

A dark blue vertical bar is positioned on the left side of the page. A blue arrow-shaped banner points to the right, containing the date. Below the banner, several thin, curved lines in dark blue and light gray extend upwards from the bottom left corner.

20-5-2023

Reporte de ejercicios

Paradigmas de programación

Integrantes:

- 1) Delgado Acosta Luis Bernardo
- 2) Díaz Jiménez Jorge Arif
- 3) Valdés Luis Eliot Fabian

Grupo: 3BV2

ESCUELA SUPERIOR DE CÓMPUTO

Parte 1

Ejercicio 1:

- Código para debug y resultados de ejecución

```
01_stack.py
64 # PROBANDO LA IMPLEMENTACIÓN
65 # Crear una nueva pila
66 stack = MinStack()
67
68 # Agregar elementos a la pila
69 stack.push(5)
70 stack.push(3)
71 stack.push(4)
72 stack.push(7)
73 stack.push(1)
74
75 # Obtener y mostrar toda la pila
76 print(f'La pila inicial es: {stack.show()}')
77
78 # Obtener el elemento mínimo
79 print(f'\nEl elemento menor de la pila es: {stack.getMin()}') # Debería imprimir 1
80
81 # Eliminar el último elemento (1 en este caso)
82 print(f'\nEliminando el elemento menor de la pila... (1)')
83 stack.pop()
84
85 # Obtener el elemento mínimo ahora
86 print(f'\nEl elemento menor de la pila ahora es: {stack.getMin()}') # Debería imprimir 3
87
88 # Obtener el último elemento (top) de la pila
89 print(f'\nEl último elemento de la pila es: {stack.top()}') # Debería imprimir 7
90
```

```
[Running] python -u
01\01_stack.py"
La pila inicial es: [5, 3, 4, 7, 1]
El elemento menor de la pila es: 1
Eliminando el elemento menor de la pila... (1)
El elemento menor de la pila ahora es: 3
El último elemento de la pila es: 7
[Done] exited with code=0 in 0.349 seconds
```

Ejercicio 2:

- Código para debug y resultados de ejecución

```
02_sortStack.py
73 # PROBANDO LA IMPLEMENTACIÓN
74 # Crear una nueva pila y agregar elementos desordenados
75 stack = Stack()
76 stack.push(3)
77 stack.push(1)
78 stack.push(4)
79 stack.push(2)
80
81 # Mostramos la pila original
82 print(f'Pila original: {stack.show()}')
83
84 # Ordenar la pila
85 sorted_stack = sort_stack(stack)
86
87 print(f'Pila ordenada:')
88 # Mostrar los elementos de la pila ordenada
89 while not sorted_stack.isEmpty():
90     print(sorted_stack.pop()) # Imprimird: 4, 3, 2, 1
91
```

```
[Running] python -u
01\02_sortStack.py"
Pila original: [3, 1, 4, 2]
Pila ordenada:
4
3
2
1
[Done] exited with code=0 in 0.357 seconds
```

Ejercicio 3:

- Código para debug y resultados de ejecución

```

13 # Probando implementación
14 # Generar 20 puntos aleatorios
15 points = [Point3D(random.random(), random.random(), random.random()) for _ in range(20)]
16
17 # Mostrar los puntos generados
18 print("Puntos generados: { ", ".join(str(p) for p in points) }")
19 print("\n-----")
20
21 # Encontrar el par de puntos más cercano
22 pair, distance = closest_pair(points)
23
24 print(f"\nEl par de puntos mas cercano es:\n(pair[0]) y (pair[1]). \nCon una distancia de {distance}")
25
26
27
28

```

[Running] python -u \\PdP_IP_2022630401\\03_distance.py"

