

MODULE 01

1. Design, Develop and Implement a menu driven Program in C for the following Array Operations

- a. Creating an Array of N Integer Elements
- b. Display of Array Elements with Suitable Headings
- c. Exit.

Support the program with functions for each of the above operations.

```
#include<stdio.h>
#include<stdlib.h>

int a[20];
int n = 0;

void create()
{
    printf("Enter the size of the array: ");
    scanf("%d", &n);
    printf("Enter the elements for the array:\n");

    for(int i=0; i<n; i++)
    {
        scanf("%d",&a[i]);
    }
}

void display()
{
    printf("The array elements are:\n");
    for(int i=0; i<n; i++)
    {
        printf("%d\n ",a[i]);
    }
}

int a[20],n, elem, pos, value;

void create();
void display();

void main()
{
```

```
int choice;
while(1)
{
    printf("A program to perform array operation\n");
    printf("1. Create\n");
    printf("2. Display\n");
    printf("3. Exit\n");

    printf("Enter your choice: \n");
    scanf("%d", &choice);

    switch(choice)
    {
        case 1: create();
        break;

        case 2: display();
        break;

        case 3: exit (0);

        default: printf("Invalid choice\n");
    }
}
}
```

OUTPUT:

```

(array)
A program to perform array operation
1. Create
2. Display
3. Exit
Enter your choice:
1
Enter the size of the array: 4
Enter the elements for the array:
3
33
44
55
A program to perform array operation
1. Create
2. Display
3. Exit
Enter your choice:
2
The array elements are:
3
33
44
55
A program to perform array operation
1. Create
2. Display
3. Exit
Enter your choice:
3
PS D:\DSA C\DSA>

```

2. Design, Develop and Implement a menu driven Program in C for the following Array operations

- a. Inserting an Element (ELEM) at a given valid Position (POS)
- b. Deleting an Element at a given valid Position POS)
- c. Display of Array Elements
- d. Exit.

Support the program with functions for each of the above operations.

```

#include <stdio.h>
#include <stdlib.h>

int a[20];
int n = 0;

void create()
{
    printf("Enter the size of the array: ");
    scanf("%d", &n);
    printf("Enter the elements for the array:\n");
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &a[i]);
    }
}

```

```
}

void display()
{
    printf("The array elements are:\n");
    for (int i = 0; i < n; i++)
    {
        printf("%d\n ", a[i]);
    }
}

void insert()
{
    int pos, value;
    printf("Enter the index position for the new element: ");
    scanf("%d", &pos);
    printf("Enter the element to be inserted : ");
    scanf("%d", &value);
    for (int i = n - 1; i >= pos; i--)
    {
        a[i + 1] = a[i];
    }
    a[pos] = value;
    n = n + 1;
}

void delete()
{
    int pos, value;
    printf("Enter the index position of the element to be deleted: ");
    scanf("%d", &pos);
    value = a[pos];
    for (int i = pos + 1; i < n; i++)
    {
        a[i - 1] = a[i];
    }
    n = n - 1;
    printf("The deleted element is = %d\n", value);
}

int a[20], n, elem, pos, value;
void create();
void display();
void insert();
void delete();

void main()
```

```
{
    int choice;
    while (1)
    {
        printf("A program to perform array operation\n");
        printf("1. Create\n");
        printf("2. Display\n");
        printf("3. Insert\n");
        printf("4. Delete\n");
        printf("5. Exit\n");
        printf("Enter your choice: \n");
        scanf("%d", &choice);

        switch (choice)
        {
            case 1:
                create();
                break;

            case 2:
                display();
                break;

            case 3:
                insert();
                break;

            case 4:
                delete ();
                break;

            default:
                printf("Invalid choice\n");
                exit(0);
        }
    }
}
```

OUTPUT:

```
A program to perform array operation
1. Create
2. Display
3. Insert
4. Delete
5. Exit
Enter your choice:
1
Enter the size of the array: 3
Enter the elements for the array:
66 77 88
A program to perform array operation
1. Create
2. Display
3. Insert
4. Delete
5. Exit
Enter your choice:
2
The array elements are:
66
77
88
A program to perform array operation
1. Create
2. Display
3. Insert
4. Delete
5. Exit
Enter your choice:
3
Enter the index position for the new element: 3
Enter the element to be inserted : 55
A program to perform array operation
1. Create
2. Display
3. Insert
4. Delete
5. Exit
Enter your choice:
2
The array elements are:
66
```

```
The array elements are:
66
77
88
55
A program to perform array operation
1. Create
2. Display
3. Insert
4. Delete
5. Exit
Enter your choice:
4
Enter the index position of the element to be deleted: 2
The deleted element is = 88
A program to perform array operation
1. Create
2. Display
3. Insert
4. Delete
5. Exit
Enter your choice:
5
Invalid choice
PS D:\DSA C> █
```

