|  |
| --- |
| INDEX |
| | **Topic No.** | **Topic** | **Description** | | --- | --- | --- | | **1** | Insertion in 1-D Arrays | Inserting an element at a specific position in a 1-D array. | | **2** | Deletion in 1-D Arrays | Deleting an element from a specific position in a 1-D array. | | **3** | Concatenate Two Arrays | Concatenating two 1-D arrays into a merged array. | | **4** | Operations on 2-D Arrays | Performing various operations on 2-D arrays: | |  |  | - Addition: Adding two matrices. | |  |  | - Subtraction: Subtracting one matrix from another. | |  |  | - Multiplication: Multiplying two matrices. | |  |  | - Transpose: Transposing a matrix (rows become columns and vice versa). | | **5** | Operations on Stack using Array | Implementing push, pop, and display operations on a stack using an array. | | **6** | Operations on Queue using Array | Implementing insert, delete, and display operations on a queue using an array. | | **7**  8. | Operations on Circular Queue using Array | Implementing insert, delete, and display operations on a circular queue using an array. | | **9.** |  |  | | **10.** |  |  | | **11.** |  |  | | **12.** |  |  | | **13.** |  |  | | **14.** |  |  | | **15.** |  |  | |

**SIGN-**

**Q1) Write a program in C to implement insertion in 1-D Arrays?**

#include <stdio.h>

int main() {

int arr[10],size,element,pos,i;

printf("Enter the number of elements in the array");

scanf ("%d", &size);

printf("Enter all the elements of array:");

for(i=0;i<size;i++){

scanf("%d", &arr[i]);

}

printf("Enter the position where new element has to be inserted:");

scanf("%d", & pos);

printf("Enter the new element:");

scanf("%d", & element);

for(i=size-1; i >= pos-1;i--){

arr[i+1] = arr[i];

}

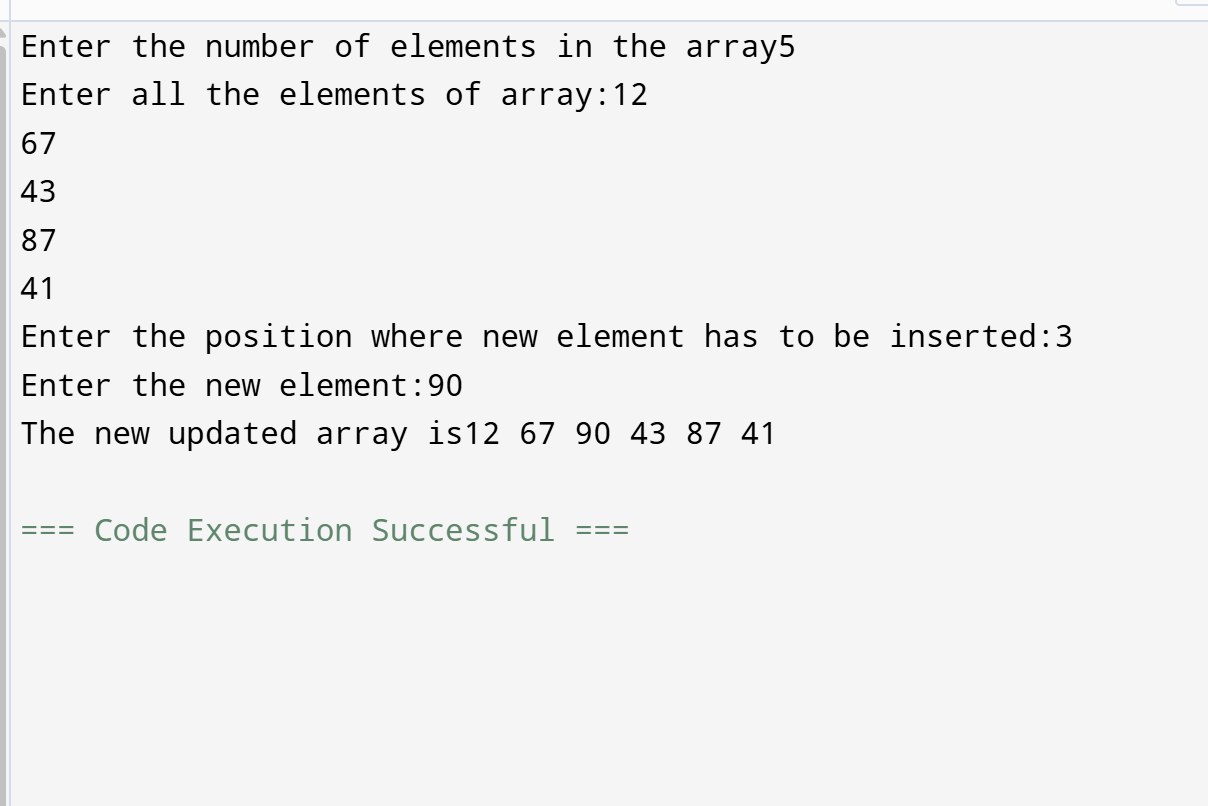
arr[pos-1] = element;

printf("The new updated array after inserting the new element");

for(i=0;i<=size;i++)

printf("%d ", arr[i]);

return 0;

}

**Q2) Write a program in C to implement deletion in 1-D Arrays ?**

#include <stdio.h>

int main() {

int a[7], size, i, position;

printf("Enter the number of elements in the array: ");

scanf("%d", &size);

printf("Enter all the elements of array: ");

for (i = 0; i < size; i++) {

scanf("%d", &a[i]);

}

printf("Enter the position which has to be removed from the array: ");

scanf("%d", &position);

if (position < 1 || position > size) {

printf("Incorrect position\n");

return 1;

}

for (i = position - 1; i < size - 1; i++) {

a[i] = a[i + 1];

} size--;

printf("The new updated array after deleting the specified element: ");

