Homework of Data Structures

Chapter 2 List

Question 1

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

Question 2

Which of the following points is/are true about Linked List data structure when it is compared with array.

A It is easy to insert and delete elements in Linked List
B Random access is not allowed in a typical implementation of Linked Lists
C The size of array has to be pre-decided, linked lists can change their size any time.
D All of the above

Question 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

Question 4

In the worst case, the number of comparisons needed to search a singly linked list of length n for a given element is

A log 2 n
B n/2
C log 2 n – 1
D n

Question 5

Given pointer to a node X in a singly linked list. Only one pointer is given, pointer to head node is not given, can we delete the node X from given linked list?

A Possible if X is not last node. Use following two steps (a) Copy the data of next of X to X. (b) Delete next of X.
B Possible if size of linked list is even.
C Possible if size of linked list is odd
D Possible if X is not first node. Use following two steps (a) Copy the data of next of X to X. (b) Delete next of X.

Question 6

Consider the following function to traverse a linked list.

```
void traverse(struct Node *head)
{
    while (head->next != NULL)
    {
        printf("%d    ", head->data);
        head = head->next;
    }
}
```

Which of the following is FALSE about above function?

A The function may crash when the linked list is empty
B The function doesn't print the last node when the linked list is not empty
C The function is implemented incorrectly because it changes head

Question 7

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(log2 n)
C O(logn)
D O(1)

Question 8

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

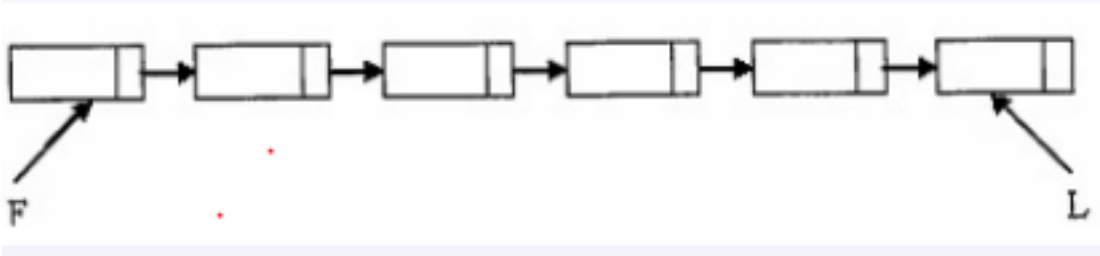
Question 9

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.

Question 10

Consider a single linked list where F and L are pointers to the first and last elements respectively of the linked list. The time for performing which of the given operations depends on the length of the linked list?



A Delete the first element of the list

B Interchange the first two elements of the list
C Delete the last element of the list
D Add an element at the end of the list

Question 11

The time required to search an element in a linked list of length n is
A O (log n)
B O (n)
C O (1)
D O(n2)

Question 12

The time required to search an element in a linked list of length n is
A O (log n)
B O (n)
C O (1)
D O(n2)

Question 13

The minimum number of fields with each node of doubly linked list is
A 1
B 2
C 3
D 4

Question 14

A doubly linked list is declared as

```
struct Node {
        int Value;
        struct Node *Fwd;
        struct Node *Bwd;
};
```

Where Fwd and Bwd represent forward and backward link to the adjacent elements of the list. Which of the following segments of code deletes the node pointed to by X from the doubly linked list, if it is assumed that X points to neither the first nor the last node of the list?

A X->Bwd->Fwd = X->Fwd; X->Fwd->Bwd = X->Bwd ;
B X->Bwd.Fwd = X->Fwd ; X.Fwd->Bwd = X->Bwd ;
C X.Bwd->Fwd = X.Bwd ; X->Fwd.Bwd = X.Bwd ;

D X->Bwd->Fwd = X->Bwd ; X->Fwd->Bwd = X->Fwd;

Question 15

A program P reads in 500 integers in the range [0..100] representing the scores of 500 students. It then prints the frequency of each score above 50. What would be the best way for P to store the frequencies?

A An array of 50 numbers
B An array of 100 numbers
C An array of 500 numbers
D A dynamically allocated array of 550 numbers

Question 16

Which of the following operations is not O(1) for an array of sorted data. You may assume that array elements are distinct.

A Find the ith largest element
B Delete an element
C Find the ith smallest element
D All of the above

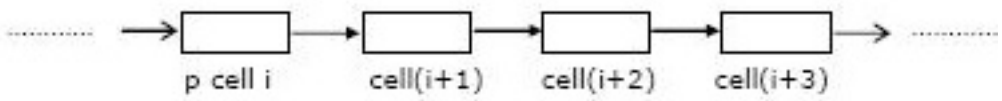
Question 17

Consider an array A[20, 10], assume 4 words per memory cell and the base address of array A is 100. What is the address of A[11, 5] ? Assume row major storage.

A 560
B 565
C 570
D 575

Question 18

Let p be a pointer as shown in the figure in a single linked list. What do the following assignment statements achieve?



q = p−>next
p−>next = q−>next
q−>next = (q−>next) −>next
(p−>next) −>next= q

Question 19

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≥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?