Sample Question Format

KIIT Deemed to be University
Online Mid Semester Examination(Spring Semester-2021)

Subject Name & Code: **Data Structure and Algorithm & CS-2001**

Applicable to Courses: **B.Tech, Sem-3rd (Regular)**

Full Marks=20 **Time:1 Hour**

SECTION-A(Answer All Questions. All questions carry 2 Marks)

Time:20 Minutes **(5×2=10 Marks)**

<table>
<tr><th>Question No</th><th>Question Type(MCQ/SAT)</th><th>Question</th><th>Answer Key(if MCQ)</th><th>CO Mapping</th></tr>
<tr><td>Q.No:1(a)</td><td></td><td>The following codes create a two dimensional matrix dynamically.
<table>
<tr><td>Code-A</td><td>Code B</td></tr>
<tr><td>int *a[4];
a[0]=malloc(12 * sizeof(int));
for(int i=0;i<3;i++)
a[i]=a[0]+(i * 4);</td><td>int *a[4];
for(int i=0;i<3;i++)
a[i]=malloc(4 * sizeof(int));</td></tr>
</table>
Which of the following statement will work properly in both of these above codes?
I) a[1][2]=7;
II) a[2]=a[1];
III) *(a[1]+3)=7;
IV) *(a[0]+7)=7;
A) I, II, III
B) I, II, III, IV
C) III, IV
D) I, III, IV</td><td>A</td><td>CO1</td></tr>
<tr><td></td><td></td><td>An array A consists of n integers in locations A[0], A[1] ....A[n-1]. It is required to shift the elements of the array cyclically to the left by k places, where 1 <= k <= (n-1). An incomplete algorithm for doing this in linear time, without using another array is given below. Complete the algorithm by filling in the blanks. Assume all the variables are suitably declared.
min = n; i = 0;
while (_____) {
temp = A[i]; j = i;
while (_____) {
A[j] = _____
j= (j + k) mod n ;</td><td>B</td><td>CO1,CO4</td></tr>
</table>

		If (j< min) then
		min =_____ ;
		}
		A[(n + i — k) mod n] =temp
		i = _____
		
		(A) i > min; j!= (n+i+k)mod n; A[(j + k)mod n]; temp; i + 1 ;
		(B) i < min; j!= (n+i-k)mod n; A[(j + k)mod n]; j; i + 1;
		(C) i > min; j!= (n+i+k)mod n; A[(j + k)mod n]; j; i + 1;
		(D) i < min; j!= (n+i-k)mod n; A[(j + k)mod n]; temp; i + 1;		

		Which of the following operations does not take O(1) time for an array of unsorted elements. Assume that array elements are distinct. (A) Find the largest element (B) Delete an element (C) Find the smallest element (D) All of the above	D	CO1,CO2
		A two dimensional array in C is initialized as int A [3][4]. What does *(*(A+3) +2), indicate? (A) A[0][3] (B) A[1][2] (C) A[3][2] (D) Compilation Error	C, D	CO1

Q.No:1(b)		What is the time complexity of following code:
		void function(int n) {
		int i, j, k = 0;
		for (i = n; i <= n/2; i--) {
		for (j = n; j <= 2; j = j / 2) {
		k = k + n / 2;
		}
		}}
		(A) O(n)
		(B) O(nLogn)
		(C) O(n^2)
		(D) O(n^2Logn)
			All	CO2

		What is the time complexity of following code:
		void fun(int n) {
		int a, b;
		for(a=1; a<=n; a=2*a)
		for(b=n/2; b<=n; b++)
		printf("%d%d", a, b);
		}
		(A) O (n $\log_2$ n)
		(B) O ($\log_2$ n)
		(C) O (n^2)
		(D) O (n)
			A	CO2

		What is the time complexity of following code:	B	CO2

		void function(int n) {
		int i, j, k;
		for(i=n/2; i<=n; i++)
		for(j=1; j + n/2<=n; j= j++)
		for(k=1; k<=n; k= k * 2)
		count++;
		}
		(A) O (n $\log_2$ n)
		(B) O ($n^2\log_2$ n)
		(C) O (n^2)
		(D) O (n)		

