Trabajo Práctico N° 3

Alcance en Lenguajes Dinámicos

Pueden ejecutar los códigos en el enlace que se encuentra en el título de cada ejercicio o en otros intérpretes online o locales de perl, python, php y scheme.

Ejercicio 1

Dado el siguiente programa en lenguaje PYTHON

```
x=1
def a():
    x=2
    b()
def b():
    print (x)
a()
```

a) ¿Qué imprime este programa y qué característica implica que el lenguaje tiene ámbito definido por el anidamiento sintáctico?
b) ¿Cuál sería la salida si el lenguaje tuviese ámbito dinámico?

Ejercicio 2

Dado el siguiente programa en lenguaje PYTHON

```
y=1
def a():
  x = 2
  def b():
     print (x)
     print (y)
  b()
  return b
a()
z=a()
y=3
x=4
z()
```

a) Realizar un diagrama de memoria que muestre las variables y la evolución de la ejecución del programa
b) ¿Qué imprime el programa? ¿Por qué?
c) ¿Por qué todas las veces que se ejecuta la instrucción “print x” imprime el mismo valor?

Ejercicio 3

Considere el siguiente fragmento de código en el lenguaje Python:

a) Indicar qué imprime el programa. Realizar un **diagrama de memoria de la tabla de símbolos representada como una pila de la ejecución del programa**.

#	Programa	Impresiones	Diagrama de memoria

1
2
3
4
5
6
	a = 1
	def f():
	  b = 2
	  def g():
	  global a
	  nonlocal b		

7 8 9 10 11 12 13	a = 2 b = 3 c = 4 print (a+b+c) g() print (a+b) f()		

b) Además, se generan 6 versiones adicionales del programa comentando para cada versión las líneas indicadas en la tabla siguiente. Indicar **qué imprime el programa en cada versión**. Fundamentar **cada versión** haciendo un diagrama de memoria de la ejecución de cada una.

V1: 5, 6, 7 y 8	V2: 5, 6 y 7	V3: 5, 6 y 8	V4: 6 y 8	V5: 5 y 7	V6: 5 y 6

c) Responder si las siguientes afirmaciones son verdaderas o falsas fundamentando las respuestas:

- **i.** Los modificadores global y nonlocal afectan el alcance de las variables indicadas en esas instrucciones
- **ii.** La versión V6 ocupa menos espacio en memoria que la versión original
- **iii.** Si hubiese una variable “a” en la función f(), por ejemplo a = 3, el modificador global de g() no permitiría escribir en la variable del ámbito global porque habría otra variable “a” en f().

d) En la versión original del programa ¿Puede lograrse el efecto de las instrucciones global y nonlocal en un lenguaje como Pascal? **Justifique su respuesta** indicando si es posible o no.
Si su respuesta es afirmativa, **transcriba el código en un pseudocódigo de Pascal** (o lenguaje/pseudocódigo con anidamiento de unidades con nombre) que justifique su respuesta.

Ejercicio 4

Considere el siguiente programa en lenguaje PERL

```
use warnings;
$x=1;
mysub();
localsub();
imprime();
sub mysub {
   my $x = 2;
   imprime();
}
sub localsub {
   local $x = 3;
   imprime();
}
sub imprime { print "$x\n";}
```

a) ¿Qué imprime este programa? Describir qué es lo que sucede en cada una de las salidas respecto del alcance de las variables.
b) ¿Cuáles son las instrucciones (y por qué) que reflejan el tipo de lenguaje que es Perl?
c) ¿Cómo influye en los resultados la diferencia entre la semántica de la palabra reservada “local” y “my” ?
d) ¿Qué cambiaría en el programa (modificando una sola instrucción, sin borrar ninguna y sin cambiar el valor literal 3), para que nunca imprima "3"?

Ejercicio 5

Dado el siguiente programa en el lenguaje Perl con sus variantes 1 a 6 como agregados al programa.

#	Programa	V 1	V2	V3	V 4	V5	V6

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
	use warnings;
	sub f1() {
	$x = 1;
	return sub() {
	print $x;
	$x = $x + 1;
	}
	}
	sub f2() {
	local $x = 2;
	return sub() {
	print $x;
	$x = $x + 1;
	}
	}
	sub f3() {
	my $x = 3;
	return sub() {
	print $x;
	$x = $x + 1;
	}
	}
	(V1 o V2 o V3 o V4 o V5 o V6)
		$c1 = f1();
		&$c1();
		$c2 = f2();
		&$c2();
		$c3 = f3();
		&$c3();
			$c1 = f1();
			&$c1();
			$c2 = f3();
			&$c2();
			$c3 = f2();
			&$c3();
				$c1 = f3();
				&$c1();
				$c2 = f1();
				&$c2();
				$c3 = f2();
				&$c3();
					$c1 = f3();
					&$c1();
					$c2 = f2();
					&$c2();
					$c3 = f1();
					&$c3();
						$c1 = f2();
						&$c1();
						$c2 = f3();
						&$c2();
						$c3 = f1();
						&$c3();
							$c1 = f2();
							&$c1();
							$c2 = f1();
							&$c2();
							$c3 = f3();
							&$c3();

a) Construir la tabla de símbolos para las variantes 1, 3, 4 y 5 colocando los nombres de variables con los símbolos de los closures respectivos.

