Assignment: Young tableaus, one of the famous applications of the heap properties

Q1. Draw 4×4 tableau containing the elements {9,16,3,2,4,8,5,14,12}

Young Tableau

[ [ 2, 3, 5, 14]

[4, 8, 9, 16]

[12, 99, 99, 99]

[99, 99, 99, 99] ]

Q2. Argue that an m×n Young tableau Y is empty if Y[1,1]=∞. Argue that Y is full (contains mn elements) if Y[m,n]<∞.

Any Element in a Young Tableau has its parents as its left and top node and its children being bottom and right nodes. Here each node is bigger than or equal to its parent node and smaller than or equal to its child nodes.

Therefore if Y[1,1] = ∞ then all elements in the first row will be ∞. And hence all elements in all the rows beneath the first row will be ∞. Therefore Y will be empty.

If the bottom right element is smaller than ∞, all the elements on the bottom row need to be smaller than ∞. But so are the other elements in the tableau, because each is smaller than the bottom element of its column. That is why Y is full (contains mn elements) if Y[m,n]<∞.

Q3 . Give an algorithm to implement EXTRACT-MIN on a nonempty m×n Young tableau that runs in O(m+n) time.

The minimum value in the tableau will obviously be the element at [1,1]. When we extract this node we replace it with infinity. However for maintaining property of tableau we will need to place infinity after all the small finite numbers. The youngify function is used to compare which rotates m\*(n-1) or (m-1)\*n elements. Being a linear recursive function its time complexity will be as follows:

T(u)=T(u-1)+O(1)

T(u)=T(u-1)+O(1)+O(1)

T(u)=O(u)

Hence we can conclude worst case time complexity as O(m+n)=O(u)

The above algorithm has been implemented in the python code in the functions extract\_minimum and youngify.

Q4. Show how to insert a new element into a nonfull m×n Young tableau in O(m+n) time

Element Insertion into Young Tableau: Initially all elements are set to a high large value. Every time an element is to be entered the tableau is checked whether its full or not. The first element will be added to the last position and compared with the each neighbor and then swapped for the tile with smaller value. In this fashion all the elements are inserted into the Young Tableau. A while loop is used for checking young tableau property which makes a linear cost to the algorithm. Therefore the time complexity will be O(m+n). This algorithm has been implemented as the function Insert in the python code.

Q5. Using no other sorting method as a subroutine, show how to use an n×n Young tableau to sort n2 numbers in O(n3) time.

We have to sort n\*n numbers in the young tableau. This is done by extracting one small element at a time and restoring the functionality of the tableau each time. After that we are extracting elements one by one and the tableau is then maintained. Therefore for n^2 numbers the complexity will be O(n^3). This algorithm has been implemented using the sort function in the python code.

Q6. Give an O(m+n)-time algorithm to determine whether a given number is stored in a given m×n Young tableau.

We have to find whether a given number is present in the Young Tableau. For this we are checking every element of the tableau whether its equal or not. If the element I greater than key then the row is reduced by one and if element is less than key then column is increased by one. Therefore in this way we find whether a given element is present in the Tableau or not. The Complexity of this function will be O(m+n) which has been implemented as the search method in the python code.

Python Code using Object Oriented Approach

import copy

class YoungTableau():

def \_\_init\_\_(self):

self.mi=0

#Function to find a given element in Youngs Tableau

def search(self,YoungTableau,key):

i=3

j=0

while i>=0 and j<=3:

if YoungTableau[i][j]==key:

print(key,'is present in the tableau at',(i,j))

return

elif YoungTableau[i][j]>key:

i=i-1

j=j

elif YoungTableau[i][j]<key:

i=i

j=j+1

if ((i==0 and j==0) or (i==3 and j==3)):

if YoungTableau[i][j]!=key:

print(key,'is not present in YoungTableau')

#Function to extract minimum element from Youngs Tableau.

#It Requires the Youngify function which has been implemented next

def extract\_minimum(self,YTab):

maxi=9999

x=YTab[0][0]

YTab[0][0]=maxi

self.Yongify(YTab,0,0)

return x

#Youngify Function required in above extract\_min function

def Yongify(self,YoungT,i,j):

x=i

y=j

if(i+1<4) and (YoungT[i][j]>YoungT[i+1][j]):

x=i+1

y=j

if(j+1<4) and (YoungT[x][y]>YoungT[i][j+1]):

x=i

y=j+1

if x!=i or y!=j:

temp= YoungT[x][y]

YoungT[x][y]=YoungT[i][j]

YoungT[i][j]=temp

self.Yongify(YoungT,x,y)

#Function to sort the elements present in the Youngs Tableau

def sort\_tab(self,YTab,A):

C=[]

#A=[9,16,3,2,4,8,5,14,12]

for i in range (len(A)):

C.append(self.extract\_minimum(YTab))

return C

#Function to Insert an element in the Tableau

def Insert(self,YTab,i,j,key):

if YTab[i][j]<key:

print('Tableau is full')

return

YTab[i][j]=key

x=i

y=j

maximum\_i=99999

m=1

while(m!=0):

m=0

if (i-1)>=0 and YTab[i][j]<YTab[i-1][j]:

x=i-1

y=j

m=m+1

if (j-1)>=0 and YTab[x][y]<YTab[i][j-1]:

x=i

y=j-1

m=m+1

temp=YTab[x][y]

YTab[x][y]=YTab[i][j]

YTab[i][j]=temp

i=x

j=y

return YTab

def Young\_Tableau(self):

A=[9,16,3,2,4,8,5,14,12]

YTab=[[99,99,99,99],[99,99,99,99],[99,99,99,99],[99,99,99,99]]

#Addition of given elements into Youngs Tableau

for i,j in enumerate(A):

y=self.Insert(YTab,3,3,j)

if i==len(A)-1:

print('Insertion of Elements:',y)

#To check if key =12 is present in the Young Tableau

self.search(YTab,12)

#Creating references of YTab to be used in sorting and extracting minimum element function

YoungTableau1=copy.deepcopy(YTab)

YoungTableau2=copy.deepcopy(YTab)

#Sorting Elements Present in the Tableau using sort function

print("Sorted Elements of Young Tableau are:",self.sort\_tab(YoungTableau2,A))

#Extracting the minimum element from the tableau

print("Minimum Element present in Young Tableau is:",self.extract\_minimum(YoungTableau1))

y = YoungTableau()

y.Young\_Tableau()

Output

Insertion of Elements: [[2, 3, 5, 14], [4, 8, 9, 16], [12, 99, 99, 99], [99, 99, 99, 99]]

12 is present in the tableau at (2, 0)

Sorted Elements of Young Tableau are: [2, 3, 4, 5, 8, 9, 12, 14, 16]

Minimum Element present in Young Tableau is: 2