Department of Computer Science & Engineering

100003/CS600F-COMPREHENSIVE COURSE WORK

Data Structures- Module 1,2 &3

1. How do you initialize an array in C?
   a) int arr[3] = (1,2,3);
   b) int arr(3) = {1,2,3};
   c) int arr[3] = {1,2,3};
   d) int arr(3) = (1,2,3);
2. Which of the following is not the application of stack?
   a) A parentheses balancing program
   b) Tracking of local variables at run time
   c) Compiler Syntax Analyzer
   d) Data Transfer between two asynchronous process
3. Consider the usual algorithm for determining whether a sequence of parentheses is balanced. The maximum number of parentheses that appear on the stack AT ANY ONE TIME when the algorithm analyzes: (()(())(()))?
   a) 1
   b) 2
   c) 3
   d) 4 or more
4. What is the value of the postfix expression 6 3 2 4 + – *?
   a) 1
   b) 40
   c) 74
   d) -18
5. Here is an infix expression: 4 + 3*(6*3-12). Suppose that we are using the usual stack algorithm to convert the expression from infix to postfix notation. The maximum number of symbols that will appear on the stack AT ONE TIME during the conversion of this expression?
   a) 1
   b) 2
   c) 3
   d) 4
6. The postfix form of the expression (A+ B)*(C*D- E)*F / G is?
   a) AB+ CD*E – FG /**
   b) AB + CD* E – F **G /
   c) AB + CD* E – *F *G /
   d) AB + CDE * – * F *G /
7. The postfix form of A*B+C/D is?

a) *AB/CD+
b) AB*CD/+
c) A*BC+/D
d) ABCD+/*

8. The prefix form of A-B/ (C * D ^ E) is?
a) -/*^ACBDE
b) -ABCD*^DE
c) -A/B*C^DE
d) -A/BC*^DE

9. What is the result of the following operation?
Top (Push (S, X))
a) X
b) X+S
c) S
d) XS

10. Consider the following operation performed on a stack of size 5.

```
Push(1);
Pop();
Push(2);
Push(3);
Pop();
Push(4);
Pop();
Pop();
Push(5);
```

After the completion of all operation, the number of elements present in stack is?
a) 1
b) 2
c) 3
d) 4

11. Assume that the operators +,-, X are left associative and ^ is right associative. The order of precedence (from highest to lowest) is ^, X, +, -. The postfix expression for the infix expression a + b X c – d ^ e ^ f is?
a) abc X+ def ^^ –
b) abc X+ de^f^ –
c) ab+c Xd – e ^f^

d) -+aXbc^ ^def

12. A normal queue, if implemented using an array of size MAX_SIZE, gets full when?
a) Rear = MAX_SIZE – 1
b) Front = (rear + 1)mod MAX_SIZE
c) Front = rear + 1
d) Rear = front

13. Minimum number of queues to implement stack is ___________
a) 3
b) 4
c) 1
d) 2

14. After performing these set of operations, what does the final list look contain?

```
InsertFront(10);
InsertFront(20);
InsertRear(30);
DeleteFront();
InsertRear(40);
InsertRear(10);
DeleteRear();
InsertRear(15);
display();
```

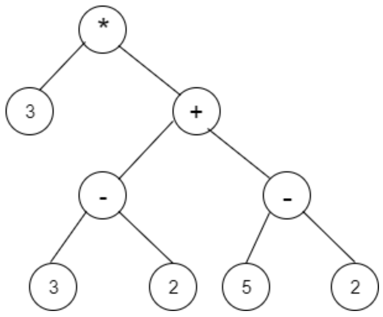
a) 10 30 10 15
b) 20 30 40 15
c) 20 30 40 10
d) 10 30 40 15

15. Which of the following statement is incorrect with respect to evaluation of infix expression algorithm?
a) Operand is pushed on to the stack
b) If the precedence of operator is higher, pop two operands and evaluate
c) If the precedence of operator is lower, pop two operands and evaluate
d) The result is pushed on to the operand stack

