DATA STRUCTURE

(PRACTICAL LAB)

Manav Rachna International Institute of Research and Studies

School of Computer Applications

Department of Computer Applications

Submitted By	
Student Name	***RIAZ MOHAMMAD***
Roll No	***24/SCA/BCA (AI&ML)/39***
Programme	***B.C.A. AI & ML***
Semester	2nd Semester
Section/Group	***II C***
Department	***Computer Applications***
Batch	***2024-2028***
Submitted To	
Faculty Name	***Ms. Parul Gandhi***

INDEX

1.Write a program in C to implement insertion in 1-D Arrays?

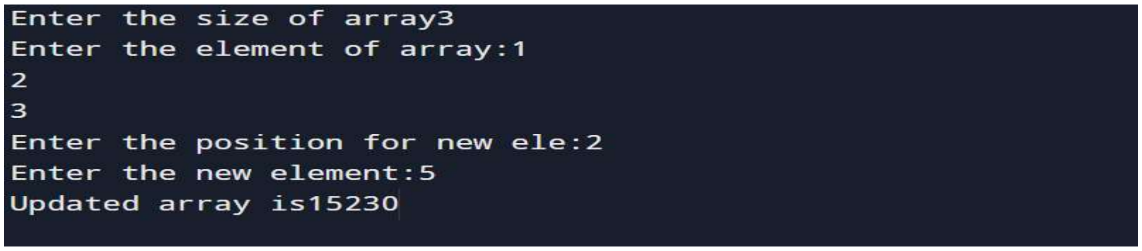
Insertion in array can be done in 3 positions in array:

1. In between first and last:

Input:

```
#include <stdio.h>
int main() {
int a[10],size,i,pos,ele;
printf("Enter the size of array");
scanf ("%d", &size);
printf("Enter the element of array:");
for(i=0;i<size;i++){
    scanf("%d", &a[i]);
}
printf("Enter the position for new ele:");
scanf("%d", & pos);
printf("Enter the new element:");
scanf("%d", & ele);
for(i=size-1; i >= pos-1;i--){
    a[i+1] = a[i];
}
a[pos - 1] = ele;
    size++;
printf("Updated array is");
for(i=0;i<=size;i++)
printf("%d", a[i]);
 return 0;
}
```

Output:



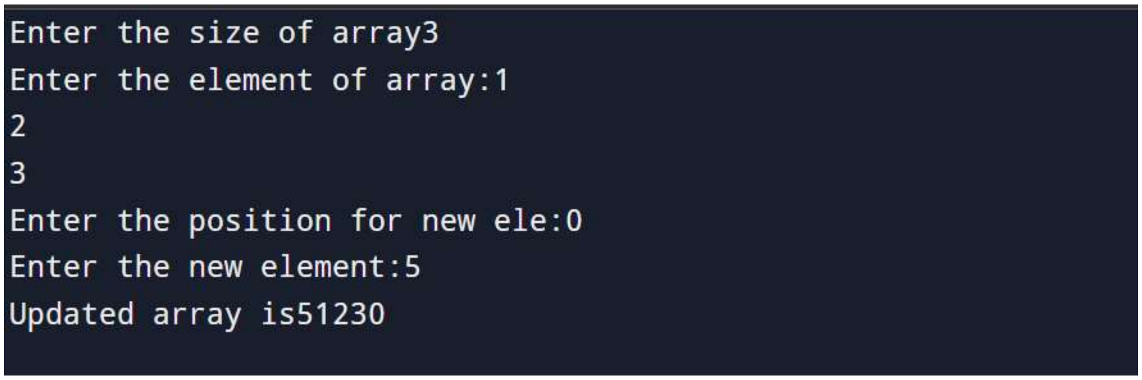
```
Enter the size of array3
Enter the element of array:1
2
3
Enter the position for new ele:2
Enter the new element:5
Updated array is15230
```

2. In First position:

Input:

```
#include <stdio.h>
int main() {
int a[10],size,i,pos,ele;
printf("Enter the size of array");
scanf ("%d", &size);
printf("Enter the element of array:");
for(i=0;i<size;i++){
    scanf("%d", &a[i]);
}
printf("Enter the position for new ele:");
scanf("%d", & pos);
printf("Enter the new element:");
scanf("%d", & ele);
for(i=size-1; i >= pos-1;i--){
    a[i+1] = a[i];
}
a[pos - 1] = ele;
    size++;
printf("Updated array is");
for(i=0;i<=size;i++)
printf("%d", a[i]);

    return 0;
}
```

Output:



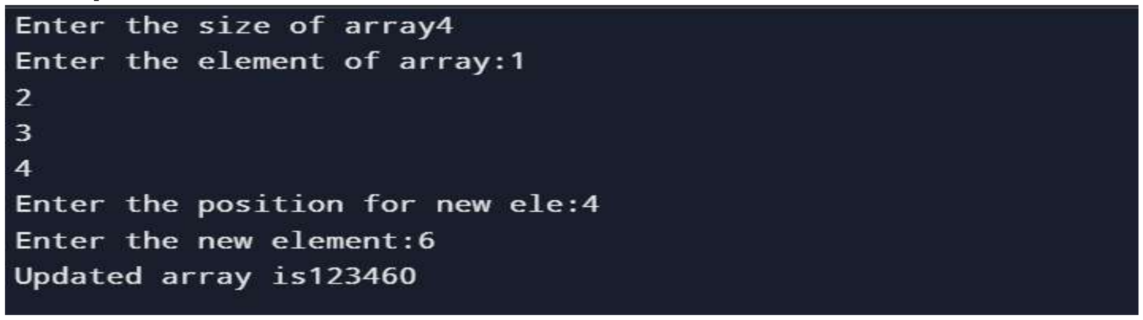
```
Enter the size of array3
Enter the element of array:1
2
3
Enter the position for new ele:0
Enter the new element:5
Updated array is51230
```

