UNSA

UNIVERSIDAD NACIONAL DE SAN AGUSTÍN DE AREQUIPA

Estructuras Discretas II

Docente: Carlo Corrales Delgado

Actividad

Ejercicios extras árboles binarios

Escuela:

Ciencia de la computación (Primer año)

Temas:

-Árboles Binarios

Alumno:

Josue Gabriel Sumare Uscca

Ejercicios

1)Bloque de ejercicios 1

a. Ejercicio 1

-Código:

```
Arbol2=["a",["b",["d",[],["h",[],[]]],["e",[],[]]],["c",["f",[],[]]
,["g",["i",[],[]],["j",["k",[],[]],[]]]]]
def raizArbin(a):
  return a[0]
def izqArbin(a):
  return a[1]
def derArbin(a):
  return a[2]
def vacioArbin(a):
  return a==[]
def preOrden(a):
  if not(vacioArbin(a)):
        print(raizArbin(a))
        preOrden(izqArbin(a))
        preOrden(derArbin(a))
def inOrden(a):
   if not(vacioArbin(a)):
        inOrden(izqArbin(a))
        print(raizArbin(a))
        inOrden(derArbin(a))
def postOrden(a):
   if not(vacioArbin(a)):
        postOrden(izqArbin(a))
        postOrden(derArbin(a))
        print(raizArbin(a))
def alturaArbin(a):
    if vacioArbin(a):
       return 0
    elif vacioArbin(izqArbin(a)) and vacioArbin(derArbin(a)):
       return 1
    else:
      izq=alturaArbin(izqArbin(a))
      der=alturaArbin(derArbin(a))
      if izq<der:
        return der+1
      else:
        return izq+1
def pesoArbol(a):
  if vacioArbin(a):
    return 0
  else:
      return 1+pesoArbol(izqArbin(a))+pesoArbol(derArbin(a))
```

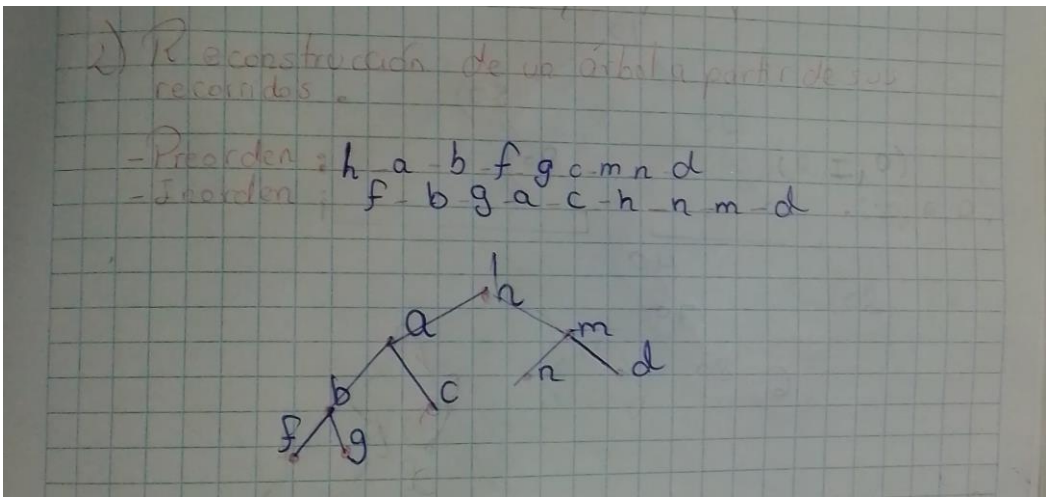
```
print("Preorden")
preOrden(Arbol2)
print("Inorden")
inOrden(Arbol2)
print("Postorden")
postOrden(Arbol2)
```

