Data Structures and applications (II Test)

Type: MCQ

Q1. A variant of the linked list in which none of the node contains NULL pointer is? (0.5)

1. Singly linked list
2. Doubly linked list
3. Singly linked list with header node
4. **Circular linked list

Q2. Consider a singly linked list containing 3 nodes with values 10, 20 and 30 respectively and Nodeptr ***first*** pointing to the first node of the list. What is the final content of the list after the assignment statement, first->next = first->next-next->next; (0.5)

1. **10
2. 10, 20
3. 20
4. Segmentation Fault

Q3. Consider a circular singly linked list containing 3 nodes with values 'I', 'U' and 'M' respectively and Nodeptr ***first*** pointing to the first node of the list (containing 'I'). What is the final content of the list after the following statements? (0.5)

Nodeptr temp = first->next;

temp = temp->next;

first = first->next;

temp->next = first;

1. I, U, M
2. I, M
3. **U, M
4. Segmentation Fault

Q4. What is the output of the following function, with Nodeptr ***first*** pointing to the first node of the following singly linked list? 1->2->3->4->5->6. (0.5)

```
void  print(Nodeptr first){
        if (first == NULL)
                return;
        printf("%d \t ", first->data);

        if (first->next != NULL )
                first=first->next;
        print(first->next);
        printf("%d\t ", first->data);
}
```

1. 2 4 6 6 4 2
2. ** 1 3 5 6 4 2
3. 2 4 6 2 4 6
4. 1 3 5 5 3 1

5. The following C function takes a singly linked list as input argument and modifies the list by moving the last element to the front of the list and returns the modified list. Some part of the code is left blank. Choose the correct alternative to replace the blank line. (0.5)

```
Nodeptr Shift(Nodeptr first) {
  Nodeptr p, q;

  if ((first == NULL || (first->next == NULL))
        return first;
  q = NULL; p = first;
  while (p-> next !=NULL) {
        q = p;
        p = p->next;
  }
  ----------------------------------------
}
```

1. p->next=first; q->next=NULL; return first;

2. q>next=first; q->next=NULL; return q;

3. p->next=first; q=NULL; return p;

4. **p->next=first; q->next=NULL; return p;

Q6. While inserting a node in the middle of a doubly linked list, the number of pointers affected will be (0.5)

1. 3
2. **4
3. 2
4. 1

Q7. A doubly linked list is declared as

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

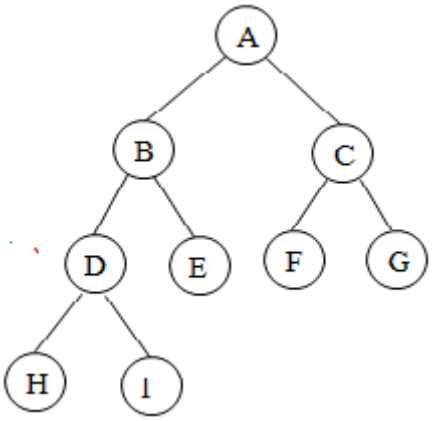
where Fwd and Bwd represent forward and backward links to the adjacent elements of the list. Which of the following segments of code deletes the node pointed to by X from the doubly linked list, if it is assumed that X points to neither the first nor the last node of the list? (0.5)

1. X->Fwd->Fwd->Bwd = X->Fwd; X->Bwd->Bwd->Fwd = X->Bwd;
2. X->Fwd->Bwd->Bwd = X->Bwd; X->Bwd->Fwd->Fwd = X->Fwd;
3. ** X->Bwd->Fwd = X->Fwd; X->Fwd->Bwd = X->Bwd;
4. X->Fwd->Bwd = X; X->Bwd->Fwd = X;

Q8. The degree of a tree is

1. the degree of its root node
2. **the maximum of the degree of the nodes in the tree
3. the maximum of the degree of the nodes in the tree + 1
4. number of levels in the tree (0.5)

Q9. The List representation of the tree shown in following figure is:



1. (A(B,C),(D,E,F,G),(H,I))
2. ((A,B,D,H),(E),(C,F),(G))
3. **(A(B(D(H,I),E),C(F,G)))
4. ((H,I),(D,E,F,G),(B,C),(A)) (0.5)

Q10. For any nonempty binary tree, if n0 is the number of leaf nodes and n2 is the number of nodes of degree 2, then

1. **n0 = n2+1.
2. n2 = n0+1.
3. n0 + n2 = 1
4. n0 = 2*n2-1. (0.5)