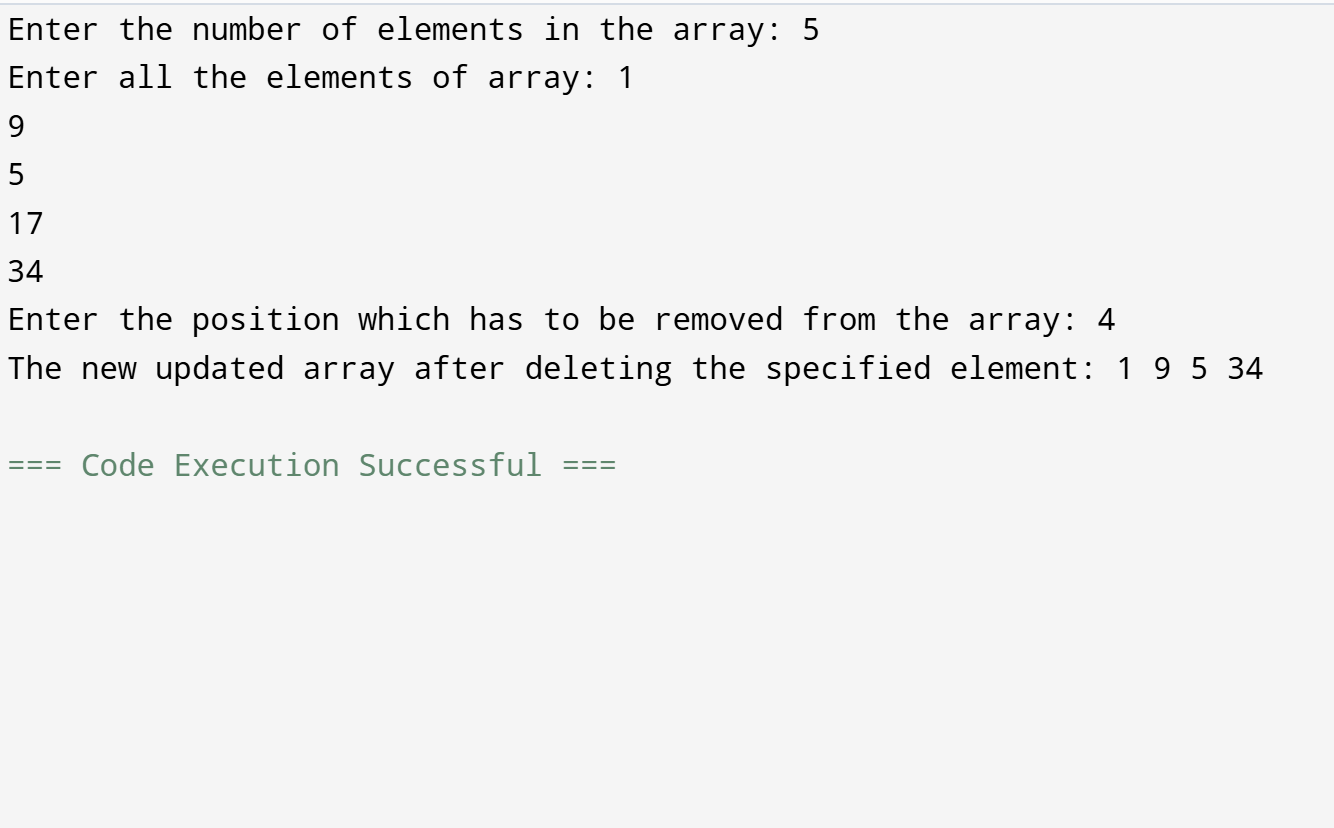
for (i = 0; i < size; i++) {

printf("%d ", a[i]);

}

return 0;

}



**Q3) Write a program in C to concatenate two arrays?**

#include <stdio.h>

int main() {

int a[50], b[50], merged[100];

int n1, n2, i, j;

printf("Enter the number of elements of the first array: ");

scanf("%d", &n1);

printf("Enter all the elements of array: ");

for(i = 0; i < n1; i++) {

scanf("%d", &a[i]);

}

printf("Enter the number of elements of the second array ");

scanf("%d", &n2);

printf("Enter all the elements of array ");

for(i = 0; i < n2; i++) {

scanf("%d", &b[i]);

}

for(i = 0; i < n1; i++) {

merged[i] = a[i];

}

for(j = 0; j < n2; j++) {

merged[i] = b[j];

i++;

}

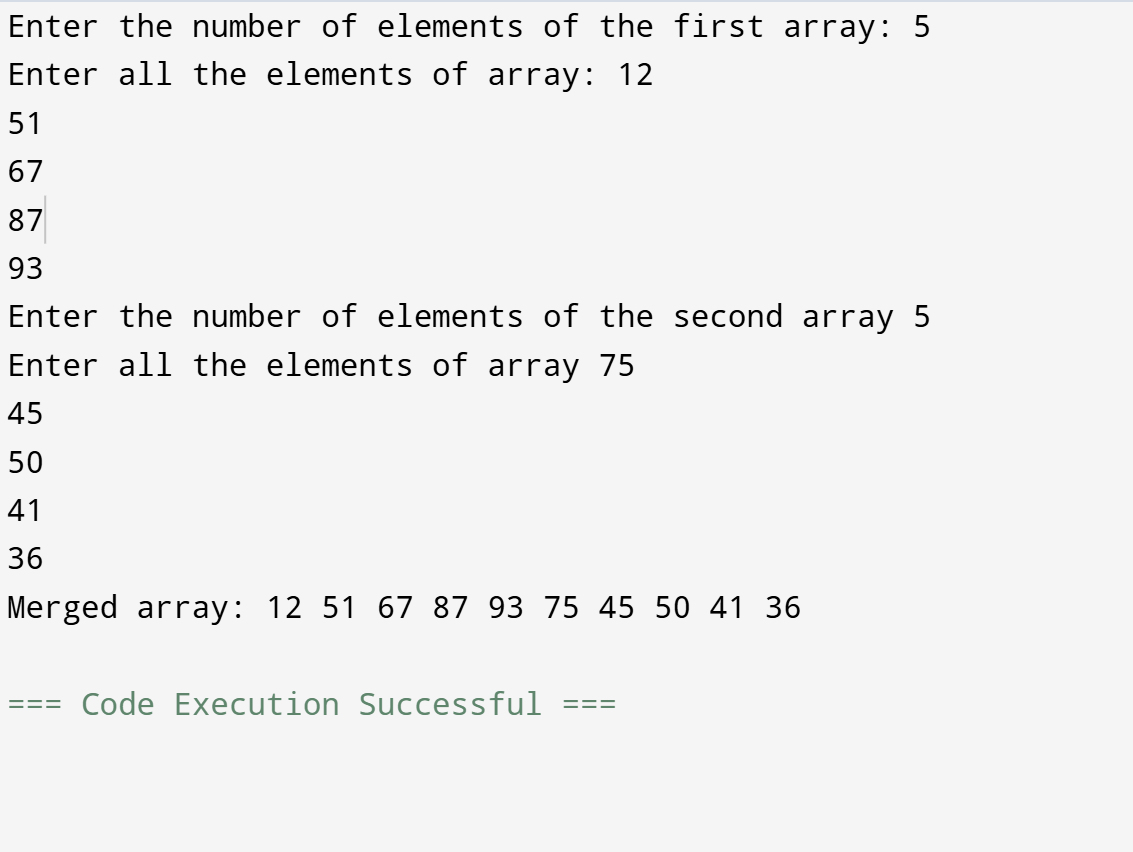
printf("Merged array: ");

for(i = 0; i < n1 + n2; i++) {

printf("%d ", merged[i]);

}

return 0;



**Q4) . Write a program in C to implement the following Operations on 2-D Array (addition; subtraction; multiplication; transpose) ?**

ADDITION

#include <stdio.h>

void main() {

int arr1[2][2], arr2[2][2], arr3[2][2], i, j;

printf("Input the elements of the first matrix:\n");

for(i = 0; i < 2; i++) {

for(j = 0; j < 2; j++) {

scanf("%d", &arr1[i][j]);

}

}

printf("Input the elements of the second matrix:\n");

for(i = 0; i < 2; i++) {

for(j = 0; j < 2; j++) {

scanf("%d", &arr2[i][j]);

}

}

for(i = 0; i < 2; i++) {

for(j = 0; j < 2; j++) {

arr3[i][j] = arr1[i][j] + arr2[i][j];

}

}

printf("Addition of both matrices:\n");

for(i = 0; i < 2; i++) {

for(j = 0; j < 2; j++) {

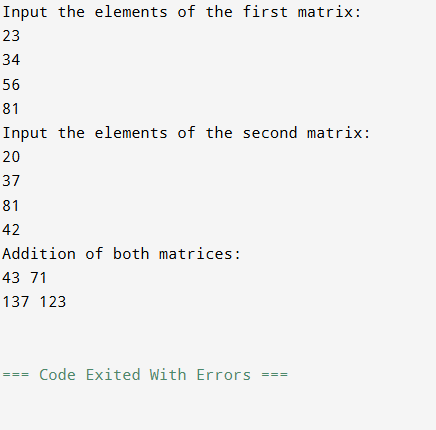
printf("%d ", arr3[i][j]);

}

printf("\n");

}

}



SUBTRACTION

#include <stdio.h>

void main() {

int arr1[3][3], arr2[3][3], arr3[3][3], i, j;

printf("Input the elements of the first matrix:\n");

for(i = 0; i < 2; i++) {

for(j = 0; j < 3; j++) {

scanf("%d", &arr1[i][j]);

