**Problem Statement:**

Design a menu-driven program that allows users to perform various operations on an array. The program should provide options to insert an element into the array, delete an element from the array, and traverse (display) the elements of the array. The user should be able to choose any of these operations from a menu and provide the required inputs.

The program should implement the following functionalities:

1. Array Initialization:

2. Insertion:

3. Deletion:

4. Traverse (Display):

5. Exit:

The program should display a menu with the above options and allow the user to select an operation by entering the corresponding menu number. After executing the selected operation, the program should return to the menu and continue until the user chooses the exit option.

**Source Code:**

#include <stdio.h>

#include <stdlib.h>

void print(int arr[], int n)

{

if (n == 0)

{

printf("No Element to Display");

return;

}

for (int i = 0; i < n; i++)

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

}

void insertAtEnd(int arr[], int \*n, int element)

{

\*n = \*n + 1;

arr[\*n - 1] = element;

}

void deleteAtEnd(int arr[], int \*n)

{

\*n = \*n - 1;

}

void insertAtPos(int arr[], int \*n, int element, int pos)

{

int m = \*n;

\*n = \*n + 1; int size = \*n; int i;

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

{

if (i == pos - 1)

{

while (m != i)

{

arr[m] = arr[m - 1];

m--;

}

arr[i] = element; break;

}

}

}

void deleteAtPos(int arr[], int \*n, int pos)

{

int m;

int size = \*n; int i;

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

{

if (i == pos - 1)

{

m = i;

while (m != size)

{

arr[m] = arr[m + 1];

m++;

}

break;

}

}

\*n = \*n - 1;

}

void updateAtPos(int arr[], int n, int element, int pos)

{

int i;

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

{

if (i == pos - 1) arr[i] = element;

}

}

void search(int arr[], int n, int element)

{

int i, flag = 0;

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

{

if (element == arr[i])

{

flag = 1; break;

}

}

if (flag == 1)

printf("%d found at %d position", element, i + 1);

else

printf("%d not found", element);

}

int main()

{

printf("Yogesh MCA-B 70 \n");

int arr[5] = {10, 20, 30, 40, 50};

int n = sizeof(arr) / sizeof(arr[0]);

int ele, pos;

printf("Default values in array\n");

print(arr, n);

while (1)

{

printf("\n\nArray\n\n");

printf("1.Traverse\n");

printf("2.Insert at End postion\n");

printf("3.Insert at position\n");

printf("4.Delete at End position\n");

printf("5.Delete at position\n");

printf("6.Update at position\n");

printf("7.Search the Element\n");

printf("8.Exit\n");

printf("Enter the choice:");

int ch;

scanf("%d", &ch);

switch (ch)

{

case 1:

printf("\n\*\*\*\*\* Traverse \*\*\*\*\*\n\n");

print(arr, n);

break;

case 2:

printf("\n\*\*\*\*\* Insertion At End\*\*\*\*\*\n");

printf("Enter the element:");

scanf("%d", &ele); insertAtEnd(arr, &n, ele);

print(arr, n);

break;

case 3:

printf("\n\*\*\*\*\* Insertion At Position\*\*\*\*\*\n");

if (n == 0)

{

printf("Please,First insert Element at End \n");

}

else

{

printf("Enter the Position to insert the element");

scanf("%d", &pos);

if (pos > n)

{

printf("position doesn't exist\n");

}

else

{

printf("Enter the element:");

scanf("%d", &ele);

insertAtPos(arr, &n, ele, pos);

print(arr, n);

}

}

break;

case 4:

printf("\n\*\*\*\*\* Deletion At End\*\*\*\*\*\n");

if (n == 0)

{

printf("No Element to delete\n");

}

else

{

deleteAtEnd(arr, &n);

print(arr, n);

}

break;

case 5:

printf("\n\*\*\*\*\* Deletion At Position\*\*\*\*\*\n");

if (n == 0)

{

printf("No element to delete \n");

}

else

{

printf("Enter the Position to delete the element");

scanf("%d", &pos);

if (pos > n)

{

printf("position doesn't exist\n");

}

else

{

deleteAtPos(arr, &n, pos);

print(arr, n);

}

}

break;

case 6:

printf("\n\*\*\*\*\* Updation At Position\*\*\*\*\*\n");

if (n == 0)

{

printf("No element to Update \n");

}

else

{

printf("Enter the Position to update the element");

scanf("%d", &pos);

if (pos > n)

{

printf("position doesn't exist\n");

}

else

{

printf("Enter the update element value:");

scanf("%d", &ele);

updateAtPos(arr, n, ele, pos);

print(arr, n);

}

}

break;

case 7:

printf("\n\*\*\*\*\* Search the Element\*\*\*\*\*\n");

if (n == 0)

{

printf("No element to Search \n");

}

else

{

printf("Enter the element to search:");

scanf("%d", &ele);

search(arr, n, ele);

}

break;

case 8:

exit(0);

default:

printf("wrong choice");

}

}

return 0;

}

**Output:**