		What is the time complexity of following code :
		void function(int n) {
		int i, j, k, p, q = 0;
		for (i = 1; i<n; ++i) {
		p = 0;
		for (j=n; j>1; j=j/2)
		++p;
		for (k=1; k<p; k=k*2)
		++q;
		}
		(A) O (n $\log_2$ n)
		(B) O ($n^2\log_2$ n)
		(C) O (n^2)
		(D) O (n)
			A	CO2

Q.No:1(c)		In a two dimensional matrix A[0...19, 0…34] is stored in the memory with each element requiring 4 bytes of storage. If the address of A[0][0] is 2148 and the location of A[k, j] is same in both row-major-order and column-major-order. Find the value of k/j? (A) 34/19 (B) 2148/2228 (C) 19/34 (D) 2228/2148	A	CO1,CO4
		In a two dimensional matrix A[0...64, 0…92] is stored in the memory with each element requiring 4 bytes of storage. If the address of A[0][0] is 1000 and the location of A[k, j] is same in both row-major-order and column-major-order. Find the value of k/j? (A) 64/92 (B) 1000/2508 (C) 2508/1000 (D) 92/64	D	CO1,CO4
		In a two dimensional matrix A[0...,29, 0..,49] is stored in the memory with each element requiring 4 bytes of storage. If the address of A[0][0] is 1756 and the location of A[k, j] is same in both row-major-order and column-major-order. Find the value of k/j? (A) 1756/2234 (B) 49/29	B	CO1,CO4

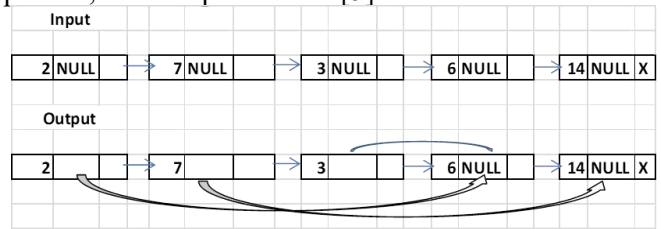
		(C) 29/49 (D) 2234/1756		
		In a two dimensional matrix A[0...54, 0…17] is stored in the memory with each element requiring 4 bytes of storage. If the address of A[0][0] is 7600 and the location of A[k, j] is same in both row-major-order and column-major-order. Find the value of k/j? (A) 7600/8540 (B) 17/54 (C) 54/17 (D) 8540/7600	B	CO1,CO4
Q.No:1(d)		Evaluate the following postfix expression using stack and indicate the content of the stack when the red marked '*' (2nd multiplication from left to right sequence)is encountered: 5 3 2 ∗ − 18 9 / 4 ∗ 2 / − 6 + 2 – (A)12, 2, 2 (B) -1, -2 4 (C) -1, 2, 4 (D) 1,-2,2	C	CO4
		Evaluate the following postfix expression using stack and indicate the content of the stack when the operand '+' is encountered: 3 5 10 - 2 + / 5 3 * / (A) 3, 5, 10 (B) 3, -5, 2 (C) 3, 5, 15 (D) -5, -2-, 3	B	CO4
		Convert the following Infix expression using stack and indicate the content of the stack when the operand '5' is encountered: ((6 + 8) *9 – (5- 4) ^ (2 + 7)) (A) (- * (- (B) (- ((C) (-(- (D) (*(-	B	CO4
		Convert the following Infix expression to Postfix using stack and indicate the content of the stack when the operand '5' is encountered: (1-(2+3)/4)^5+6/7) (A) (-(^ (B) (- (^ (C) (^ (D) ((-/^	All	CO4
Q.No:1(e)		What is the time complexity of the EnQueue and DeQueue operation in a Linear queue? (A) O (1) and O(1) (B) O (n) and) O (n) (C) O (1) and O (n) (D) O (n) and) O (1)	A	CO2
		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	A	CO4

		(C) Front=rear+1 (D) Rear=front		
		What is the worst case time complexity for a consecutive n EnQueue operation in a linear Queue? (A) O (n $\log_2$ n) (B) O ($n^2\log_2$ n) (C) O (n^2) (D) O (n)	D	CO2
		What is the worst case time complexity for a consecutive n DeQueue operation in a linear Queue? (A) O (n $\log_2$ n) (B) O (n) (C) O ($\log_2$ n) (D) O (n^2)	B	CO2