-Compilación:

```
Preorden
a
b
d
h
e
c
f
g
i
j
k
Inorden
d
h
b
e
a
f
c
i
g
k
j
Postorden
h
d
e
b
f
i
k
j
g
c
a
```

b. ejercicio 2

```
Arbol2=[8,[3,[1,[],[]],[6,[4,[],[]],[7,[],[]]]],[10,[],[14,[13,[],[
]],[]]]]
```




```
def raizArbin(a):
  return a[0]
def izqArbin(a):
  return a[1]
```

```
def derArbin(a):
  return a[2]
def vacioArbin(a):
  return a==[]
def preOrden(a):
  if not(vacioArbin(a)):
        print(raizArbin(a))
        preOrden(izqArbin(a))
        preOrden(derArbin(a))
def inOrden(a):
   if not(vacioArbin(a)):
        inOrden(izqArbin(a))
        print(raizArbin(a))
        inOrden(derArbin(a))
def postOrden(a):
   if not(vacioArbin(a)):
        postOrden(izqArbin(a))
        postOrden(derArbin(a))
        print(raizArbin(a))
def alturaArbin(a):
    if vacioArbin(a):
       return 0
    elif vacioArbin(izqArbin(a)) and vacioArbin(derArbin(a)):
       return 1
    else:
      izq=alturaArbin(izqArbin(a))
      der=alturaArbin(derArbin(a))
      if izq<der:
        return der+1
      else:
        return izq+1
def pesoArbol(a):
  if vacioArbin(a):
    return 0
  else:
      return 1+pesoArbol(izqArbin(a))+pesoArbol(derArbin(a))
print("Preorden")
preOrden(Arbol2)
print("Inorden")
inOrden(Arbol2)
print("Postorden")
postOrden(Arbol2)
```

-Compilación :

```
Preorden
8
3
1
6
4
7
10
14
13
Inorden
1
3
4
6
7
8
10
13
14
Postorden
1
4
7
6
3
13
14
10
8
```


c. ejercicio 3

-Código:

```
Arbol2=["A",["B",["D",["G",[],[]],[]],["E",["H",[],[]],["I",[],[]]]
],["C",[],["F",["J",[],["K",[],[]]],[]]]]
def raizArbin(a):
  return a[0]
def izqArbin(a):
  return a[1]
def derArbin(a):
  return a[2]
def vacioArbin(a):
  return a==[]
def preOrden(a):
  if not(vacioArbin(a)):
        print(raizArbin(a))
        preOrden(izqArbin(a))
        preOrden(derArbin(a))
def inOrden(a):
   if not(vacioArbin(a)):
        inOrden(izqArbin(a))
        print(raizArbin(a))
        inOrden(derArbin(a))
def postOrden(a):
   if not(vacioArbin(a)):
        postOrden(izqArbin(a))
        postOrden(derArbin(a))
        print(raizArbin(a))
def alturaArbin(a):
    if vacioArbin(a):
```

```
        return 0
    elif vacioArbin(izqArbin(a)) and vacioArbin(derArbin(a)):
        return 1
    else:
      izq=alturaArbin(izqArbin(a))
      der=alturaArbin(derArbin(a))
      if izq<der:
        return der+1
      else:
        return izq+1
def pesoArbol(a):
  if vacioArbin(a):
    return 0
  else:
      return 1+pesoArbol(izqArbin(a))+pesoArbol(derArbin(a))
print("Preorden")
preOrden(Arbol2)
print("Inorden")
inOrden(Arbol2)
print("Postorden")
postOrden(Arbol2)
```

-Compilación:

```
Preorden
A
B
D
G
E
H
I
C
F
J
K
Inorden
G
D
B
H
E
I
A
C
J
K
F
Postorden
G
D
H
I
E
B
K
J
F
C
A
```

2)Bloque de ejercicios 2

a. Encontrar la raíz y subdividir los recorridos.