}

}

printf("Input the elements of the second matrix:\n");

for(i = 0; i < 2; i++) {

for(j = 0; j < 3; j++) {

scanf("%d", &arr2[i][j]);

}

}

for(i = 0; i < 2; i++) {

for(j = 0; j < 3; j++) {

arr3[i][j] = arr1[i][j] - arr2[i][j];

}

}

printf("Subtraction of both matrices:\n");

for(i = 0; i < 2; i++) {

for(j = 0; j < 3; j++) {

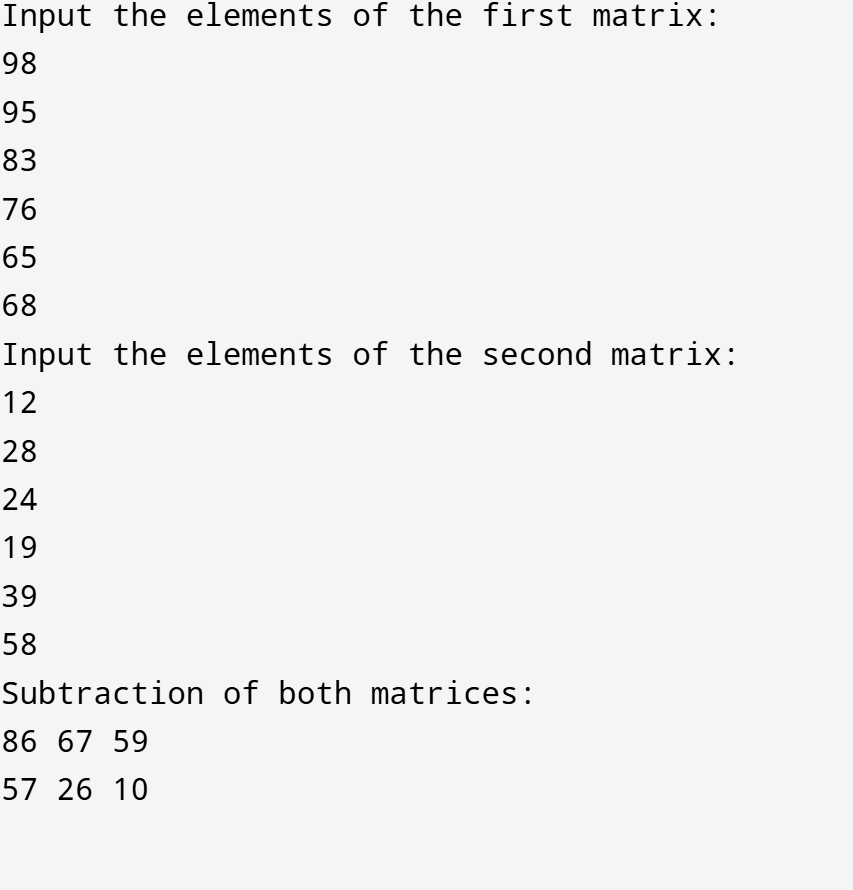
printf("%d ", arr3[i][j]);

}

printf("\n");

}

}



MULTIPLICATION

#include <stdio.h>

int main() {

int mat1[10][10], mat2[10][10], product[10][10];

int row1, col1, row2, col2, i, j, k;

printf("Enter rows and columns for first matrix: ");

scanf("%d %d", &row1, &col1);

printf("Enter rows and columns for second matrix: ");

scanf("%d %d", &row2, &col2);

if (col1 != row2) {

printf("Multiplication not possible.\n");

return 0;

}

printf("Provide elements of first matrix:\n");

for(i = 0; i < row1; i++) {

for(j = 0; j < col1; j++) {

scanf("%d", &mat1[i][j]);

}

}

printf("Provide elements of second matrix:\n");

for(i = 0; i < row2; i++) {

for(j = 0; j < col2; j++) {

scanf("%d", &mat2[i][j]);

}

}

for(i = 0; i < row1; i++) {

for(j = 0; j < col2; j++) {

product[i][j] = 0;

for(k = 0; k < col1; k++) {

product[i][j] += mat1[i][k] \* mat2[k][j];

}

}

}

printf("Resultant matrix:\n");

for(i = 0; i < row1; i++) {

for(j = 0; j < col2; j++) {

printf("%d ", product[i][j]);

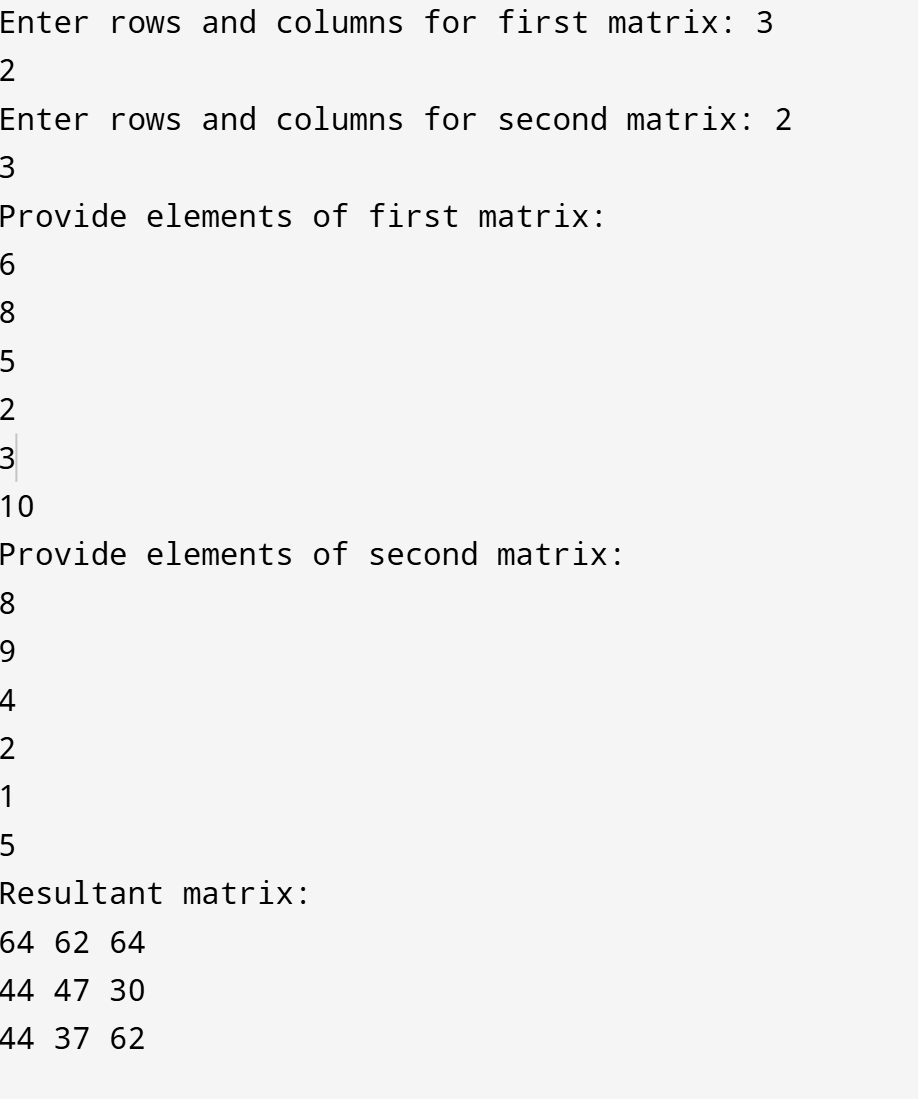
}

printf("\n");

}

return 0;

}



TRANSPOSE

#include <stdio.h>

int main() {

int r, c;

printf("Enter dimensions of matrix (rows cols): ");

scanf("%d %d", &r, &c);

int mat[r][c], trans[c][r];

printf("Fill the matrix:\n");

for(int i = 0; i < r; i++) {

for(int j = 0; j < c; j++) {

scanf("%d", &mat[i][j]);

