Paradigmas de la Programación –Primer Parcial

18 de Abril de 2024

Apellido y Nombre: _____

1. Diagrame los estados por los que pasa la pila de ejecución en el siguiente programa a) en un lenguaje con pasaje de parámetros por valor **[5 pt.]** y b) en un lenguaje con pasaje de parámetros por referencia **[5 pt.]**, de forma que se vea claramente cómo se modifican las variables en cada caso. En cada caso, indique qué imprime el programa.

```
1 def incrementa(num):
2     num += 1
3
4 x = 5
5 incrementa(x)
6 print(x)
```

2. **A ELEGIR UNO ENTRE EL 2 Y EL 6 [10 pt.]** Transforme el programa del ejercicio anterior para que la función `incrementa` sea declarativa.

3. En el siguiente programa identifique en qué línea se daría un comportamiento diferente en un lenguaje con alcance estático y en un lenguaje con alcance dinámico **[5 pt.]** y describa por qué **[10 pt.]**, usando los conceptos de *contexto de definición* y *contexto de ejecución*. Indique qué escribiría el programa en cada uno de esos casos.

```
1  object Main extends App {
2    var x: Int = 0
3
4    def q(): Unit = {
5      x += 1
6    }
7
8    def r(): Unit = {
9      var x: Int = 1
10     q()
11     println(x)
12   }
13
14   x = 2
15   r()
16 }
```

4. La siguiente expresión está mal tipada: `f(x) = x+2 && x`. Diagrame el grafo de tipado y el sistema de ecuaciones correspondiente [**5 pt.**], describa dónde se encuentra el problema [**5 pt.**] y explique cómo lo trataría un lenguaje de tipado fuerte y cómo podría tratarlo un lenguaje de tipado no fuerte [**5 pt.**].

5. [**10 pt.**] Describa cómo es la ejecución del siguiente programa si el archivo `"conteo_de_palabras.txt"` no existe. Describa con el detalle de qué activation records se van apilando y desapilando. Puede complementar su explicación con diagramas de la pila de ejecución si le resulta más claro.

```
1  def contar_palabras(archivo_entrada, archivo_salida):
2      try:
3          with open(archivo_entrada, 'r') as archivo:
4              text = archivo.read()
5              conteo_palabras = len(text.split())
6              with open(archivo_salida, 'w') as salida:
7                  salida.write(str(conteo_palabras))
8              print(archivo"Conteo de palabras: {conteo_palabras}.")
9      except FileNotFoundError:
10         print("No se encontro el archivo.")
11     finally:
12         archivo.close()
13         salida.close()
14
15 archivo_entrada = "texto_de_entrada.txt"
16 archivo_salida = "conteo_de_palabras.txt"
17 contar_palabras(archivo_entrada, archivo_salida)
```

6. **A ELEGIR UNO ENTRE EL 2 Y EL 6 [10 pt.]** En el programa del ejercicio anterior ¿hay alguna porción con propiedades no declarativas? si es así, indique en qué línea(s) se encuentra(n) y descríbala(s).

HOJA N°

FECHA

PARCIAL I - PARADIGMAS

ACHÁVAL BERZERO TOMÁS
45085146

EJERCICIO Nº1:

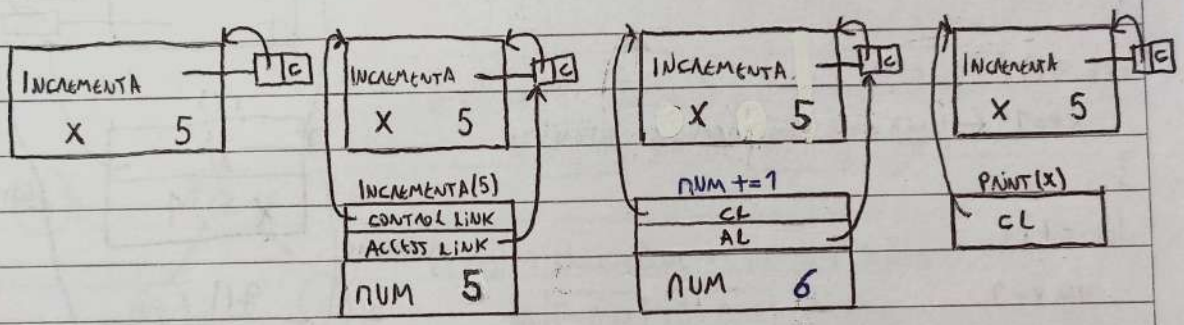
1 100
2 ~~40~~ 100
3 100
4 100
5 100

```
def INCREMENTA(NUM):
    NUM += 1
X=5
INCREMENTA(X)
PRINT(X)
```

