```
# TD_2_Arbres_binaires.py

001| from graphviz import Digraph
002|
003| graphe = Digraph(strict = True)
004|
005|
006|
007|
008| class Arbre :
009|     nb_arbre = 0;
010|     def __init__(self, val=None, fg =
None,fd=None) -> None:
011|         self.val = str(val)
012|         self.fg = fg
013|         self.fd = fd
014|         self.__class__.nb_arbre += 1
015|         self.num =
str(self.__class__.nb_arbre)
016|         graphe.node(self.num,self.val) #trace
le noeud
017|
018|
019|     def trace_graphique(self):
020|         """    Trace les ponts de l'arbre sous
le self    """
021|
022|         l = [self]
023|         while l!= [] :
024|             nd =  l.pop(0)
025|             if nd.fg != None:
026|                 graphe.edge(nd.num,nd.fg.num)
027|                 l.append(nd.fg)
028|             if nd.fd != None :
029|                 graphe.edge(nd.num,nd.fd.num)
030|                 l.append(nd.fd)
031|
032|
033|     def inserer_droite(self,valeur):
034|         if self.fd == None :
```

```
035|             self.fd =  Arbre(str(valeur))
036|         else :
037|             nouvel_arbre = Arbre(str(valeur))
038|             nouvel_arbre.fd = self.fd
039|             self.fd = nouvel_arbre
040|
041|     def inserer_gauche(self,valeur):
042|         if self.fg == None:
043|             self.fg = Arbre(str(valeur))
044|         else :
045|             nouvel_arbre = Arbre(str(valeur))
046|             nouvel_arbre.fg = self.fg
047|             self.fg = nouvel_arbre
048|
049|     def  get_valeur(self) :
050|         return self.val
051|
052|     def get_gauche(self):
053|         return self.fg
054|
055|     def get_droite(self):
056|         return self.fd
057|
058|
059|
060|
061|
062|
063| class File:
064|     def __init__(self):
065|         self.file = []
066|
067|     def enfiler(self, valeur):
068|         self.file.append(valeur)
069|
070|     def est_Vide(self):
071|         return self.file == []
072|
073|     def taille(self):
074|         return len(self.file)
```

```
075|
076|
077|     def defiler(self):
078|         if self.file:
079|             return self.file.pop(0)
080|
081| class Pile:
082|
083|     def __init__(self):
084|         self.valeurs = []
085|
086|     def empiler(self, valeur):
087|         self.valeurs.append(valeur)
088|
089|     def depiler(self):
090|         if self.valeurs:
091|             return self.valeurs.pop()
092|
093|     def est_Vide(self):
094|         return self.valeurs == []
095|
096|     def taille(self):
097|         return len(self.valeurs)
098|
099|
100| def parcours_largeur(A):
101|     if A == None :
102|         return None
103|     else:
104|         F= File()
105|         F.enfiler(A)
106|         print("Parcours largeur ",end='')
107|         while not( F.est_Vide()) :
108|             elt = F.defiler()
109|             print(f" {elt.val}  ",end='')
110|             if elt.fg != None :
F.enfiler(elt.fg)
111|             if elt.fd != None :
F.enfiler(elt.fd)
112|
```

```
113|
114| def hauteur(A):
115|     if A == None : return 0
116|     else: return 1+
max(hauteur(A.fg),hauteur(A.fd))
117|
118|
119| def afficheNiveau(A, niveau):
120|         if A != None:
121|             if niveau == 1:
122|                 print( A.val, end=" ")
123|             else:
124|                 afficheNiveau(A.get_gauche(),
niveau-1)
125|                 afficheNiveau(A.get_droite(),
niveau-1)
126|
127| def parcoursLargeurs_n2(A):
128|         h = hauteur(A)
129|         for i in range(1, h+1):
130|             afficheNiveau(A, i)
131|
132|
133| def trouve(A, valeur) :
134|     if A == None :
135|         return False
136|     else:
137|         F= File()
138|         F.enfiler(A)
139|
140|         while not( F.est_Vide()) :
141|             elt = F.defiler()
142|
143|             if elt .val == valeur  : return
True
144|             if elt.fg != None :
F.enfiler(elt.fg)
145|             if elt.fd != None :
F.enfiler(elt.fd)
146|         return False
```

```
147|
148|
149|
150|
151|
152|
153| A=Arbre(1)
154| A.inserer_gauche(2)
155| A.inserer_droite(3)
156| A.fg.inserer_gauche(4)
157| A.fg.inserer_droite(5)
158| A.fd.inserer_gauche(6)
159| A.fd.inserer_droite(7)
160|
161|
162| A.trace_graphique()
163| parcours_largeur(A)
164|
165| print(trouve(A, '3'))
166| print(trouve(A,'10'))
167| graphe.view()
```