16. Evaluate the following and choose the correct answer.
a/b+c*d where a=4, b=2, c=2, d=1.
a) 1
b) 4

c) 5
d) 2

17. From the given expression tree, identify the infix expression, evaluate it and choose the correct result.



a) 5
b) 10
c) 12
d) 16

18. Consider an implementation of unsorted singly linked list. Suppose it has its representation with a head pointer only. Given the representation, which of the following operation can be implemented in O(1) time?

```
i) Insertion at the front of the linked list
ii) Insertion at the end of the linked list
iii) Deletion of the front node of the linked list
iv) Deletion of the last node of the linked list
```

a) I and II
b) I and III
c) I, II and III
d) I, II and IV

19. What would be the asymptotic time complexity to find an element in the linked list?

a) O(1)
b) O(n)
c) O(n^2)
d) O(n^4)

20. What does the following function do for a given Linked List with first node as head?

```
void fun1(struct node* head)
{
    if(head == NULL)
    return;
    fun1(head->next);
    printf("%d  ", head->data);
}
```

a) Prints all nodes of linked lists
b**)** Prints all nodes of linked list in reverse order
c) Prints alternate nodes of Linked List
d) Prints alternate nodes in reverse order

21. What is the functionality of the following piece of code?

```
public int function(int data)
{
            Node temp = head;
            int var = 0;
            while(temp != null)
            {
                        if(temp.getData() == data)
                        {
                                    return var;
                        }
                        var = var+1;
                        temp = temp.getNext();
            }
            return Integer.MIN_VALUE;
}
```

a) Find and delete a given element in the list
b) Find and return the given element in the list
c) Find and return the position of the given element in the list
d) Find and insert a new element in the list

22. How do you insert a node at the beginning of the list?
a)

```
public class insertFront(int data)
{
            Node node = new Node(data, head, head.getNext());
            node.getNext().setPrev(node);
            head.setNext(node);
            size++;
}
```

b)

```
public class insertFront(int data)
{
            Node node = new Node(data, head, head);
            node.getNext().setPrev(node);
            head.setNext(node);
            size++;
}
```

c)

```
public class insertFront(int data)
{
            Node node = new Node(data, head, head.getNext());
            node.getNext().setPrev(head);
            head.setNext(node);
```

```
        size++;
}
```

d)

```
public class insertFront(int data)
{
        Node node = new Node(data, head, head.getNext());
        node.getNext().setPrev(node);
        head.setNext(node.getNext());
        size++;
}
```

23. Consider the following doubly linked list: head-1-2-3-4-5-tail. What will be the list after performing the given sequence of operations?

```
        Node temp = new Node(6,head,head.getNext());
        Node temp1 = new Node(0,tail.getPrev(),tail);
        head.setNext(temp);
        temp.getNext().setPrev(temp);
        tail.setPrev(temp1);
        temp1.getPrev().setNext(temp1);
```

a) head-0-1-2-3-4-5-6-tail
b) head-1-2-3-4-5-6-tail
c) head-6-1-2-3-4-5-0-tail
d) head-0-1-2-3-4-5-tail

24. Consider a small circular linked list. How to detect the presence of cycles in this list effectively?
a) Keep one node as head and traverse another temp node till the end to check if its 'next points to head
b) Have fast and slow pointers with the fast pointer advancing two nodes at a time and slow pointer advancing by one node at a time
c) Cannot determine, you have to pre-define if the list contains cycles
d) Circular linked list itself represents a cycle. So no new cycles cannot be generated

25. What is the time complexity of following code:

```
int a = 0, i = N;
while (i > 0)
{
a += i;
i /= 2;
}
```

A. O(N)
B. O(Sqrt(N))
C. O(N / 2)
D. O(log N)

Answer Key

1. **C**
2. **D**
3. **C**
4. **D**
5. **D**
6. **C**
7. **B**
8. **C**
9. **A**
10. **A**
11. **B**
12. **A**
13. **C**
14. **D**
15. **B**
16. **B**
17. **C**
18. **B**
19. **B**
20. **B**
21. **B**
22. **A**
23. **C**
24. **B**
25. **D**