SECTION-B(Answer Any One Question. Each Question carries 10 Marks)

Time: 30 Minutes **(1×10=10 Marks)**

Question No	Question	CO Mapping

Q.No:2	a) Differentiate between array and linked list. An array has n positive integers. Write a function of O (n) order for removing all the odd numbers from the array. Example, the array contains 10, 2, 3, 7, 8, 6, and 11. The output should be 10, 2, 8, and 6. [5]
	Ans: Differentiate between array and linked list. [0.5 mark] , 4.5 marks will be given for the correct answer and partial marks to be awarded depending on the correctness of the steps.
	void odd-remove(int A[], int n) {
	int i, k=0;
	for (i=0, i< n; i++) {
	if (A[i] %2 == 0) {
	A[k]=A[i];
	k++;
	}
	}
	for (i=0, i< k; i++)
	printf ("%d", A[i]);
	}
	
	b) Write the difference between Array and Linked List. Given singly linked list with every node having an additional pointer named as 'multiply' that currently points to NULL. Need to make the "Multiply" pointer point to the next multiplied value of the current node. If multiplied value is not present, then keep it NULL. [5]
	 
	
		CO1,CO3,CO4

	Ans: Differentiate between array and linked list. [0.5 mark] , 4.5 marks will be given for the correct answer and partial marks to be awarded depending on the correctness of the steps.
	
	typedef struct node{
	int data;
	struct node *next;
	struct node *multiply;
	} Node;
	Node * start=NULL;
	
	void MUL-Link () {
	Node *temp2, * temp1=start;
	while (temp1 !=NULL) {
	temp2=temp1;
	while (temp2 !=NULL) {
	if (temp2->data % temp1->data==0) {
	temp1->multiply=temp2;
	break;
	}
	else
	temp1->multiply=NULL;
	temp2=temp2->next;
	}
	temp1=temp1->next;
	}
	}	

	a) How do we represent a polynomial expression using single linked list? Write a pseudo code to add two polynomial having two numbers of unknown variables. [5]
	Ans: Polynomial representation using single linked list [1 mark]. Addition is of 4 marks. Step marks will be given depending on the correctness of the steps.
	
	struct node { int coff, exp1, exp2; struct node *next;};
	
	void add(struct node *poly1, struct node *poly2) {
	struct node *p1, *p2, *prev, *q;
	if (poly1== NULL) //join two linked list
	poly1=poly2;
	else {
	q = poly1;
	while(q→next != NULL)
	q = q→next;
	q→next = poly2;
	}
	for(p1=poly1; p1!=NULL; p1=p1→next) {
	// duplicate remove
	prev = p1;
	for(p2 = p1→ next; p2 != NULL;)
	if(p1→exp1==p2→exp1 &&
	p1→exp2==p2→exp2) {
	p1→coff = p1→coff + p2→coff
	prev→next = p2→next;
		CO1,CO3,CO4

```
                free(p2);
                p2 = prev → next;
            }
            else {
                prev=p2;
                p2 = p2 → next;
            }
    }
}
```

b) Let a linked list consists of n number of nodes, where each node consists of an unique character represents the grades of the students (O, E, A, B, C), and pointer to next node. Write pseudo code/ C code to group the students having same grade in consecutive place and also finally all the nodes should be in sorting order as per their grade value. (O>E>A>B>C) [5]