Preorden: h – a – b – f – g – c – m – n – d

Inorden:

f – b – g – a – c –h – n – m – d

Código:

```
Arbol2=["h",["a",["b",["f",[],[]],["g",[],[]]],["c",[],[]]],["m",["
n",[],[]],["d",[],[]]]]
def raizArbin(a):
  return a[0]
def izqArbin(a):
  return a[1]
def derArbin(a):
  return a[2]
def vacioArbin(a):
  return a==[]
def preOrden(a):
  if not(vacioArbin(a)):
        print(raizArbin(a))
        preOrden(izqArbin(a))
        preOrden(derArbin(a))
def inOrden(a):
   if not(vacioArbin(a)):
        inOrden(izqArbin(a))
        print(raizArbin(a))
        inOrden(derArbin(a))
def postOrden(a):
   if not(vacioArbin(a)):
        postOrden(izqArbin(a))
        postOrden(derArbin(a))
        print(raizArbin(a))
def alturaArbin(a):
    if vacioArbin(a):
       return 0
    elif vacioArbin(izqArbin(a)) and vacioArbin(derArbin(a)):
       return 1
    else:
      izq=alturaArbin(izqArbin(a))
      der=alturaArbin(derArbin(a))
      if izq<der:
        return der+1
      else:
        return izq+1
def pesoArbol(a):
  if vacioArbin(a):
    return 0
  else:
      return 1+pesoArbol(izqArbin(a))+pesoArbol(derArbin(a))
#Ejercicio A
print("-La raiz del arbol es  ")
print(raizArbin(Arbol2))
#recorridos
print("Recorrido preorden")
print("Recorrido hijo izquierdo")
preOrden(izqArbin(Arbol2))
```

```
print("Recorrido hijo derecho")
preOrden(derArbin(Arbol2))
print("Recorrido inorden")
print("Recorrido hijo izquierdo")
inOrden(izqArbin(Arbol2))
print("Recorrido hijo derecho")
inOrden(derArbin(Arbol2))
print("Recorrido postorden")
print("Recorrido hijo izquierdo")
postOrden(izqArbin(Arbol2))
print("Recorrido hijo derecho")
postOrden(derArbin(Arbol2))
#Ejercicio B
print("-
Los pesos del subarbol izquierdo del arbol con raiz en h con 5 vert
ices ")
#raiz a
print("1.El peso del subarbol con raiz en a es")
print(pesoArbol(izqArbin(Arbol2)))

#raiz b
print("2.El peso del subarbol b es")
print(pesoArbol(izqArbin(izqArbin(Arbol2))))
#raiz c
print("3.El peso del subarbol c es")
print(pesoArbol(derArbin(izqArbin(Arbol2))))
#raiz f
print("4.El peso del subarbol f es")
print(pesoArbol(izqArbin(izqArbin(izqArbin(Arbol2)))))
#raiz g
print("5.El peso del subarbol g es")
print(pesoArbol(derArbin(izqArbin(izqArbin(Arbol2)))))
print("-
Los pesos del subarbol derecho del arbol con raiz en h con 3 vertic
es ")
#raiz m
print("1.El peso del subarbol con raiz en m es")
print(pesoArbol(derArbin(Arbol2)))

#raiz n
print("2.El peso del subarbol n es")
print(pesoArbol(izqArbin(derArbin(Arbol2))))
#raiz d
print("3.El peso del subarbol c es")
print(pesoArbol(derArbin(derArbin(Arbol2))))
#recorridos preorden subarboles
```