}

}

printf("Matrix entered:\n");

for(int i = 0; i < r; i++) {

for(int j = 0; j < c; j++) {

printf("%d ", mat[i][j]);

}

printf("\n");

}

for(int i = 0; i < r; i++) {

for(int j = 0; j < c; j++) {

trans[j][i] = mat[i][j];

}

}

printf("Transposed matrix:\n");

for(int i = 0; i < c; i++) {

for(int j = 0; j < r; j++) {

printf("%d ", trans[i][j]);

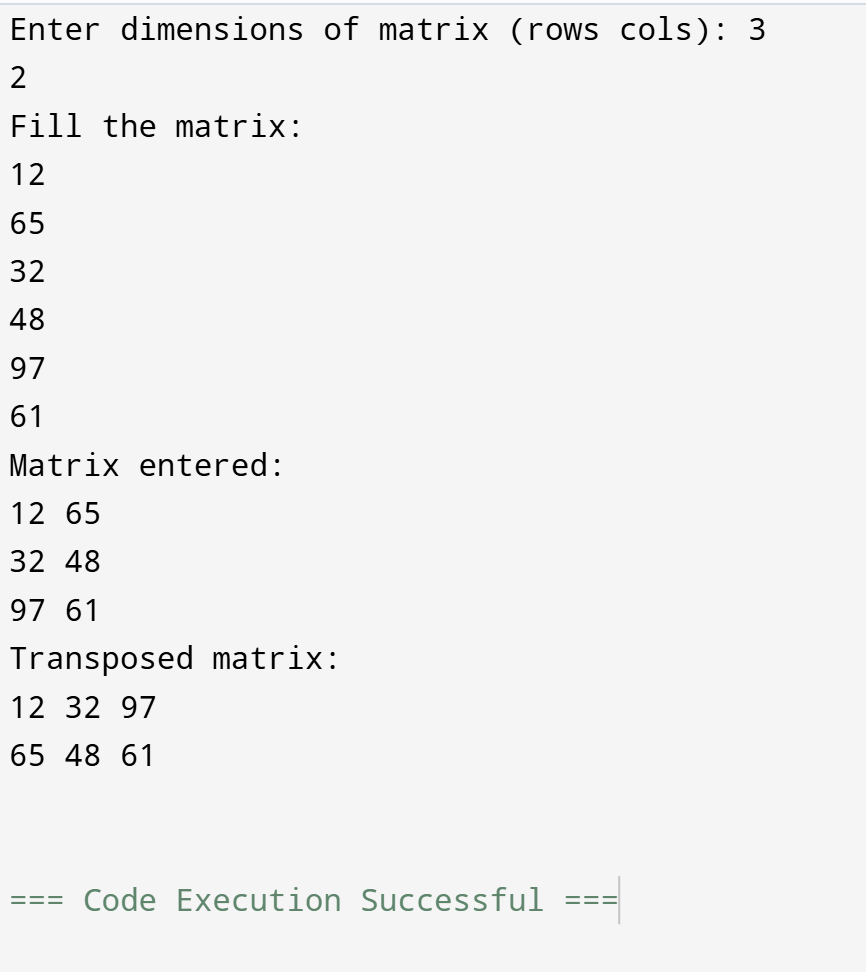
}

printf("\n");

}

return 0;

}



**Q5) Write a program in C to implement operations on Stack using array ?**

#include <stdio.h>

#include <stdlib.h>

void push();

void pop();

void display();

int maxstack, top = -1, stack[8];

void main() {

int choice;

printf("Enter the number of elements to be added in a stack: ");

scanf("%d", &maxstack);

while (1) {

printf("\n\n1: Push 2: Pop 3: Display 4: Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

push();

break;

case 2:

pop();

break;

case 3:

display();

break;

case 4:

exit(0);

break;

default:

printf("You chose an invalid option. Please try again.\n");

}

}

}

void push() {

int element;

if (top == maxstack - 1) {

printf("\nStack Overflow! Cannot insert more elements.\n");

} else {

printf("\nInput the new element: ");

scanf("%d", &element);

top = top + 1;

stack[top] = element;

printf("Element %d has been inserted successfully.\n", element);

}

}

void pop() {

if (top == -1) {

printf("\nStack Underflow! No elements to remove.\n");

} else {

printf("\nElement %d has been removed.\n", stack[top]);

top = top - 1;

}

}

void display() {

int i;

if (top == -1) {

printf("\nStack is empty! No elements to display.\n");

} else {

printf("\nStack elements are:\n");

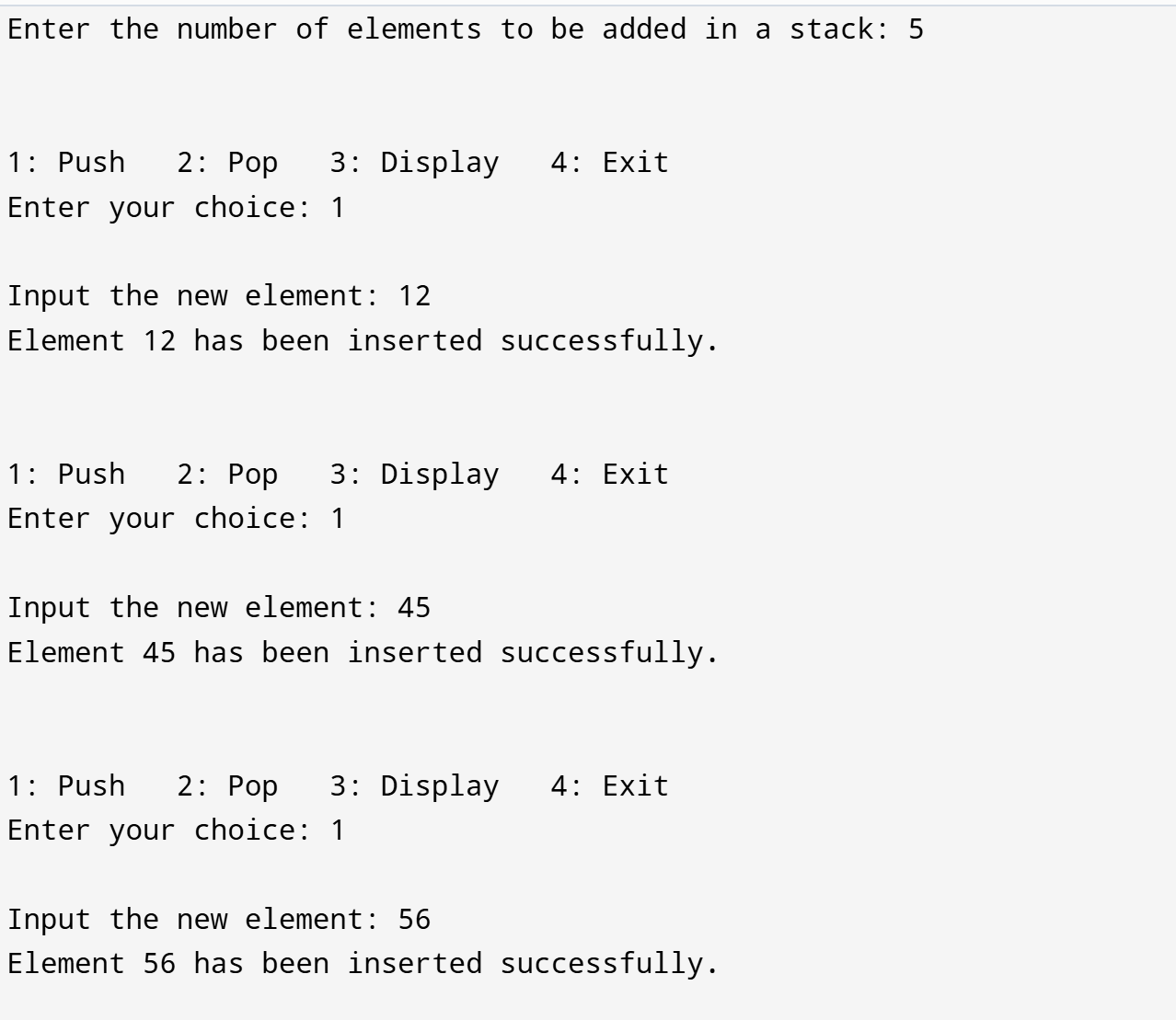
for (i = top; i >= 0; i--)

