United International University (UIU)

Dept. of Computer Science & Engineering (CSE)

Final Exam Year: 2021 Trimester: Spring

Course: CSE 2215/CSI 217 Data Structure and Algorithms I,

Total Marks: 40, Time: 1 hour 30 min, Upload & Download: 15 min

(Any examinee found adopting unfair means will be expelled from the trimester / program as per UIU disciplinary rules)

There are FOUR questions. Answer all of them. Figures in the right-hand margin indicate full marks.

---

1. a) Show the status of a STACK implemented by an array of size, m=2 for the operations: push(10), push(20), pop(), push(30), push(40), pop(), pop(), pop(). [3]

   b) How is overflow checking in a QUEUE is done when it is implemented by an array? [2]

   c) Show the mechanism of the following algorithm using the Queue of size 3. Here, Queue is a FIFO data structure, and m, f and r are size, front and rear of the Queue, respectively. What is the purpose of the algorithm? [5]

Queue	30	31		32
	0	r=1	f=2	3

```
i=(f+1)%(m+1);
while( i!=f ){
    printf("%d  ", Queue[i]);
    i=(i+1)%(m+1);
 }
```

2. a) Convert the following infix expression into postfix using a STACK. [3]

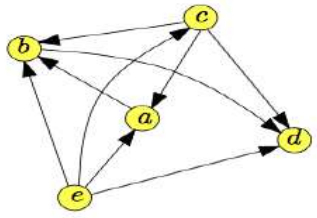
   Infix expression: a↑2-b+c/d

   b) Evaluate the postfix expression, a b – c d * + for a=2, b=3, c=2 and d=1 using STACK [3]

   c)Design a recursive/iterative algorithm for TOWER OF HANOI using one intermediate pillar/peg and show simulation for n = 3, where n is the number of disks. [4]

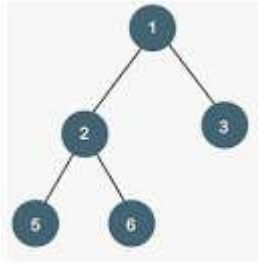
3. a) Represent the graph of Ques. 3(b) using a 2D array and a linkedlist. [2]

   b) Show the mechanism of topological ordering algorithm for the following directed acyclic graph. [5]



   c) Write an algorithm to find adjacent vertices of a given vertex from an undirected graph, G represented by a two-dimensional array. [3]

4. a) How does the following algorithm work for the given binary tree? Here, **Queue** is a FIFO data structure. [4]




```
Queue← root                    // Insert root into Queue
while(Queue !=Empty){
      v←Queue                  // Delete a vertex from Queue and store it into v
      print*, v                // Display v
      if (Left_Child(v) exists)
         Queue←Left_Child(v)   // Insert left child of v into Queue
      if (Right_Child(v) exists)
         Queue←Right_Child(v)  // Insert right child of v into Queue
}
```

b) Construct a unique binary tree from the following tree traversal sequences. [3]

Inorder: SBHEMLICG
Postorder: SHEBILGCM

c) Draw a binary search tree (BST) for the data given below. Sow the steps to delete 40 from the BST. [3]

10, 20, 30, 40, 50, 60