Manav Rachna International Institute of Research and Studies

School of Computer Applications

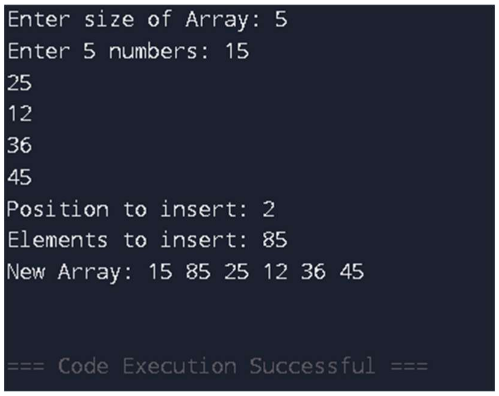
Department of Computer Applications

Submitted By	
Student Name	KHUSHI SHARMA
Roll No	24/SCA/BCA(AI/ML)/031
Program	BCA(AI&ML)
Semester	2
Section/Group	C
Department	School of Computer Applications
Session / Batch	2024-27
Submitted To	
Faculty Name	Dr.Parul Gandhi/Dr.Sakshi Gupta

INDEX

Q1 - write a c program to implement insertion in 1 d array?

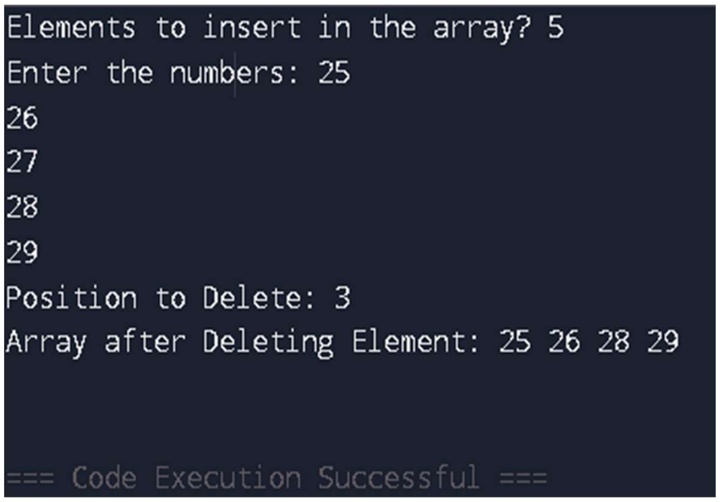
```
#include <stdio.h>
int main() {
    int nums[11];
    int n, insertAt, newVal;
    printf("Enter size of Array: ");
    scanf("%d", &n);
    printf("Enter %d numbers: ", n);
    for (int i = 0; i < n; i++) {
        scanf("%d", &nums[i]);
    }
    printf("Position to insert: ", n + 1);
    scanf("%d", &insertAt);
    printf("Elements to insert: ");
    scanf("%d", &newVal);
    for (int i = n; i >= insertAt; i--) {
        nums[i] = nums[i - 1];
    }
    nums[insertAt - 1] = newVal;
    printf("New Array: ");
    for (int i = 0; i <= n; i++) {
        printf("%d ", nums[i]);
    }
    printf("\n");
    return 0;
}
```



```
Enter size of Array: 5
Enter 5 numbers: 15
25
12
36
45
Position to insert: 2
Elements to insert: 85
New Array: 15 85 25 12 36 45


=== Code Execution Successful ===
```

Q2 – write a c program to implement deletion in 1D array?

