(B) 20 30 40 15
(C) 20 30 40 10
(D) 10 30 40 15

46. Which of the following is not an application of priority queue?
(A) Huffman codes
(B) Interrupt handling in operating system
(C) Undo operation in text editors
(D) Bayesian spam filter

47. What is the worst case time complexity to insert a node in a priority queue?
a) O(nlogn)
b) O(logn)
c) O(n)
d) O(n^2)

4 Linked List

1. What is the following function doing for a given singly Linked List with first node pointed by a pointer named 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

2. What is the following function doing for a given singly Linked List with first node pointed by a pointer named as *head*?

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

(A) Prints all nodes of linked lists
(B) Prints alternate nodes of Linked List
(C) Prints all nodes of Linked List except first node
(D) Prints all nodes of Linked List except first and last nodes

3. Which of the following point/s is/are true about Linked List data structure with respect to array.
(A) Due to better cache locality Arrays are better than linked lists in terms of performance.
(B)Insert and deletion of elements in a singly Linked List can be done in O(1) time.
(C) Linked list allows accessing elements in random fashion
(D) The size of array has to be pre-decided, linked lists can change their size any time as they take space from stack.
(E) All of the above

4. Consider the following function.
It accepts a pointer to the head of a standard Doubly LL.

```
void foo(struct node **hp){
struct node *p = NULL;
struct node *curr = *hp;
while (curr != NULL)
{
p = curr->prev;
curr->prev = curr->next;
curr->next = p;
curr = curr->prev;
}
if(p != NULL )
hp = p->prev;
}
```

Assume that doubly linked list is: 10 < −− > 20 < −− > 30 < −− > 50 < −− > 40 < −− > 60 and a pointer to its head is passed to foo() as a parameter. What should be the modified linked list after the function call?

(A) 60 < −− > 50 < −− > 40 < −− > 30 < −− > 20 < −− > 10
(B) 50 < −− > 60 < −− > 30 < −− > 40 < −− > 10 < −− > 20
(C) 60 < −− > 50 < −− > 40 < −− > 30 < −− > 20 < −− > 10
(D) 60 < −− > 40 < −− > 50 < −− > 30 < −− > 20 < −− > 10

5. Which of the following sorting algorithms can be used to sort a singly linked list with minimum time complexity?

   (A) Insertion Sort
   (B) Quick Sort
   (C) Heap Sort
   (D) Merge Sort

6. What is the output of following function given a linked list whose head is pointed by *start*?
   $10- > 20- > 40- > 30- > 50- > 60$

   ```
   void foo(struct node* start){
   if(start == NULL)
   return;
   printf("%d ", start− >data);
   if(start->next != NULL )
   fun(start->next->next);
   printf("%d ", start->data);
   }
   ```
   (A) $10, 40, 60, 60, 40, 10$
   (B) $10, 30, 50, 20, 40, 60$
   (A) $10, 30, 50, 50, 30, 10$
   (B) $10, 40, 50, 50, 40, 10$

7. In the worst case, the minimum number of comparisons needed to search a singly linked list of length n for a given element can be given by:
   (A) $O(log_2n)$
   (B) $O(n^2)$
   (C) $O(log_2log_2n)$
   (D) $O(n)$

8. Suppose each set is represented as a linked list with elements in arbitrary order. Which of the operations among union, intersection, membership, cardinality will be the slowest?
   (A) union only
   (B) intersection and membership
   (C) membership and cardinality
   (D) union and intersection

9. Consider the function foo() defined as follows:

```
struct node {
int data;
struct node * next;
};
int f(struct node *head){
return ((head == NULL) || (head->next == NULL) || (( head->data >=
head->next->data) && f(head->next)));
}
```

For a given linked list pointed by *head*, the function foo() returns 1 iff:
(A) the list is empty or has exactly one element
(B) the elements in the list are sorted in non-decreasing order of data value
(C) the elements in the list are sorted in non-increasing order of data value
(D) not all elements in the list have the same data value

10. To represent a Queue a linked list is used whose end node,i.e. rear, points to its first node,i.e. front. A pointer *temp* to the nodes of linked list is used to access elements of queue. In order to have $O(1)$ time complexity for both insertion and deletion in queue which node should *temp* point to?
(A) Front node
(B) Rear node
(C) The node at mid point of the Linked list
(D) Node after Front
(E) Not possible with single pointer.

