}
void fun3(void) {
int d,e,f;
……..
}
```

Asumiendo alcance dinámico, determine qué variables son visibles en la última función llamada. Referencie para cada variable dónde fue definida.

a. main llama a fun1, fun1 llama a fun2, fun2 llama a fun3
b. main llama a fun1, fun1 llama a fun3
c. main llama a fun2, fun2 llama a fun3, fun3 llama a fun1
d. main llama a fun3, fun3 llama a fun1
e. main llama a fun1, fun1 llama a fun3, fun3 llama a fun2
f. main llama a fun3, fun3 llama a fun2, fun2 llama a fun1