ADAMA SCIENCE AND TECHNOLOGY UNIVERSITY

School of Electrical Engineering and Computing

Computer Science & Engineering Department

Data Structure and Algorithm Worksheet

Part I. Write the answer of the following questions accordingly

1. There is an array of non-negative integers. A second array is formed by shuffling the elements of the first array and deleting a random element. Given these two arrays, write a function/an algorithm to find which element missing in the second array.
2. Consider the ADT List operations
   a) Create an empty list.
   b) Destroy a list.
   c) Determine whether a list is empty.
   d) Determine the number of items in a list.
   e) Insert an item at a given position in the list.
   f) Delete the item at a given position in the list.
   g) Look at (retrieve) the item at a given position in the list.

   Using only the ADT List operations (a)-(g) above, describe how to replace an item at a given position on a list with a new item. (So, for example, given the list of integers 0,3,8,9,4,10, if the fourth item is to be replaced with a 7, the new list will be 0, 3, 8, 7, 4, 10.) You should write your answer in (precise) words, in pseudocode, or in representative C++ code.
3. Given the following Node definition for a singly linked list

```
Struct Node{
int data;
Node *next=NULL;
}list=NULL;
```

   Answer the following questions
   a) Write a function that inserts in the front
   b) Write a function that inserts in the end

c) Write a function that inserts after the n^{th} element

d) Write a function that deletes all elements that has data= item

e) Write a function that deletes n^{th} element

f) Write a function that copy the list

g) Write a function that appends list2 to list1

4. Assume that the node of a linked list is in the usual nodeType form with the info of type int and link of type nodeType pointer. The following data, as described in parts (a) to (d), is to be inserted into an initially empty list: 72, 43, 8 ,12. Suppose that head is a pointer of type nodeType. After the linked list is created, head should point to the first node of the list. Declare additional variables as you need them. Write the C++ code to create the linked list. After the linked list is created, write a code to print the list.

   a) Insert 72 into an empty linked list.

   b) Insert 43 before 72.

   c) Insert 8 at the end of the list.

   d) Insert 12 after 43.

5. **A doubly linked list is based on the following data:**

```
class DoubleList{
 // public function declarations here ...
 private:
   struct DoubleNode{
   ListItemType item;   // data in the node
   DoubleNode* next;    // points to next node
   DoubleNode* precede; // points to previous node
};
 DoubleNode* head;      // points to first node
};
```

   a) Write a member function

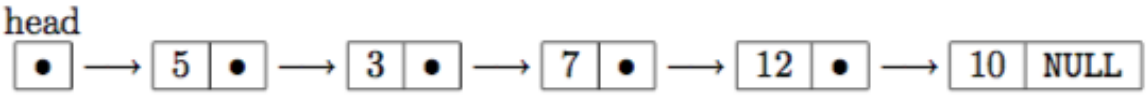
      void DoubleList::removeFirstNode() which deletes the first node from a doubly linked list.

   b) Assume the DoubleNode *curr points to a node in our doubly linked list. Write a few lines of code to remove this node curr from the list. (Consider cases: what should happen if curr is at the beginning, in the middle, or end of the list?)

6. **Consider the Node with the following members:**

```
struct Node
 {
 int item;          // the data of the node
 Node* next;        // points to the next node of the list
} Node* head;       // point to first node in the list
```

Consider the linked list of integers represented by the following diagram:



a) Draw a diagram of the above list after the following lines of code have been executed:

```
Node* prev = head->next;
Node* nodeToInsert = new Node;
nodeToInsert->item = 4;
nodeToInsert->next = prev->next;
prev->next = nodeToInsert;
```

b) Assume that the code represented above in part (a) has executed. What is the value of prev->item?

c) In addition to the code above, assume the following code executes. Draw a diagram of the list after this code executes as well.

```
prev = prev->next;
prev = prev->next;
Node* curr = prev->next;
prev->next = curr->next;
delete curr;
curr = NULL;
```