11. What are the time complexities of finding K^{th} element from the beginning and K^{th} element from the end in a singly linked list containing n elements where K is some integer and n$\geq$ 8?
(A) O(K) and O(n)
(B) O(1) and O(1)
(C) O(K) and O(K)
(D) O(K) and O(n$\log n$)

12. Which of the following is true?
(A) A singly linked list can not be implemented with a node structure originally defined for a doubly linked list with some extra constant space.
(B) A doubly linked list can not be implemented with a node structure originally defined for a singly linked list with some extra constant space.
(C) Both (A) and (B) are true.
(D) Both (A) and (B) are False.

13. The concatenation of two lists is to be performed in O(1) time. Which of the following implementations of a list should be used?

(A) singly linked list
(B) doubly linked list
(C) circular doubly linked list
(D array implementation of lists

14. What is the following function *foo()* doing?
```
int foo(struct node* head) {
int count = 0;
struct node* current = head;
while (current ! = NULL || current->next ! = NULL) {
count++;
current=current->next->next;
}
return(count*2);}
```
(A) returns the length of list.
(B) returns half of number of nodes in list
(C) returns length of the list when number of elements in list is even.
(D) returns length of the list when number of elements in list is odd.

15. If pointers to first and penultimate nodes of a singly linked list are provided, which of the following operations need time as $\Theta(n)$?
(A) Delete the first element
(B) Insert new element in the beginning of list
(C) Delete the middle element of first 10 elements in the list
(D) Add a new element before the last node of the list
(E) All of the above
(F) None of the above

16. Let P be a singly linked list. Let Q be the pointer to an intermediate node x in the list. What is the worst-case time complexity of the best-known algorithm to delete the node x from the list ?
(A) O(n)
(B) O($log_2 n$)
(C) O(logn)
(D) O(1)

17. Consider the following function on a linked list of integers:
```
int foo(struct node** pointer_to_head) {
struct node* head;
int result;
head = *pointer_to_head;
assert(head != NULL);
result = head->data;
pointer_to_head = head->next;
free(head);
return(result);
}
```

To which of the following operations does foo() resemble?
(A) Push into stack
(B) delete first element from the queue
(C) Pop from the stack
(D) insert an element into the queue

18. Insertion sort performs better than many comparison sorting algorithms when the input size is small and the list is nearly sorted. Given a linked list of n integers while n≤50, what is the best time complexity for sorting this list using insertion sort while it is known that the list is at least 60% sorted.
(A) O(n)
(B) O(nlogn)
(C) O(n^2)
(D) O($n^2 logn$)
(E) O(1)

19. Consider the following function *Append(List1,List2)* which takes two linked lists and append the second list at the end of first list. There are three lines *Line 1, Line 2* and *Line 3* which are left blank.

```
void Append(struct node** L1, struct node** L2) {
struct node* curr;
if (– Line 1 –) {
L1 = *L2;
}
else {
curr = *L1;
while (– Line 2 –) {
– Line 3 –
}
curr->next = *L2; }
L2=NULL; }
```

Which of the following can be a replacement of these blank lines:
(A) *L1 == NULL, curr->next ! = NULL, curr = curr->next;
(B) *L2 == NULL, curr->next == NULL, curr = curr->next;
(C) *L1 == NULL, curr->next == NULL, curr->next = curr->next->next;
(D) *L1 == NULL, curr->next != NULL, curr->next = curr->next->next;

20. Following is a function *Mystery()* which takes a list and sorts it. *Line 1* and *Line 2* are left blank intentionally.

```
void Mystery(struct node** headRef) {
struct node* head = *headRef;
struct node* a;
struct node* b;
if ((head == NULL) —— (head->next == NULL)){
return;}
— Line 1 —;
Mystery(&a);
Mystery(&b);
— Line 2 —
}
```

Which of the following best represents the sorting algorithm used along with the best replacement of above blank lines.
(A) Quick sort, ListSplit(head, **a, **b), *headRef = Partition(a, b);
(B) Merge sort, ListSplit(head, &a, &b), *headRef = Merge(a, b);
(C) Quick sort, ListSplit(head, &a, &b), *headRef = Partition(a, b);
(D) Merge sort, ListSplit(head, **a, **b), *headRef = Merge(a, b);

