1	a B	b B				
2	a R-	b B				
3	a B	b B				
4	a M	b B	c B	d B	e B	
					Resultado Final	Aprobado

Ejercicio 1 (30p)
Realice la pila de ejecución para el siguiente código:

a) por cadena estática
b) por cadena dinámica

```
Program Main;
Var x, y, z:integer;
a, b: array[1..6] of integer;

Procedure D(var x: integer; nombre y: integer);
  var h:integer;
  begin
     x:= f + 5;
     y:= y + 2;
     h:= x + 15;
  end;

Function F: integer;
  Var y:integer;
  Begin
    y:=1;
    b(x):= b(x) + 1;

    if(x < 6) then
        x:= x + 1;

    a(y):=a(y)+b(x)+3;
    a(x)=a(x) + 2;
    y:=y+3;
    return b(y);
  end

begin
     x:= 1;   y:= 1;
     for z:=1 to 6 do begin
              a(z):= abs(7-z);
              b(z):= z * 2;
     end;
     D(a(z), b(abs(6-x+y)));
     for z:=1 to 6 do  write (a(z), b(z));
     end.
```

Nota: La forma de evaluación de este lenguaje es de izquierda a derecha. Abs es una función que retorna el valor absoluto de la operación recibida.

Ejercicio 2

a) (10pts) Clasifique las siguientes estructuras de datos de acuerdo a lo visto en la práctica.Justifique en cada caso:

i) **Java**

```
class Alumno {
  String nombre;
  String apellido;
  int edad;
  float promedio;
  String domicilio
```

Escaneado con CamScanner

Realice el parcial con lapicera, de otra forma se desaprobará el/los ejercicio/s.
Se considera presentismo cuando se realiza completamente un ejercicio.

```
        public float getPromedio(){
                return this.promedio;
        }
}
```

ii) C

```
typedef struct _nodoArbol {
  void *info;
  struct _nodoArbol *hijoIzq;
  struct _nodoArbol *hijoDer;
} nodoArbol;

typedef struct _arbolBinario {
  int valor_guardado;
  nodoArbol *raiz
} arbolBinario;
```

b) (10 pts) Responda si las siguientes afirmaciones son V o F. Justifique en cada caso

i) Los lenguajes con sistema de tipos fuerte son siempre compilados F
ii) Las tuplas de python son un ejemplo de producto cartesiano F
iii) La unión y la unión discriminada no son seguras en ejecución V

Ejercicio 3

a) (15 pts) Dado el siguiente código en Java, establezca cuáles de las opciones indicadas más abajo son válidas como camino de ejecución. Justifique con una breve descripción del flujo de ejecución, caso contrario no se considerará válida la respuesta)

```
1 public class Java7MultiplesExceptions {
2
3   public static void main(String[] args) {
4         try{
5               for (int i = 1; i < 4; i++) {
6                     if(i==1){
7                           System.out.println(Integer.toString(i));
8                           rethrow("Primera");
9                     }
10                    else{
11                          if(i==2){
12                                System.out.println(Integer.toString(i));
13                                rethrow("Segunda");
14                          }
15                          else{
16                                if(i==3){
17                                      System.out.println(Integer.toString(i));
18                                      rethrow("Tercera");
19                                }
20                          }
21                    }
22              }
23        }catch(ThirdException e){
24              System.out.println(e.getMessage());
25        }
26 }
27
28 static void rethrow(String s) throws ThirdException {
29     try {
30           try {
31                 if (s.equals("Primera")){
32                       throw new FirstException("Primera excepción");
33                 }
34                 else{
35                       if (s.equals("Segunda")){
36                             throw new SecondException("Segunda excepción");
37                       }
38                       else{
39                             throw new ThirdException("Tercera excepción");
40                       }
41                 }
42           }catch (SecondException e) {
43                 ThirdException e1=new ThirdException("Tercera excepción");
```