3. In last position

Input:

```
#include <stdio.h>
int main() {
int a[10],size,i,pos,ele;
printf("Enter the size of array");
scanf ("%d", &size);
printf("Enter the element of array:");
for(i=0;i<size;i++){
    scanf("%d", &a[i]);
}
printf("Enter the position for new ele:");
scanf("%d", & pos);
printf("Enter the new element:");
scanf("%d", & ele);
for(i=size-1; i >= pos-1;i--){
    a[i+1] = a[i];
}
a[pos - 1] = ele;
    size++;
printf("Updated array is");
for(i=0;i<=size;i++)
printf("%d", a[i]);

    return 0;
}
```

Output:



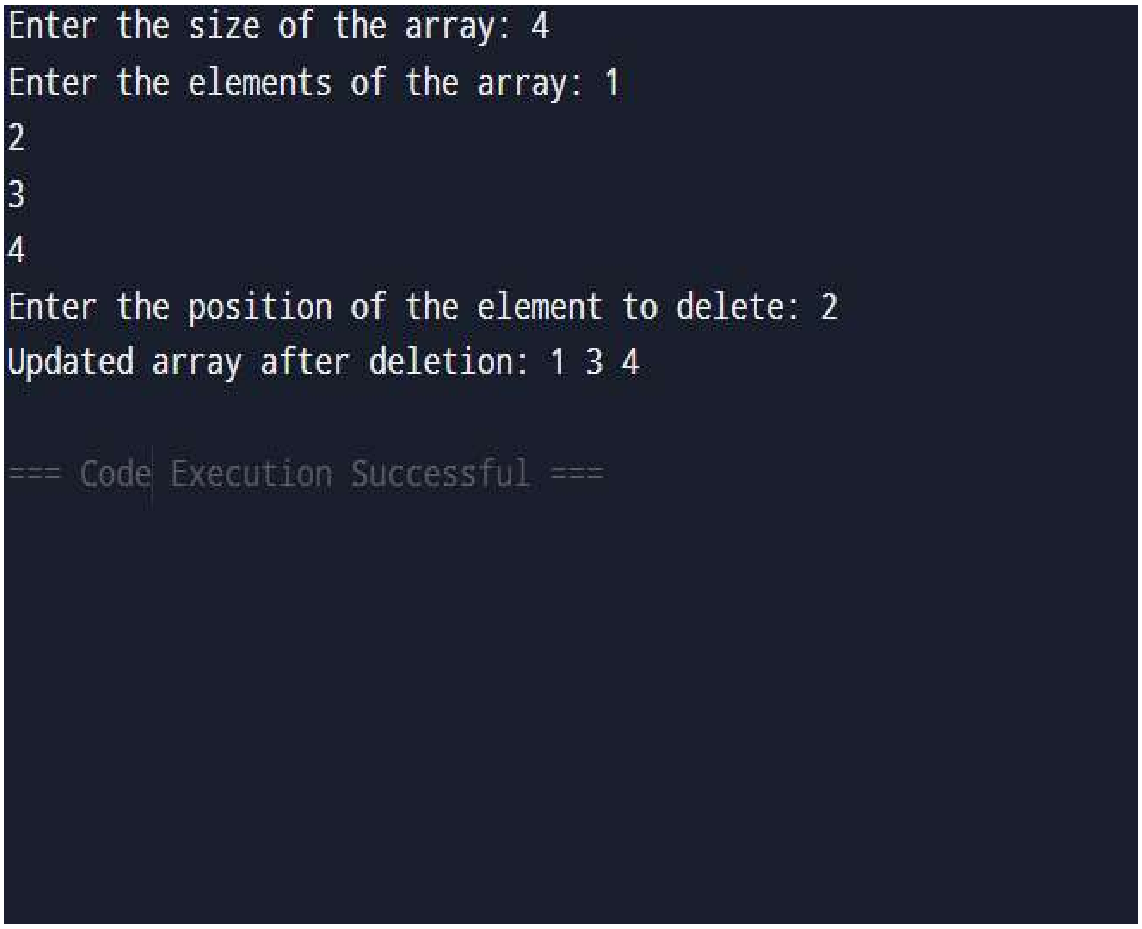
```
Enter the size of array4
Enter the element of array:1
2
3
4
Enter the position for new ele:4
Enter the new element:6
Updated array is123460
```

2.Write a program in C to implement deletion in 1-D Array?

INPUT:

```
#include <stdio.h>
int main() {
    int a[10], size, i, pos;
    printf("Enter the size of the array: ");
    scanf("%d", &size);
    printf("Enter the elements of the array: ");
    for (i = 0; i < size; i++) {
        scanf("%d", &a[i]);
    }
    printf("Enter the position of the element to delete: ");
    scanf("%d", &pos);

    if (pos < 1 || pos > size) {
        printf("Invalid position!\n");
        return 1;
    }
    for (i = pos - 1; i < size - 1; i++) {
        a[i] = a[i + 1];
    } size--;
    printf("Updated array after deletion: ");
    for (i = 0; i < size; i++) {
        printf("%d ", a[i]);
    }
 return 0;
}
```

Output:



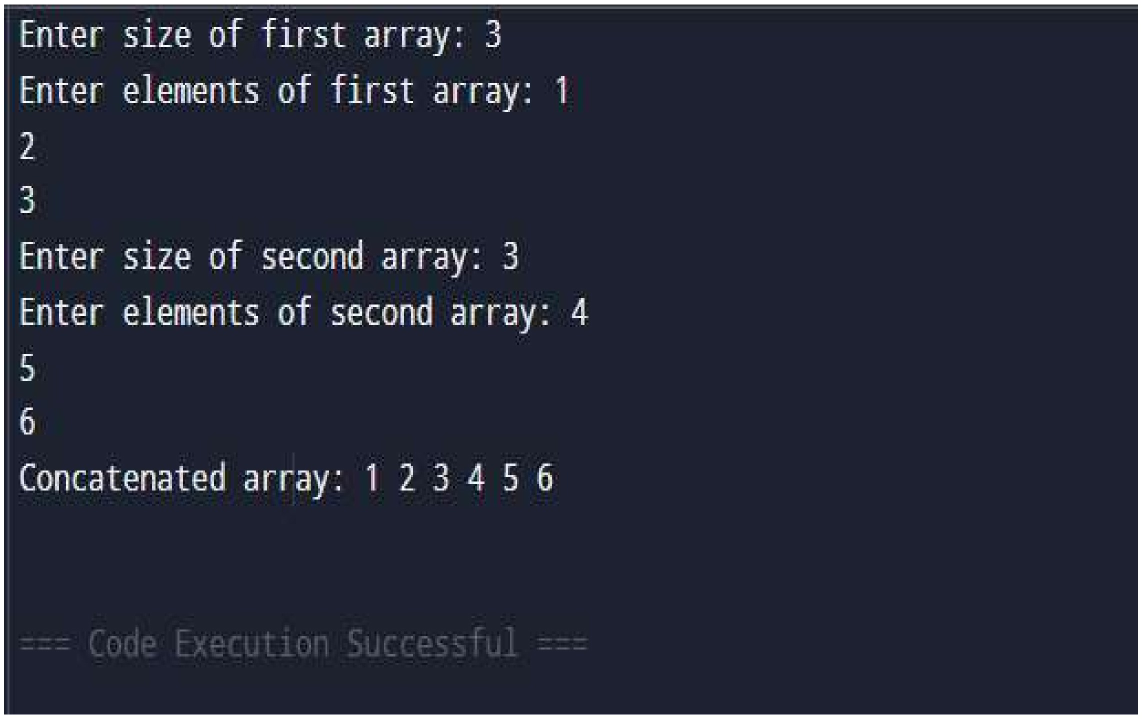
```
Enter the size of the array: 4
Enter the elements of the array: 1
2
3
4
Enter the position of the element to delete: 2
Updated array after deletion: 1 3 4


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

3. Write a program in C to concatenate two arrays?

INPUT:

```
#include <stdio.h>
int main() {
    int a[10], b[10], c[20];
    int size1, size2, i, j;
     printf("Enter size of first array: ");
    scanf("%d", &size1);
    printf("Enter elements of first array: ");
    for (i = 0; i < size1; i++) {
        scanf("%d", &a[i]);
    }
    printf("Enter size of second array: ");
    scanf("%d", &size2);
     printf("Enter elements of second array: ");
    for (i = 0; i < size2; i++) {
        scanf("%d", &b[i]);
    }
    for (i = 0; i < size1; i++) {
        c[i] = a[i];
    }
    for (j = 0; j < size2; j++) {
        c[size1 + j] = b[j];
    }
    printf("Concatenated array: ");
    for (i = 0; i < size1 + size2; i++) {
        printf("%d ", c[i]);
    }
  printf("\n");
    return 0;
}
```

<u>Output:</u>



```
Enter size of first array: 3
Enter elements of first array: 1
2
3
Enter size of second array: 3
Enter elements of second array: 4
5
6
Concatenated array: 1 2 3 4 5 6


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

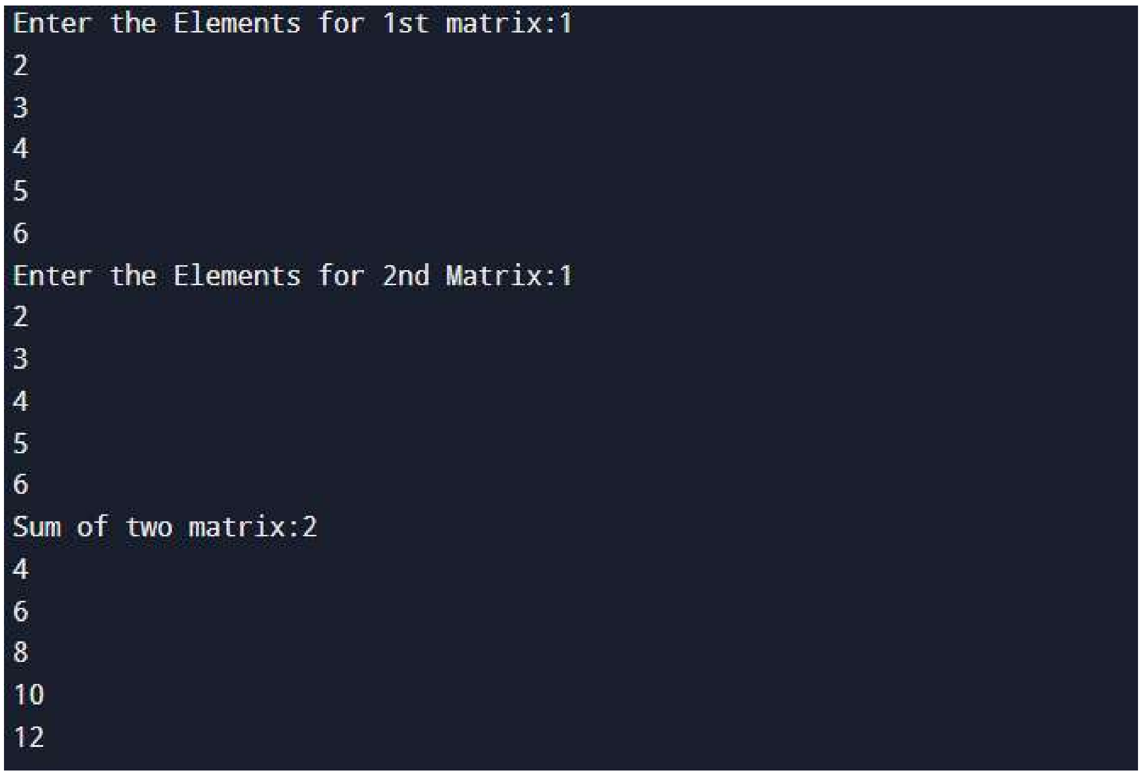
4. Write a program in C to implement the following Operations on 2-D Array (addition; subtraction; multiplication; transpose)?