21. Consider the following piece of 'C' code fragment that removes duplicates from an ordered list of integers.

```
Node *remove-duplicates(Node *head, int *j)
{
Node *t1, *t2;
j=0;
t1 = head;
if (t1! = NULL) t2 = t1 − >next;
else return head;
j = 1;
if(t2 == NULL)
return head;
while t2 != NULL)
{
if (t1.val != t2.val) ———————————— (S1)
{
(*j)++; t1 − > next = t2; t1 = t2: ——— (S2)
}
t2 = t2 − >next;
}
t1 − >next = NULL;
return head;}
```

Assume the list contains n elements ($n \geq 2$) in the following questions.
(A). How many times is the comparison in statement S1 made?
(B). What is the minimum and the maximum number of times statements marked S2 get executed?
(C). What is the significance of the value in the integer pointed to by j when the function completes?

22. Suppose there are two singly linked lists both of which intersect at some point and become a single linked list onward. The head or start pointers of both the lists are known, but the intersecting node and lengths of lists are not known.
What is worst case time complexity of optimal algorithm to find intersecting node from two intersecting linked lists?
(A) O(n*m), where m, n are lengths of given lists
(B) O(n^2), where m>n and m, n are lengths of given lists
(C) O(m+n), where m, n are lengths of given lists
(D) O(min(n, m)), where m, n are lengths of given lists

23. Suppose there is a singly linked list which may contain a loop. The head or start pointers of the list is known, but the address of node where the loop starts is not known.
What is worst case time complexity of optimal algorithm to detect a loop in linked list ?
(A) O(n^2),
(B) O(n^3),
(C) O(n),
(D) O(nlogn)),

24. Consider an implementation of unsorted single linked list. Suppose it has its representation with a head and a tail pointer (i.e. pointers to the first and last nodes of the linked list). Given the representation, which of the following operation can not be implemented in O(1) time ?
(A) Insertion at the front of the linked list.
(B) Insertion at the end of the linked list.
(C) Deletion of the front node of the linked list.
(D) Deletion of the last node of the linked list.

25. What does the following function do?

```
int foo(Node head){
int temp= 0;
if( head == null)
return 0;
Node item = head.getNext();
while(item != head)
{
item = item.getNext();
temp+=1;
}
return temp;
}
```

(A) Returns the length of the singly linked list.
(B) Returns the length of the doubly linked list.
(C) Returns the length of the circular linked list.
(D) Returns the number of elements in dequeue.

26. Which of the following applications makes use of circular linked list?
(A) Undo operation in a text editor
(B) Recursive function calls
(C) Allocating CPU to resources
(D) All of the above

27. What is the time complexity of inserting and deleting a node in a doubly linked list?
(A) O(nlogn), O(nlogn)
(B) O(logn), O(n)
(C) O(n), O(n)
(D) O(1), O(n)

28. Select among the options the best explanation about the behavior of following function:
Assume *node* is a node in linked list and is globally defined data structure along with its standard operations.

```
int foo(){
if(head == null)
return -1;
int var;
node temp = head;
while(temp.getNext() != head)
temp = temp.getNext();
if(temp == head)
{
var = head.getItem();
head = null;
return var;
}
temp.setNext(head.getNext());
var = head.getItem();
head = head.getNext();
return var;
}
```

(A) Return data from the end of the list
(B) Returns the data and deletes the node at the end of the list
(C) Returns the data from the beginning of the list
(D) Returns the data and deletes the node from the beginning of the list

29. Which of the following is not TRUE about a circular linked list?
(A) Every node has a successor
(B) Time complexity of inserting a new node at the head of the list is O(1)
(C) Time complexity for deleting the last node is O(n)
(D) None of the mentioned

5 Trees

1. Which of the following is a false statement about Binary Trees
(A) Every binary tree is either complete or full.
(B) Every complete binary tree is also a full binary tree.
(C) Every full binary tree is also a complete binary tree.
(D) No binary tree is both complete and full.

2. The maximum number of binary trees that can be formed with three unlabeled nodes is:
(A) 10
(B) 5
(C) 6
(D) 12

3. The number of leaf nodes in a rooted tree of n nodes, with each node having 0 or 3 children is:
(A) $\frac{n}{2}$
(B) $\frac{(n-1)}{3}$
(C) $\frac{(n-1)}{2}$
(D) $\frac{(2n+1)}{3}$

4. The maximum possible height of a weight-balanced tree with n nodes is best described by:
(A) $\log_2 n$