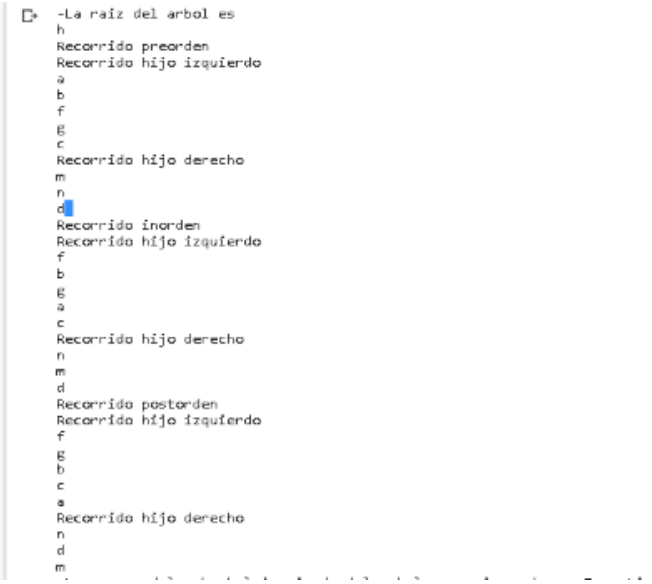
```
print("-
Los recorridos preorden del subarbol izquierdo del arbol con raiz e
n h con 5 vertices ")
#raiz a
print("1.El recorrido preorden subarbol con raiz en a es")
preOrden(izqArbin(Arbol2))

#raiz b
print("2.El recorrido preorden subarbol con raiz en b es")
preOrden(izqArbin(izqArbin(Arbol2)))
#raiz c
print("3.El recorrido preorden subarbol con raiz en c es")
preOrden(derArbin(izqArbin(Arbol2)))
#raiz f
print("4.El recorrido preorden subarbol con raiz en f es")
preOrden(izqArbin(izqArbin(izqArbin(Arbol2))))
#raiz g
print("5.El recorrido preorden subarbol con raiz en g es")
preOrden(derArbin(izqArbin(izqArbin(Arbol2))))
print("-
Los recorridos preorden del subarbol derecho del arbol con raiz en
h con 5 vertices  ")
#raiz m
print("1.El recorrido preorden subarbol con raiz en m es")
preOrden(derArbin(Arbol2))
#raiz n
print("2.El recorrido preorden subarbol con raiz en n es")
preOrden(izqArbin(derArbin(Arbol2)))
#raiz d
print("3.El recorrido preorden subarbol con raiz en d es")
preOrden(derArbin(derArbin(Arbol2)))
#Ejercicio C
#recorridos en h a y m
print("Recorrido en h ")
print("Preorden ")
preOrden(Arbol2)
print("Inorden ")
inOrden(Arbol2)
print("Postorden ")
postOrden(Arbol2)
print("Recorrido en a ")
print("Preorden ")
preOrden(izqArbin(Arbol2))
print("Inorden ")
inOrden(izqArbin(Arbol2))
print("Postorden ")
postOrden(izqArbin(Arbol2))
print("Recorrido en m ")
print("Preorden ")
```

```
preOrden(derArbin(Arbol2))
print("Inorden ")
inOrden(derArbin(Arbol2))
print("Postorden ")
postOrden(derArbin(Arbol2))
#Ejercicio D
#Recorrido preorden con h a b m n d
#raiz h
print("El recorrido preorden con raiz en h es ")
preOrden(Arbol2)
#raiz a
print("El recorrido preorden con raiz en a es ")
preOrden(izqArbin(Arbol2))
#raiz b
print("El recorrido preorden con raiz en b es ")
preOrden(izqArbin(izqArbin(Arbol2)))
#raiz m
print("El recorrido preorden con raiz en m es ")
preOrden(derArbin(Arbol2))
#raiz n
print("El recorrido preorden con raiz en n es ")
preOrden(izqArbin(derArbin(Arbol2)))
#raiz d
print("El recorrido preorden con raiz en d es ")

preOrden(derArbin(derArbin(Arbol2)))
#Ejercicio E
```

Compilación:



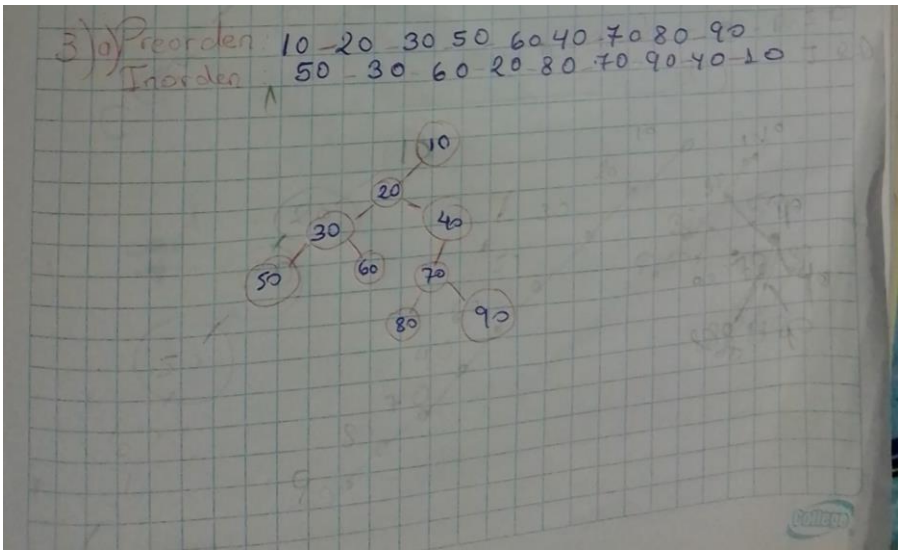
```
-Los pesos del subarbol izquierdo del arbol con raiz en h con 5 vertices
1.El peso del subarbol con raiz en a es
5
2.El peso del subarbol b es
3
3.El peso del subarbol c es
1
4.El peso del subarbol f es
1
5.El peso del subarbol g es
1
-Los pesos del subarbol derecho del arbol con raiz en h con 3 vertices
1.El peso del subarbol con raiz en m es
3
2.El peso del subarbol n es
1
3.El peso del subarbol c es
1
```

```
-Los recorridos preorden del subarbol izquierdo del arbol con raiz en h con 5 vertices
1.El recorrido preorden subarbol con raiz en a es
a
b
f
g
c
2.El recorrido preorden subarbol con raiz en b es
b
f
g
3.El recorrido preorden subarbol con raiz en c es
c
4.El recorrido preorden subarbol con raiz en f es
f
5.El recorrido preorden subarbol con raiz en g es
g
-Los recorridos preorden del subarbol derecho del arbol con raiz en h con 5 vertices
1.El recorrido preorden subarbol con raiz en m es
m
n
d
2.El recorrido preorden subarbol con raiz en n es
n
3.El recorrido preorden subarbol con raiz en d es
d
```

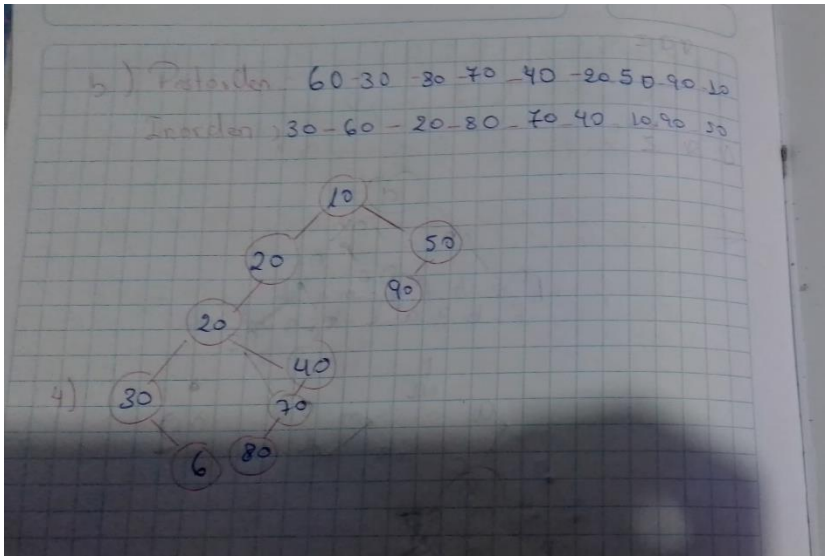
```
El recorrido preorden con raiz en h es
h
a
b
f
g
c
m
n
d
El recorrido preorden con raiz en a es
a
b
f
g
c
El recorrido preorden con raiz en b es
b
f
g
El recorrido preorden con raiz en m es
m
n
d
El recorrido preorden con raiz en n es
n
El recorrido preorden con raiz en d es
d
```