Escaneado con CamScanner

Realice el parcial con lapicera, de otra forma se desaprobará el/los ejercicio/s.
Se considera presentismo cuando se realiza completamente un ejercicio.

```
44                  throw e1;
45            }
46      }catch (FirstException e) {
47            ThirdException e1=new ThirdException("Tercera excepción");
48            throw e1;
49            }
50      }
51  }
```

i) Se imprime en pantalla "1" y luego "tercera excepción" y luego termina → varia
ii) Se imprime en pantalla "1", "Primera excepción", "2", "segunda excepción", "3", "tercera excepción" y luego termina
iii) Se imprime en pantalla "1", "Tercera excepción", "2", "Tercera excepción", "3", "tercera excepción" y luego termina
iv) Ninguna de las anteriores

b) (10 pts) Indique si el resultado de intercambiar las líneas 4 y 5 (es decir, el try fuera del for) genera el mismo resultado de impresión. Justifique

Ejercicio 4

(25pts). Marcar si son verdaderas o falsas las siguientes afirmaciones. Acompañar la respuesta con una justificación, caso contrario, NO se tomarán cómo válidas

a. La sentencia for de ADA y Pascal son igualmente seguras V F
b. El if de circuito corto puede prevenir errores en ejecución V F
c. La sentencia yield de python equivale a hacer return V F
d. En PL/1 si se genera una excepción, se ejecuta el manejador correspondiente y el control es pasado inmediatamente al programa principal V F
e. La sentencia else de python en el manejo de excepciones se ejecuta solo si no se encontró ningún manejador asociado a la excepción en cuestión V F

Escaneado con CamScanner

1) cadena estatica

*1	RA MAIN PR X = ~~1~~ 2 Y = 1 Z = ~~1..6~~ 1..6 a(1) = ~~6~~ 13 a(2) = ~~8~~ 4 a(3) = 4 a(4) = 3 a(5) = 2 a(6) = ~~X~~ 13 b(1) = ~~Z~~ 3 b(2) = 4 b(3) = 6 b(4) = 8 b(5) = ~~10~~ 12 b(6) = 12 procedure D Function F VR
*2	RA D PR EE(*1) ED(*1) n = 28 X = Y = ↑ b(abs(6-x+y)) VR 8
*3	RA F PR EE(*1) ED(*2) Y = ~~1~~ 4 VR

main
D F

a(6)
D(a(z), b(abs(6-x+y)));

imprime
13 , 3
4 , 4
4 , 6
3 , 8
2 , 12
13 , 12

$x = \overset{8}{F} + 5 = 13$

$y = y + 2$

$b(abs(6-x+y)) =$

$abs(6-2+1)$ ⟸ $b(abs(6-x+y)) +$

$b(5) = b(5) + 2 = 12$

$n = x + 15 = 13 + 15 = 28$

$b(x) = b(x) + 1 = 3$

$x = x + 1 = 2$

$a(y) = a(y) + b(x) + 3$

$\underbrace{1 \quad 1 \quad 2}$

$a(1) = a(1) + b(2) + 3 =$

$6 + 4 + 3 = 13$

$\underbrace{a(x) = a(x) + 2 =}$

$a(2) = a(2) + 2 =$

$5 \quad T2 = 7$

$y = y + 3 = 4$

return $\overbrace{b(y)}^{b(4) \to 8}$

Escaneado con CamScanner

cadena dinamica

*1	RA MAIN PR X = 1 ✓ Y = 1 Z = 1 ~~6~~ 1..6 ✓ a(1) = ~~6~~ 13 ✓ a(2) = ~~8~~ 4 ✓ a(3) = 4 a(4) = 3 a(5) = 2 a(6) = ~~1~~ ~~Z~~ 13 ✓ b(1) = ~~Z~~ 3 ✓ b(2) = 4 b(3) = 6 b(4) = 8 b(5) = 10 b(6) = ~~12~~ 14 ✓ Procedure D Function F ✓ VR
*2	RA D PR EE(*1) ED(*1) ✓ n = 28 ✓ X = Y = ↑ b(abs(6 - x + y)) VR 8 ✓
*3	RA F PR EE(*1) ✓ ED(*2) Y = ~~*~~ 4 ✓ VR

a(6)

D(a(2) , b(abs(6 - x + y)))

imprime

13	3 ✓
4	4 ✓
4	6 ✓
3	8 ✓
2	10 ✓
13	14 ✓

D

$x = \overset{8}{\uparrow} F + 5 = 13$

$y = y + 2 \Rightarrow b(abs(6 - x + y)) = b(abs(6 - x + y) + 2$

$\Rightarrow b(abs(6 - 1 + 1)) = b(abs(6 - 1 + 1)) + 2$

$\Rightarrow b(6) = b(6) + 2 = 14$ $\quad n = x + 15 = 13 + 15 =$

$b(x) = b(x) + 1 = 3$

$x = x + 1 = 2$

a(6)