b) Indicar cuáles resultados (impresiones y/o errores) produce la ejecución de cada variante. Justifique sus respuestas indicando el por qué de cada impresión, a qué variable corresponde, cuál es su ámbito y modo.

Nombre	Modo	Ámbito	Dirección	Instrucción en la que la variable sale de la TS

Dirección	Valor

Ejercicio 6

Dados los siguientes códigos en los lenguajes PERL y PYTHON, responder cada uno de los incisos.

PERL

```
use warnings;
$a=1;
sub s1() {
   local $a = 2;
   print $a;
   s2();
   return sub() {
       print $a;
   }
}
sub s2() {
   print $a;
}
$d=s1();
&$d();
s2();
```

PYTHON

```
a=1
def s1():
    a = 2
    print (a)
    s2()
    def retorna():
        print (a)
    return retorna

def s2():
    print (a)

d=s1()
d()
s2()
```

a) Realizar un diagrama *(tabla de símbolos)* donde se muestre la ubicación de las variables en memoria

b) ¿Cuáles son las impresiones de los respectivos programas? **Justificar cada impresión**

c) Indicar cuál/es impresiones evidencian diferencias de binding de alcance entre ambos lenguajes y **justificar** la respuesta.

d) Si en el programa en Perl se cambia la declaración de la variable $a de sub1 utilizando el modificador “my”, indicar si hay cambios en las impresiones y explicar por qué se producen los mismos respecto del programa actual.

Ejercicio 7

Dados los siguientes códigos en el lenguaje PHP, responder cada uno de los incisos.

PROG 1

```
function f() {
 echo "en f";
  function g() {
     echo "en g";
  }
  function h() {
     echo "en h";
  }
}
g();
h();
```

PROG 2

```
function f() {
 echo "en f";
  function g() {
     echo "en g";
  }
  function h() {
     echo "en h";
     }

}
f();
g();
h();
```

PROG 3

```
function f() {
 echo "en f";
  function g() {
     echo "en g";
     function h() {
      echo "en h";
     }
  }
}
f();
h();
```

PROG 4

```
function f() {
 echo "en f";
  function g() {
 echo "en g";
  function h() {
 echo "en h";
    }
 h();
 }
}
f();
g();
```

PROG 5

```
function f() {
 echo "en f";
 function g() {
  echo "en g";
 function h() {
  echo "en h";
     }
 }
}
f();
g();
h();
```

a) Indicar la salida de cada programa.
b) Construir la tabla de símbolos en cada caso.
c) En el ejemplo 2 reemplazar la instrucción “f();” por “if (rand(1,10) > 5) f();” e indicar cuál es la salida (ejecutarlo varias veces para observar mejores resultados).
d) ¿El lenguaje tiene binding de alcance estático o dinámico? Fundamentar la respuesta.

Ejercicio 8

Dados los siguientes programas en los lenguajes Scheme y Python:

#	Scheme	Python

1
2
3
4
5
6
7
8
9
10
11
12
	(define x 1)
	x
	(define (z n3) (x n3))
	(define (x n1) (define (x n2) (+ n1 n2)) (display n1) (x 3))
	x
	(x 5)
	(z 9)
		x = 1
		print (x)
		def z(n3):
		  x(n3)
		def x(n1):
		  print (n1)
		  def x(n2):
		  print(n1+n2)
		  x(2)
		print (x)
		x(3)
		z(5)

a) Ejecutar los programas, explicando las salidas.
b) Construir la tabla de símbolos
c) Indicar si la función x interna es local a la externa o global

Fundamentar las respuestas

Ejercicio 9

Dado el siguiente programa en Python,

1
2
3
4
5
6
7
8
9
	x = 1
	def x():
	  def x():
	  print (x)
	  x()
	x()
	print (x)
	x = 2
	x()
		a) construir la tabla de símbolos y explicar la salida
		b) Insertar “global x” entre las líneas 3 y 4 y responder a)
		c) Insertar “global x” y “x=2” entre las líneas 3 y 4, responder a)