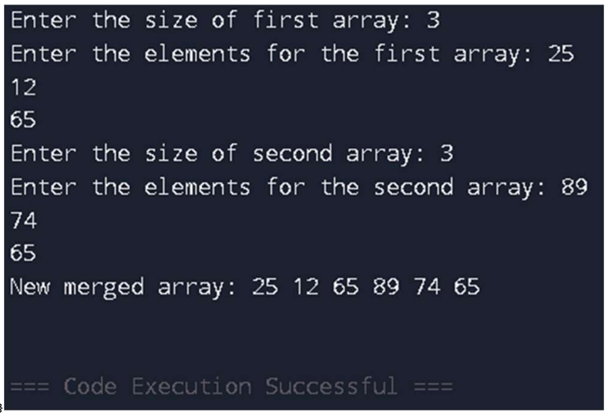
```
#include <stdio.h>
int main() {
    int numbers[7];
    int n, removePos;
    printf("How many numbers in the array? ");
    scanf("%d", &n);
    printf("Enter the %d numbers: ", n);
    for (int i = 0; i < n; i++) {
        scanf("%d", &numbers[i]);
    }
    printf("Position to remove (1 to %d): ", n);
    scanf("%d", &removePos);
    if (removePos < 1 || removePos > n) {
        printf("Invalid position!\n");
        return 1;
    }
    for (int i = removePos - 1; i < n - 1; i++) {
        numbers[i] = numbers[i + 1];
    }
    n--;
    printf("Array after removing element at position %d: ",
removePos);
    for (int i = 0; i < n; i++) {
        printf("%d ", numbers[i]);
    }
    printf("\n");
    return 0;
}
```



```
Elements to insert in the array? 5
Enter the numbers: 25
26
27
28
29
Position to Delete: 3
Array after Deleting Element: 25 26 28 29


=== Code Execution Successful ===
```

Q3 - write a c program to concatenate two arrays?

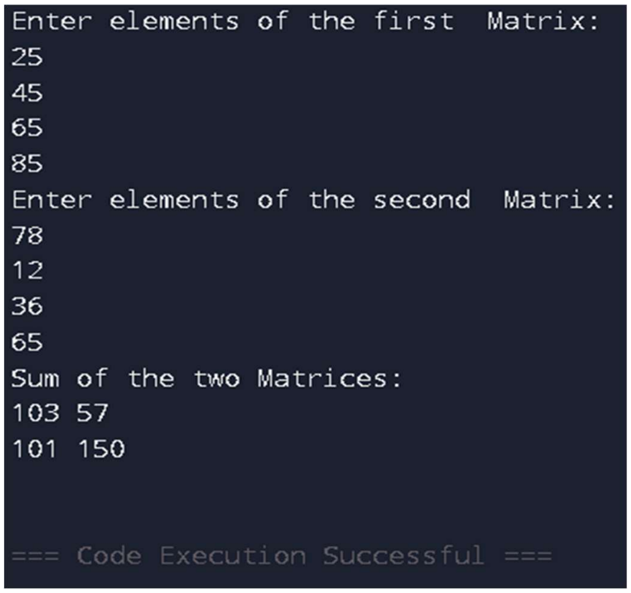
```
#include <stdio.h>
int main() {
    int arr1[50], arr2[50], combined[100];
    int size1, size2, i, k = 0;
    printf("Enter the size of first array: ");
    scanf("%d", &size1);
    printf("Enter the elements for the first array: ", size1);
    for (i = 0; i < size1; i++) {
        scanf("%d", &arr1[i]);
        combined[k++] = arr1[i];
    }
    printf("Enter the size of second array: ");
    scanf("%d", &size2);
    printf("Enter the elements for the second array: ", size2);
    for (i = 0; i < size2; i++) {
        scanf("%d", &arr2[i]);
        combined[k++] = arr2[i];
    }
    printf("New merged array: ");
    for (i = 0; i < size1 + size2; i++) {
        printf("%d ", combined[i]);
    }
    printf("\n");
    return 0; }
```



```
Enter the size of first array: 3
Enter the elements for the first array: 25
12
65
Enter the size of second array: 3
Enter the elements for the second array: 89
74
65
New merged array: 25 12 65 89 74 65


=== Code Execution Successful ===
```

Q4 – write a c program to implement the operations of 2D Array(addition, subtraction, multiplication, transpose)?

