DS LAB

NAME: SOUMYADIP MAITY

ROLL NO.: 22053029

SECTION: CSE 49

YEAR: 2022-23

ASSIGNMENT- 1

DATE: 18.08.2023

// Q1. WAP to find whether an array is subset of another array.

```
#include <stdio.h>
int main()
{
    int a[] = {1, 2, 3, 4, 5, 6};
    int m = sizeof(a) / sizeof(a[0]);
    int b[] = {4, 50, 3};
    int n = sizeof(b) / sizeof(b[0]);
    int subset = 1;
    for (int i = 0; i < n; i++)
    {
   int y = 0;
        for (int j = 0; j < m; j++)
       {
            if ( b[i] == a[j])
          {
                y = 1;
                break;
          }
        }
        if (!y)
        {
            subset = 0;
            break;
         }
    }
    if (subset)
        printf("b is a subset of a");
    else
        printf("b is not a subset of a");
    return 0;
}
```

// Q2. Given an array of integers, and a number 'sum'. WAP to find the number of pairs of integers in the array whose sum is equal to 'sum'.

```
#include <stdio.h>
int main()
{
   int arr[] = {1, 5, 8, -2,8,1,5,8};
   int n = sizeof(arr) / sizeof(arr[0]);
   int sum = 6;
   int count = 0;
   for (int i = 0; i < n - 1; i++)
 {
      for (int j = i + 1; j < n; j++)
      {
         if (arr[i] + arr[j] == sum)
         {
            count++;
            printf("(%d, %d)\n", arr[i], arr[j]);
         }
       }
   }
   printf("Number of pairs with sum %d: %d\n", sum, count);
   return 0;
}
```

// Q3. Write a program to read two numbers and compare the numbers using function call by address.

```
#include <stdio.h>
int compare(int *a, int *b) {
   if (*a > *b) {
      printf("%d is greater than %d\n", *a, *b);
   }
   else if (*a < *b) {
      printf("%d is less than %d\n", *a, *b);
   }
   else {
      printf("%d is equal to %d\n", *a, *b);
   }
   return *a,*b;
}
int main() {
   int a, b;
```

```
    printf("Enter the first & second number: ");
    scanf("%d %d", &a, &b);
    compare(&a, &b);
    return 0;
}
```

// Q4. WAP to create a 1-D array of n elements and perform the following menu based operations using function.
a. insert a given element at specific position.
b. delete an element from a specific position of the array.
c. linear search to search an element
d. traversal of the array.

```
#include <stdio.h>
#define MAX_SIZE 100
void insertElement(int arr[], int *size, int element, int position) {
    if (*size >= MAX_SIZE) {
        printf("Array is full. Cannot insert.\n");
        return;
    }
    if (position < 0 || position > *size) {
        printf("Invalid position for insertion.\n");
        return;
    }
    for (int i = *size; i > position; i--) {
        arr[i] = arr[i - 1];
    }
    arr[position] = element;
    (*size)++;
}
void deleteElement(int arr[], int *size, int position) {
    if (*size <= 0) {
        printf("Array is empty. Cannot delete.\n");
        return;
    }
    if (position < 0 || position >= *size) {
        printf("Invalid position for deletion.\n");
        return;
    }
    for (int i = position; i < *size - 1; i++) {
        arr[i] = arr[i + 1];
    }
    (*size)--;
```

```
}
int linearSearch(int arr[], int size, int key) {
   for (int i = 0; i < size; i++) {
      if (arr[i] == key) {
         return i;
      }
   }
   return -1;
}
void traverseArray(int arr[], int size) {
   printf("Array elements: ");
   for (int i = 0; i < size; i++) {
      printf("%d ", arr[i]);
   }
   printf("\n");
}
int main() {
   int arr[MAX_SIZE];
   int size = 0;
   int choice, element, position, searchKey, result;
   while (1) {
      printf("\nMenu:\n");
      printf("1. Insert element\n");
      printf("2. Delete element\n");
      printf("3. Linear search\n");
      printf("4. Traverse array\n");
      printf("5. Exit\n");
      printf("Enter your choice: ");
      scanf("%d", &choice);
      switch (choice) {
         case 1:
            printf("Enter element to insert: ");
            scanf("%d", &element);
            printf("Enter position for insertion: ");
            scanf("%d", &position);
            insertElement(arr, &size, element, position);
            break;
         case 2:
            printf("Enter position for deletion: ");
            scanf("%d", &position);
            deleteElement(arr, &size, position);
            break;
         case 3:
                printf("Enter element to search: ");
                scanf("%d", &searchKey);
                result = linearSearch(arr, size, searchKey);
                if (result != -1) {
                    printf("Element found at index %d\n", result);
                } else {
                    printf("Element not found\n");
                }
                break;
            case 4:
                traverseArray(arr, size);
                break;
            case 5:
                printf("Exiting...\n");
            default:
                printf("Invalid choice\n");
        }
    }
    return 0;
}
```

// Q5   Write a program to perform the following operations on a given square matrix using functions:

i.   Find the no.of nonzero elements

ii. Display upper triangular matrix

iii. Display the elements of just above and below the main diagonal.

```
#include <stdio.h>
#define N 3
void readMatrix(int matrix[N][N]) {
    printf("Enter matrix elements:\n");
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            scanf("%d", &matrix[i][j]);
        }
    }
}
int countNonZero(int matrix[N][N]) {
    int count = 0;
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            if (matrix[i][j] != 0) {
                count++;
            }
        }
    }
    return count;
}
void displayUpperTriangular(int matrix[N][N]) {
    printf("Upper triangular matrix:\n");
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            if (i <= j) {
                printf("%d ", matrix[i][j]);
            } else {
                printf("0 ");
            }
        }
        printf("\n");
    }
}
void displayAboveAndBelowDiagonal(int matrix[N][N]) {
    printf("Elements above and below the main diagonal:\n");
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++) {
            if (i == j - 1) {
                printf("Above: %d, ", matrix[i][j]);
            }
            if (i == j + 1) {
                printf("Below: %d\n", matrix[i][j]);
            }
        }
    }
}
int main() {
    int matrix[N][N];
    readMatrix(matrix);
    printf("No. of nonzero elements: %d\n", countNonZero(matrix));
    displayUpperTriangular(matrix);
    displayAboveAndBelowDiagonal(matrix);
    return 0;
}
```