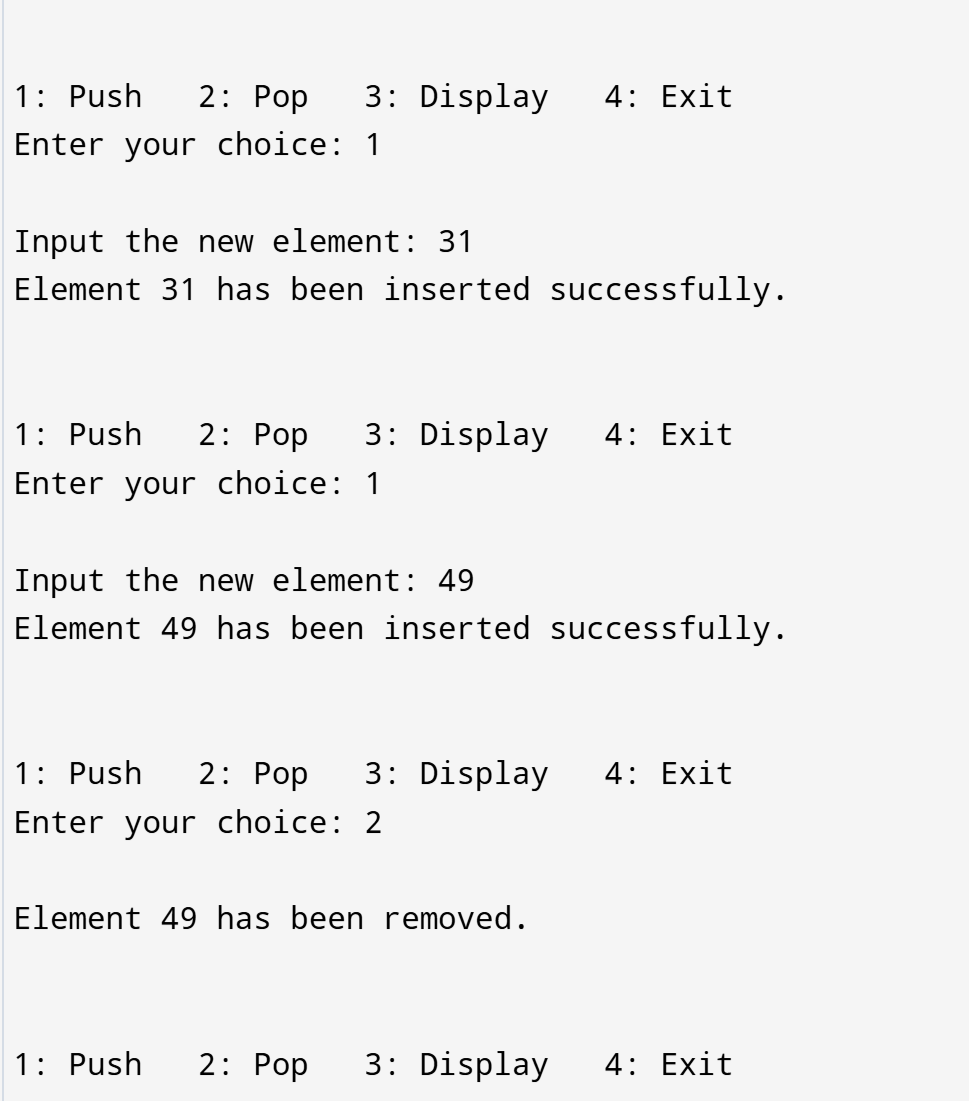
printf("%d ", stack[i]);

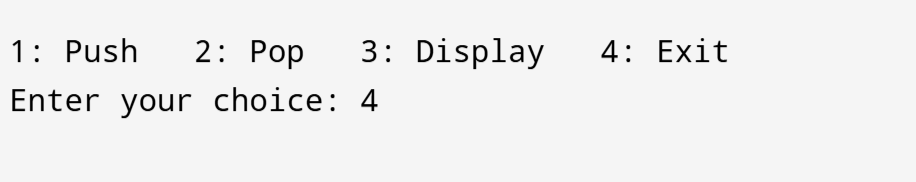
printf("\n");

}

}







**Q6) Write a program in C to implement operations on 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 the Queue: ");

scanf("%d", &size);

while (1) {

printf("\n1 - Insert 2 - Delete 3 - Display 4 - Exit\n");

printf("Enter your choice: ");

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! Please try again.\n");

}

}

}

void insert() {

int ele;

if (r == size - 1) {

printf("\nQueue Overflow! Cannot insert more elements.\n");

} else {

printf("\nEnter the element to insert: ");

scanf("%d", &ele);

if (f == -1 && r == -1) { // If queue is empty

f = r = 0;

} else {

r = r + 1;

}

Queue[r] = ele;

printf("Element %d has been inserted successfully.\n", ele);

}

}

void Delete() {

if (f == -1 && r == -1) {

printf("\nQueue Underflow! No elements to delete.\n");

} else {

printf("\nElement %d has been deleted.\n", Queue[f]);

if (f == r) { // If only one element was present, reset the queue

f = r = -1;

} else {

f = f + 1;

}

}

}

void display() {

int i;

if (f == -1) {

printf("\nQueue is empty! No elements to display.\n");

} else {

printf("\nQueue elements (Front to Rear):\n");