Ans: List grouping: 3.5 marks, Sorting ascending order: 1.5 marks

```
#include<stdio.h>
#include<stdlib.h>
struct node {
    char ch;
    int imp;
    struct node *next;
};
struct node *head = NULL;
struct node *p1 = NULL;

void swap(struct node *a, struct node *b) {
    char temp = a->ch;
    a->ch = b->ch;
    b->ch = temp;
}

void sort(struct node *start) {
    int swapped, i;
    struct node *ptr1;
    struct node *lptr = NULL;
    if(start == NULL) return;
    do {
        swapped = 0;
        ptr1 = start;
        while(ptr1->next != lptr) {
            if(ptr1->imp > ptr1->next->imp) {
                swap(ptr1, ptr1->next);
                swapped = 1;
            }
            ptr1 = ptr1-next;
        }
        lptr = ptr1;
    } while(swapped);
}

void addnode(char ch) {
    struct node *p1;
```

```
    struct node *nd=(struct node*) malloc(sizeof(struct node));
    nd->ch = ch;
    nd->next = NULL;
    if(head == NULL) head = nd;
    else {
        for(p1=head; p1->next != NULL; p1=p1->next);
        p1->next = nd;
    }
}

void main() {
    char ch, from, to;
    printf("\nEnter the string: ");
    while((ch=getchar()) != '\n')
        addnode(ch);
    struct node *tmp = head;
    while(tmp != NULL) {
        if(tmp->ch == 'O') tmp->imp=5;
        else if(tmp->ch == 'E') tmp->imp=4;
        else if(tmp->ch == 'A') tmp->imp=3;
        else if(tmp->ch == 'B') tmp->imp=2;
        else if(tmp->ch == 'C') tmp->imp=1;
        else if(tmp->ch == 'D') tmp->imp=0;
        else if(tmp->ch == 'F') tmp->imp=-1;
        tmp=tmp->next;
    }
    sort(head);
    for(p1=head; p1!=NULL; p1=p1->next)
        printf("%c", p1-ch);
}
```

a) Write a C program to add two triplets and print the result in another triplet format using the array.[5]

Ans: Triplet node structure: 1 marks, Addition: 4 marks

CO1,CO3,CO4

```
int ** add_sparse (int **M1, int **M2) {
    if ((M1[0][0]!=M2[0][0]) || (M1[0][1]!=M2[0][1]))
        return;
    int i=1, j=1, k=1, t1 = M1[0][2], t2= M2[0][2] ;
    int **M3;
    M3=(int **) malloc ((t1+t2 +1) * sizeof(int *));
    for (i=0; i<m+n+1; i++)
        M[3]= (int *) malloc (3 * sizeof(int *));
    M3[0][0] = M1[0][0];
    M3[0][1] = M1 [0][1];
    while( i<=t1 && j<=t2) {
        if( M1[i][0]==M2[j][0] && M1[i][1]==M2[j][1]) {
            M3[k][0]= M1[i][0];
            M3[k][1] = M1[[i][1];
            M3[k][2] = M1[[i][2] + M2[j][2];
            i++; j++; k++;
        }
        else if( M1[i][0]==M2[j][0] && M1[i][1]<M2[j][1]) {
            M3[k][0]= M1[i][0];
            M3[k][1] = M1[[i][1];
            M3[k][2] = M1[[i][2];
```

```
            i++; k++;
        }
        else if( M1[i][0]==M2[j][0] && M1[i][1] >M2[j][1]) {
            M3[k][0]= M2[j][0];
            M3[k][1] = M2[j][1];
            M3[k][2] = M2[j][2];
            j++; k++;
        }
        else if( M1[i][0] < M2[j][0]) {
            M3[k][0]= M1[i][0];
            M3[k][1] = M1[[i][1];
            M3[k][2] = M1[[i][2];
            i++; k++;
        }
        else {
            M3[k][0]= M2[j][0];
            M3[k][1] = M2[j][1];
            M3[k][2] = M2[j][2];
            j++; k++;
        }
    }
    while ( i<= t1) {
        M3[k][0]= M1[i][0];
        M3[k][1] = M1[[i][1];
        M3[k][2] = M1[[i][2];
        i++; k++;
    }
    while ( j<= t2) {
        M3[k][0]= M2[j][0];
        M3[k][1] = M2[j][1];
        M3[k][2] = M2[j][2];
        j++; k++;
    }
    M3[0][2] =k-1;
    return (M3);
}
```

b) Write a function in C or Pseudo code: DeleteFromEnd() in a header linked list ,where one Node structure to store an integer value and the special designated node (i.e. header node) contains three information: number of nodes in the list and the maximum, minimum among the list of values. These values must be updated, if required, in every function call. At the beginning define the structure of both the nodes. [5]