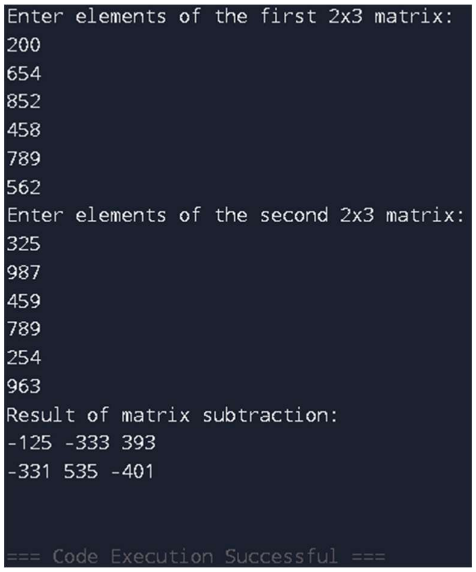
```c
#include <stdio.h>
int main() {
    int matrix1[2][2], matrix2[2][2], sum[2][2];
    printf("Enter elements of the first  Matrix:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            scanf("%d", &matrix1[i][j]);
        }
    }
    printf("Enter elements of the second  Matrix:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            scanf("%d", &matrix2[i][j]);
        }
    }
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            sum[i][j] = matrix1[i][j] + matrix2[i][j];
        }
    }
    printf("Sum of the two Matrices:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 2; j++) {
            printf("%d ", sum[i][j]);
        }
        printf("\n");
    }
    return 0;}
```



```
Enter elements of the first  Matrix:
25
45
65
85
Enter elements of the second  Matrix:
78
12
36
65
Sum of the two Matrices:
103 57
101 150


=== Code Execution Successful ===
```

Subtraction

```
#include <stdio.h>
int main() {
    int mat1[2][3], mat2[2][3], diff[2][3];
    printf("Enter elements of the first 2x3 matrix:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 3; j++) {
            scanf("%d", &mat1[i][j]);
        }
    }
    printf("Enter elements of the second 2x3 matrix:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 3; j++) {
            scanf("%d", &mat2[i][j]);
        }
    }
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 3; j++) {
            diff[i][j] = mat1[i][j] - mat2[i][j];
        }
    }
    printf("Result of matrix subtraction:\n");
    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 3; j++) {
            printf("%d ", diff[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

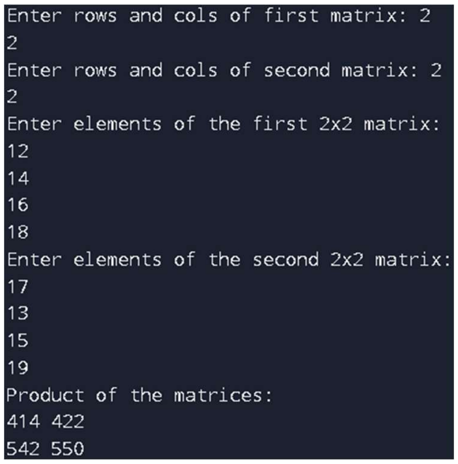


```
Enter elements of the first 2x3 matrix:
200
654
852
458
789
562
Enter elements of the second 2x3 matrix:
325
987
459
789
254
963
Result of matrix subtraction:
-125 -333 393
-331 535 -401