MODULE 02:

1. Design, Develop and Implement a menu driven Program in C for the following operations on

STACK of Integers (Array Implementation of Stack with maximum size MAX)

- a. Push an Element on to Stack
- b. Pop an Element from Stack
- c. Demonstrate Overflow and Underflow situations on Stack
- d. Display the status of Stack
- e. Exit

Support the program with appropriate functions for each of the above operations

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define max_size 5
```

```
int stack[max_size], top = -1;

void push()
{
    int item;
    if (top == max_size - 1)
    {
        printf("Stack Overflow\n");
    }
    else
    {
        printf("Enter the element to be inserted\n");
        scanf("%d", &item);

        top = top + 1;
        stack[top] = item;
    }
}

void pop()
{
    int item;
    if (top == -1)
    {
        printf("Stack Underflow\n");
    }
    else
    {
        printf("The popped element : %d\t", stack[top--]);
    }
}

void display()
{
    if (top == -1)
    {
        printf("Stack is empty\n");
    }
    else
    {
        printf("The stack elements are :\n\n");
        for (int i = 0; i <= top; i++)
        {
            printf("%d\n", stack[i]);
        }
    }
    printf("\n\n");
}
```



```
void main()
{
    int choice;
    while (choice)
    {
        printf("\n\n-----STACK OPERATIONS-----\n");
        printf("1. push\n 2. Pop\n 3.Display\n 4. Exit\n");
        printf("Enter your choice:\n");
        scanf("%d", &choice);

        switch (choice)
        {
            case 1:
                push();
                break;
            case 2:
                pop();
                break;
            case 3:
                display();
                break;
            case 4:
                exit(0);
                break;

            default:
                printf("Invalid choice\n");
                break;
        }
    }
}
```

OUTPUT:

```
-----STACK OPERATIONS-----
1. push
2. Pop
3.Display
4. Exit
Enter your choice:
1
Enter the element to be inserted
23

-----STACK OPERATIONS-----
1. push
2. Pop
3.Display
4. Exit
Enter your choice:
3
The stack elements are :
23

-----STACK OPERATIONS-----
1. push
2. Pop
3.Display
4. Exit
Enter your choice:
2
The popped element : 23

-----STACK OPERATIONS-----
1. push
2. Pop
3.Display
4. Exit
Enter your choice:
4
PS D:\DSA C> |
```

2. Design, Develop and Implement a Program in C for the following Stack Applications
- Evaluation of Suffix expression with single digit operands and operators: +, -, *, /, %, ^
 - Solving Tower of Hanoi problem with n disks

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>
#define MAX 50

int stack[MAX];
char post[MAX];
int top = -1;

/*FUNCTION PROTOYPE */
void pushstack(int tmp);
void calculator(char c);

void main()
{
    int i;
    printf("Insert a postfix notation :: ");
    scanf("%s", post);
    for (i = 0; i < strlen(post); i++)
    {
        if (post[i] >= '0' && post[i] <= '9')
        {
            pushstack(i);
        }
        if (post[i] == '+' || post[i] == '-' || post[i] == '*' || post[i] ==
        '/' || post[i] == '^')
        {
            calculator(post[i]);
        }
    }
    printf("\nResult :: %d", stack[top]);
}

void pushstack(int tmp)
{
    stack[++top] = (int)(post[tmp] - 48);
}

void calculator(char c)
{
    int a, b, ans;
    a = stack[top--];
    b = stack[top--];

    switch (c)
    {
```

```
    case '+':
        ans = b + a;
        break;
    case '-':
        ans = b - a;
        break;
    case '*':
        ans = b * a;
        break;
    case '/':
        ans = b / a;
        break;
    case '^':
        ans = pow(b, a);
        break;
    default:
        ans = 0;
    }

    top++;
    stack[top] = ans;
}
```