1. Addition:

Input:

```
#include <stdio.h>
void main() {
int a[2][3],b[2][3],i,j,c[2][3];
    printf("Enter the Elements for 1st matrix:");
    for(i=0;i<2;i++){
        for(j=0;j<3;j++){
            scanf("%d", &a[i][j]);
        }
     }
     printf("Enter the Elements for 2nd Matrix:");
     for(i=0;i<2;i++){
        for(j=0;j<3;j++){
            scanf("%d", &b[i][j]);
        }
     }
    for(i=0;i<2;i++){
        for(j=0;j<3;j++){
            c[i][j] = a[i][j] + b[i][j];
        }
    }
    printf("Sum of two matrix:");
    for(i=0;i<2;i++){
        for(j=0;j<3;j++){
            printf("%d", c[i][j]);
            printf("\n");
        }
    }
}
```

Output:



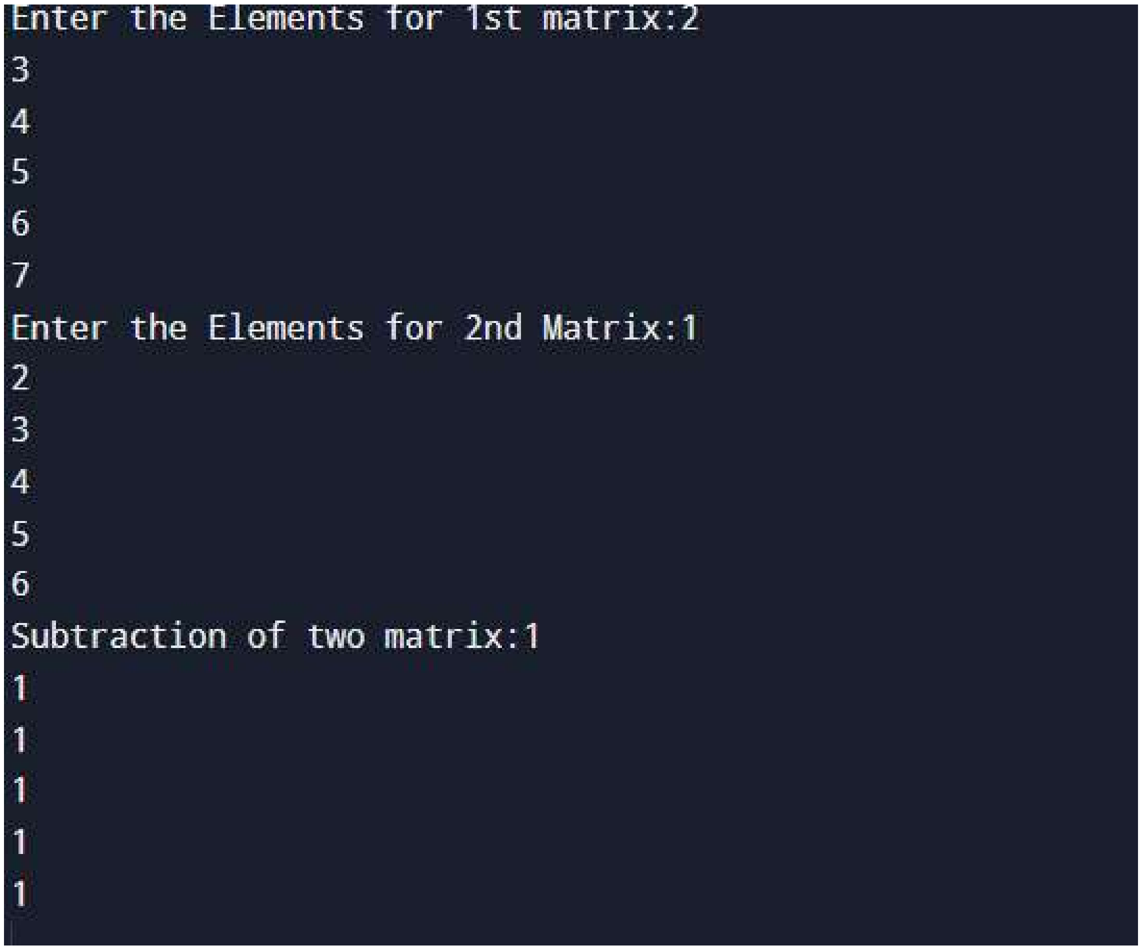
```
Enter the Elements for 1st matrix:1
2
3
4
5
6
Enter the Elements for 2nd Matrix:1
2
3
4
5
6
Sum of two matrix:2
4
6
8
10
12
```

2. Subtraction:

Input:

```
#include <stdio.h>
void main() {
int a[2][3],b[2][3],i,j,c[2][3];
    printf("Enter the Elements for 1st matrix:");
    for(i=0;i<2;i++){
        for(j=0;j<3;j++){
            scanf("%d", &a[i][j]);
        }
     }
     printf("Enter the Elements for 2nd Matrix:");
     for(i=0;i<2;i++){
        for(j=0;j<3;j++){
            scanf("%d", &b[i][j]);
        }
     }
    for(i=0;i<2;i++){
        for(j=0;j<3;j++){
            c[i][j] = a[i][j] - b[i][j];
        }
    }
    printf("Subtraction of two matrix:");
    for(i=0;i<2;i++){
        for(j=0;j<3;j++){
            printf("%d", c[i][j]);
            printf("\n");
        }
    }
}
```

Output:



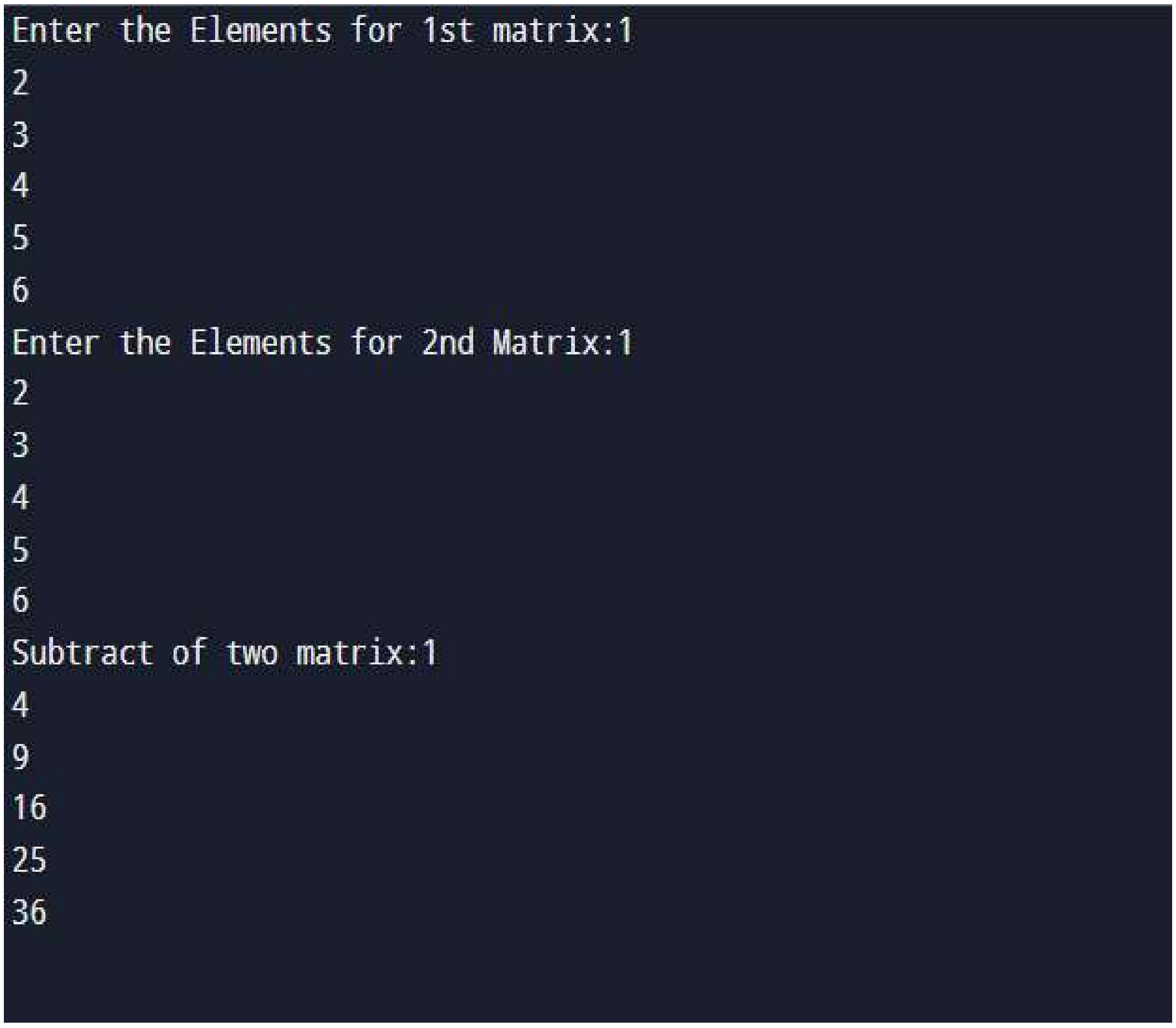
```
Enter the Elements for 1st matrix:2
3
4
5
6
7
Enter the Elements for 2nd Matrix:1
2
3
4
5
6
Subtraction of two matrix:1
1
1
1
1
1
```

3.Multiplication

Input:

```
#include <stdio.h>
void main() {
int a[2][3],b[2][3],i,j,c[2][3];
    printf("Enter the Elements for 1st matrix:");
    for(i=0;i<2;i++){
        for(j=0;j<3;j++){
            scanf("%d", &a[i][j]);
        }
     }
     printf("Enter the Elements for 2nd Matrix:");
     for(i=0;i<2;i++){
        for(j=0;j<3;j++){
            scanf("%d", &b[i][j]);
        }
     }
    for(i=0;i<2;i++){
        for(j=0;j<3;j++){
            c[i][j] = a[i][j] * b[i][j];
        }
    }
    printf("Subtract of two matrix:");
    for(i=0;i<2;i++){
        for(j=0;j<3;j++){
            printf("%d", c[i][j]);
            printf("\n");
        }
    }
}
```

Output:



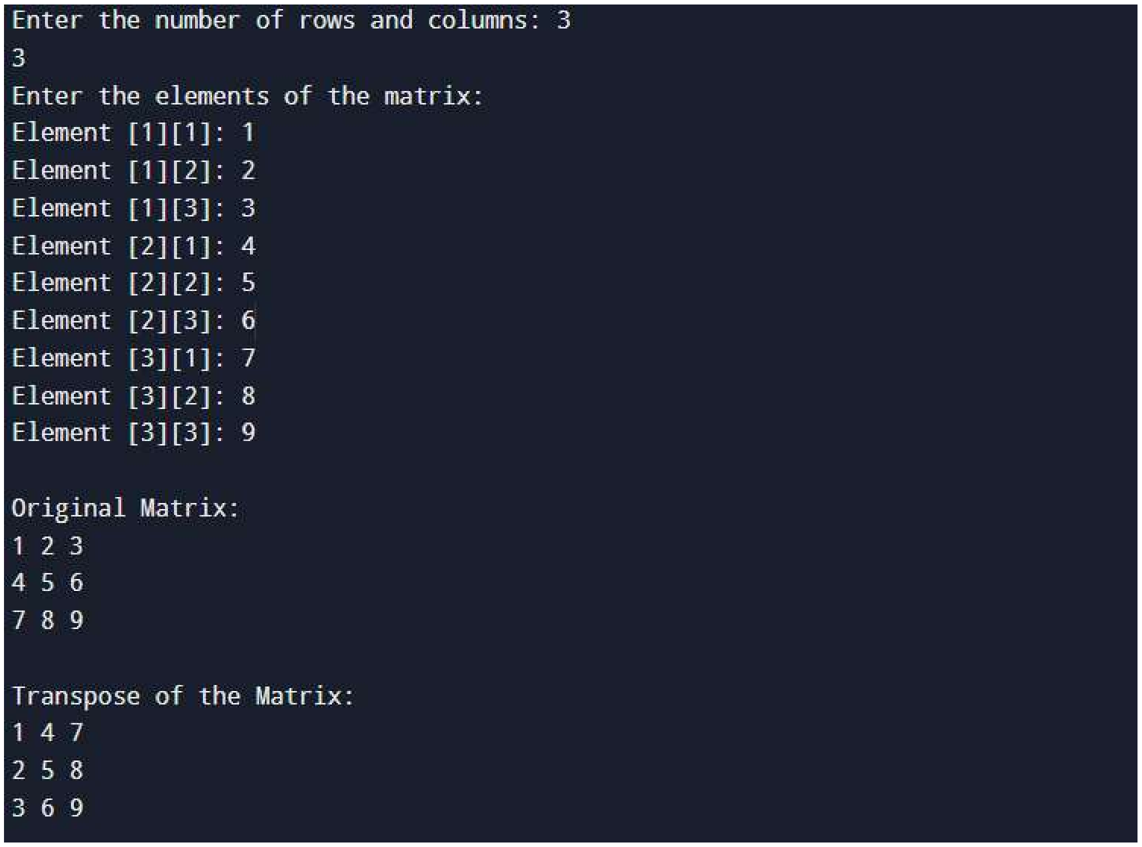
```
Enter the Elements for 1st matrix:1
2
3
4
5
6
Enter the Elements for 2nd Matrix:1
2
3
4
5
6
Subtract of two matrix:1
4
9
16
25
36
```