7. Which of the following operations is performed more efficiently by doubly linked list than by singly linked list?

   A. Deleting a node whose location is given

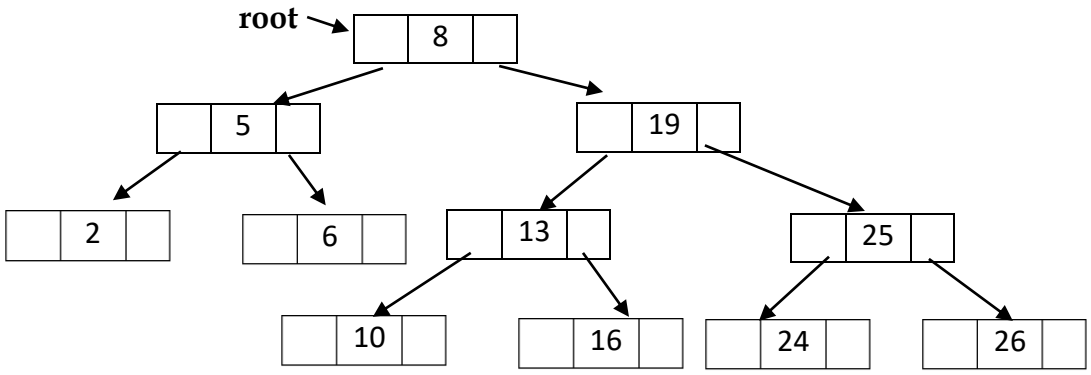
   B. Searching of an unsorted list for a given item

C. Inserting a node after the node with given location

D. Traversing a list to process each node

E. Appending a node to the tail

8. Suppose we begin with an empty stack, and perform the following operations: push 7, push 2, push 9, push 6, pop, pop, top, push 1, push 3, top, push 8, pop, top, pop, pop, push 5, push 4, pop, pop, pop, push 8. Then write the contents of a stack from top to bottom.
9. Given the following linked list representation of binary search tree:



Given the following structure based on which the tree is created:

```
struct bstree{
      int data;
      bstree *left;
      bstree *right; }
```

A. What is the output of the following code segments? **[2pts]**

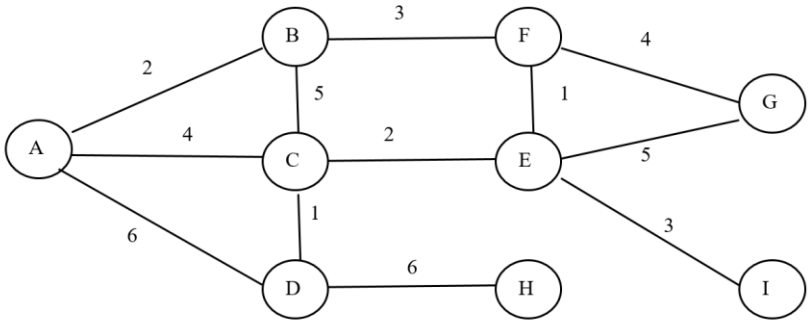
```
bstree *temp;
temp = root;
while(temp!=NULL)
{
      if(temp->data % 2 == 0){
            cout << temp->data;
            temp = temp->right;
          }
    else{
            cout << temp->data;
            temp = temp->left; }
```

```
        }
```

B. Redraw and show what the tree looks like at the end of execution of the following code segment?

```
bstree *P;
bstree *temp;
temp = root;
while(temp!=NULL)
{
   if(temp->data % 2 == 0){
        P = temp;
        temp = temp->right;
   }
   else{
                  P = temp;
                  temp = temp->left;
             }
}
bstree *t = root->right;
P->right = t->right;
P->left = t->left;
t->left->left = NULL;
delete t;
root->right = P;
```

10. Briefly explain what a binary search tree (BST) is, listing its properties, and Construct a BST of list of input in the order given A= {25, 20, 29, 100, 12, 29, 40, 1, 26, 4}
11. Explain with C++ code for deletion cases in binary search tree (BST)?
    a) deleting a leaf node
    b) deleting a node with two children
    c) deletion a root node
12. For the following graph give both adjacency matrix and adjacency list representation. In what order vertices will be visited using Depth First Search? Breadth First Search? Start at A.