OUTPUT:

```
Insert a postfix notation :: 456*+
Result :: 34
PS D:\DSA C> cd "d:\DSA C\" ; if ($?)
Insert a postfix notation :: 37+
Result :: 10
PS D:\DSA C> █
```

b. Tower of Hanoi

```
#include <stdio.h>
#include <math.h>

void tower(int n, int source, int temp, int destination)
{
    if (n == 0)
        return;
    tower(n - 1, source, destination, temp);
    printf("\nMove disc %d from %c to %c", n, source, destination);
    tower(n - 1, temp, source, destination);
}

void main()
{
    int n;
    printf("\nEnter the number of discs: \n");
    scanf("%d", &n);
    tower(n, 'A', 'B', 'C');
    printf("\n\nTotal Number of moves are: %d", (int)pow(2, n) - 1);
}
```

OUTPUT:

```
Enter the number of discs:
3

Move disc 1 from A to C
Move disc 2 from A to B
Move disc 1 from C to B
Move disc 3 from A to C
Move disc 1 from B to A
Move disc 2 from B to C
Move disc 1 from A to C

Total Number of moves are: 7
```

MODULE 03:

1. Singly Linked List (SLL) of Integer Data

- a. Create a SLL stack of N integer.
- b. Display of SLL
- c. Linear search. Create a SLL queue of N Students Data Concatenation of two SLL of integers.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
struct node
{
    int info;
    struct node *link;
};
typedef struct node * NODE;
struct nodes
{
    char usn[100],name[100],branch[100];
    struct nodes *next;
};
typedef struct nodes * NODES;

NODES insertRear(NODES first)
{
    NODES temp,cur;
    char usn[100],branch[100],name[100];
    temp=(NODES)malloc(sizeof(struct nodes));
    printf("Enter student's name,usn,branch respectively\n");
    scanf("%s%s%s",name,usn,branch);
    strcpy(temp->name,name);
    strcpy(temp->usn,usn);
    strcpy(temp->branch,branch);
    temp->next=NULL;
    if(first==NULL)
        return temp;
    cur = first;
    while(cur->next!=NULL)
        cur = cur->next;
    cur->next=temp;
    return first;
}

NODES DeleteFront(NODES first)
{
    NODES temp = first;
```

```

    if(first == NULL)
    {
        printf("No Student data is present in linked list\n");
        return first;
    }
    printf("\nDeleted students data is\n");
    printf("%s\t%s\t%s\t",first->name,first->usn,first->branch);
    first = temp->next;
    free(temp);
    return first;
}

void displayq(NODES first)
{
    NODES cur = first;
    if(first == NULL)
    {
        printf("No students data is present in linked list\n");
        return;
    }
    printf("\nStudents data is\nname\t\tusn\t\tbranch\t\t\n-----\t\t\t-----\t\t\t-
-----\t\t\n");
    while(cur!=NULL)
    {
        printf("%s\t\t%s\t\t%s\t\t\n",cur->name,cur->usn,cur->branch);
        cur = cur->next;
    }
}

NODES createq()
{
    int n;
    NODES temp = NULL;
    printf("Enter number of nodes\n");
    scanf("%d",&n);
    for(int i=0;i<n;i++)
    {
        printf("\nEnter student %d details\n",i+1);
        temp = insertRear(temp);
    }
    return temp;
}

NODE insertFront(NODE first,int item)
{
    NODE temp = (NODE)malloc(sizeof(struct node));
    temp->info = item;

```

```
temp->link = NULL;
if(first==NULL)
    return temp;
temp->link = first;
first = temp;
return first;
}

NODE deleteFront(NODE first)
{
    NODE temp = first;
    if(first == NULL)
    {
        printf("No Node is present in linked list\n");
        return first;
    }
    printf("Deleted item is %d\n",temp->info);
    first = temp->link;
    free(temp);
    return first;
}

void display(NODE first)
{
    NODE cur = first;
    if(first == NULL)
    {
        printf("No Node is present in linked list\n");
        return;
    }
    while(cur!=NULL)
    {
        printf("Info is %d\n",cur->info);
        cur = cur->link;
    }
}

void linearSearch(NODE first,int key)
{
    NODE cur = first;
    int count = 0;
    if(first == NULL)
    {
        printf("No Node is present in linked list\n");
        return;
    }
    while(cur!=NULL&&key!=cur->info)
```



```
{
    cur = cur->link;
    count++;
}
if(cur==NULL)
{
    printf("Unsuccessful search\n");
    return;
}
printf("Element found at location %d\n",count+1);
return;
}