$a(y) = a(y) + b(y) + 3$

$\underbrace{\quad}_{1 \quad 1 \quad 2}$

$a(1) = a(1) + b(2) + 3 =$

$6 + 4 + 3 = 13$ ✓

$a(x) = a(x) + 2$

$2 \quad 2$

$a(2) = a(2) + 2 =$

$5 + 2 = 7$

$y = y + 3 = 4$

return $b(y) \rightarrow b(4)$

8

Escaneado con CamScanner

ROJAS AGUSTINA | 50

2)a)i) Se trata de un tipo de dato abstracto ya que es una clase. Las clases en Java son un TAD ya que cumplen con el encapsulamiento y el ocultamiento de la información. Con respecto al encapsulamiento, la representación del tipo y las operaciones que puede realizar se describen en una única unidad sintáctica, es decir, el estado interno y el comportamiento en este caso. Con respecto al ocultamiento es que estas cuestiones antes mencionadas permanecen ocultas.

M. X

Es producto cartesiano.

Esta clase sí cumple con TAD.

ii) Se trata de un tipo de dato definido por el usuario compuesto recursivo, ya que se trata de un tipo nodo cuya estructura puede contener otros datos del mismo tipo nodo. La recursión se puede realizar mediante punteros. El tamaño de esta estructura puede crecer arbitrariamente y además es de tipo producto cartesiano.

R

b) i) F Esto es falso ya que, por ejemplo, Python es interpretado y su sistema de tipos es fuerte. En Python las ligaduras se realizan en ejecución y su sistema de tipos provee restricciones con respecto a la forma en que las operaciones involucran valores de diferentes tipos.

B

ii) F Las tuplas en Python son un ejemplo de correspondencia finita, ya que se trata de un conjunto de valores en un conjunto de dominio.

B

iii) V La unión y la unión discriminada no son seguras en ejecución ya que puede ocurrir que se intente acceder al valor que no se debe acceder y esto provocaría un error. Problemas en el flujo del programa.

B

3) a) La opción valida es la i) ya que en el por dentro del try del main cuando i es igual a uno se va a imprimir "1", luego se va a llamar al método rethrow con el parámetro "primera", en este método el try de más adentro va a lanzar una excepción de tipo FirstException que no va a coincidir con ninguno manejador de ese try por lo que se va a propagar al de más afuera, la excepción será capturada y manejada por el catch (FirstException e), que lanzará una excepción de tipo ThirdException, esta misma se propagará finalmente y será capturada y manejada por el

Es importante recalcar que cuando se genera la excepción el bloque try donde se produce es terminado

Escaneado con CamScanner

catch (ThirdException e) del main que imprimirá el mensaje de la excepción "Tercera Excepción".
El for no se seguirá ejecutando ya que, como mencione anteriormente, cuando se genera una excepción, el bloque donde se genero la misma es terminado.

b) No, no genera el mismo resultado ya que al estar el for afuera del try se van a realizar todas las iteraciones, si estuviera dentro del try, en cambio, como mencione con anterioridad, el for no se seguiría ejecutando ya que se produciría una excepción en la primera iteración y el bloque sería finalizado (el try, finalizaría)

4) a. (V) F si, con igual de seguras ya que tanto ADA como Pascal, en forma estandar, no permiten que la variable de control sea modificada.
en PASCAL es posible modificarla.

b. V puede presentar errores en ejecución como por ejemplo en el siguiente caso:

```
if ((b != 0) && (a/b > 5))
```

ya que con la primera parte de la expresión logica se puede determinar el resultado final, por lo que no se seguiría evaluando el resto, evitando así un error de división por cero. Si se tratara de circuito largo, si se produciría un error al intentar dividir por cero

c. F la sentencia yield de python no es lo mismo que return ya que yield genera una secuencia de valores, puede mantener su estado y continuar desde donde se detuvo.

d. F una vez que se maneja una excepción, el control es devuelto a la sentencia siguiente a la que genero la excepción ya que pl/1 tiene como criterio de ejecución la "reanudación".

e. F la sentencia else de python en el manejo de excepciones se ejecuta solo si NO se genero una excepción.

NOTA