Puntos generados: (0.4483377883498224, 0.5492424618854836, 0.6281991682540394), (0.7723534587465824, 0.200115783357126, 0.88544161933697), (0.5903871575289091, 0.29550883358800486, 0.9575153735705524), (0.5466872450160604, 0.7082817303902013, 0.6442964349439834), (0.5476319844016314, 0.835186756566094, 0.9866108795842233), (0.426082668775741, 0.4762715134834672, 0.6469230541199599), (0.8936388411089503, 0.48184870500800414, 0.3703353753932777), (0.13762344659486692, 0.07394217715064832, 0.5664637254246204), (0.3704527402107979, 0.0762382056025421, 0.95136877088516), (0.4219745285214836, 0.879206031120196, 0.5287228613955011), (0.29369710407407923, 0.3485952683596044, 0.31269586925074607), (0.4155505357441617, 0.28457656915803553, 0.11110213332442154), (0.29778425137064224, 0.15818224928793612, 0.9146943212686343), (0.6969193952282093, 0.8633141136849066, 0.9652267345216692), (0.3105729328968211, 0.5691044355482157, 0.2763568244472421), (0.9848502870500256, 0.14819721578103806, 0.3895524521207355), (0.21758622571559516, 0.5673715321623816, 0.5438514111305202), (0.6046902876977209, 0.14835874466777266, 0.9324337412996665), (0.7602597189464341, 0.6127597330963298, 0.3981489111785014), (0.49594231196582692, 0.6460313878659362, 0.7197701053426416).

El par de puntos mas cercano es:
(0.4483377883498224, 0.5492424618854836, 0.6281991682540394) y (0.426082668775741, 0.4762715134834672, 0.6469230541199599).
Con una distancia de 0.07855338258756528

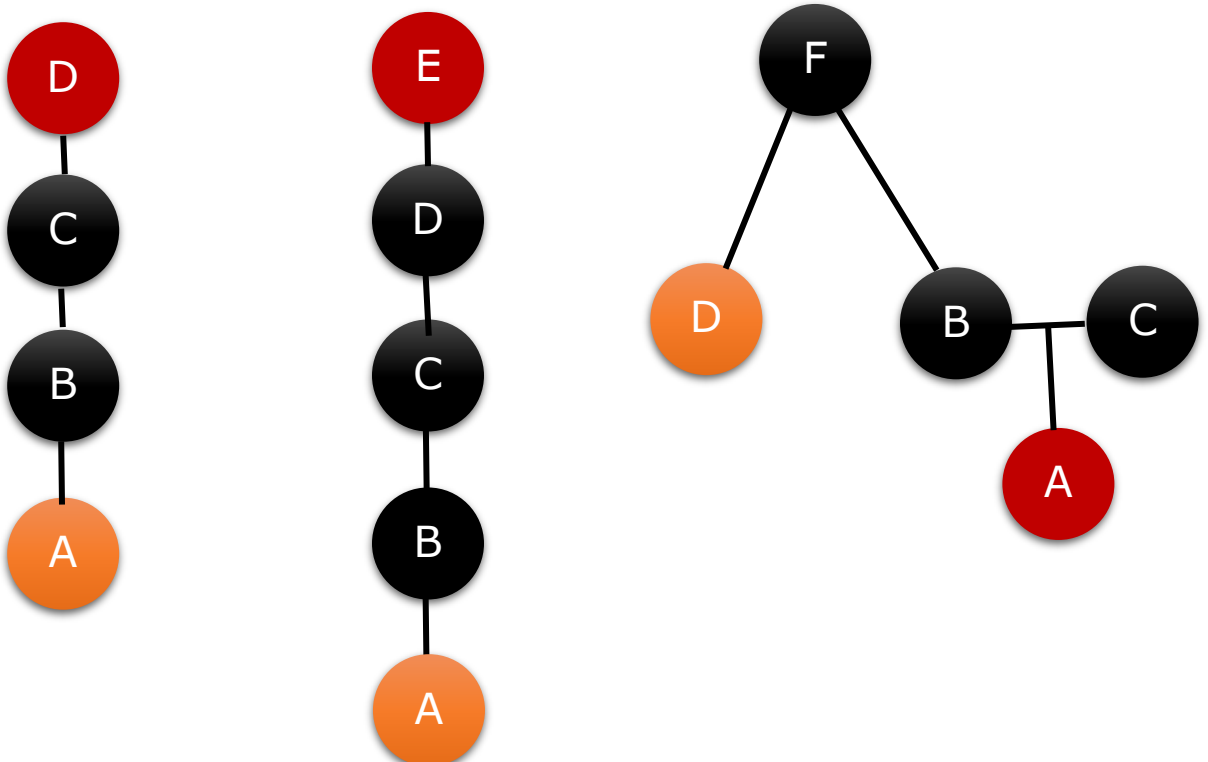
[Done] exited with code=0 in 0.209 seconds

Ejercicio 4:

- Diagramas, programa y ejecución

Bisabuelo(D de A) Tatarabuelo(E de A)