Ans: Step marks will be given depending on the correctness of the steps.

```
typedef struct node{
      int data;
      struct node *next;
} Node;

struct hNode {
      int count;
      int max;
      int min;
      Node *next;
```

	}*hnode;
	
	hnode=(struct hNode*) malloc(sizeof(struct hNode));
	hnode->count = 0;
	hnode->max = INT_MIN;
	hnode->min = INT_MAX;
	hnode->next = NULL;
	
	Node *node, *last;
	
	void deleteFromEnd(struct hNode *head) {
	if(last->data == head->min) {
	int newMin = INT_MAX;
	Node *start = head->next;
	while(start->next->next!=NULL) {
	start = start->next;
	newMin = newMin < start->data? newMin: start->data;
	}
	head->min = newMin;
	temp = start->next;
	start->next = NULL;
	last = start;
	}
	else if(last->data == head->max) {
	int newMax = INT_MIN;
	Node *start = head->next;
	while(start->next->next!=NULL) {
	start = start->next;
	newMax = newMax > start->data? newMax: start->data;
	}
	head->max = newMax;
	temp = start->next;
	start->next = NULL;
	last = start;
	}
	else {
	struct hNode *curr = head;
	Node *start = curr->next;
	while(start->next->next!=NULL)
	start=start->next;
	temp = start->next;
	start->next = NULL;
	last = start;
	}
	head->count--;
	free(temp);
	}	

Q.No:3	a) Let 'm' number of stacks are implemented in one array where m_i is the size of each i^{th} stack . Write a pseudo code /function for the push() and pop() operations on i^{th} stack. [5] Ans: Push operation: 2.5 marks, Pop Operation: 2.5 Marks /*the array size[m] holds the value of size of each stack size[0] will hold the value of size of first stack*/	CO1,CO4

```
/*the array top[m] holds the value of index of each stack
topmost element top[0] will hold the value of index
position of topmost element of first stack*/

int top[m], size[m];
void push(int stack[], int i, int item) {
    int max_upperbound=-1;
    for(int k=0; k<i; k++)
        max_upperbound=max_upperbound+size[k];
    if(top[i-1]==max_upperbound) {
        printf("\nOverflow condition");
        return;
    }
    top[i-1]++;
    stack[top[i-1]]=item;
}

void pop(int stack[],int i) {
    int min_lowerbound=-1;
    for(int k=0; k<i-1; k++)
    min_lowerbound=min_lowerbound+size[k];
    if(top[i-1]==min_lowerbound) {
        printf("\nUnderflow condition");
        return;
    }
    printf("\nDeleted element is %d", stack[top[i-1]]);
    top[i-1]--;
}

void display(int stack[], int i) {
    int j;
    int min_lowerbound=-1;
    for(int k=0; k<i-1; k++)
        min_lowerbound=min_lowerbound+size[k];
    if(top[i-1]==min_lowerbound) {
        printf("\n stack is empty");
        return;
    }
    for(j=top[i-1]; j>min_lowerbound; j--)
        printf("\n %d", stack[j]);
}
```

b) Write a function/Pseudo code to swap the following nodes in a circular single Linked List with minimum number of pointers and having only one pointer head/start to indicate first node address.

i. 1st node with 2nd node

ii. Last node with its previous node

(Note: Swap the node/structure node) [5]

Ans: 1st node with 2nd node- 2.5 Marks, Last node with its previous node- 2.5 marks

i) <u>1st node with 2nd node:</u>

head pointer is pointing to the 1st node.

Initially temp pointer is pointing to the 1st node.

```
void swap() {
```

	struct node *temp=head;
	temp = temp->next;
	head-> next = temp-> next;
	temp-> next = head;
	head=temp;
	while(temp->next != head)
	temp= temp->next;
	temp->next = head;
	}
	
	ii) Last node with its previous node:
	head pointer is pointing to the 1st node.
	Initially temp pointer is pointing to the 1st node.
	void swap() {
	struct node *temp=head;
	while(temp->next != head)
	temp= temp->next;
	temp->next = head->next;
	head->next->next->next = head;
	head->next= temp;
	head = temp;
	}	