=== Code Execution Successful ===
```

Multiplication

```
#include <stdio.h>
int main() {
    int m1[10][10], m2[10][10], result[10][10];
    int r1, c1, r2, c2;
    printf("Enter rows and cols of first matrix: ");
    scanf("%d %d", &r1, &c1);
    printf("Enter rows and cols of second matrix: ");
    scanf("%d %d", &r2, &c2);
    if (c1 != r2) {
        printf("Matrix multiplication isn't possible (columns of first must
equal rows of second).\n");
        return 1;
    }
    printf("Enter elements of the first %dx%d matrix:\n", r1, c1);
    for (int i = 0; i < r1; i++) {
        for (int j = 0; j < c1; j++) {
            scanf("%d", &m1[i][j]);
        } }
    printf("Enter elements of the second %dx%d matrix:\n", r2, c2);
    for (int i = 0; i < r2; i++) {
        for (int j = 0; j < c2; j++) {
            scanf("%d", &m2[i][j]);
        } }
    for (int i = 0; i < r1; i++) {
        for (int j = 0; j < c2; j++) {
            result[i][j] = 0;
            for (int k = 0; k < c1; k++) {
                result[i][j] += m1[i][k] * m2[k][j];
            } } }
    printf("Product of the matrices:\n");
    for (int i = 0; i < r1; i++) {
```

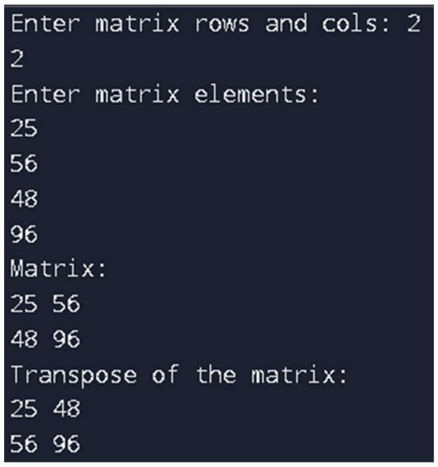
```c
        for (int j = 0; j < c2; j++) {
            printf("%d ", result[i][j]);
        }
        printf("\n");
    }
    return 0; }
```



```
Enter rows and cols of first matrix: 2
2
Enter rows and cols of second matrix: 2
2
Enter elements of the first 2x2 matrix:
12
14
16
18
Enter elements of the second 2x2 matrix:
17
13
15
19
Product of the matrices:
414 422
542 550
```

Transpose

```c
#include <stdio.h>
int main() {
    int rows, cols;
    printf("Enter matrix rows and cols: ");
    scanf("%d %d", &rows, &cols);
    int matrix[rows][cols], transpose[cols][rows];
    printf("Enter matrix elements:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            scanf("%d", &matrix[i][j]);
        }
    }
    printf("Matrix:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            transpose[j][i] = matrix[i][j];
        }
    }
    printf("Transpose of the matrix:\n");
    for (int i = 0; i < cols; i++) {
        for (int j = 0; j < rows; j++) {
            printf("%d ", transpose[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

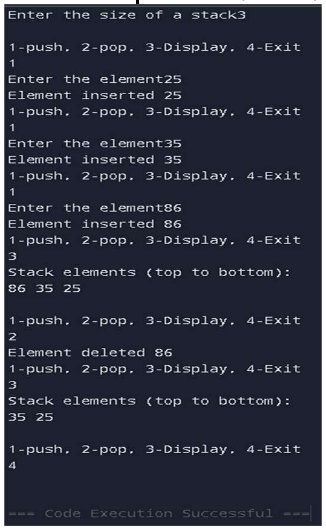


```
Enter matrix rows and cols: 2
2
Enter matrix elements:
25
56
48
96
Matrix:
25 56
48 96
Transpose of the matrix:
25 48
56 96
```

Q5 – Write a c program to implement Stacks using Array?

```
#include <stdio.h>
#include<stdlib.h>
void push();
void pop();
void display();
int maxstk, stack[10],top = -1;
int main() {
    int ch;
    printf("Enter the size of a stack");
    scanf("%d", & maxstk);
    while(1) {
    printf("\n1-push, 2-pop, 3-Display, 4-Exit\n");
    scanf("%d", &ch);
    switch(ch)
    {
    case 1:
    push();
    break;
    case 2:
    pop();
    break;
case 3:
    display();
    break;
    case 4:
    exit(0);
    break;
    default:
    printf("Wrong choice");
     }
  }
```

