

# 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).

# PARCIAL I - PARADIGMAS

HOJA N°

FECHA

ACHAM BAZENO TOMÁS

45085146

Ejercicio N°1:

1 100  
2 40  
3 100  
4 100  
5 100

claramente mal corregido

def INCREMENTA (NUM):

NUM += 1

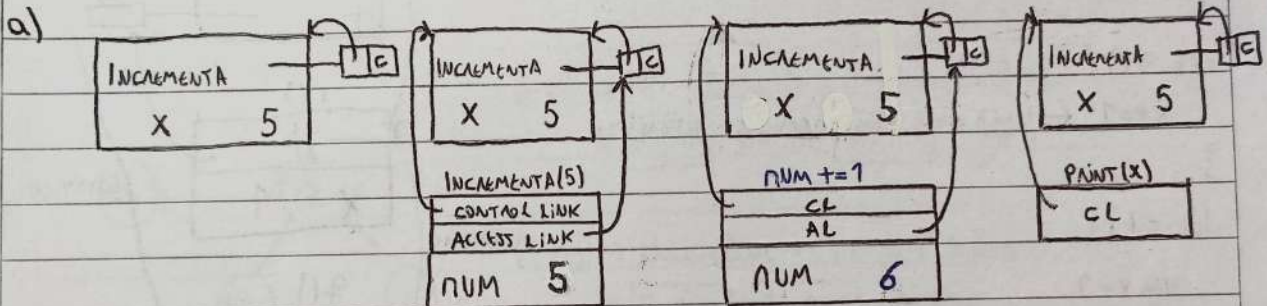
X=5

INCREMENTA(X)

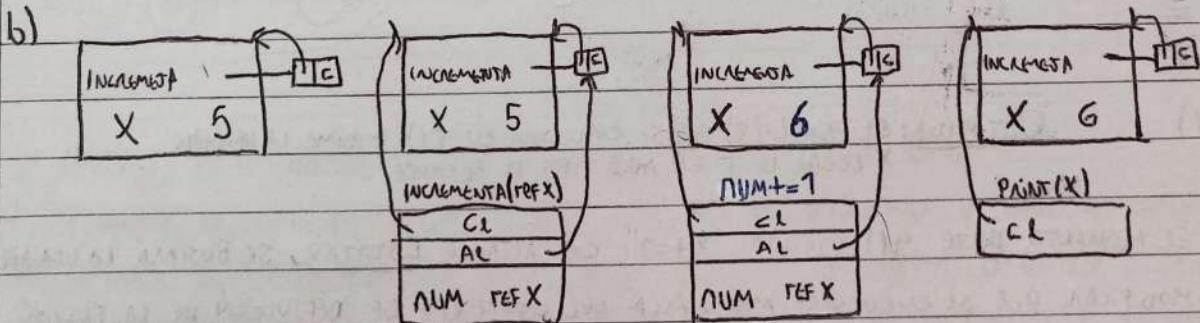
PRINT(X)

Obs: ☒ es un puntero al código de la función

TIEMPO



EN PASAJE POR VALOR SE IMPRIME 5.



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

PILA EN EL MOMENTO DE LA LÍNEA

```
VAR X=0
```

```
def q() {
```

```
    X+=1 ← LÍNEA CON COMPORTAMIENTO DIFERENTE.
```

```
}
```

```
def r() {
```

```
    VAR X=1
```

```
    q()
```

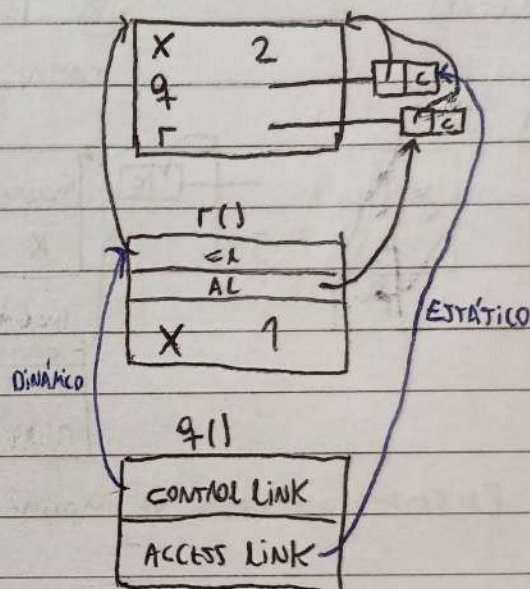
```
    PRINTLN(X)
```

```
}
```

```
X=2
```

```
r()
```

