Université d'Ottawa
Faculté de génie

École de science informatique
et de génie électrique

uOttawa

L'Université canadienne
Canada's university

University of Ottawa
Faculty of Engineering

School of Electrical Engineering
and Computer Science

CSI2110/CSI2510

Data Structures and Algorithms

Final Exam

Duration: 3 hours **December 15th, 2016**
Professors: L. Moura, M. Hoda, Y. Sami **Page 1 of 19**

Last name: ______________________

First Name: ______________________

Student number: ______________________

Signature: ______________________

Question	points
Q1-Q32	out of 32
Q33	out of 4
Q34	out of 4
TOTAL	out of 40

Closed Book.

No electronic devices allowed, except for a simple non-programmable calculator.

This exam consists of 32 multiple choice questions and 2 written questions.

Multiple choice questions must be entered in the optically-read answer sheet (Scantron form).
Written questions must be completed in the space provided (in this questionnaire).
Please add your name and student number at the top of both written question pages.

Throughout this exam, unless specified, every logarithm is in base 2.

Question 1 [1 point] Consider the following Java code:

```
public boolean search(int[] data, int key) {
     int low = 0;
     int high = data.length - 1;
     while (high >= low) {
          int m = high;
          if (data[m] == key) { return true; }
          else if (data[m] < key) { low = m + 1; }
          else  { high = m - 1; }
     }
     return false;
}
```

Of the following, which best describes the worst case running time of the algorithm as a function of the array size `data.length`:

(a) constant (b) logarithmic (c) linear (d) quadratic (e) none of the above

Question 2 [1 point] Consider the following Java code of a well known sorting method

```
public static void mySort( int [ ] num ) {
     int j;
     boolean flag = true;
     int temp;
     while ( flag ) {
            flag = false;
            for( j=0;  j < num.length -1;  j++ ) {
                   if ( num[j] < num[j+1] )   {
                           temp = num[j];
                           num[j] = num[j+1];
                           num[j+1] = temp;
                           flag = true;
                  }
            }
      }
}
```

Give the exact asymptotic complexity for the best case running time and worst case running time of `mySort` as a function of $n =$`num.length`, the number of elements on input array `num`:

(a) best case running time: $\Theta(n)$ worst case running time: $\Theta(n^2)$

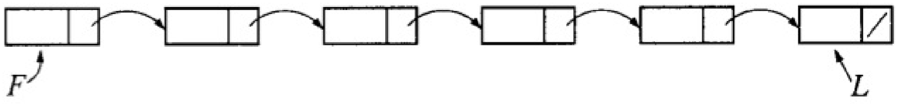
(b) best case running time: $\Theta(\log n)$ worst case running time: $\Theta(n)$

(c) best case running time: $\Theta(n)$ worst case running time: $\Theta(n \log n)$

(d) best case running time: $\Theta(n \log n)$ worst case running time: $\Theta(n^2)$

(e) none of the above

Question 3 [1 point] Consider a singly-linked list of the form



where F is a reference to the first element and L is a reference to the last element in the list. For which of the following operations the running time depends on the length of the list?

(a) Delete the last element of the list.

(b) Delete the first element of the list.

(c) Add an element after the last element of the list.

(d) Add an element before the first element of the list.

(e) Interchange the first two elements of the list.

Question 4 [1 point]
Consider the function $f(n) = 3n^2 + 100$. Which of the following assertions are TRUE:

I. $f(n)$ is $O(n^2)$.

II. $f(n)$ is $O(n^4)$.

III. $f(n)$ is $\Omega(n^4)$.

(a) I only (b) I and II only (c) I and III only (d) I, II, and III (e) None

Question 5 [1 point] Which data structure would be most appropriate to implement a collection of values with the following three characteristics?

- Items are retrieved and removed from the collection in FIFO (first in first out) order.
- There is no a priori limit on the number of items in the collection.
- The size of an item is large relative to the storage required for a memory address.