3)

a)



b)



4)

Código:

```
def unirArboles(raiz,arbol1,arbol2):
  return [raiz,[arbol1,[],[]],[arbol2,[],[]]]
Arbol2=["h",["a",["b",["f",[],[]],["g",[],[]]],["c",[],[]]],["m",["
n",[],[]],["d",[],[]]]]

Arbol=["a",["b",[],[]],["c",[],[]]]
a=unirArboles("n",Arbol,Arbol2)
print(a)
```

Compilación:

```
def unirArboles(raiz,arbol1,arbol2):
  return [raiz,[arbol1,[],[]],[arbol2,[],[]]]
Arbol2=["h",["a",["b",["f",[],[]],["g",[],[]]],["c",[],[]]],["m",["n",[],[]],["d",[],[]]]]
Arbol=["a",["b",[],[]],["c",[],[]]]
a=unirArboles("n",Arbol,Arbol2)
print(a)
```

```
['n', [['a', ['b', [], []], ['c', [], []]], [], []], [['h', ['a', ['b', ['f', [], []], ['g', [], []]], ['c', [], []]], ['m', ['n', [], []], ['d', [], []]]], [], []]]
```

10)

Código:

```
Arbol2=["h",["a",["b",["f",[],[]],["g",[],[]]],["c",[],[]]],["m",["
n",[],[]],["d",[],[]]]]
def raizArbin(a):
  return a[0]
def izqArbin(a):
  return a[1]
def derArbin(a):
  return a[2]
def vacioArbin(a):
  return a==[]
def preOrden(a):
  if not(vacioArbin(a)):
        print(raizArbin(a))
        preOrden(izqArbin(a))
        preOrden(derArbin(a))
def inOrden(a):
   if not(vacioArbin(a)):
        inOrden(izqArbin(a))
        print(raizArbin(a))
        inOrden(derArbin(a))
def postOrden(a):
   if not(vacioArbin(a)):
        postOrden(izqArbin(a))
        postOrden(derArbin(a))
        print(raizArbin(a))
def alturaArbin(a):
    if vacioArbin(a):
       return 0
    elif vacioArbin(izqArbin(a)) and vacioArbin(derArbin(a)):
       return 1
    else:
      izq=alturaArbin(izqArbin(a))
      der=alturaArbin(derArbin(a))
      if izq<der:
        return der+1
      else:
        return izq+1
```

```
def pesoArbol(a):
  if vacioArbin(a):
    return 0
  else:
      return 1+pesoArbol(izqArbin(a))+pesoArbol(derArbin(a))
print("El peso del arbol es Arbol 2 ",end=" ")
print(pesoArbol(Arbol2))
```

Compilación:

```
El peso del arbol es  9
```

Cuestionario:

1.¿Cuáles son las principales implementaciones de Árboles en Python?

-En algoritmos de estructuras de datos como en la Busqueda binaria, ademas de una forma mas clara de estos algoritmos

2.¿Cuál es tu opinión sobre la importancia de los Árboles para la resolución de problemas?

-Son estructuras que cuando son gráficas facilitan la comprensión del problema ya que podemos ver ramificaciones como conexiones , caminos , costos ,etc dependiendo del problema.

3.¿Cómo son los algoritmos de las aplicaciones de Árboles en Python?

-En su mayoria son de tipo recursivo debido a que se necesita entrar hasta una profundidad muchas veces desconocida por lo que bucles anidados en estos problemas serian muy poco útiles , por lo que usamos la recursividad.

4.¿Cual es tu opinión sobre la importancia de las aplicaciones de Árboles?

-Es un campo aun por mayor exploración ya que se le puede sacar muchas utilidades y mas aun aplicado a algoritmos que me permitiran la automatización de la resolución de este problema.