OBS: [C] ES UN PUNTERO AL CÓDIGO DE LA FUNCIÓN

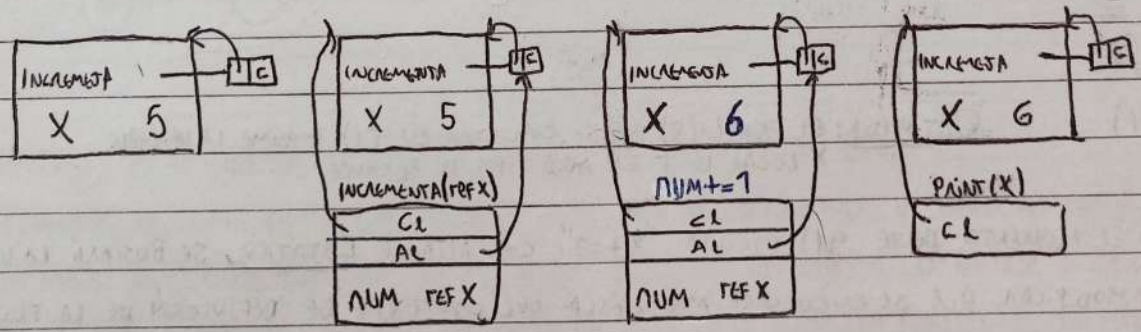
TIEMPO →

a)



EN PASAJE POR VALOR SE IMPRIME 5.

b)



EN PASAJE POR REFERENCIA SE IMPRIME 6.

NOTA

Ejercicio N°2

Programa del 1, declarativo

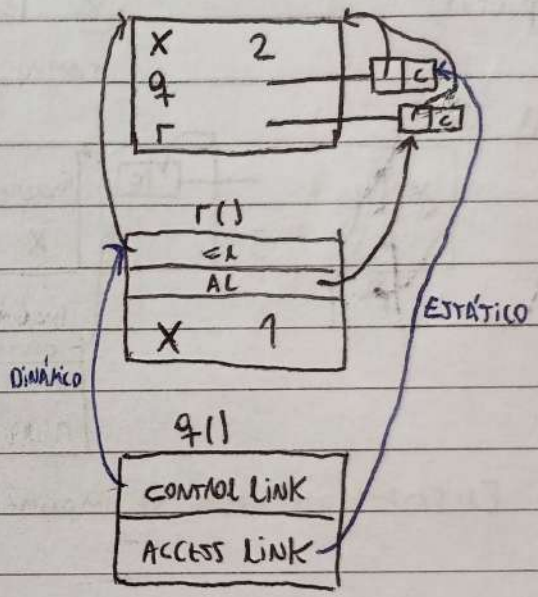
```
def incrementa(num)
    return num+1
x=5
print(incrementa(x))
```

Ejercicio N°3

```
var x=0

def q(){
   x+=1 ← Línea con comportamiento diferente.
}
def r(){
   var x=1
   q()
   println(x)
}
x=2
r()
```

Pila en el momento de la línea



Obs previa: El println(x) que se encuentra en r() imprime la variable x local de r en ambos tipos de alcance.