(a) Singly-linked list, with head and tail pointers

(b) Doubly-linked list, with only a head pointer

(c) Array

(d) Binary tree

(e) Hash table

Question 6 [1 point] A full binary tree is a rooted tree in which every internal node has exactly two children. How many internal nodes are there in a full binary tree with 500 leaves?

(a) 250 (b) 499 (c) 500 (d) 501 (e) 1,000

Question 7 [1 point] Select the answer that gives the most precise estimate of the worst-case running time of Mergesort, Heapsort and Quicksort, respectively.

(a) $O(n \log n), O(n \log n), O(n \log n)$

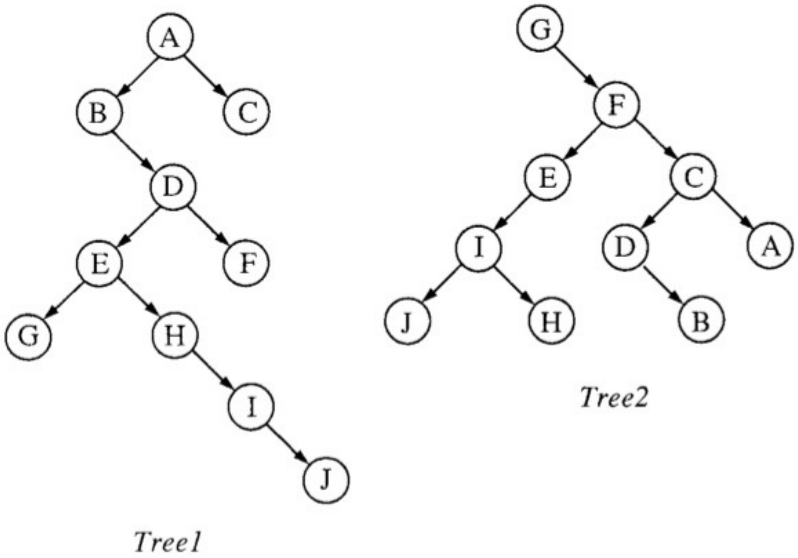
(b) $O(n \log n), O(\log n), O(n \log n)$

(c) $O(n \log n), O(n \log n), O(n^2)$

(d) $O(n^2), O(n \log n), O(n \log n)$

(e) $O(n^2), O(n^2), O(n \log n)$

Question 8 [1 point] If $Tree1$ and $Tree2$ are the trees indicated below,



which traversals of $Tree1$ and $Tree2$, respectively, will produce the same sequence of node names?

(a) Preorder, postorder.

(b) Postorder, inorder.

(c) Postorder, postorder.

(d) Inorder, inorder.

(e) Postorder, preorder.

Question 9 [1 point]
Consider the binary heap that uses the following array `a[ ]=[ 9, 7, 4, 2, 6, 1, 3]` to store its elements.
What will be the values of the remaining elements in the array after one `removeMax()` operation from the heap?

(a) `[7, 4, 2, 6, 1, 3]`

(b) `[7, 4, 6, 2, 1, 3]`

(c) `[7, 4, 6, 2, 3, 1]`

(d) `[7, 6, 4, 2, 1, 3]`

(e) `[7, 6, 4, 2, 3, 1]`

Question 10 [1 point] Consider a min-heap. Which of the following answers gives the worst-case running time of the operations: insert(), removeMin() and min(), respectively:

(a) $\Theta(\log n)$, $\Theta(\log n)$, $\Theta(1)$

(b) $\Theta(\log n)$, $\Theta(1)$, $\Theta(1)$

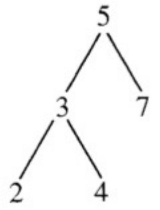
(c) $\Theta(n)$, $\Theta(\log n)$, $\Theta(1)$

(d) $\Theta(\log n)$, $\Theta(\log n)$, $\Theta(\log n)$

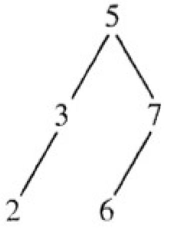
(e) $\Theta(n)$, $\Theta(n)$, $\Theta(\log n)$

Question 11 [1 point] Which of the following is not a binary search tree?