NODE create()
{
    int n,item;
    NODE temp = NULL;
    printf("Enter number of nodes\n");
    scanf("%d",&n);
    for(int i=0;i<n;i++)
    {
        printf("Enter %d item\n",i+1);
        scanf("%d",&item);
        temp = insertFront(temp,item);
    }
    return temp;
}

NODE concatenate(NODE first,NODE second)
{
    NODE cur = first;
    if(first == NULL)
        return second;
    while(cur->link!=NULL)
        cur = cur->link;
    cur->link = second;
    return first;
}

void stack()
{
    int ch, n, i, key, item;
    NODE first=NULL,second=NULL;
    printf("\n-----Stack of integers using linked list-----\n");

    while (1)
    {
        printf("\n-----menu-----\n");
```

```

        printf("1.create SLL stack of
integers\n2.display\n3.insert_front\n4.delete_front\n5.linear
search\n6.Concatenation of 2 lists\n7.Exit\n");
        printf("\nenter your choice:\n");
        scanf("%d", &ch);
        switch (ch)
        {
        case 1: printf("Enter details of first linked list\n");
                first = create();
                break;
        case 2: display(first);
                break;
        case 3: printf("Enter item to insert at first\n");
                scanf("%d",&item);
                first = insertFront(first,item);
                break;
        case 4: first = deleteFront(first);
                break;
        case 5: printf("Enter the key to be searched:\n");
                scanf("%d", &key);
                linearSearch(first,key);
                break;
        case 6: printf("Enter details of second linked list\n");
                second = create();
                first = concatenate(first,second);
                break;
        case 7: exit(0);
        default:printf("invalid choice");
        }
    }
    return;
}

void queue()
{
    int ch, n, i, key, item;
    NODES first=NULL;
    printf("\n-----Queue of students data using linked list-----\n");

    while (1)
    {
        printf("\n-----menu-----\n");
        printf("1.create SLL queue of students data\n2.display\n3.Insert
Rear\n4.Delete front\n5.Exit\n");
        printf("\nenter your choice:\n");
        scanf("%d", &ch);
        switch (ch)
        {

```

```
        case 1: first = createq();
                break;
        case 2: displayq(first);
                break;
        case 3: first = insertRear(first);
                break;
        case 4: first = DeleteFront(first);
                break;
        case 5: exit(0);
        default:printf("invalid choice");
    }
}

int main()
{
    int op;
    printf("\nEnter 1 for SLL stack of integers\nEnter 2 for SLL queue of
students data\nEnter your choice\n");
    scanf("%d",&op);
    switch(op)
    {
        case 1: stack();
                break;
        case 2: queue();
                break;
    }
}
```

OUTPUT:

//OUTPUT OF PART 1

```
Enter 1 for SLL stack of integers
Enter 2 for SLL queue of students data
Enter your choice
1

-----Stack of integers using linked list-----

-----menu-----
1.create SLL stack of integers
2.display
3.insert_front
4.delete_front
5.linear_search
6.Concatenation of 2 lists
7.Exit

enter your choice:
1
Enter details of first linked list
Enter number of nodes
3
Enter 1 item
22
Enter 2 item
44
Enter 3 item
55

-----menu-----
1.create SLL stack of integers
2.display
3.insert_front
4.delete_front
5.linear_search
6.Concatenation of 2 lists
7.Exit

enter your choice:
2
Info is 55
Info is 44
Info is 22
```

```
-----menu-----
1.create SLL stack of integers
2.display
3.insert_front
4.delete_front
5.linear_search
6.Concatenation of 2 lists
7.Exit
```

enter your choice:

3

Enter item to insert at first

99

```
-----menu-----
1.create SLL stack of integers
2.display
3.insert_front
4.delete_front
5.linear_search
6.Concatenation of 2 lists
7.Exit
```

enter your choice:

2

Info is 99

Info is 55

Info is 44

Info is 22

```
-----menu-----
1.create SLL stack of integers
2.display
3.insert_front
4.delete_front
5.linear_search
6.Concatenation of 2 lists
7.Exit
```

enter your choice:

4

Deleted item is 99

```
-----menu-----
1.create SLL stack of integers
2.display
3.insert_front
4.delete_front
5.linear_search
6.Concatenation of 2 lists
7.Exit

enter your choice:
5
Enter the key to be searched:
55
Element found at location 1