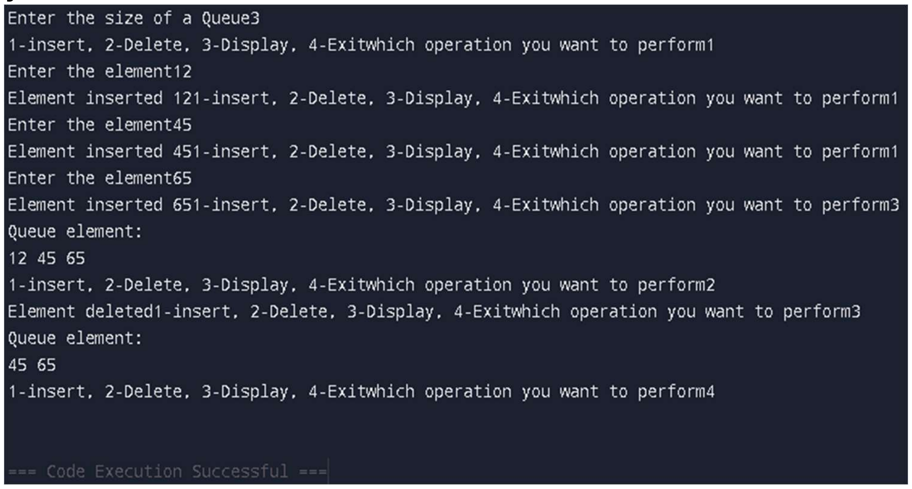
```
}
void push()
{
int ele;
    if (top == maxstk-1){
    printf("Overflow\n");
    }
else
{
    printf("Enter the element");
    scanf("%d", & ele);
    top = top + 1;
    stack[top] = ele;
    printf("Element inserted %d", ele);
    }
}
void pop()
{
    if (top == -1) {
    printf("Underflow");
    }
    else
    {
         printf("Element deleted %d", stack[top]);
        top = top -1;
    }
}
void display()
{
    int i;
    if (top == -1){
        printf("underflow condition");
    }
    else {
    printf("Stack elements (top to bottom):\n");
        for(i = top; i >= 0; i--)
            printf("%d ", stack[i]);
        printf("\n");
```



```
Enter the size of a stack3

1-push, 2-pop, 3-Display, 4-Exit
1
Enter the element25
Element inserted 25
1-push, 2-pop, 3-Display, 4-Exit
1
Enter the element35
Element inserted 35
1-push, 2-pop, 3-Display, 4-Exit
1
Enter the element86
Element inserted 86
1-push, 2-pop, 3-Display, 4-Exit
3
Stack elements (top to bottom):
86 35 25

1-push, 2-pop, 3-Display, 4-Exit
2
Element deleted 86
1-push, 2-pop, 3-Display, 4-Exit
3
Stack elements (top to bottom):
35 25

1-push, 2-pop, 3-Display, 4-Exit
4


=== Code Execution Successful ===
```

Q6 – Write a c program to implement linear queue using Array?

```
#include <stdio.h>
#include<stdlib.h>
void insert();
void Delete();
void display();
int size, Queue[10],r=-1,f=-1;
void main() {
    int ch;
    printf("Enter the size of a Queue");
    scanf("%d", & size);
    while(1) {
    printf("1-insert, 2-Delete, 3-Display, 4-Exit");
    printf("which operation you want to perform");
    scanf("%d", &ch);
    switch(ch)
    {
    case 1:
    insert();
    break;
    case 2:
    Delete();
    break;
    case 3:
    display();
    break;
    case 4:
    exit(1);
    break;
    default:
    printf("Wrong choice");
     }
  }
}
void insert()
{
int ele;

    if (r == size -1){

    printf("Overflow\n");
    }
    else{
        printf("Enter the element");
    scanf("%d", & ele);
    if ((f==-1) && (r== -1)){
f=r=0;
}else
{
    r = r + 1;}

    Queue[r] = ele;
    printf("Element inserted %d", ele);
        }    }

void Delete()
{
    if (f == -1) {
    printf("Underflow");
    }
    else{
    if (f==r){
    f=r = -1;
}
    else
    {

        f = f +1;
    }      printf("Element deleted");
}
}

void display()
{
    int i;
    if (f == -1){
        printf("underflow condition");
    }
    else {
    printf("Queue element:\n");
        for(i = f; i <= r; i++)
            printf("%d ", Queue[i]);
        printf("\n");
}
}
```