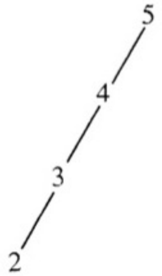
(a)



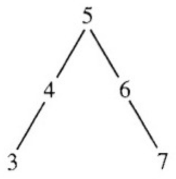
(b)



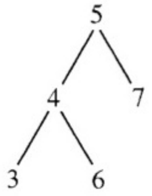
(c)



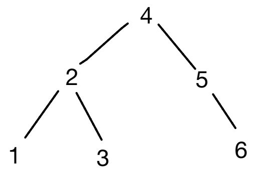
(d)



(e)



Question 12 [1 point] Consider the following AVL tree.



Which of the following sequence of insertions **DOES NOT** produce the AVL tree above:

(a) 2, 1, 4, 5, 3, 6

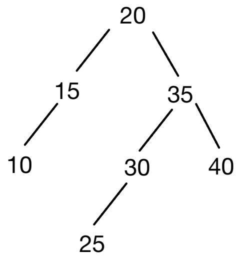
(b) 4, 2, 5, 6, 1, 3

(c) 4, 1, 5, 6, 2, 3

(d) 4, 3, 2, 1, 5, 6

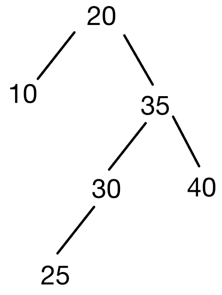
(e) 2, 4, 1, 3, 5, 6

Question 13 [1 point] Consider the following AVL tree.

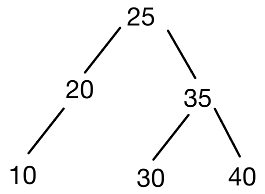


After the operation `delete(15)` the AVL tree is the following:

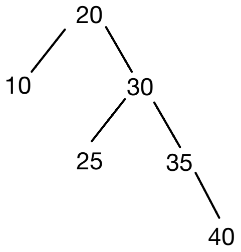
(a)



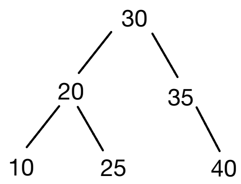
(b)



(c)



(d)



(e) None of the above.

Question 14 [1 point] Consider all the possible different (2,4)-trees that store keys 1,2,3,4,5. How many distinct such trees do exist?

(a) 1 (b) 2 (c) 3 (d) 4 (e) 5 or more.

Question 15 [1 point] Consider a (2,4)-tree with the following characteristics:

- The tree has exactly 2 levels not counting the dummy leaves (its height is $h = 2$)
- The tree stores 15 keys in total.

The following two operations are performed **one after the other**:
I. an element with a key larger than any key in the tree is inserted;
II. the element with the smallest key in the tree is deleted.

What is the height of the tree right after operation I and right after operation II, respectively?

(a) 3 and 3

(b) 3 and 2

(c) 2 and 2

(d) 2 and 1

(e) It is not possible to determine the height based on the information provided.

Question 16 [1 point] A program to check spelling has been designed in the following way. A hash table has been defined in which each entry is a Boolean variable initialized to *false*. A hash function has been applied to every word in an English dictionary, and the appropriate entry in the hash table (home address) has been set to *true*. There may be collisions but collision resolution is NOT applied. To check the spelling in a document, the hash function is applied to every word in the document, and the appropriate entry in the hash table is examined: if the entry is true the word is considered to be correct and if the entry is false the word is considered to be misspelled. One of the following alternatives is CORRECT regarding possible mistakes in this spell checker:

(a) The spell checker works correctly.

(b) There may exist a misspelled word that goes undetected while other misspelled words may be detected.

(c) There may exist a correct word that is considered to be misspelled while other correct words may be considered correct.