-----menu-----
1.create SLL stack of integers
2.display
3.insert_front
4.delete_front
5.linear_search
6.Concatenation of 2 lists
7.Exit

enter your choice:
6
Enter details of second linked list
Enter number of nodes
2
Enter 1 item
88
Enter 2 item
65

-----menu-----
1.create SLL stack of integers
2.display
3.insert front
4.delete_front
5.linear_search
6.Concatenation of 2 lists
7.Exit

enter your choice:
```

```
-----menu-----
1.create SLL stack of integers
2.display
3.insert_front
4.delete_front
5.linear search
6.Concatenation of 2 lists
7.Exit

enter your choice:
2
Info is 55
Info is 44
Info is 22
Info is 65
Info is 88

-----menu-----
1.create SLL stack of integers
2.display
3.insert_front
4.delete_front
5.linear search
6.Concatenation of 2 lists
7.Exit

enter your choice:
7
PS D:\DSA C\DSA> █
```

// OUTPUT FOR PART 2

```
Enter 1 for SLL stack of integers
Enter 2 for SLL queue of students data
Enter your choice
2

-----Queue of students data using linked list-----

-----menu-----
1.create SLL queue of students data
2.display
3.Insert Rear
4.Delete front
5.Exit

enter your choice:
1
Enter number of nodes
2

Enter student 1 details
Enter student's name,usn,branch respectively
Ketan
1AY21IS000
ISE

Enter student 2 details
Enter student's name,usn,branch respectively
Girish
1AY21IS999
ECE

-----menu-----
1.create SLL queue of students data
2.display
3.Insert Rear
4.Delete front
5.Exit

enter your choice:
2

Students data is
name          usn          branch
-----
```



```
Students data is
name          usn          branch
-----
Ketan         1AY21IS000      ISE
Girish        1AY21IS999      ECE

-----menu-----
1.create SLL queue of students data
2.display
3.Insert Rear
4.Delete front
5.Exit

enter your choice:
3
Enter student's name,usn,branch respectively
David
1AY21IS777
CSE

-----menu-----
1.create SLL queue of students data
2.display
3.Insert Rear
4.Delete front
5.Exit

enter your choice:
2

Students data is
name          usn          branch
-----
Ketan         1AY21IS000      ISE
Girish        1AY21IS999      ECE
David         1AY21IS777      CSE

-----menu-----
1.create SLL queue of students data
2.display
3.Insert Rear
4.Delete front
5.Exit
```

```
enter your choice:
4

Deleted students data is
Ketan  1AY21IS000      ISE
-----menu-----
1.create SLL queue of students data
2.display
3.Insert Rear
4.Delete front
5.Exit

enter your choice:
2

Students data is
name          usn          branch
-----
Girish        1AY21IS999      ECE
David         1AY21IS777      CSE

-----menu-----
1.create SLL queue of students data
2.display
3.Insert Rear
4.Delete front
5.Exit

