3rd Semester (Regular)
Sub & Code DSA/CS-2001
Branch (s)
CSE/IT/ETC/EEE/AEI/EE

AUTUMN MID SEMESTER EXAMINATION-2016

Data Structure and Algorithm

[CS-2001]

Full Marks: 25 **Time: 2 Hours**

Answer any five questions including question No.1 which is compulsory.
The figures in the margin indicate full marks.
Candidates are required to give their answers in their own words as far as practicable and
<u>all parts of a question should be answered at one place only</u>.

1. 5×1

a) *What is time complexity of the function func()?*

```
void func(int arr[], int n)
{
    int i, j = 0;
    for(i=0; i<n; i++)
            while(j < n && arr[i] < arr[j])
                    j++;
}
```

Ans:

O(n) (Note: Since the *j* is not initialized inside the outer while loop, the inner while loop will be executed nearly *n* number of times)

b) *A matrix M[1..n, 1..n] is initialized as: M[i,j]=i-j for all i,j, 1 ≤ I ≤ n;1 ≤ j ≤n. Write a code for initializing M and find the sum of all the elements of the matrix M. If the value of n is 100, then what is the sum of all the elements of the matrix M.*

Ans:

```
for(i=1,sum=0;i<=n;i++)
    for(j=1;j<=n;j++){
            M[i][j]=i-j; sum +=M[i][j];
    }
```

The sum of all the elements of the matrix M is **0**.

c) *Let a doubly linked list consists of 20 elements. If a pointer called* ***ptr*** *is pointing to the 5th node of the list, then write a code to swap 5th and 6th node of the list without using any additional pointer variable and without changing the* ***ptr****'s position.*

Ans:

```
ptr->next=ptr->next->next;
ptr->prv->next=ptr->next->prv;
ptr->next->prv->prv=ptr->prv;
ptr->next->prv->next=ptr;
ptr->prv=ptr->next->prv;
ptr->next->prv=ptr;
```

d) *Define data structure and Abstract Data Type. Implement PUSH and POP operation of STACK ADT.*

Ans:

Definition (½ mark)

ADT declaration (1 mark)

```
typedef struct{
            int a[MAX];
            int top;
}STACK;
void PUSH(STACK,int);
int POP(STACK);
```

e) *struct node{ int data; struct node *next; };*

What does the following function do for the linked list: 10->20->30->40->50?

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

Ans:

50 40 50 30 50 40 50 20 50 40 50 30 50 40 50 10 50 40 50 30 50 40 50 20 50 40 50 30 50 40 50

(Note: Let the address of nodes are 100,200,300,400,500

fun(500)=> 50

fun(400)=> 50 40 50

fun(300)=> 50 40 50 30 50 40 50

fun(200)=> 50 40 50 30 50 40 50 20 50 40 50 30 50 40 50

fun(100)=> 50 40 50 30 50 40 50 20 50 40 50 30 50 40 50 10 50 40 50 30 50 40 50 20 50 40 50 30 50 40 50)

2. ***a)*** *Design a suitable data structure to efficiently represent a sparse matrix? Write an algorithm to add the original sparse matrix with the transpose of the same matrix.* 3

Ans:

Representation of sparse matrix (1 mark)

```
struct node{
            int row,col,val;
            struct node * next;
            };
      struct node * head;
      void add(){
            struct node *ptr1,*ptr2, *cur;
            if(head->row!=head->col) return;
            for(ptr1=head->next; ptr1!=NULL; ptr1=ptr1->next){
                  if(ptr1->row==ptr1->col)
                        ptr1->val *=2;
                  else{
                        flag=1;
                        for(ptr2=ptr1->next; ptr2!=NULL; ptr2=ptr2->next){
                              if(ptr1->row==ptr2->col && ptr1->col==ptr2->row){
                                    ptr1->val +=ptr2->val;
                        ptr2->val=ptr1->val;
                        flag=0;
                    }
                }
                if(flag){
                    cur=(struct node *) malloc(sizeof(struct node));
                    cur->row=ptr1->col;
                    cur->col=ptr1->row;
                    cur-val=ptr1->val;
                    cur->next=head->next;
                    head->next=cur;
                    head->val++;
                }
            }
        }
    }
```

b) *Write a pseudo code/function to reverse only even position nodes value in a Singly Linked List.* 2

Ans:

Assume that first node is the even position node;

```
rev_even(){
        struct node *ptr1, *ptr2, *prv;
        if(head==NULL)return;
        for(ptr2=head;  ptr2->next!=NULL  &&  ptr2->next->next!=NULL  ;ptr2=ptr2-
            >next->next);
        for(ptr1=head1 ; (ptr1!=ptr2)&&(ptr2->next==NULL || ptr2->next->next!=ptr1) ;
            ptr1=ptr1->next->next){
                int tmp=ptr1->val;
                ptr1->val=ptr2->val;
                ptr2->val=tmp;
                for(prv=ptr1;prv->next->next!=ptr2;prv=prv->next->next);
                ptr2=prv;
        }
}
```

3.

a) *Let a QUEUE ADT is implemented using an array having front and rear index. The criteria for insertion and deletion of the queue are as follows:* 4

- *Element should be less than the size of the array*
- *If an element is **m** then the same **m** value will be added to **m** blocks of the array.*
- *If the front data element in the queue is **m** then all the consecutive **m** blocks must be deleted.*

Write the insertion and deletion operation of the above mentioned QUEUE ADT with necessary overflow and underflow conditions.