	a) WAP to find maximum element of stack at a particular instant when any number of push and pop operations are allowed using linked list such that each top node will contain the maximum element from all elements below to it. [5]
	Ans: Push operation: 2.5 Marks, Pop Operation 2.5 Marks
	struct node {
	int info;
	int current_max ;
	struct node *next;
	}*top=NULL;
	
	void push(int item) {
	struct node *temp;
	temp=(struct node*)malloc(sizeof(struct node));
	if(!temp) {
	printf("\n Heap is full :Overflow condition");
	return;
	}
	temp->info=item;
	if(top==NULL)
	temp->current_max=temp->info;
	else if(top->current_max < temp->info)
	temp->current_max=temp->info;
	else
	temp->current_max=top->current_max;
	temp->next=top;
	top=temp;
	}
	
	void pop() {
	struct node *temp;
	if(top==NULL){
	printf("\nstack underflow");
		CO1,CO4

```
        return;
    }
    temp=top;
    top=top->next;
    temp->next=NULL;
    free(temp);
}

void max_elem() {
    if(top==NULL)
        printf("stack is empty");
    else
        printf("Max element is %d", top->current_max);
}
```

b) Design pseudo code/function to add a given value K to each element in the double linked list and if it becomes greater than M, then convert it to 0…M-1 by doing modulo operation with M.(K<M). Then if the element in the current node is equal to any other node previous to this, delete the current node. [5]

Ans: Adding K and converting it into M: 2.5 Marks, Deleting Node: 2.5 Marks

```
void func1(struct node *start, int K, int M) {
    struct node *temp=start;
    while(temp!= NULL) {
        temp->info = temp->info + K;
        if (temp->info > M) {
            temp->info = (temp->info) % M;
            temp= temp ->next;
        }
        else
            temp= temp->next;
    }
}

void delNode(struct node *start) {
    struct node *tmp, *temp1, *temp2;
    temp1 = start;
    while (temp1 != NULL) {
        temp1 = temp1->next;
        if (temp1 == NULL) temp2 = NULL;
        else {
            temp2 = temp1->prev;
            tmp = temp1;
        }
        while (temp2 != NULL)          {
            if (temp1->info == temp2->info) {
                tmp->prev->next = tmp->next;
                if (tmp->next != NULL)
                    tmp->next->prev = tmp->prev;
                else
                    temp1 = NULL;
                free(tmp);
                break;
            }
```

	temp2 = temp2->prev;
	}
	}
	}	

	a) Using basic stack push() and pop() operation implement the insert() function as follows.
	Insert(): Insert function will insert the new element if the element doesn't exist and the insertion will happen using push() such that after insertion the stack elements will be in a sorted manner. [5]
	Ans: Push operation with Sorting: 5 Marks
	int top;
	void insert(int ele) {
	if((top==-1) || stack[top] <ele) {
	push(ele)
	return;
	}
	else if(stack[top] == ele)
	return;
	else {
	x= pop();
	insert(ele);
	push(x);
	}
	}
	
	void push (int x, int S[], int n) {
	if (top == n-1)
	return;
	else
	top++;
	S[top] = x;
	}
	
	int pop (int S[], int n) {
	if (top == -1)
	return;
	else
	x = s[top];
	top--;
	return x;
	}
	
	b) Write a function to delete all prime numbers present in a doubly linked list. For example, if input: 5->6->11->4->12->16, then output 6->4->12->16. [5]
	Ans: Full marks will be given for the correct answer and partial marks to be awarded depending on the correctness of the steps.
	struct node {
	int info;
	struct node * next;
	struct node *prev;
	}*start=NULL;
	
	void Del-prime() {
		CO1,CO4

```
    struct node * temp=start. *t1;
    while( temp ! = NULL) {
        if( isPrime (temp->info) )
            temp=temp->next;
        else{
            t1=temp->next;
            temp->prev->next=temp->next;
            temp->next->prev=temp->prev;
            free(temp);
            temp=t1;
        }
    }
}

int isPrime (int n) {
    int flag=0, i;
    for(i=2; i<= sqrt(n); i++) {
        if(n%i == 0) {
            flag=1;
            break;
        }
    }
    if (flag==0) return 1;
    else return 0;
}
```