KATHMANDU UNIVERSITY
End Semester Examination
February/March, 2019

Level : B. E./B. Sc. | Course : COMP 202
Year : II | Semester : I
Exam Roll No. : | Time: 30 mins. | F. M. : 10
Registration No.: | Date : FEB 25 2019

SECTION "A"
[20 Q. × 0.5 = 10 marks]

Tick [√] the most appropriate answer from the alternatives given. All symbols have their usual meanings.

1. Suppose we are debugging a quick sort implementation that is supposed to sort an array in ascending order. After the first partition step has been completed, the contents of the array are in the following order:
9 1 14 17 24 22 20
Which of the following statements is correct about the partition step?
[ ] The pivot could have been either 14 or 17
[ ] The pivot could have been 14, but could not have been 17
[ ] The pivot could have been 17, but could not have been 14 3
[ ] Neither 14 nor 17 could have been the pivot

2. If the binary search algorithm determines that the search argument is in the lower half of the array, which of the following statements will set the appropriate variable to the appropriate value?
[ ] start sub = middle sub-1
[ ] start sub = middle sub + 1
[ ] stop sub = middle sub – 1
[ ] stop sub = middle sub + 1

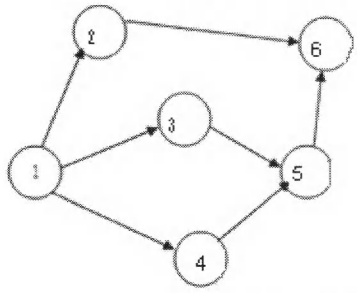
3. What is the output of following function for start pointing to first node of following linked list?
1->2->3->4->5->6

```
void fun(struct node* start)
{
  if(start == NULL)
  return;
  printf("%d  ", start->data);
  if(start->next != NULL )
  fun(start->next->next);
  printf("%d  ", start->data);
}
```

A. 1 4 6 6 4 1 B. 1 3 5 1 3 5 C. 1 2 3 5 D. 1 3 5 5 3 1
[ ] A, B and C
[ ] A, B, C and D
[ ] A, B and D
[ ] A and D

4. What is the infix version of the following postfix expression?
[ ] x 12 + z 17 y + 42 * / +
[ ] (x + 12 + z) / (17 + y * 42)
[ ] x + 12 + z / 17 + y * 42
[ ] x + 12 + z / ((17 + y) * 42)

5. Identify the binary search first path for the following graph.



[ ] 1-2-3-4-6-5 [ ] 1-4-3-2-6-5
[ ] 1-2-3-4-5-6 [ ] 1-3-2-4-5-6

6. Assume a set A of arrays of size N all of which are sorted already. Which one of the following statements it correct?
[ ] Quick sort on elements of A has a runtime in Ω (N^2)
[ ] Merge sort on elements of A has a runtime in Ω (N^2)
[ ] Insertion sort on elements of A has a runtime in Ω (N^2)
[ ] Heap sort on elements of A has a runtime in Ω (N^2)

7. Which is the correct order of the following algorithms with respect to their time complexity in the best case ?
[ ] Merge sort > Quick sort >Insertion sort > selection sort
[ ] insertion sort < Quick sort < Merge sort < selection sort
[ ] Merge sort > selection sort > quick sort > insertion sort
[ ] Merge sort > Quick sort > selection sort > insertion sort

8. What is the running time of the following code fragment?

```
for(int i=0; i<10; i++)
  for(int j=0; j<N; j++)
    for(int k=N-2; k<N+2; k++)
        cout << i << " " << j << endl;
```

[ ] (log N) [ ] O(N log N) [ ] O(N) [ ] $O(N^2)$

9. Maximum number of nodes in a binary tree with height k, where root is height 0, is:
[ ] $2^k - 1$ [ ] $2^{k+1} - 1$ [ ] $2^{k-1} + 1$ [ ] $2^k - 1$

10. In the worst case, the number of comparisons needed to search a singly linked list of length n for a given element is:
[ ] log 2 n [ ] n/2 [ ] log 2 n – 1 [ ] n

11. What may be the psuedo code for finding the size of a tree?
[ ] find_size(root_node->left_node) + 1 + find_size(root_node->right_node)
[ ] find_size(root_node->left_node) + find_size(root_node->right_node)
[ ] find_size(root_node->right_node) – 1
[ ] find_size(root_node->left_node + 1

12. The complexity of merge sort algorithm is
[ ] O(n) [ ] O(log n) [ ] $O(n^2)$ [ ] O(n log n)

13. What is missing in this logic of finding a path in the tree for a given sum (i.e. checking whether there will be a path from roots to leaf nodes with given sum)?

```
checkSum(struct bin-treenode *root , int sum) :
  if(root==null)
   return sum as 0
  else :
   leftover_sum=sum-root_node-->value
   //missing
```

[ ] code for having recursive calls to either only left tree or right trees or to both subtrees depending on their existence
[ ] code for having recursive calls to either only left tree or right trees
[ ] code for having recursive calls to either only left tree
[ ] code for having recursive calls to either only right trees

14. Suppose the elements 7, 2, 10 and 4 are inserted, in that order, into the valid 3- ary max heap found in the above question, which one of the following is the sequence of items in the array representing the resultant heap?