4.Transpose

Input:

```
#include <stdio.h>
        int main() {
    int matrix[10][10], transpose[10][10];
    int row, col, i, j;
        printf("Enter the number of rows and columns: ");
    scanf("%d %d", &row, &col);
printf("Enter the elements of the matrix:\n");
    for (i = 0; i < row; i++) {
        for (j = 0; j < col; j++) {
            printf("Element [%d][%d]: ", i + 1, j + 1);
            scanf("%d", &matrix[i][j]);
        }
    }
    for (i = 0; i < row; i++) {
        for (j = 0; j < col; j++) {
            transpose[j][i] = matrix[i][j];
        }
    }
   printf("\nOriginal Matrix:\n");
    for (i = 0; i < row; i++) {
        for (j = 0; j < col; j++) {
            printf("%d ", matrix[i][j]);
        }
        printf("\n");
    }
   printf("\nTranspose of the Matrix:\n");
    for (i = 0; i < col; i++) {
        for (j = 0; j < row; j++) {
            printf("%d ", transpose[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

Output:



```
Enter the number of rows and columns: 3
3
Enter the elements of the matrix:
Element [1][1]: 1
Element [1][2]: 2
Element [1][3]: 3
Element [2][1]: 4
Element [2][2]: 5
Element [2][3]: 6
Element [3][1]: 7
Element [3][2]: 8
Element [3][3]: 9

Original Matrix:
1 2 3
4 5 6
7 8 9

Transpose of the Matrix:
1 4 7
2 5 8
3 6 9
```

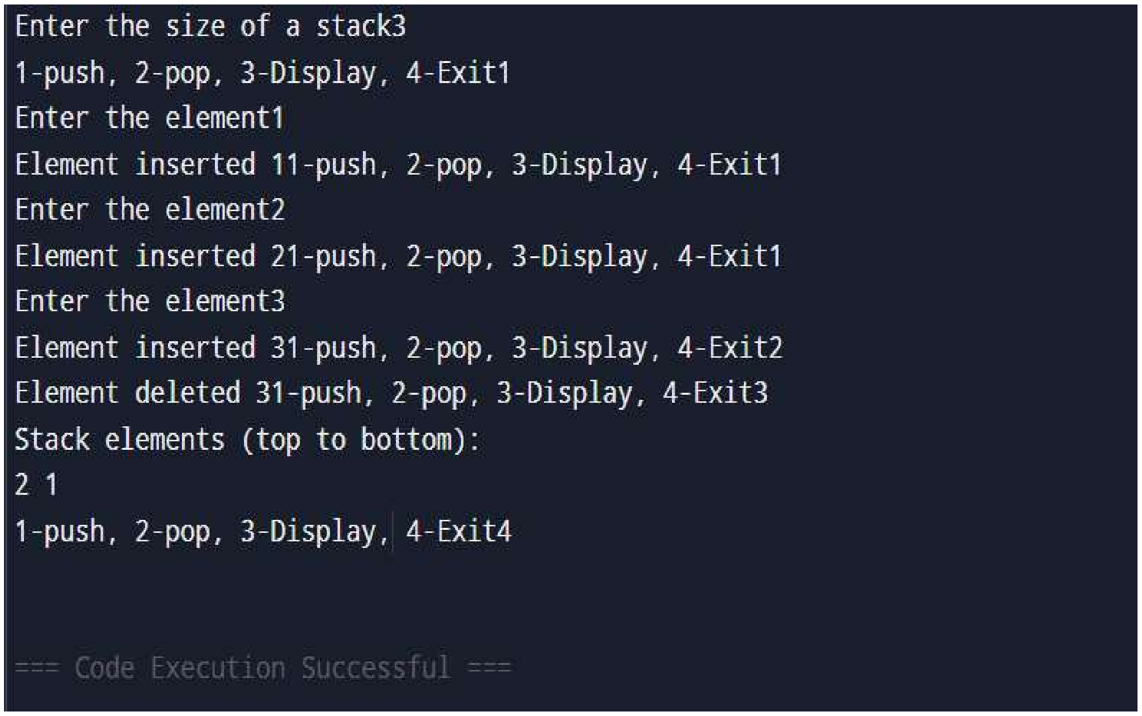
5.Write a program in C to implement operations on Stack using array ?

Input:

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

Output:



```
Enter the size of a stack3
1-push, 2-pop, 3-Display, 4-Exit1
Enter the element1
Element inserted 11-push, 2-pop, 3-Display, 4-Exit1
Enter the element2
Element inserted 21-push, 2-pop, 3-Display, 4-Exit1
Enter the element3
Element inserted 31-push, 2-pop, 3-Display, 4-Exit2
Element deleted 31-push, 2-pop, 3-Display, 4-Exit3
Stack elements (top to bottom):
2 1
1-push, 2-pop, 3-Display, 4-Exit4