for (i = f; i <= r; i++) {

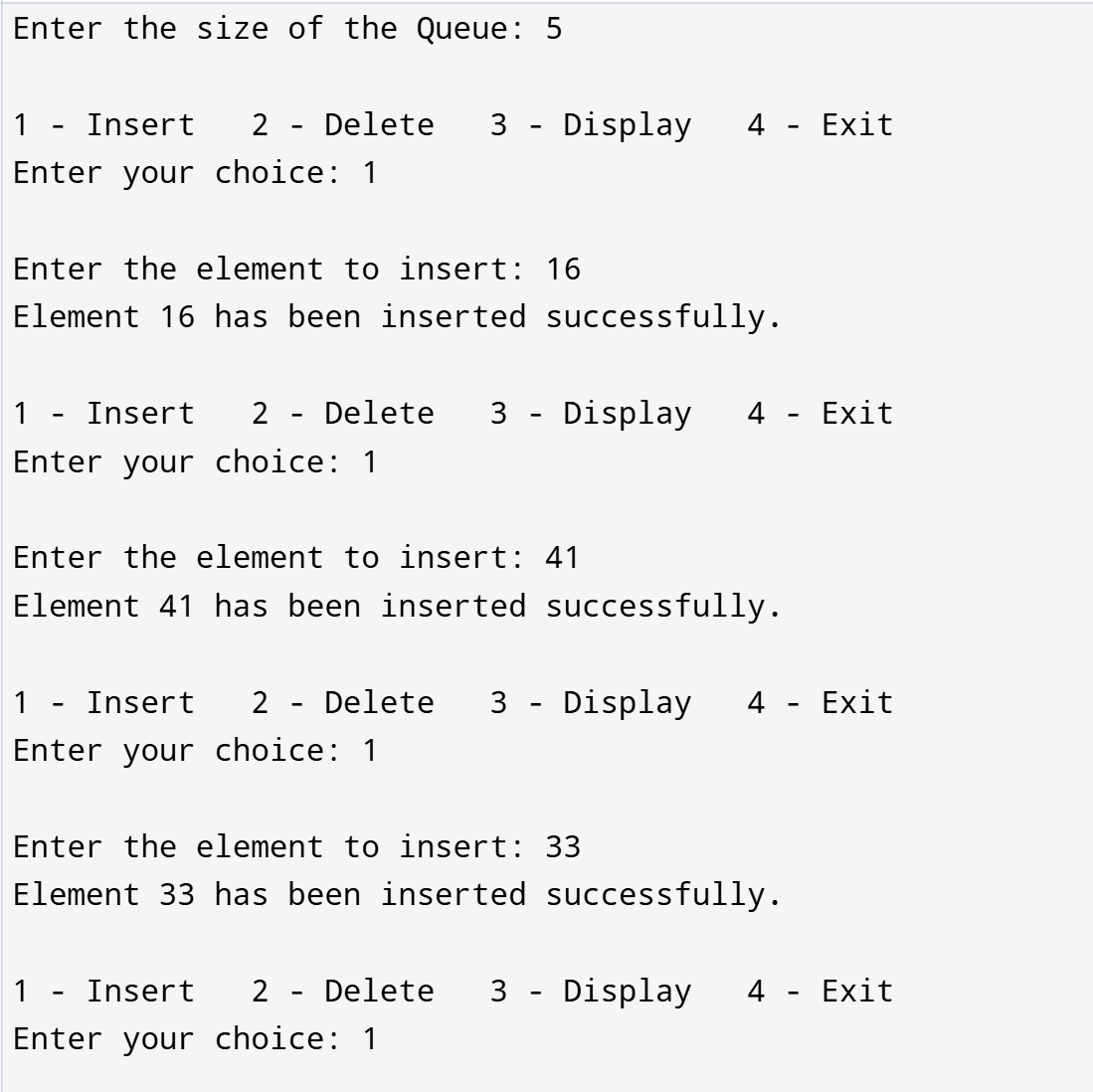
printf("%d ", Queue[i]);

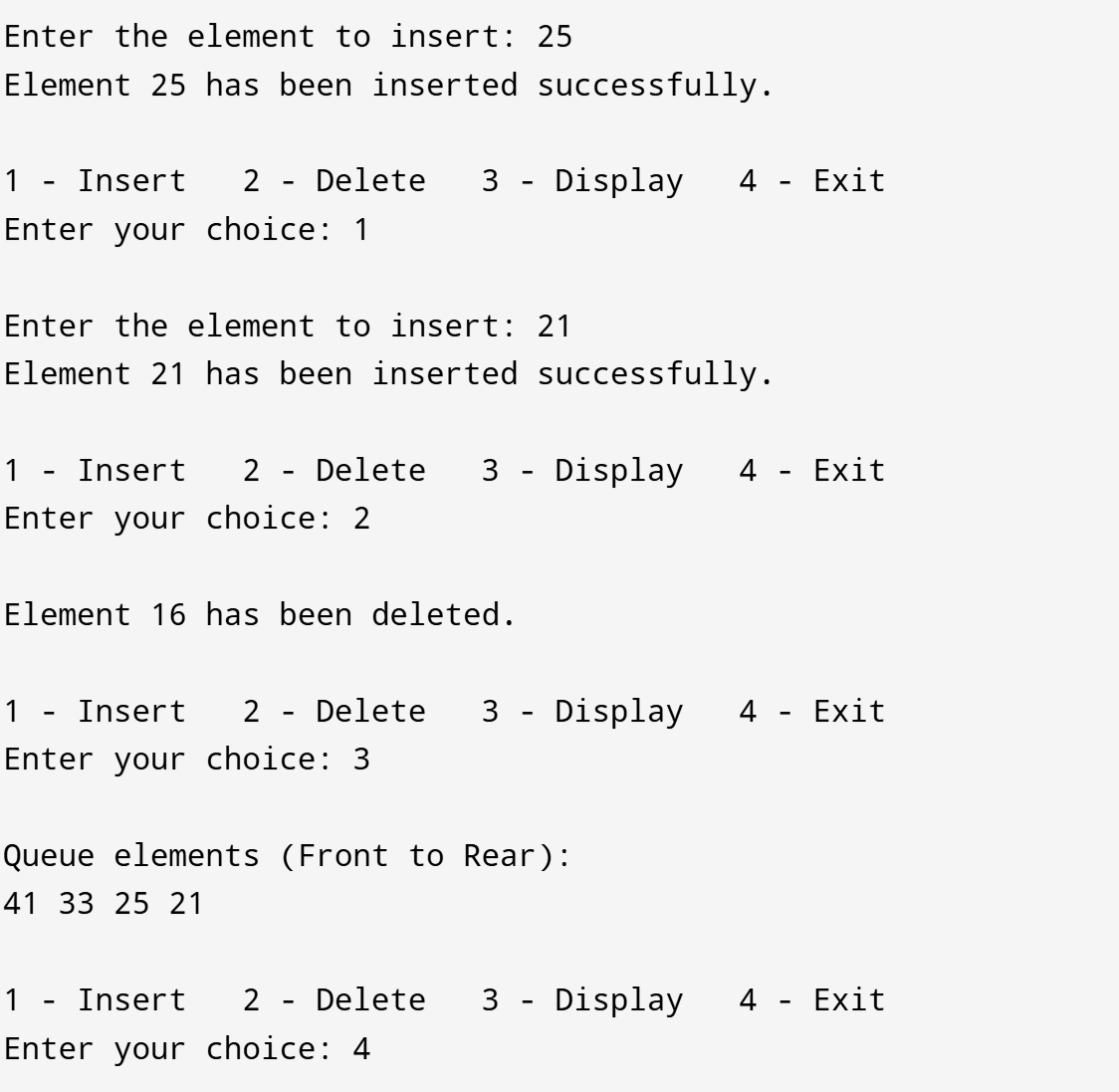
}

printf("\n");

}

}





**Q7) Write a program in C to implement operations on circular queue using array?**

#include <stdio.h>

#include <stdlib.h>

void insert();

void Delete();

void display();

int size, Queue[10], front = -1, rear = -1;

void main() {

int ch;

printf("Enter the size of the Queue: ");

scanf("%d", &size);

while (1) {

printf("\n1 - Insert 2 - Delete 3 - Display 4 - Exit\n");

printf("Enter your choice: ");

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! Please try again.\n");

}

}

}

void insert() {

int ele;

if ((front == 0 && rear == size - 1) || (front == rear + 1)) {

printf("\nQueue Overflow! Cannot insert more elements.\n");

} else {

printf("\nEnter the element to insert: ");

scanf("%d", &ele);

if (front == -1) {

front = rear = 0;

} else if (rear == size - 1) {

rear = 0;

} else {

rear++;

}

Queue[rear] = ele;

printf("Element %d has been inserted successfully.\n", ele);

}

}

void Delete() {

if (front == -1) {

printf("\nQueue Underflow! No elements to delete.\n");

} else {

printf("\nElement %d has been deleted.\n", Queue[front]);

if (front == rear) {

front = rear = -1;

} else if (front == size - 1) {

front = 0;

} else {

front++;

}

}

}

void display() {

int i;

if (front == -1) {

printf("\nQueue is empty! No elements to display.\n");

} else {

printf("\nQueue elements (Front to Rear):\n");

if (rear >= front) {

for (i = front; i <= rear; i++) {

printf("%d ", Queue[i]);