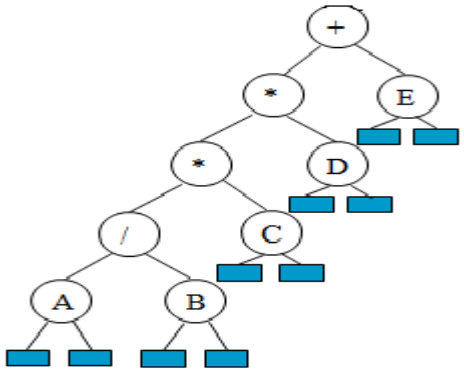
Type: DES

Q11. Given two singly linked lists representing long integer numbers in reverse order shown as L1 and L2 below, write a function ***Nodeptr AddLongInteger(Nodeptr L1, Nodeptr L2)***; which finds the sum and returns the new list. Assume function ***Nodeptr insert_rear (Nodeptr first, int data)*** which inserts the node at the rear of a list is available. The function insert_rear takes ***first*** as a pointer to the first node of the list and data and returns a pointer to the first node of the updated list. The node structure consists of {data (int), next(Nodeptr) }.
If first number is 7898 (L1=8->9->8->7), and if second number is 2345 (L2=5->4->3->2), then sum is 10243 (L3=3->4->2->0->1) (3)

Solution:

```
Nodeptr AddLongInteger(Nodeptr A, Nodeptr B){
   int digit,sum,carry;
   Nodeptr L3=NULL, r, R, a, b;
   carry = 0;
   while(A!=NULL && B!=NULL){                          --1M
      sum = A->data + B->data + carry;
      digit = sum % 10;
      carry = sum/10;
      L3=Insert_Rear(L3,digit);
      A = A->next;
      B = B->next;
   }
   if (A!=NULL) r=A;
      else     r=B;
   //add carry to remaining digits of bigger number
   while(r!= NULL){                                    --1M
      sum = r->data + carry;
      digit = sum%10;
      carry = sum/10;
      L3=Insert_Rear(L3,digit);
      r = r->next;
   }
   //Insert the last carry, if present.
   if (carry)
      L3=Insert_Front(L3,carry);                       - ½ M
   return L3;                                          -- ½ M
}
```

Q12. Write the recursive function for preorder traversal of a binary tree with the function having the signature ***void preorder(Nodeptr root); .*** Trace the operations of preorder traversal for the binary tree shown in figure by filling the below given template (The first entry is already shown).



Call of preorder	Value in root	Action
1	+	Printf

(3)

Ans:

Preorder traversal:

```
void preorder(Nodeptr root){
   if (root) {
      printf("%d", root->data);
      preorder(root->left_child);
      predorder(root->right_child);
   }
}
```
--1M

Preorder traversal on given binary tree: --2M

Call of preorder	Value in root	Action	Call of preorder	Value in root	Action
1	+	printf	11	C	printf
2	*	printf	12	NULL	
3	*	printf	11	C	
4	/	printf	13	NULL	
5	A		2	*	
6	NULL		14	D	printf
5	A		15	NULL	
7	NULL		14	D	
4	/		16	NULL	
8	B	printf	1	+	
9	NULL		17	E	printf
8	B		18	NULL	
10	NULL		17	E	
3	*		19	NULL	

Q13. Given the pointer to the last node of a *circular singly linked list* of integers, (Note: the list may be empty), write the function **Insert** to insert a given node at the beginning of the list. The function has the signature ***void Insert(Nodeptr *last, Nodeptr temp);*** where ***last*** points to the last node and **temp** is the node to be inserted. The node structure consists of {data (int), next(Nodeptr) } (2)

Solution:

```
void Insert(Nodeptr *last, Nodeptr  temp)
{
        if (*last == NULL) {
                temp->next = temp;
                *last = temp;
        }                                               -1M
        else{
                temp->next = (*last)->next;
                (*last)->next = temp;
        }                                               -1M
}
```

Q14. Given a *doubly linked list without header node* representing a word with each node containing one character. The node structure consists of {data (char), left (Nodeptr), right (Nodeptr)}. Write a function IsPalindrome having the signature: ***int IsPalindrome(Nodeptr first, Nodeptr last);*** where **first** points to the first node of the list (which contains the first character) and **last** points to the last node of the list (which contains the last character of the word). Also, given that the word contains an odd number of characters, with a minimum of one character. The function should return 1, if the word is a palindrome, else it should return 0. The function should **NOT** find the length of the list. (2)

```
int IsPalindrome(Nodeptr first, Nodeptr last){
   int flag=1;
   Nodeptr temp=first;
   while(temp!=last){                    -0.5M
     if(temp->info!=last->info) {
         flag=0;
         break;
      }
      temp=temp->right;
      last=last->left;
   }                                     -1M
   return flag;
}                                        -0.5M
```