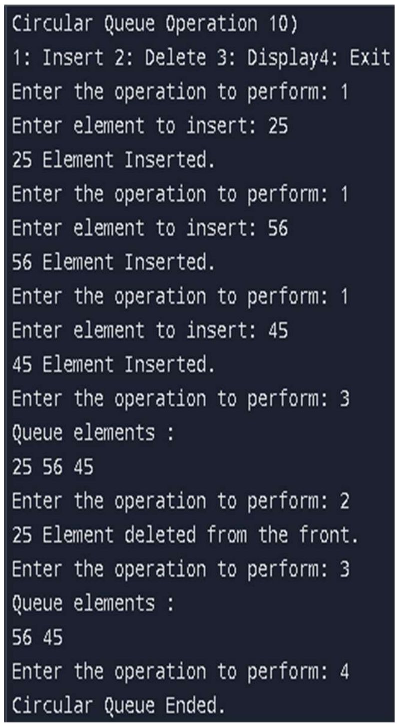


```
Enter the size of a Queue3
1-insert, 2-Delete, 3-Display, 4-Exitwhich operation you want to perform1
Enter the element12
Element inserted 121-insert, 2-Delete, 3-Display, 4-Exitwhich operation you want to perform1
Enter the element45
Element inserted 451-insert, 2-Delete, 3-Display, 4-Exitwhich operation you want to perform1
Enter the element65
Element inserted 651-insert, 2-Delete, 3-Display, 4-Exitwhich operation you want to perform3
Queue element:
12 45 65
1-insert, 2-Delete, 3-Display, 4-Exitwhich operation you want to perform2
Element deleted1-insert, 2-Delete, 3-Display, 4-Exitwhich operation you want to perform3
Queue element:
45 65
1-insert, 2-Delete, 3-Display, 4-Exitwhich operation you want to perform4



=== Code Execution Successful ===
```

Q7 – Write a c program to implement circular Queue using Array?

```
#include <stdio.h>
#include <stdlib.h>
#define MAX_QUEUE_SIZE 10
int queue[MAX_QUEUE_SIZE];
int front = -1;
int rear = -1;
void enqueue(int element);
void dequeue();
void display();
int main() {
    int choice;
    printf("Circular Queue Operation %d)\n", MAX_QUEUE_SIZE);
    printf("1: Insert 2: Delete 3: Display4: Exit\n");
    while (1) {
        printf("Enter the operation to perform: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1: {
                int element;
                printf("Enter element to insert: ");
                scanf("%d", &element);
                enqueue(element);
                break;
            }
            case 2:
                dequeue();
                break;
            case 3:
                display();
                break;
            case 4:
                printf("Circular Queue Ended.\n");
                exit(0);
            default:
                printf("Invalid choice. Please try again.\n");
        }
    }
    return 0;
}
void enqueue(int element) {
    if ((front == 0 && rear == MAX_QUEUE_SIZE - 1) || (front == rear + 1)) {
        printf("Queue Overflow!  %d.\n", element);
    } else {
        if (front == -1) {
            front = 0;
        }
        rear = (rear + 1) % MAX_QUEUE_SIZE;
```