(d) All the misspelled words will always go undetected.

(e) All the correct words are always considered to be misspelled.

Question 17 [1 point]
Consider the following hash function $h(k) = k \bmod 9$ and a hash table of size 9.
Collisions are resolved using **quadratic probing**.
What is the table resulting from the following sequence of insertions into an initially empty table:

$$11, 21, 1, 6, 8, 16, 19$$

(a)

index:	0	1	2	3	4	5	6	7	8
hash table:	1	6	8	11	16	19	21	-	-

(b)

index:	0	1	2	3	4	5	6	7	8
hash table:	-	1	11	21	19	-	6	16	8

(c)

index:	0	1	2	3	4	5	6	7	8
hash table:	-	1	11	21	-	19	6	16	8

(d)

index:	0	1	2	3	4	5	6	7	8
hash table:	-	1	11 19	21	-	-	-	16	8

(e)

index:	0	1	2	3	4	5	6	7	8
hash table:	19	1	11	21	-	-	6	16	8

Question 18 [1 point] Consider a hash table with 10,000 addresses that hold 8,000 keys and collisions are resolved using linear probing. Assume the hash function is nearly random.
Which of the following is **FALSE**:

(a) The load factor of the table is 0.8

(b) If we insert a new key, a collision is likely to happen.

(c) Resolving collisions via chaining with a separate overflow area would not increase the average time to search for a key in the table.

(d) As the load factor increases, the hash efficiency improves and the average search time decreases.

(e) It is possible that the the hash function produces no collisions for the keys that have been inserted into the table.

Question 19 [1 point] All of the following data structures are implementations of the MAP abstract data type, except:

(a) (2,4)-tree.

(b) Heap.

(c) Hash table.

(d) AVL tree.

(e) Binary search tree.

Question 20 [1 point]

Consider a binary search tree (BST), **without** dummy leaves.
The references to left and right subtrees of a node `x` are given as `x.left` and `x.right`.
The variable `root` is a reference to the node that is the root of the tree.
Consider the following piece of code that performs `operationX()` on the BST.

```
public void operationX() {
    if (isEmpty()) return;
    root = operationY(root);
}

private Node operationY(Node x) {
    if (x.left == null) return x.right;
    x.left = operationY(x.left);
    return x;
}
```

What does `operationX()` do?

(a) If the BST is not empty, it always changes the value of the variable `root`.

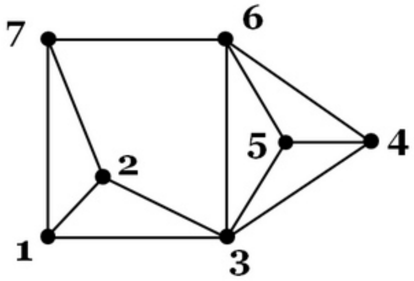
(b) It removes every node that is a left leaf in the tree.

(c) It deletes the node containing the maximum element of the BST.

(d) It deletes the node containing the minimum element of the BST.

(e) None of the above.

The graph below will be used for questions 21 and 22.
Let G be the following graph with list of adjacent vertices to each vertex given in the table below:



vertex	list of adjacent vertices
1	(2,3,7)
2	(1,3,7)
3	(1,2,4,5,6)
4	(3,5,6)
5	(3,4,6)
6	(3,4,5,7)
7	(1,2,6)

Assume that, in a traversal of G, the adjacent vertices of a given vertex are returned in the same order as they are listed in the table above.

Question 21 [1 point] Give the sequence of vertices of G visited using a Depth-First Search (DFS) traversal starting at vertex 6.

(a) 6, 3, 4, 5, 7, 1, 2

(b) 6, 4, 5, 3, 1, 2, 7

(c) 6, 3, 1, 2, 7, 4, 5

(d) 6, 5, 4, 3, 2, 1, 7

(e) None of the above.

Question 22 [1 point] Give the sequence of vertices of G visited using a Breadth-First Search (BFS) traversal starting at vertex 6.