}

} else {

for (i = front; i < size; i++) {

printf("%d ", Queue[i]);

}

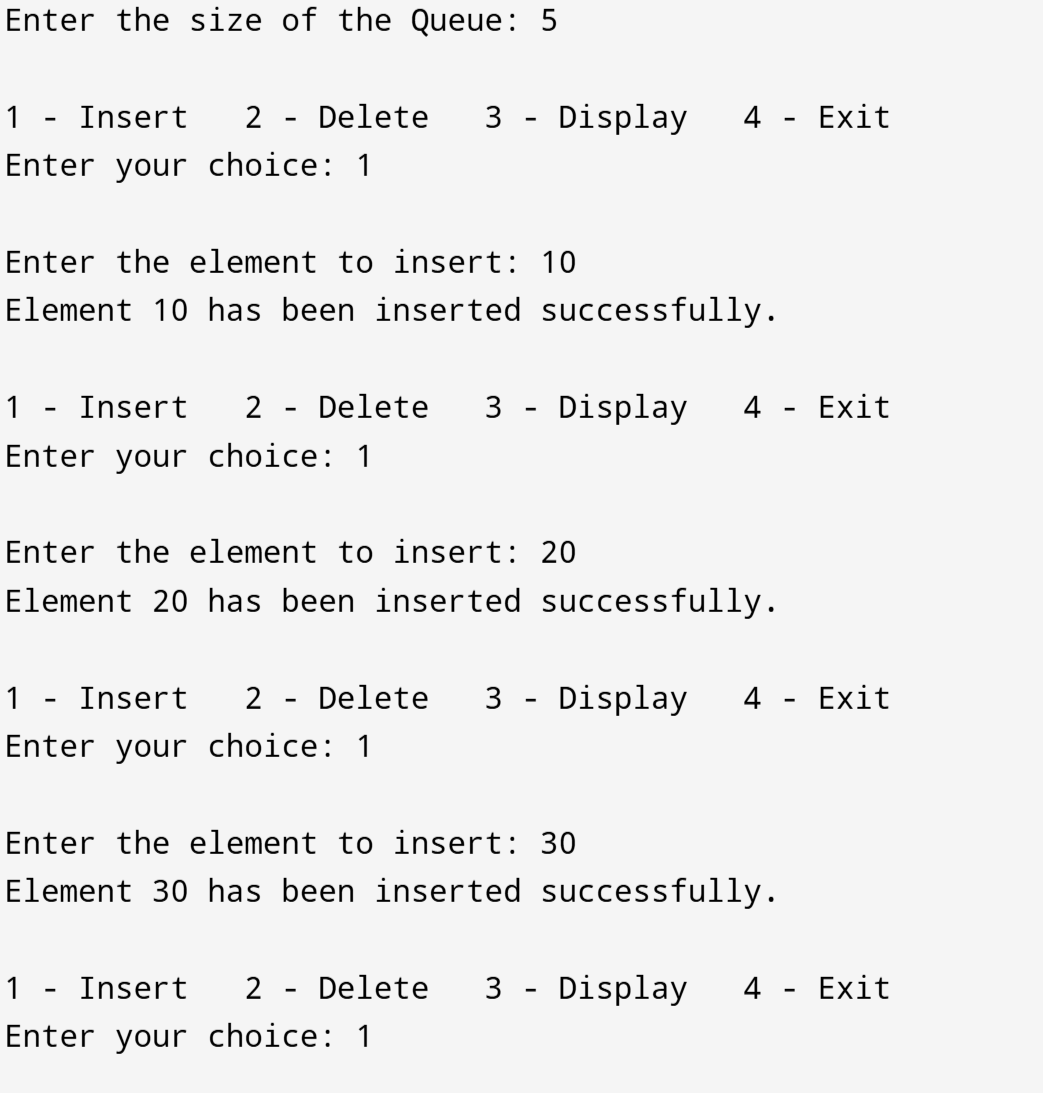
for (i = 0; i <= rear; i++) {

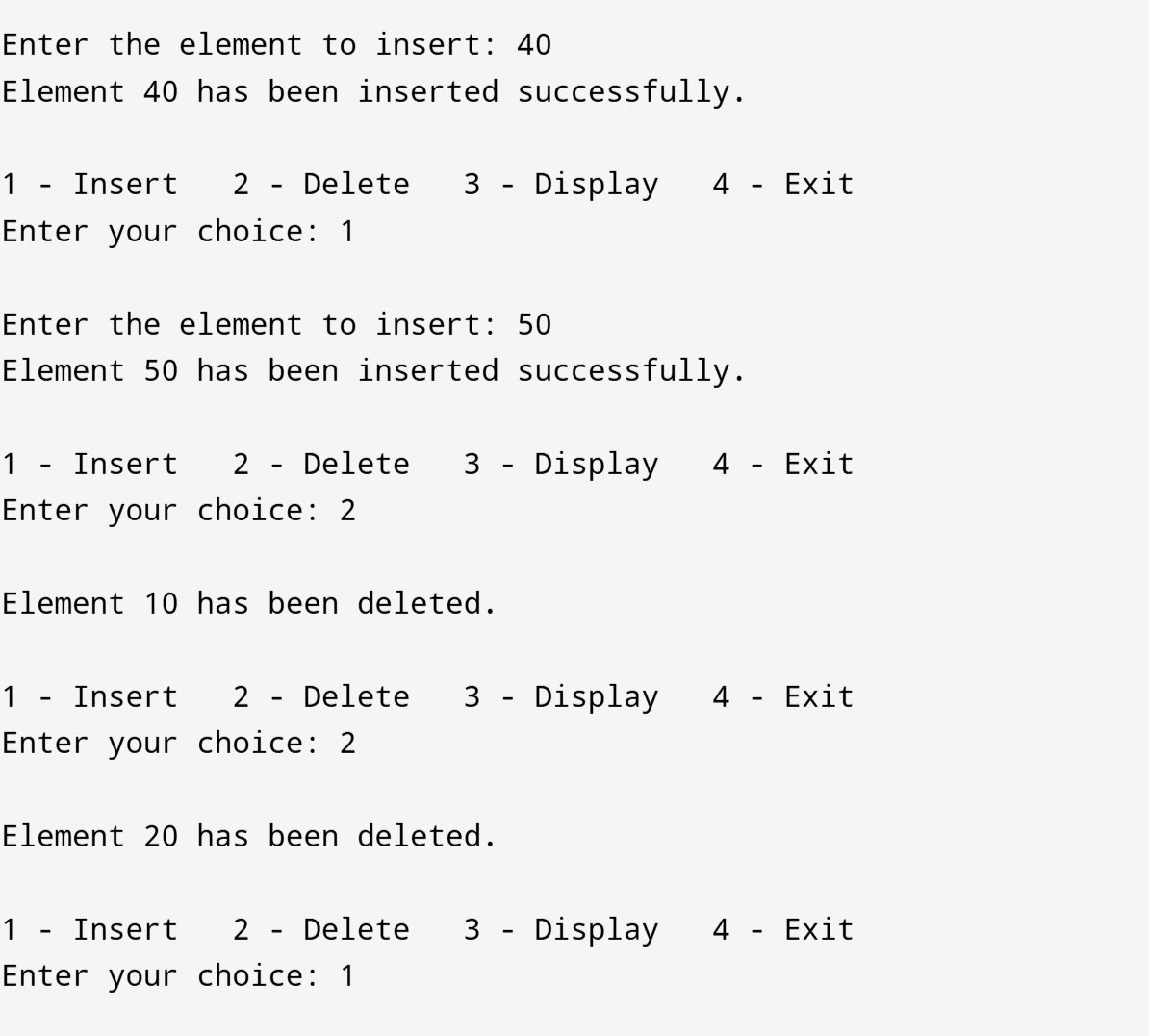
printf("%d ", Queue[i]);