Sobrino(A de D)



```

parent(james, daniel).
parent(juan, julio).
parent(julio, anam).
parent(maria, anam).
parent(anam, tabo).
parent(tabo, adan).

% Rule: grandparent(Grandchild, Grandparent)
grandparent(Grandchild, Grandparent) :-
    parent(Grandchild, Parent),
    parent(Parent, Grandparent).

% Rule: sibling(Person1, Person2)
sibling(Person1, Person2) :-
    parent(Person1, Parent),
    parent(Person2, Parent),
    Person1 \= Person2.

% Rule: cousin(Person1, Person2)
cousin(Person1, Person2) :-
    parent(Person1, Parent1),
    parent(Person2, Parent2),
    sibling(Parent1, Parent2).

% Rule: greatgrandparent(Greatgrandchild, Greatgrandparent)
greatgrandparent(Greatgrandchild, Greatgrandparent) :-
    parent(Greatgrandchild, Parent),
    parent(Parent, Grandparent),
    parent(Grandparent, Greatgrandparent).

% Rule: greatgreatgrandparent(Greatgreatgrandchild, Greatgreatgrandparent)
greatgreatgrandparent(Greatgreatgrandchild, Greatgreatgrandparent) :-
    greatgrandparent(Greatgreatgrandchild, Parent1),
    parent(Parent1, Greatgreatgrandparent).

% Rule: nephew(Nephew, Uncle)
nephew(Nephew, Uncle) :-
    parent(Nephew, Parent1),
    sibling(Uncle, Parent1).

```

```
% c:/users/jorge.desktop-rl89eh2/onedrive/documentos/escom/examenespp/parcial1/partel1/prologo/familiaextendida compiled 0.00 sec, -2 clauses
?- greatgrandparent(juan, tabo).
true.

?- greatgrandparent(juan, X).
X = tabo.

?- greatgrandparent(X, tabo).
X = juan.

?- greatgreatgrandparent(juan, adan).
true.

?- greatgreatgrandparent(juan, X).
X = adan.

?- greatgreatgrandparent(X, adan).
X = juan.

?- nephew(juan, maria).
true.

?- nephew(juan, X).
X = maria
Unknown action: [] (h for help)
Action?
Unknown action: [] (h for help)
Action? .

?- nephew(X, maria).
X = juan.

?-
```

Ejercicio 5:

- Código y ejecución del programa

```
factorial.lisp X
C: > Users > jorge.DESKTOP-RL89EH2 > OneDrive > Documentos > ESCOM > examenesPP > parcial1 > Parte1 > lisp > factorial.lisp
2  Autores:
3  1) Delgado Acosta Luis Bernardo
4  2) Díaz Jiménez Jorge Arif
5  3) Valdés Luis Eliot Fabian
6
7  Grupo: 3BV2
8
9  Fecha de realización: 20/05/2023
10
11 Planteamiento del problema:
12 A. Calcular el factorial de un número 'N'
13 B. Calcular el valor de la posición 'N' en la serie fibonacci (la serie tendra esta forma: 1, 2, 3, 5, 8, ...)
14
15 /#
16
17 (defun factorial(numero) ;Funcion que permite obtener el factorial de un número
18   (if (= numero 0) 1 ;Si el numero llega a 0, este devolvera '1'
19       (* numero (factorial (- numero 1))) ;Retorna el numero multiplicado por su valor menos 1
20   )
21 )
22
23 (defun fibonacci(pos) ;Función que permite obtener el valor de una posición 'N' en la serie Fibonacci
24   (if (> pos 2) (+ (fibonacci (- pos 2)) (fibonacci (- pos 1))) ;Devuelve la suma de los dos valores anteriores al valor actual
25       pos ;Devuelve el valor actual si 'pos' es menor a 2
26   )
27 )
28
29 (defun menu() ;Funcion del menu principal
30   (princ "Ingrese el numero para calcular factorial: ")
31   (setq valor (read)) ;Obtenemos el valor de factorial que desea calcular el usuario
32   (print (factorial valor)) ;Imprime el factorial
33   (terpri);Salto de linea
34   (princ "Ingrese la posición de fibonacci: ")
35   (setq valor2 (read));Obtenemos el valor de la posición en la serie Fibonacci que el usuario desea obtener
36   (print (fibonacci valor2)) ;Imprime la posición de Fibonacci
37 )
38 (menu) ;Ejecución del programa completo
```