(a) 6, 3, 4, 5, 7, 1, 2

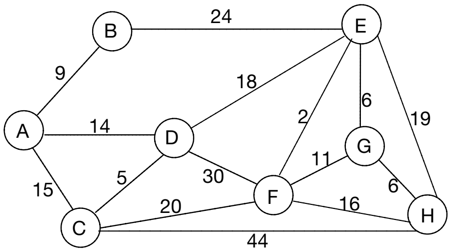
(b) 6, 4, 5, 3, 1, 2, 7

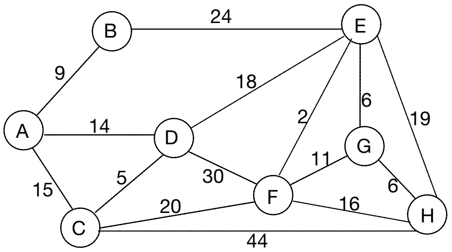
(c) 6, 3, 1, 2, 7, 4, 5

(d) 6, 5, 4, 3, 2, 1, 7

(e) None of the above.

The next three questions use the following graph G_1 with given weights (distances) on the edges (the graph is repeated for your simulations)





Question 23 [1 point] What is the total weight of a minimum spanning tree (MST) of G_1?
(a) 48 (b) 60 (c) 65 (d) 70 (e) More than 70.

Question 24 [1 point] Let T_{SP} be the tree of **shortest paths** of G_1 **starting from vertex A** and let T_{MST} be a **minimum spanning tree** of G_1. Circle the only assertion that is **FALSE**.

(a) Both T_{SP} and T_{MST} contain the two edges of weight 6.

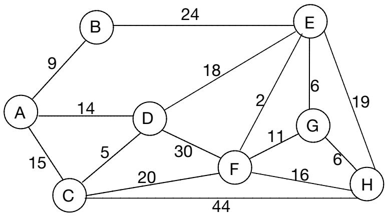
(b) T_{SP} and T_{MST} have the same number of edges.

(c) T_{SP} and T_{MST} have six edges in common.

(d) The shortest path between A and H uses exactly 4 edges.

(e) For graph G_1 and starting from vertex A, the order in which vertices join the cloud in Dijkstra algorithm and in Prim-Jarnik algorithm is NOT the same.

Question 25 [1 point] Consider the same weighted graph G_1 as the previous question.
Suppose we run Dijkstra algorithm for finding the shortest paths starting from vertex F.



The resulting distance dist[v] for every vertex v returned by Dijkstra Algorithm is:

	dist[A]	dist[B]	dist[C]	dist[D]	dist[E]	dist[F]	dist[G]	dist[H]
(a)	35	26	20	30	2	0	11	15
(b)	14	24	20	18	2	0	6	6
(c)	34	26	20	20	2	0	8	14
(d)	14	9	5	18	2	0	6	6
(e)	34	26	25	20	2	0	8	14

Question 26 [1 point]
Consider a **connected** graph G with at least 4 edges that has all **distinct** edge weights. Which of the following properties must be true of a Minimum Spanning Tree (MST) of G ?

I. The MST must contain the shortest edge of G.

II. The MST must contain the second-shortest edge of G.

III. The MST can never contain the longest edge of G.

(a) None (b) I only (c) I and II only (d) I and III only (e) I, II, and III

Question 27 [1 point] Consider the following pseudocode.

```
public static void MyAlgorithm(Graph<V,E> g) {

     OperationA();
     for (Vertex<V> v : g.vertices()) { // for every vertex v in the graph
         OperationB();
         for (Edge<E> e : g.incidentEdges(v)) { //for every edge incident to vertex v
               OperationC();
         }
     }
}
```

Suppose **g** is a connected graph with n vertices and m edges, and the graph is represented with the adjacency list data structure. Suppose the called methods have the following running times:

Method	Running time
OperationA()	$O(n)$
OperationB()	$O(\log n)$
OperationC()	$O(\log n)$