[ ] 10, 7, 9, 8, 3, 1, 5, 2, 6, 4
[ ] 10, 9, 8, 7, 6, 5, 4, 3, 2, 1
[ ] 10, 9, 4, 5, 7, 6, 8, 2, 1, 3
[ ] 10, 8, 6, 9, 7, 2, 3, 4, 1, 5

15. What type of sorting technique is applied in the given code?

```
int num[] = { 3, 4, 6, 7, 8 }; int i, j, temp;
for(i=0; i<n;i++)
        for(j=0;j<n; j++ )
        {  if (num[j]>num[j+1]
        { temp = num[j];
          num[j]=num[j+1];
        num[j+1]=temp;
            }}
```

[ ] Selection sort [ ] Bubble sort [ ] Insertion sort [ ] Merge sort

16. How many undirected graphs (not necessarily connected) can be constructed out of a given set V = {v1, v2, … vn} of n vertices?

[ ] n(n-1)/2 [ ] 2^n [ ] n! [ ] $2^{n(n-1)/2}$

17. In simple chaining, what data structure is appropriate?

[ ] Singly linked list
[ ] Doubly linked list
[ ] Circular linked list
[ ] Binary trees

18. What does the following function do for a given binary tree?

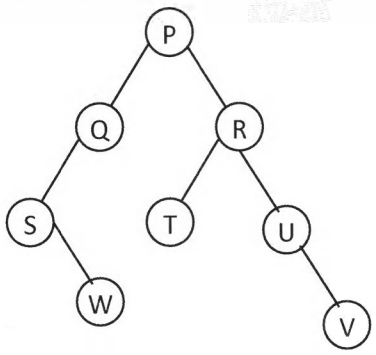
```
 int fun(struct node *root)
{
  if (root == NULL)
    return 0;
  if (root->left == NULL && root->right == NULL)
    return 0;
  return 1 + fun(root->left) + fun(root->right);
}
```

[ ] Counts leaf nodes
[ ] Counts internal nodes
[ ] Returns height where height is defined as number of edges on the path from root to deepest node
[ ] Return diameter where diameter is number of edges on the longest path between any two nodes

19. Suppose a circular queue of capacity (n – 1) elements is implemented with an array of n elements. Assume that the insertion and deletion operation are carried out using REAR and FRONT as array index variables, respectively. Initially, REAR = FRONT = 0. The conditions to detect queue full and queue empty are

[ ] Full: (REAR+1) mod n == FRONT, empty: REAR == FRONT
[ ] Full: (REAR+1) mod n == FRONT, empty: (FRONT+1) mod n == REAR
[ ] Full: REAR == FRONT, empty: (REAR+1) mod n == FRONT
[ ] Full: (FRONT+1) mod n == REAR, empty: REAR == FRONT

20. Provide the breath first traversal for the following tree in fig.



[ ] PQSWRTUV [ ] WSQTVURP [ ] PQRSTUWV [ ] SWQPTRUV

KATHMANDU UNIVERSITY
End Semester Examination
February/March, 2019

FEB 25 2019

Level : B. E./B. Sc.
Year : II
Time : 2 hrs. 30 mins.

Course : COMP 202
Semester : I
F. M. : 40

SECTION "B"
[6Q. × 4 = 24 marks]

Attempt *ANY SIX* questions.

1. The following algorithm takes as input an array, and returns the array with all the duplicate elements removed. For example, if the input array is {1, 3, 3, 2, 4, 2}, the algorithm returns {1, 3, 2,4}.

```
S = new empty set
A = new empty dynamic array
for every element x in input array
if not S.member(x) then
S.insert(x)
A.append(x)
return A
```

What is the big-O complexity of this algorithm, if the set is implemented as:
a) an AVL tree
b) a hash table
Make explanation.

2. Here is an array of ten integers: 6 3 8 9 2 7 0 3 6 5
Suppose we partition this array using quick sort's partition function and using 5 for the pivot. Draw the resulting array after the partition finishes.

3. What is Prim's Algorithms? Explain with the help of an example. [1+3]

4. Explain Binary Search Tree. How does dynamic memory allocation help in managing data? [2+2]

5. Explain the following with diagram and also write algorithm:
   (a) Insert a node at the beginning, at the end and at the specified position of singly link list.
   (b) Deleting the first node, last node and a node from a specified position in case of doubly linked list.
   (c) Deleting the first node, last node and a node from a specified position in case of circular linked list.

6. Explain the insertion sort algorithm and discuss the worst case and best case time complexities. Explain the time complexities with the suitable case scenarios. [2+2]

7. Define hashing. The table size is 10 (0.....9), hash function: h(x) = x mod 10, Insert keys 21, 32, 3, 14, 22, 27, 8. [1+3]

SECTION-"C"

[2Q. × 8 = 16 marks]

Attempt *ANY TWO* questions.

8. a. What are the two ways of representing binary trees in the memory? Which one do you prefer and why? [2]

   b. Quick sort can be described as a recursive in-place sorting algorithm that performs a partition () operation on the given array and then invokes itself twice on two distinct subranges of the array.
      i. Describe the purpose, I/O parameters and effect of the partition() procedure and explain what the pivot is. Pseudocode is not required. [2]
      ii. Give pseudocode for the quicksort () procedure that would call the partition () procedure [2]
      you described in (a). Prove that your quick sort () will always terminate. [2]
      iii. Analyse the worst-case behaviour of Quick sort and discuss possible ways of improving it. [2]

9. a. Give a concise and accurate description of a good way for selection sort to improve its performance by using insertion sort. [4]
   b. What is sorting and what are its types? Arrange following elements in insertion sort with step-by-step procedure: [4]

      14 17 19 44 22 24 32 26

10. Describe how binary search works. Describe its main idea, what kind of data it can be used on, its time complexity and its pseudo code. Give an example of how the algorithm works. [2+3+3]