Ans:

```
typedef struct{
        int a[MAX];
        int F,R;
    }QUEUE;
    QUEUE q1;
    void insertion(int val){
        if(val>=MAX) return;
        if(q1.R+val>=MAX){
            printf(“CANNOT insert”); return;
        }else{
            for(i=val;i>0;i--)
                q1.a[++q1.R]=val;
        }
    }
    void deletion(){
        if(q1.F==-1){
            printf(“CANNOT delete”); return;
        }else{
            int val=q1.a[q1.F];
            for(i=val;i<=q1.R;i++)
                q1.a[i-val]=q1.a[i];
            q1.R -=val;
        }
    }
```

b) *What do you mean by dynamic memory allocation? How to allocate memory for a 2D array dynamically.* 1

Ans:

Dynamic memory allocation definition(1 mark)

```
int **b=(int **)malloc(sizeof(int *) * 3);
b[0]=(int *)malloc(sizeof(int) * 4);
b[1]=(int *)malloc(sizeof(int) * 4);
b[2]=(int *)malloc(sizeof(int) * 4);
```

4.

a) *Write an algorithm to convert infix expression to its equivalent postfix expression. Translate infix expression: (A-(B+C)/F)^D+G/E into its equivalent postfix expression using STACK.* 3

Ans:

Infix to postfix algorithm(2 marks)

post fix using stack step by step(1 mark)

ABC+F/-D^GE/+

b) *Write a pseudo code/function to replace every element in the array with the next greatest element presented in the same array.* 2

Ans:

```
void replace(int a[], int size){
    for(i=0;i<size-1;i++){
        flag=0;
        for(j=i+1;j<size;j++)
            if(a[j]>a[i]){
                flag=1; break;
            }
```

```
                        if(flag)
                                a[i]=a[j];
                }
        }
```

5. ***a)*** *Write an algorithm to multiply two polynomials.* 3

Ans:

polynomial multiplication algorithm (3 marks)

b) *Write a function to identify all the nodes which are keeping the address of any one of its previous nodes in a single linked list and find out the corresponding previous node.* 2

Ans:

```
struct node *head;
identify_loop(){
        struct node * ptr=head, *ptr1;
        while(ptr!=NULL){
                ptr1=head;
                while(ptr1!=ptr){
                        if(ptr->next==ptr1)
                                break;
                        else
                                ptr1=ptr1->nex;
                }
                if(ptr->next==ptr1){
                        printf("FOUND");break;
                }else
                        ptr=ptr->nex;
        }
}
```

6. ***a)*** *Suppose a current STACK contains **n** number of elements which are stored in ascending order. Write a function to insert an element into the STACK using its PUSH and POP operation, so that the final STACK will also be sorted in same order.* 3

Ans:

```
STACK s1,s2;// Lets assume that s1 stack contains n number of elements
void copy(){
            int no;
            if(s1.top == -1) return;
            no=pop(s1);
            copy();
            push(s2,no);
}
```

b) *Write a function to delete all the nodes in a doubly linked list, where the data element of the node is greater than data element of all its previous nodes and is less than data element of all the next nodes.* 2

Ans:

```
struct node *head;
del(){
        struct node *ptr=head;
        while(ptr!=NULL){
                flag1=flag2=1;
                for(ptr1=ptr->prv;ptr1!=NULL;ptr1=ptr1->prv)
                    if(ptr1->data>ptr->data){
                        flag1=0; break;
                    }
                for(ptr1=ptr->next;ptr1!=NULL;ptr1=ptr1->next)
                    if(ptr1->data<ptr->data){
                        flag2=0; break;
                    }
                if(flag1 && flag2){
                    if(head==ptr){
                        head=head->next;
                        free(ptr);
                        ptr=head;
                        ptr->prv=NULL;
                    }else if(ptr->next==NULL){
                        ptr->prv->next=NULL;
                        free(ptr);
                        ptr=NULL;
                    }else{
                        ptr=ptr->next;
                        ptr->prv=ptr->prv->prv;
                        free(ptr->prv->next);
                        ptr->prv->next=ptr;
                    }
                }
        }
}
```

7.

a) *Let a 2-dimensional matrix of size* ***m X n*** *contains elements either 0 or 1. Write a function to replace an element 0 with 1, if all its surrounding elements are 1.* 3

Ans:

```
void fill_one(int a[][], int m, int n){
        while(1){
        flag=0;
        for(i=1;i<m-1;i++)
                for(j=1;j<n-1;j++)
                if(a[i][j]==0 && a[i-1][j]==1 && a[i+1][j]==1 && a[i][j-1]==1 && a[i]
                    [j+1]==1){
                        a[i][j]=1;
                        flag=1;
                }
        if(flag==0) break;
        }
}
```

2

b) *Compare and contrast LIST, STACK, and QUEUE ADT along with their applications.*

Ans:

Compare 3 ADTs.