Which of the alternatives is the most precise estimate of the worst case running time of `MyAlgorithm(g)`:

(a) $O(nm \log n)$

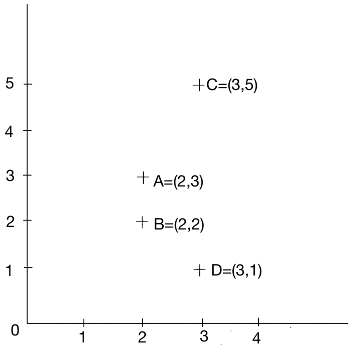
(b) $O(n^2)$

(c) $O(n^2 \log n)$

(d) $O(n^2 m)$

(e) $O((n + m) \log n)$

For the next two questions, you will use the k-clustering algorithm employed in assignment#4 to split the following data into $k = 2$ clusters.
The data consists of the following points: $A = (2, 3)$, $B = (2, 2)$, $C = (3, 5)$, $D = (3, 1)$



The clustering algorithm builds a complete graph with vertices being the data points and the weight on the edges being the distance between the corresponding points. Then, the algorithm runs some steps of Kruskal algorithm, until we have exactly 2 clusters.

Question 28 [1 point] In this question, the distance between two points is the usual distance given by the length of the segment connecting them in the plane. More precisely, the distance between $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ is $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. For example, distance((0,0),(1,1))=$\sqrt{2}$. In this case, the 2 clusters returned by the algorithm are:

(a) $\{A, B\}$ and $\{C, D\}$

(b) $\{A, B, D\}$ and $\{C\}$

(c) $\{A, C\}$ and $\{B, D\}$

(d) $\{A, B, C\}$ and $\{D\}$

(e) None of the above.

Question 29 [1 point]

In this question, we use a different notion of distance to place weight on the edges. Here the distance between two points is the number of coordinates in which they differ. More precisely, the distance between $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ is the number of nonzero coordinates in $(x_1 - x_2, y_1 - y_2)$. For example, the distance((2,3),(2,4))=1 and distance((1,5),(2,4))=2.
In this case, the 2 clusters returned by the algorithm are:

(a) $\{A, B\}$ and $\{C, D\}$

(b) $\{A, B, D\}$ and $\{C\}$

(c) $\{A, C\}$ and $\{B, D\}$

(d) $\{A, B, C\}$ and $\{D\}$

(e) None of the above.

Question 30 [1 point] Consider the following array as an input to Quicksort algorithm.

```
a=[9,5,1,4,6,7,8,2,3]
```

After partition is complete using the last element as the pivot, two recursive calls are performed involving subarrays with n_1 and n_2 elements. If the first call involves the smaller indices and the second call involves the larger indices, then n_1 and n_2 are, respectively, equal to

(a) 4, 4 (b) 4, 5 (c) 5, 3 (d) 9, 0 (e) 2, 6

Question 31 [1 point] If the following sequence (2,2) (0,4) (1,4) (0,1) (1,2) (0,6) (2,1) (1,1) is sorted using a stable sort by ascending order on the first key and then using the same method on the second key, the result would be:

(a) (0,4) (0,1) (0,6) (1,4) (1,2) (1,1) (2,2) (2,1)

(b) (0,1) (2,1) (1,1) (2,2) (1,2) (0,4) (1,4) (0,6)

(c) (2,1) (1,1) (0,1) (2,2) (1,2) (1,4) (0,4) (0,6)

(d) (0,1) (1,1) (2,1) (1,2) (2,2) (0,4) (1,4) (0,6)

(e) (0,1) (0,4) (0,6) (1,1) (1,2) (1,4) (2,1) (2,2)

Question 32 [1 point]
Consider an abstract data type whose elements are integers and whose operations are `INSERT(`y`)`, `DELETE(`y`)`, and `FINDCLOSEST(`y`)` defined to be the element x that is closest to y in the data structure (i.e. return x such that the absolute value of the difference $|x - y|$ is as small as possible). Suppose we need to minimize the worst case running time of the following method when n elements are stored in the data structure `T`:

```
void myMethod(int a, int b) {
                T.INSERT(a);
                int x=T.FINDCLOSEST(b);
                T.DELETE(x);
}
```

Which of the following data structures would be the best implementation for T in order to minimize the worst case running time of `myMethod` ?