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

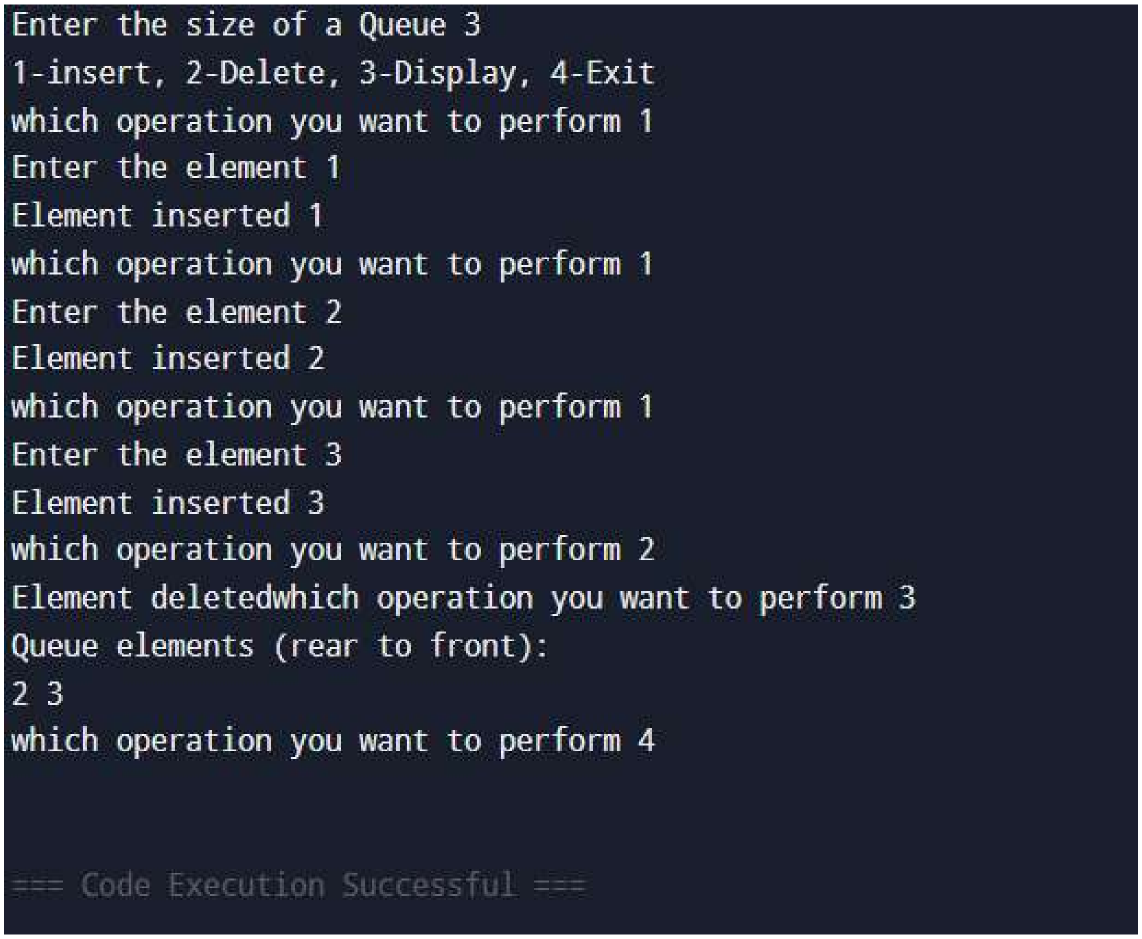
6. Write a program in C to implement operations on a queue using an array?

Input:

```
 #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);
    printf("1-insert, 2-Delete, 3-Display, 4-Exit\n");
    while(1) {
    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(0);
    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\n", 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 elements (rear to front):\n");
        for(i = f; i <= r; i++)
            printf("%d ", Queue[i]);
        printf("\n");
}
}
```

OUTPUT:



```
Enter the size of a Queue 3
1-insert, 2-Delete, 3-Display, 4-Exit
which operation you want to perform 1
Enter the element 1
Element inserted 1
which operation you want to perform 1
Enter the element 2
Element inserted 2
which operation you want to perform 1
Enter the element 3
Element inserted 3
which operation you want to perform 2
Element deletedwhich operation you want to perform 3
Queue elements (rear to front):
2 3
which operation you want to perform 4


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

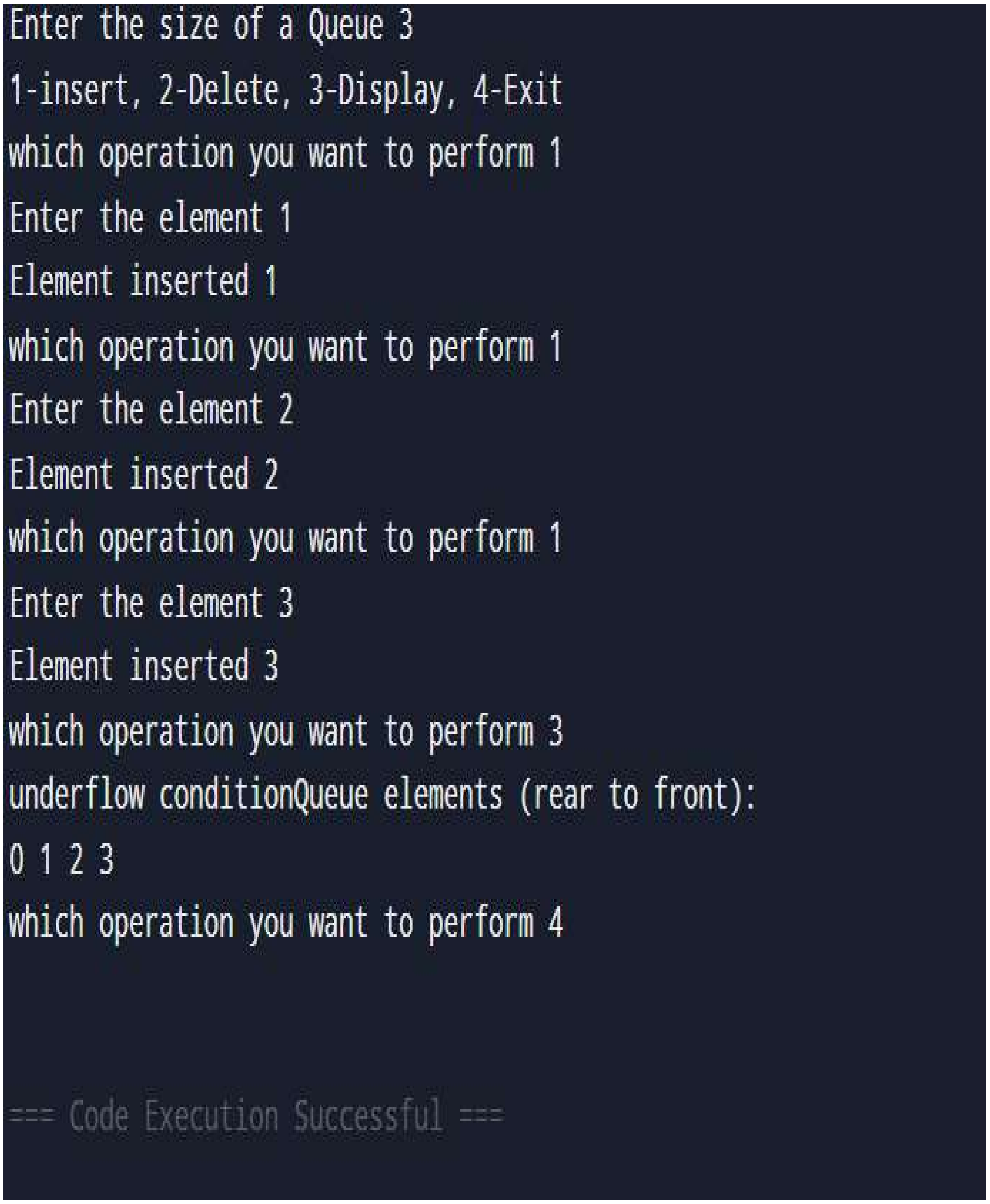
7. Write a program in C to implement operations on a circular queue using an array?

Input:

```
#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);

    printf("1-insert, 2-Delete, 3-Display, 4-Exit\n");

    while(1) {

    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(0);
    break;
    default:
    printf("Wrong choice");
     }
  }
}
void insert()
{
int ele;

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

```
}else
{

    r = r + 1;}

    Queue[r] = ele;
    printf("Element inserted %d\n", 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");
    }
     if(f<=r) {
    printf("Queue elements (rear to front):\n");
        for(i = f; i <= r; i++)
            printf("%d ", Queue[i]);
        printf("\n");
}

     else {
    for (i=f;i<=size-1;i++){
        printf("%d", &Queue[i]);
        for(i=0;i<=r;i++){
            printf("%d",Queue[i]);
        }
    }
}
}
```

OUTPUT:



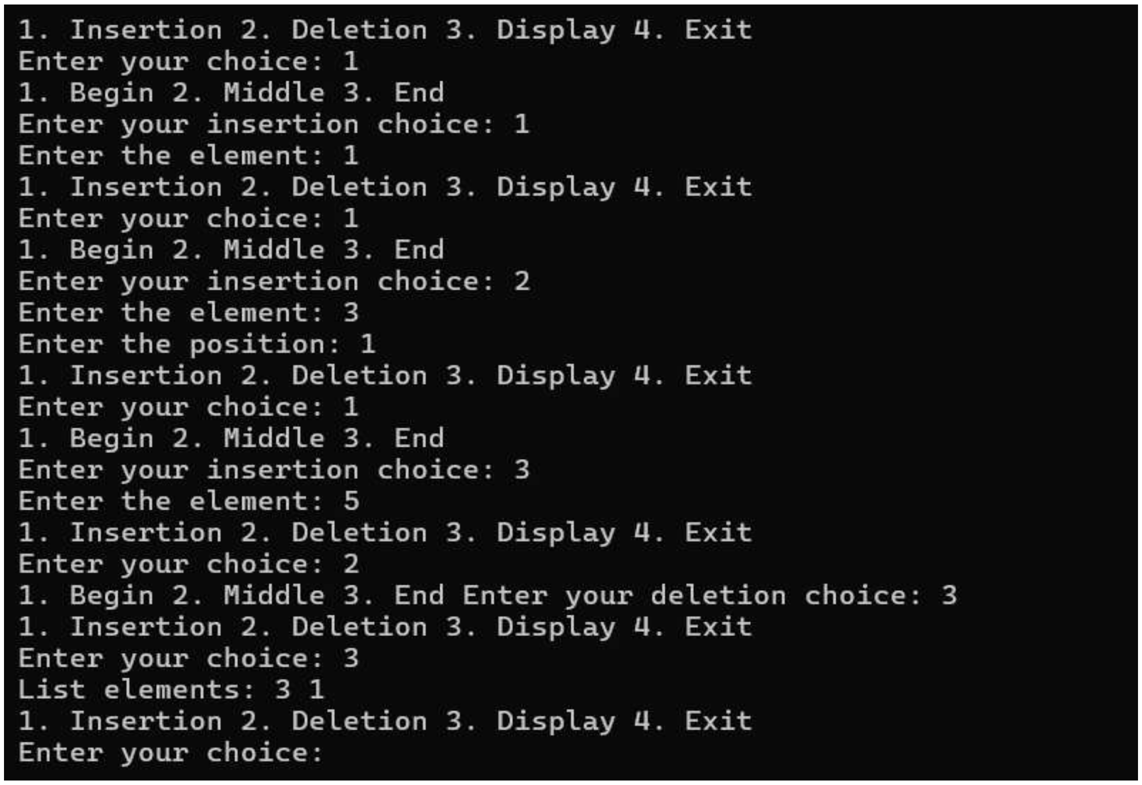
```
Enter the size of a Queue 3
1-insert, 2-Delete, 3-Display, 4-Exit
which operation you want to perform 1
Enter the element 1
Element inserted 1
which operation you want to perform 1
Enter the element 2
Element inserted 2
which operation you want to perform 1
Enter the element 3
Element inserted 3
which operation you want to perform 3
underflow conditionQueue elements (rear to front):
0 1 2 3
which operation you want to perform 4



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

Q-8,9. Write a program in C to implement insertion and deletion in a linked list(beg; mid; end)?

```
#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: 1
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: 3
Enter the position: 1
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: 5
1. Insertion 2. Deletion 3. Display 4. Exit
Enter your choice: 2
1. Begin 2. Middle 3. End Enter your deletion choice: 3
1. Insertion 2. Deletion 3. Display 4. Exit
Enter your choice: 3
List elements: 3 1
1. Insertion 2. Deletion 3. Display 4. Exit
Enter your choice:
```