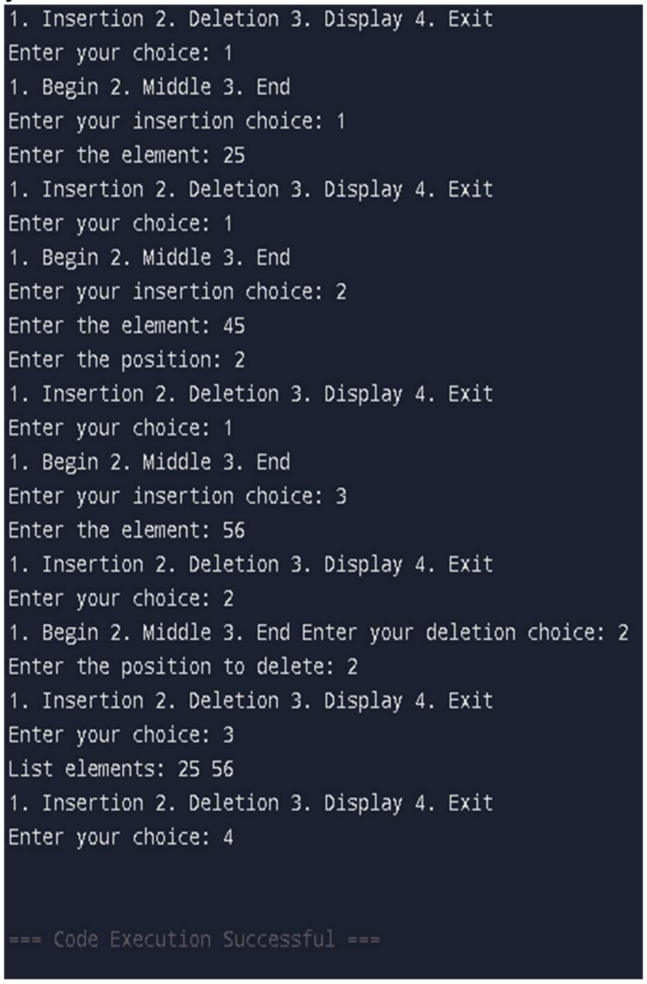
```c
        queue[rear] = element;
        printf("%d Element Inserted.\n", element);
    }
}
void dequeue() {
    if (front == -1) {
        printf("Queue Underflow!.\n");
    } else {
        printf("%d Element deleted from the front.\n", queue[front]);
        if (front == rear) {
            front = rear = -1;
        } else {
            front = (front + 1) % MAX_QUEUE_SIZE;
        }
    }
}
void display() {
    if (front == -1) {
        printf("Queue is empty.\n");
    } else {
        printf("Queue elements :\n");
        int i = front;
        do {
            printf("%d ", queue[i]);
            i = (i + 1) % MAX_QUEUE_SIZE;
        } while (i != (rear + 1) % MAX_QUEUE_SIZE);
        printf("\n");
    }
}
```



```
Circular Queue Operation 10)
1: Insert 2: Delete 3: Display4: Exit
Enter the operation to perform: 1
Enter element to insert: 25
25 Element Inserted.
Enter the operation to perform: 1
Enter element to insert: 56
56 Element Inserted.
Enter the operation to perform: 1
Enter element to insert: 45
45 Element Inserted.
Enter the operation to perform: 3
Queue elements :
25 56 45
Enter the operation to perform: 2
25 Element deleted from the front.
Enter the operation to perform: 3
Queue elements :
56 45
Enter the operation to perform: 4
Circular Queue Ended.
```