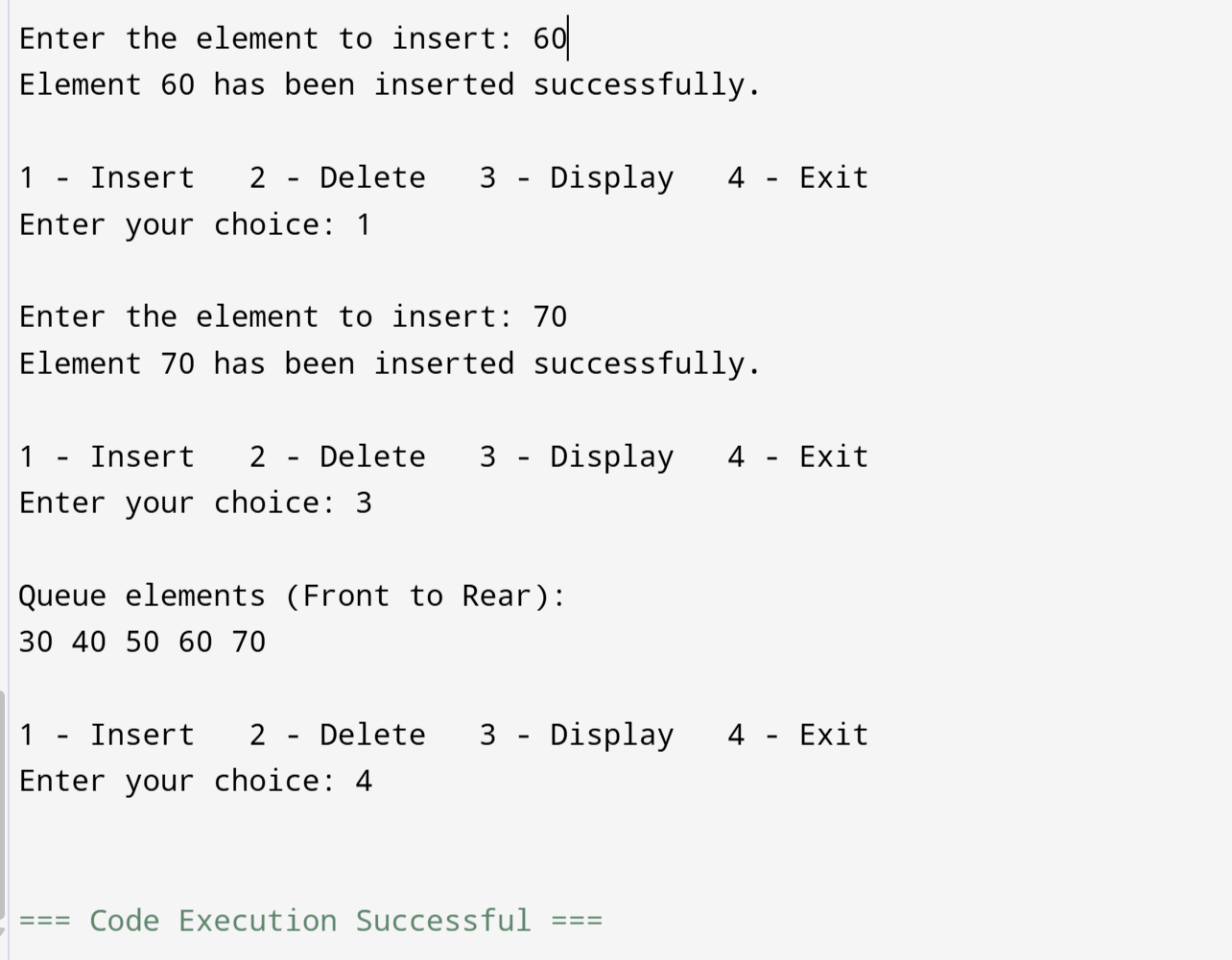
} printf("\n");

}

}







9. Write a program in C to implement deletion from a linked list(beg; mid; end)?

#include <stdio.h>

#include <stdlib.h>

typedef struct node {

int info;

struct node \*next;

} Node1;

Node1 \*start = NULL, \*ptr, \*temp;

void ins\_beg();

void ins\_mid();

void ins\_end();

void del\_beg();

void del\_mid();

void del\_end();

void display();

int main() {

int ch, ch1, ele, pos;

while (1) {

printf("1. Insertion\n2. Deletion\n3. Display\n4. Exit\n");

printf("Select the Operation to perform (1-4): ");

scanf("%d", &ch);

switch (ch) {

case 1:

printf("1. Beg 2. Mid 3. End\n");

scanf("%d", &ch1);

switch (ch1) {

case 1:

ins\_beg();

break;

case 2:

ins\_mid();

break;

case 3:

ins\_end();

break;

default:

printf("Invalid choice\n");

break;

}

break;

case 2:

printf("1. Beg 2. Mid 3. End\n");

scanf("%d", &ch1);

switch (ch1) {

case 1:

del\_beg();

break;

case 2:

del\_mid();

break;

case 3:

del\_end();

break;

default:

printf("Invalid choice\n");

break;

}

break;

case 3:

display();

break;

case 4:

exit(0);

default:

printf("Invalid choice\n");

break;

}

}

}

void ins\_beg() {

int ele;

printf("Enter element to insert: ");

scanf("%d", &ele);

temp = (Node1 \*)malloc(sizeof(Node1));

temp->info = ele;

if (start == NULL) {

temp->next = NULL;

} else {

temp->next = start;

}

start = temp;

}

void ins\_mid() {

int ele, pos;

printf("Enter element to insert: ");

scanf("%d", &ele);

printf("Enter position: ");

scanf("%d", &pos);

temp = (Node1 \*)malloc(sizeof(Node1));

temp->info = ele;

ptr = start;

for (int i = 1; i < pos - 1 && ptr != NULL; i++) {

ptr = ptr->next;

}

if (ptr == NULL) {

printf("Position out of bounds\n");

free(temp);

return;

}

temp->next = ptr->next;

ptr->next = temp;

}

void ins\_end() {

int ele;

printf("Enter element to insert: ");

scanf("%d", &ele);

temp = (Node1 \*)malloc(sizeof(Node1));

temp->info = ele;

temp->next = NULL;

if (start == NULL) {

start = temp;

} else {

ptr = start;

while (ptr->next != NULL) {

ptr = ptr->next;

}

ptr->next = temp;

}

}

void del\_beg() {

if (start == NULL) {

printf("Underflow\n");

return;

}

ptr = start;

start = start->next;

free(ptr);

}

void del\_end() {

if (start == NULL) {

printf("Underflow\n");

return;

}

ptr = start;

Node1 \*temp = NULL;

while (ptr->next != NULL) {

temp = ptr;

ptr = ptr->next;

}

if (temp != NULL) {

temp->next = NULL;

} else {

start = NULL;

}

free(ptr);

}

void del\_mid() {

int pos;

printf("Enter position to delete: ");

scanf("%d", &pos);

if (start == NULL) {

printf("Underflow\n");

return;

}

ptr = start;

Node1 \*temp = NULL;

for (int i = 1; i < pos && ptr != NULL; i++) {

temp = ptr;

ptr = ptr->next;

}

if (ptr == NULL) {

printf("Position out of bounds\n");

return;

}

if (temp != NULL) {

temp->next = ptr->next;

} else {

start = start->next;

}

free(ptr);

}

void display() {

ptr = start;

if (ptr == NULL) {

printf("List is empty\n");

return;

}

while (ptr != NULL) {

printf("%d\n", ptr->info);

ptr = ptr->next;

}

}