enter your choice:
5
PS D:\DSA C\DSA> |
```

4. Design, Develop and Implement a menu driven Program in C for the following operations on Doubly Linked List (DLL) of Professor Data with the fields: ID, Name, Branch, Area of Specialization

a. Create a DLL stack of N Professor's Data.

Display the status of DLL and count the number of nodes in it.

b. Create a DLL queue of N Professor's Data

Display the status of DLL and count the number of nodes in it.

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
int count=0;

struct node
{
    int id;
    char name[20],branch[10],aos[10];
    struct node *next;
    struct node *prev;
}*first=NULL,*last=NULL,*temp=NULL,*cur=NULL;

void create()
{
    int id;
    char name[20],branch[10],aos[10];
    temp=(struct node *)malloc(sizeof(struct node));
    printf("Enter ID,NAME,BRANCH,AREA OF SPECIALIZATION:\n");
    scanf("%d%s%s%s",&id,name,branch,aos);
    strcpy(temp->aos,aos);
    strcpy(temp->name,name);
    strcpy(temp->branch,branch);
    temp->id=id;
    temp->next=NULL;
    temp->prev=NULL;
    count++;
}

void insert_end()
{
    if(first==NULL)
    {
        create();
        first=last=temp;
    }
}
```

```
else
{
    create();
    last->next=temp;
    temp->prev=last;
    last=temp;
}
}
void display()
{
    temp=first;
    if(first==NULL)
    {
        printf("List is empty\n");
        return;
    }
    else
    {
        printf("\nDetails of the professors are:\n");
        printf("ID\t\t NAME\t\tBRANCH\t\tAREA OF SPECIALIZATION:\n");
        while(temp!=NULL)
        {
            printf("%d\t\t%s\t\t%s\t\t%s\n",temp->id,temp->name,temp->branch,temp->aos);
            temp=temp->next;
        }
        printf("No. of professors = %d\n\n",count);
    }
}
void delete_end(){
    if(first==NULL){
        printf("list is empty\n");
        return;
    }
    else if(first->next==NULL){
        printf("The deleted details of student are:\n");
        printf("%d\t%s\t%s\t%s\n",temp->id,temp->name,temp->branch,temp->aos);
        free(first);
        first=NULL;
    }
    else{
        temp=last;
        last=last->prev;
        last->next=NULL;
        printf("The deleted details of student are:\n");
        printf("%d\t%s\t%s\t%s\n",temp->id,temp->name,temp->branch,temp->aos);
        free(temp);
    }
}
```

```

        count--;
    }
void delete_front()
{
    if(first==NULL)
    {
        printf("The list is empty\n");
        return;
    }
    temp=first;
    if(first->next==NULL)
    {
        printf("The deleted details of student are:\n");
        printf("%d\t%s\t%s\t%s\n",temp->id,temp->name,temp->branch,temp->aos);
        free(first);
        first=NULL;
    }
    else
    {
        first=first->next;
        first->prev=NULL;
        printf("The deleted details of student are:\n");
        printf("%d\t%s\t%s\t%s\n",temp->id,temp->name,temp->branch,temp->aos);
        free(temp);
    }
    count--;
}
void queuedemo(){
    printf("=====Queue demo =====\n");
    int ch,n,i;
    while(1)
    {
        printf("-----QUEUE of Professors using DLL-----\n");
        printf("1.CREATE DLL QUEUE of n professors\n");
        printf("2.DISPLAY the QUEUE of professors\n");
        printf("3.INSERT END\n");
        printf("4.DELETE FRONT\n");
        printf("5.EXIT\n");
        printf("\nEnter choice:\n");
        scanf("%d",&ch);
        switch(ch)
        {
            case 1:printf("Enter number of professors:\n");
                    scanf("%d",&n);
                    for(i=0;i<n;i++)
                    {
                        insert_end();

```

```

        }
        break;
    case 2:display();
    break;
    case 3:insert_end();
    break;
    case 4:delete_front();
    break;
    case 5:
    exit(1);
    default:printf("Wrong choice\n");
    }
}
}

void stackdemo(){
    // STACK IS NOT ONLY INSERT FRONT AND DELETE FRONT ITS ALSO INSERT REAR
    AND DELETE REAR
    // IF YOU WANT INSETR FORNT AND DELETE FRONT APPLY YOURSELVES
    printf("=====stack demo =====\n");
    int ch,n,i;
    while(1)
    {
        printf("-----Stack of Professors using DLL-----\n");
        printf("1.CREATE DLL stack of n professors\n");
        printf("2.DISPLAY the stack of professors\n");
        printf("3.INSERT END\n");
        printf("4.DELETE FRONT\n");
        printf("5.EXIT\n");
        printf("\nEnter choice:\n");
        scanf("%d",&ch);
        switch(ch)
        {
            case 1:printf("Enter number of professors:\n");
                    scanf("%d",&n);
                    for(i=0;i<n;i++)
                    {
                        insert_end();
                    }
                    break;
            case 2:display();
            break;
            case 3:insert_end();
            break;
            case 4:delete_end();
            break;
            case 5:
            exit(1);
            default:printf("Wrong choice\n");
        }
    }
}

```

```
    }  
  }  
}  
int main()  
{  
    int ch;  
    printf("Enter 1 to get stack demo using professor data\n");  
    printf("Enter 2 to get queue demo using professor data\n");  
    scanf("%d",&ch);  
    if(ch==1){  
        stackdemo();  
    }  
    if(ch==2){  
        queuedemo();  
    }  
    return 0;  
}
```

OUTPUT:

// Output of part 1 i.e. Stack demo

```
Enter 1 to get stack demo using professor data
Enter 2 to get queue demo using professor data
1
=====stack demo =====
-----Stack of Professors using DLL-----
1.CREATE DLL stack of n professors
2.DISPLAY the stack of professors
3.INSERT END
4.DELETE FRONT
5.EXIT

Enter choice:
1
Enter number of professors:
2
Enter ID,NAME,BRANCH,AREA OF SPECIALIZATION:
123
Ketan
ISE
Python
Enter ID,NAME,BRANCH,AREA OF SPECIALIZATION:
345
Abhi
ISE
C
-----Stack of Professors using DLL-----
1.CREATE DLL stack of n professors
2.DISPLAY the stack of professors
3.INSERT END
4.DELETE FRONT
5.EXIT

Enter choice:
2

Details of the professors are:
ID          NAME          BRANCH          AREA OF SPECIALIZATION:
123         Ketan           ISE             Python
345         Abhi            ISE             C
No. of professors = 2
```



```

-----Stack of Professors using DLL-----
1.CREATE DLL stack of n professors
2.DISPLAY the stack of professors
3.INSERT END
4.DELETE FRONT
5.EXIT

Enter choice:
3
Enter ID,NAME,BRANCH,AREA OF SPECIALIZATION:
567
Aamir
ISE
C++
-----Stack of Professors using DLL-----
1.CREATE DLL stack of n professors
2.DISPLAY the stack of professors
3.INSERT END
4.DELETE FRONT
5.EXIT

Enter choice:
2

Details of the professors are:
ID          NAME          BRANCH          AREA OF SPECIALIZATION:
123          Ketan          ISE          Python
345          Abhi          ISE          C
567          Aamir          ISE          C++
No. of professors = 3

-----Stack of Professors using DLL-----
1.CREATE DLL stack of n professors
2.DISPLAY the stack of professors
3.INSERT END
4.DELETE FRONT
5.EXIT

Enter choice:
4
The deleted details of student are:
567    Aamir    ISE    C++

-----Stack of Professors using DLL-----
1.CREATE DLL stack of n professors
2.DISPLAY the stack of professors
3.INSERT END
4.DELETE FRONT
5.EXIT

Enter choice:
5
PS D:\DSA C\DSA>

```

// Output of DLL using Queue

```
Enter 1 to get stack demo using professor data
Enter 2 to get queue demo using professor data
2
=====Queue demo =====
-----QUEUE of Professors using DLL-----
2.DISPLAY the QUEUE of professors
3.INSERT END
4.DELETE FRONT
5.EXIT

Enter choice:
1
Enter number of professors:
2
Enter ID,NAME,BRANCH,AREA OF SPECIALIZATION:
987
David
ECE
Cloud
Enter ID,NAME,BRANCH,AREA OF SPECIALIZATION:
765
John
MECH
DevOps
-----QUEUE of Professors using DLL-----
1.CREATE DLL QUEUE of n professors
2.DISPLAY the QUEUE of professors
3.INSERT END
4.DELETE FRONT
5.EXIT

Enter choice:
2

Details of the professors are:
ID          NAME          BRANCH          AREA OF SPECIALIZATION:
987         David          ECE             Cloud
765         John           MECH            DevOps
No. of professors = 2
```

```

-----QUEUE of Professors using DLL-----
1.CREATE DLL QUEUE of n professors
2.DISPLAY the QUEUE of professors
3.INSERT END
4.DELETE FRONT
5.EXIT

Enter choice:
3
Enter ID,NAME,BRANCH,AREA OF SPECIALIZATION:
321
Param
ISE
Everything
-----QUEUE of Professors using DLL-----
1.CREATE DLL QUEUE of n professors
2.DISPLAY the QUEUE of professors
3.INSERT END
4.DELETE FRONT
5.EXIT

Enter choice:
2

Details of the professors are:
ID          NAME          BRANCH          AREA OF SPECIALIZATION:
987         David          ECE             Cloud
765         John           MECH            DevOps
321         Param          Everything
No. of professors = 3

-----QUEUE of Professors using DLL-----
1.CREATE DLL QUEUE of n professors
2.DISPLAY the QUEUE of professors
3.INSERT END
4.DELETE FRONT
5.EXIT

Enter choice:
4
The deleted details of student are:
987      David      ECE      Cloud

-----QUEUE of Professors using DLL-----
1.CREATE DLL QUEUE of n professors
2.DISPLAY the QUEUE of professors
3.INSERT END
4.DELETE FRONT
5.EXIT

Enter choice:
5
PS D:\DSA C\DSA>

```

// Other program will update soon...