13. Which best graph should you use to store a simple undirected graph with 10k vertices, 10k edges, and the neighbors are frequently enumerated? Suppose your computer only has enough memory to store 30k entries [Explain].
    a) Adjacency Matrix
    b) Adjacency List
    c) Edge List
    d) None of it
14. Here is a given binary tree:

```
        14
       /  \
      2    11
     /\   /  \
    1  3 10  30
        /      \
       7        40
```

Write the pseudocode and output of the nodes visited in following traversal methods:

a) An in-order traversal:
b) A pre-order traversal:
c) A post-order traversal:

15. Given a **Preorder**= [10,2,1,5,3,4,7,6,8,9,12,11,13,14]

    Inorder=[1,2,3,4,5,6,7,8,9,10,11,13,14] construct a binary search tree and write the postorder traversal of the tree?
16. How do we find the node with the smallest, or largest values in a binary search tree? What is the runtime complexity in both unbalanced and balanced binary search tree, in the worst case?
17. In a binary search tree, the successor of some node x is the next largest node after x. For example, in a binary search tree containing the keys 24, 39, 41, 55, 87, 92, the successor of 41

is 55. How do we find the successor of a node in a binary search tree? What is the runtime complexity of this operation?

18. Evaluate the following postfix expression using stack. At each iteration show the content of the stack. **6 5 2 3 + 8 ∗ +3 + ∗**
19. Convert the following infix expression to postfix expression using stack. At each iteration show the content of the stack and the postfix output (a + (b * (c/d))) + ((e * f + g) * h)
20. Briefly
    a) Explain limitation of the normal queue data structure with suitable examples.
    b) Describe how Queue data structure limitation can be solved.
    c) Explain clearly the advantages of implementing a queue in the form of a circular array, instead of a linear array structure?
    d) Construct a max-heap tree for list A= {100,50,25,150,1,200,250,10,75}
21. a) What strategy is being used in Merge and quick sort techniques?
    b) Explain the merge sort technique to sort the given array
    A= {54, 26, 93, 17, 77, 31, 44, 55, 20}
    c) Explain the quick sort technique to sort the given array
    A= {54, 26, 93, 17, 77, 31, 44, 55, 20}
    d) Compare and analyze the number of iteration in the two sorting algorithms above?

Part II. Develop appropriate data structure for the following problems

1. A spell checker is a program that looks at a document (file) and compares each word in the document to words stored in a dictionary. If it finds words in the dictionary, it moves on the next word, if it does not find the word, it reports the user about the misspelled (possibly) word. Formulate the problem and write a program to solve the problem by using appropriate data structures and algorithms.
2. A postfix expression of the form ab+cd-*ab/ is to be evaluated after accepting the values of a, b, c and d. The value should be accepted only once and the same value is to be used for repeated occurrence of same symbol in the expression. Formulate the problem and write a program to solve the problem by using appropriate data structures and algorithms.

3. Suppose that we are selling the services of a machine. Each user pays a fixed amount per use. However, the time needed by each user is different. We wish to maximize the returns from this

machine under the assumption that the machine is not to be kept idle unless no user is available. Whenever the machine becomes available, the user with the smallest time requirement is selected. When a new user requests the machine, he has to wait if there are pending requests. Formulate the problem and write a program to solve the problem by using appropriate data structures and algorithms.

4. Consider the database of books maintained in a library system When a user wants to check whether a particular book is available, a search operation is called for. If the book is available and is issued to the user, a delete operation can be performed to remove this book from the set of available books. When the user returns the book, it can be inserted back into the set of available books. It is essential that we are able to support the above mentioned operations as efficiently as possible as since these operations are performed quite frequently. Formulate the problem and write a program to solve the problem by using appropriate data structures and algorithms.
5. In breadth first search (BFS) of a Graph, we start at vertex ***v*** and mark it as having been visited. All unvisited vertices adjacent from *v* are visited next. The *v* is thus completely explored. The visited but at unexplored vertices are taken up next for exploration. Exploration continues until no unexplored vertex is left. If BFS is used on a connected undirected graph G, then all vertices in G get visited and the graph is completely traversed. Thus BFS can be used to check whether graph is connected. Formulate the problem and write a program to solve the problem by using appropriate data structures and algorithms.