En el momento donde q() ejecuta "x+=1", con alcance Estático, se buscará la variable x a modificar que se encuentre más cerca del contexto de definición de la función q, es decir, subiendo por los Access Links. En este caso se imprime el valor 1 pues la variable local de r() no ha sido modificada (se modificó la x declarada antes de q).

En cambio, con alcance dinámico, se sigue el contexto de ejecución del programa, por lo tanto la x a modificar es encontrada a través de los control links. Como la última x declarada antes del llamado a q() fue la x local de r, entonces esta aumenta en 1 y el programa imprime el valor 2.

NOTA

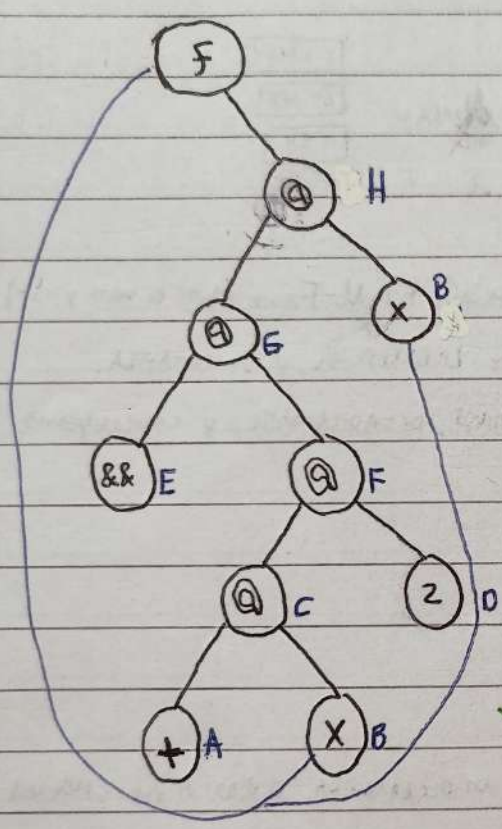
HOJA N°

FECHA

Ejercicio N°4:

Achával Benzeo Tomás
45085746

$f(x) = x+2 \;\&\&\; x$



① $A = Int \to Int \to Int$

$A = B \to C$

② $C = D \to F$ y $D = Int$

③ $E = Bool \to Bool \to Bool$

$E = F \to G$

④ $G = B \to H$

①y② ⇒ $Int \to Int \to Int = B \to D \to F = B \to Int \to F$

⇒ $\boxed{B = Int}$ $D = Int$ $\boxed{F = Int}$ ✓

③y④ ⇒ $Bool \to Bool \to Bool = F \to B \to H$

⇒ $\boxed{F = Bool}$ $\boxed{B = Bool}$ $H = Bool$

✓

El conflicto se encuentra en el tipo de x y del resultado de x+2. ✓

Un lenguaje de tipado fuerte no permitiría este tipo de operaciones, y "tiraría" un error.

Un lenguaje de tipado débil, en cambio, podría realizar un casteo de entero a booleano, donde $X: Int$, $(X+2): Int$, pero el programa se vería como $f(Int\ x) = (bool)(X+2)\ \&\&\ (bool)X$

En el cual, por ejemplo, $(bool)0 = False$

$(bool)n = True \quad n \neq 0$

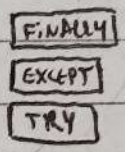
Ejercicio Nº 5:

Línea de tiempo de apilar/desapilar Activation Records. Notar que siempre el último Activation Record apilado es aquel que se está ejecutando.

Tiempo

- Se apila la función contar_palabras y las variables "globales" de archivos.
- (Llamado a función contar_palabras(...))
  - Se apila el bloque finally
  - Se apila el bloque except
  - Se apila el bloque try

  } A la vez y en este orden



- (Falla open(archivo_salida, 'w') pues no existe)
- Se desapila el bloque try por causa de excepción FileNotFound. (Antes de write y print)
- Se ejecuta el bloque except, pues hace match con la excepción, y se desapila.
- Se ejecuta el bloque finally, cerrando los archivos, desapilándose, y concluyendo con la ejecución del programa.