DATE: / /

SUBJECT:

على مصطفى السيد النكلاوى 19016024

[1] 32 - (15 - 5) = 22 elements

[2] @Void enqueue (object data) {
Q [r] ← data;
r++; }
Void dequeue { object front ← Q [0];
for (j = 0; j < r-1; j++){
Q [j] ← Q [j+1]; }
r--;
return front }

* The dequeue operation has O(n) Time Complexity. The size of the Q is fixed

(b) i. Q is empty : f - r = 0
ii. Q is full : r = N

* Fixed size of the array
* waste of memory

(c) i. Q is empty : f = r
ii. Q is full : f = r

* When f = r, we can't tell whether the Q is full or empty

(d) i. Q is empty : f = r
Q is full : f = r+1
Void enqueue (object data) {
if (f == r+1) { raise FullQ }
Q [r] ← data;
r ← (r+1) % N; }

RS

DATE: / / SUBJECT:

[3] void find (Q, S, X) {
found ← false;
for (j=0; j<=n-1; j++) {
curr ← S.POP();
if curr == X {
found ← true;}
Q.enqueue (curr); }
for (j=0; j<=n-1; j++) {
S.push(Q.dequeue());}
for (j=0; j<=n-1; j++) {
Q.enqueue(S.POP());}
for (j=0; j<=n-1; j++) {
S.push(Q.dequeue());}
return found; }

[4] Void Push(object data) {
Q1.enqueue(data); }
object POP() {
if isEmpty() { raise EmptyQException;}
for (j=0; j<Q1.size(); j++) {
Q2.enqueue(Q1.dequeue());}
object curr ← Q1.dequeue()
for (j=0; j<Q2.size(); j++) {
Q1.enqueue(Q2.dequeue());}
return curr

DATE: / / SUBJECT:

5 Void Push(Item i) { int Priority ← 0;
Q.insert(i, Priority);
Priority ++; }
item Pop() { if Qsize == 0 {raise EmptyStack}
Priority --; return Q.DeleteMax(); }

6 A Priority queue

7 A Priority queue will be useful in this Problem because the events will be automatically sorted if the time stamp is provided as a key which means that the smallest time stamp will be always found at the front.

8 int cntLeafNodes(Node root) {
if root is Null { return 0 }
if root.left() is Null and root.right() is Null {
return 1 }
return cntLeafNodes(root.right())
+ cntLeafNodes(root.left()) + 1
}

9 bool isEq(Node root1, Node root2) {
if root1 is Null and root2 is Null {
return True }
if root1.data() != root2.data() {
return false }
return isEq(root1.right(), root2.right())
and isEq(root1.left(), root2.left())
}

RS

DATE: / /

SUBJECT:

10. Void SwapTree(Node T) {

if T is Null { return }

temp = T.left().data()

T.left().data() = T.right().data()

T.right().data() = temp

SwapTree(T.left())

SwapTree(T.Right())

}