Q8- write a c program to implement link list(Insertion,Deletion,Display)?

```c
#include <stdio.h>
#include <stdlib.h>
typedef struct node {
    int info;
    struct node *next;
} Node;

Node *start = NULL;
void insbeg();
void insmid();
void insend();
void delbeg();
void delmid();
void delend();
void display();

int main() {
    int ch, ch1;
    while (1)
    {
        printf("1. Insertion 2. Deletion 3. Display 4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &ch);
        switch (ch) {
            case 1:
                printf("1. Begin 2. Middle 3. End\n");
                printf("Enter your insertion choice: ");
                scanf("%d", &ch1);
                switch (ch1) {
                    case 1:
                     insbeg();
                    break;
                    case 2:
                     insmid();
                    break;
                    case 3:
                     insend();
                    break;
                    default:
                     printf("Invalid insertion choice\n");
                      break;
                }
                break;
            case 2:
                printf("1. Begin 2. Middle 3. End ");
                printf("Enter your deletion choice: ");
                scanf("%d", &ch1);
                switch (ch1) {
                    case 1:
                     delbeg();
                     break;
                    case 2:
                     delmid();
                     break;
                    case 3:
                     delend();
                     break;
                    default:
                     printf("Invalid deletion choice\n");
                      break;
                }
                break;
            case 3:
                display();
                break;
            case 4:
                exit(0);
            default:
                printf("Invalid choice\n");
        }
    }
    return 0;
}

void insbeg() {
    Node *temp = (Node *)malloc(sizeof(Node));
    int ele;
    printf("Enter the element: ");
    scanf("%d", &ele);
    temp->info = ele;
    temp->next = start;
    start = temp;
}

void insmid() {
    Node *temp = (Node *)malloc(sizeof(Node));
    int ele, pos, i;
    printf("Enter the element: ");
    scanf("%d", &ele);
    printf("Enter the position: ");
    scanf("%d", &pos);
    temp->info = ele;

    if (pos == 1) {
        temp->next = start;
        start = temp;
        return;
    }

    Node *ptr = start;
    for (i = 1; i < pos - 1 && ptr != NULL; i++) {
        ptr = ptr->next;
    }

    if (ptr == NULL) {
        printf("Position out of range\n");
        free(temp);
        return;
    }

    temp->next = ptr->next;
    ptr->next = temp;
}

void insend() {
    Node *temp = (Node *)malloc(sizeof(Node));
    int ele;
    printf("Enter the element: ");
    scanf("%d", &ele);
    temp->info = ele;
    temp->next = NULL;

    if (start == NULL) {
        start = temp;
        return;
    }

    Node *ptr = start;
    while (ptr->next != NULL) {
        ptr = ptr->next;
    }
    ptr->next = temp;
}

void delbeg() {
    if (start == NULL) {
        printf("Underflow\n");
        return;
    }
    Node *ptr = start;
    start = start->next;
    free(ptr);
}

void delmid() {
    int pos, i;
    if (start == NULL) {
        printf("Underflow\n");
        return;
    }
    printf("Enter the position to delete: ");
    scanf("%d", &pos);
    if (pos == 1) {
        delbeg();
        return;
    }

    Node *ptr = start;
    Node *temp = NULL;
    for (i = 1; i < pos && ptr != NULL; i++) {
        temp = ptr;
        ptr = ptr->next;
    }

    if (ptr == NULL) {
        printf("Position out of range\n");
        return;
    }

    temp->next = ptr->next;
    free(ptr);
}

void delend() {
    if (start == NULL) {
        printf("Underflow\n");
        return;
    }

    if (start->next == NULL) {
        free(start);
        start = NULL;
        return;
    }

    Node *ptr = start;
    Node *temp = NULL;
    while (ptr->next != NULL) {
        temp = ptr;
        ptr = ptr->next;
    }

    temp->next = NULL;
    free(ptr);
}

void display() {
    if (start == NULL) {
        printf("List is empty\n");
        return;
    }

    Node *ptr = start;
    printf("List elements: ");
    while (ptr != NULL) {
        printf("%d ", ptr->info);
        ptr = ptr->next;
    }
    printf("\n");
}
```



```
1. Insertion 2. Deletion 3. Display 4. Exit
Enter your choice: 1
1. Begin 2. Middle 3. End
Enter your insertion choice: 1
Enter the element: 25
1. Insertion 2. Deletion 3. Display 4. Exit
Enter your choice: 1
1. Begin 2. Middle 3. End
Enter your insertion choice: 2
Enter the element: 45
Enter the position: 2
1. Insertion 2. Deletion 3. Display 4. Exit
Enter your choice: 1
1. Begin 2. Middle 3. End
Enter your insertion choice: 3
Enter the element: 56
1. Insertion 2. Deletion 3. Display 4. Exit
Enter your choice: 2
1. Begin 2. Middle 3. End Enter your deletion choice: 2
Enter the position to delete: 2
1. Insertion 2. Deletion 3. Display 4. Exit
Enter your choice: 3
List elements: 25 56
1. Insertion 2. Deletion 3. Display 4. Exit
Enter your choice: 4



=== Code Execution Successful ===
```