(a) A sorted list.

(b) An unordered list.

(c) A heap.

(d) A hash table with collisions resolved by a linked list.

(e) An AVL tree.

Question 33 [4 point] **WRITTEN QUESTION**

Consider the abstract data type Deque (double ended queue) storing integers and implemented by a Java class `Deque` providing the following methods, each running in $O(1)$ time:

```
int size ();
boolean isEmpty();
int first();
int last();
void addFirst (int e);
void addLast (int e);
int removeFirst();
int removeLast();
```

Suppose you are given two deques `A` and `B`; each of these deques contain **distinct** integers sorted in ascending order. Even though each deque does not contain repeats (duplicates), there may be common elements in both deques.
Write a method with signature `public Deque combine(Deque A, Deque B)` which returns a Deque with elements sorted in ascending order that combines all elements of `A` and `B` **without duplicates**.
Your method must run in $O(n)$ where $n =$`A.size()+B.size()`.

The example below shows the effect of `combine(A,B)`

```
A:        1, 3, 5, 6, 9,10
B:        2, 3, 4, 9, 20

return:  1, 2, 3, 4, 5, 6, 9, 10, 20
```

INSTRUCTIONS:
Please, use the space provided in the next page to write your final answer to this question.
You may use the back of other pages for draft.

This space is reserved for your solution to **Question 33**.

Write your full name: ____________________
Write your student number: _______________

```
public Deque combine(Deque A, Deque B) {
```

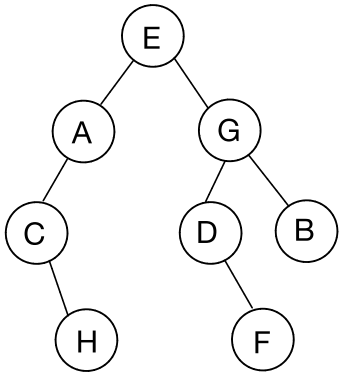
Question 34 [4 points] **WRITTEN QUESTION**

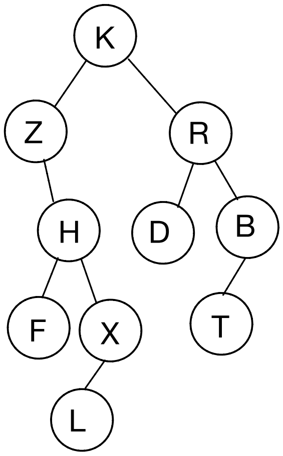
Consider a binary tree `T` with root node given by `T.root()`, where each node `q` has reference to children nodes `T.leftChild(q)` and `T.rightChild(q)`, and stores a character `T.element(q)`. The class name for the binary tree is `BinaryTree` and the class name for a node is `Node`.

Write a Java method that **prints the nodes of the binary tree `T` level-by-level from left to right**, with running time in $O(n)$ where n is the number of nodes in the tree.

You are free to use the following auxiliary data structures which you can declare and call their appropriate methods without giving implementation: stacks, queues, lists.

Examples:





For the first tree the output is: E A G C D B H F
For the second tree the output is: K Z R H D B F X T L

INSTRUCTIONS:
Please, use the space provided in the next page to write your final answer to this question.
You may use the back of other pages for draft.

This space is reserved for your solution to **question 34**.

Write your full name: ____________________
Write your student number: ______________

```
public void printByLevel(BinaryTree T) {
```

[this page is left blank for draft]

References

Questions 3, 5, 6, 8, 9, 11, 16, 26, 32 were adapted from The ETS$^{®}$ Major Field Test for Computer Science.