¿DÓNDE PASA? 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 ENCUENTRA 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. ✓

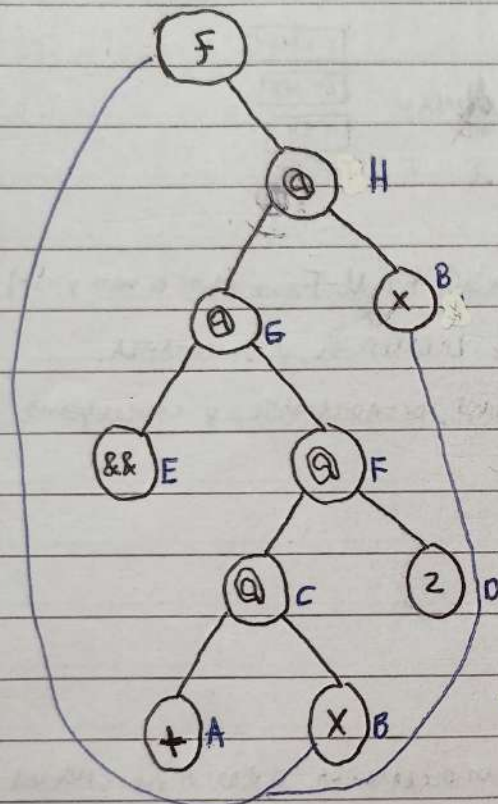


Ejercicio N° 4:

ACHAR BENZED TONAS

45085746 A.

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



$$\textcircled{1} \ A = \text{Int} \rightarrow \text{Int} \rightarrow \text{Int}$$

$$A = B \rightarrow C$$

$$\textcircled{2} \ C = D \rightarrow F \ \text{ y } D = \text{Int}$$

$$\textcircled{3} \ E = \text{Bool} \rightarrow \text{Bool} \rightarrow \text{Bool}$$

$$E = F \rightarrow G$$

$$\textcircled{4} \ G = B \rightarrow H$$

$$\textcircled{1} \text{ y } \textcircled{2} \Rightarrow \text{Int} \rightarrow \text{Int} \rightarrow \text{Int} = B \rightarrow D \rightarrow F = B \rightarrow \text{Int} \rightarrow F$$

$$\Rightarrow \boxed{B = \text{Int}} \quad D = \text{Int} \quad \boxed{F = \text{Int}} \quad \checkmark$$

$$\textcircled{3} \text{ y } \textcircled{4} \Rightarrow \text{Bool} \rightarrow \text{Bool} \rightarrow \text{Bool} = F \rightarrow B \rightarrow H$$

$$\Rightarrow \boxed{F = \text{Bool}} \quad \boxed{B = \text{Bool}} \quad H = \text{Bool}$$

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 REMITIR UN CASTEO DE ENTERO A BOOLEANO, DONDE  $x: \text{Int}$ ,  $(x+2): \text{Int}$ , PERO EL PROGRAMA SE VERÍA COMO  $f(\text{int } x) = (\text{bool})(x+2) \ \&\& \ (\text{bool})x$

EN EL CUAL, POR EJEMPLO,  $(\text{bool})0 = \text{False}$

$(\text{bool})n = \text{True} \ n \neq 0$

## Ejercicio N° 5:

LÍNEA DE TIEMPO DE APILAR/DESAPILAR ACTIVATION RECORDS. NOTAR QUE SIEMPRE EL ÚLTIMO ACTIVATION RECORD APILADO ES QUIÉN SE ESTÁ EJECUTANDO.

TIEMPO

- SE APILA LA FUNCIÓN COUNT-PARABAS Y LAS VARIABLES "GLOBALES" DE ARCHIVOS.

- (LLAMADO A FUNCIÓN COUNT-PARABAS(...))

- SE APILA EL BLOQUE FINALLY

- SE APILA EL BLOQUE EXCEPT

- SE APILA EL BLOQUE TRY

} A LA VEZ Y EN ESTE ORDEN

FINALLY  
EXCEPT  
TRY

- (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.