EXPERIMENT 5 – ARRAYS

Q1. Read a list of integers into a 1-D array & print the second largest number

Aim

To find the **second largest element** in a list of integers.

Algorithm

1. Start
2. Input number of elements
3. Read all elements into array
4. Find largest and second largest
5. Print second largest
6. End

Pseudocode

```
read n
read array A[n]
largest = second = -∞
for i in 0..n-1:
    if A[i] > largest:
        second = largest
        largest = A[i]
    else if A[i] > second AND A[i] != largest:
        second = A[i]
```

```
print second
```

Flowchart

```
START
 ↓
Read n
 ↓
Read array
 ↓
Find largest & second largest
 ↓
Print second largest
 ↓
END
```

C Program

```
#include <stdio.h>
#include <limits.h>

int main() {
    int n, i, arr[100];
    int largest = INT_MIN, second = INT_MIN;

    printf("Enter number of integers: ");
    scanf("%d", &n);

    printf("Enter %d integers:\n", n);
    for(i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    for(i = 0; i < n; i++) {
        if(arr[i] > largest) {
            second = largest;
            largest = arr[i];
```

```
        } else if(arr[i] > second && arr[i] != largest) {
            second = arr[i];
        }
    }

    printf("Second largest = %d\n", second);
    return 0;
}
```

Output

```
PS C:\Users\ASUS\Desktop\C Exp> gcc exp5.1.c
PS C:\Users\ASUS\Desktop\C Exp> ./a.exe
Enter number of elements: 5
Enter 5 integers:
1
3
4
5
6
The second largest number is: 5
PS C:\Users\ASUS\Desktop\C Exp>
```

Q2. Count positive, negative, odd & even numbers in an array

Aim

To count **positive**, **negative**, **odd**, and **even** integers in a 1D array.

Algorithm

1. Read n
2. Read array
3. Initialize count variables
4. Traverse array and update counts
5. Display counts

Pseudocode

```
read n
read array A[n]

pos=neg=odd=even=0

for each element:
    if >0 pos++
    else if <0 neg++
    if %2==0 even++
    else odd++

print pos,neg,odd,even
```

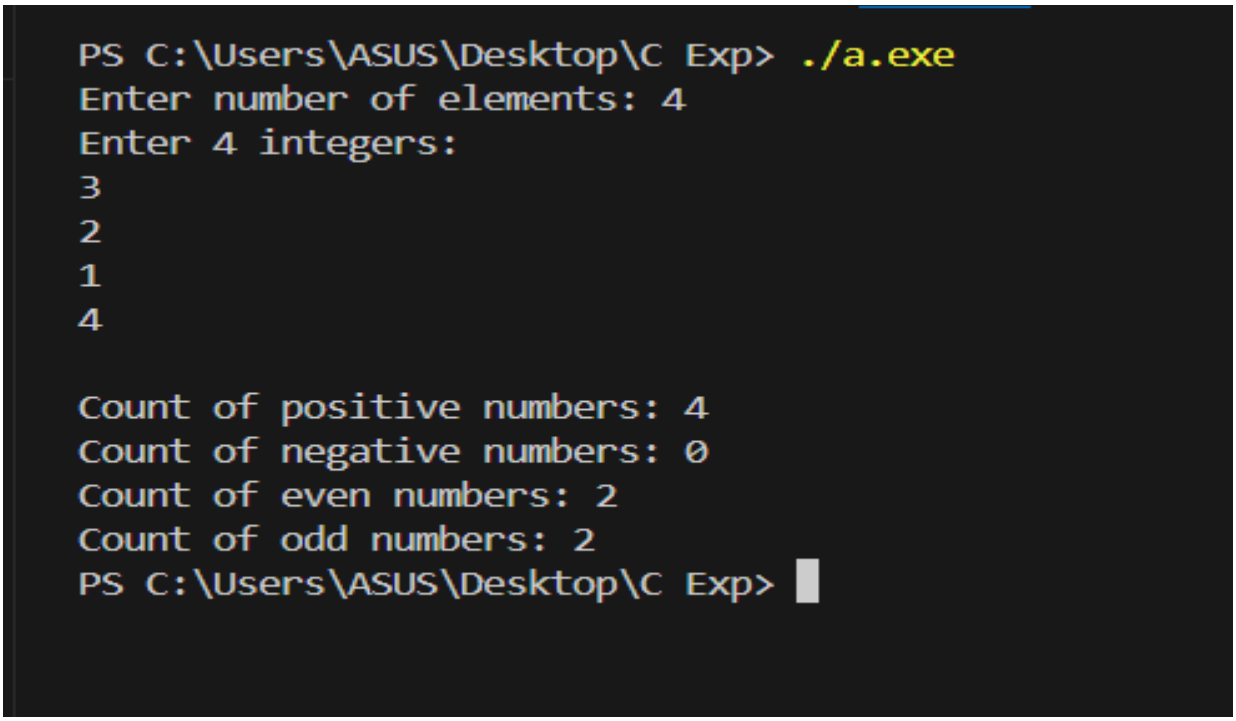
Flowchart

```
START → Read array → For each element:
      → Update pos/neg/odd/even → Print results → END
```

C Program

```
#include <stdio.h>

int main() {
    int n, i, arr[100];
    int pos = 0, neg = 0, odd = 0, even = 0;

    printf("Enter number of integers: ");
    scanf("%d", &n);

    printf("Enter %d integers:\n", n);
    for(i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    for(i = 0; i < n; i++) {
        if(arr[i] > 0) pos++;
        else if(arr[i] < 0) neg++;

        if(arr[i] % 2 == 0) even++;
        else odd++;
    }

    printf("Positive = %d\nNegative = %d\nOdd = %d\nEven = %d\n",
           pos, neg, odd, even);

    return 0;
}
```

Output



Q3. Find the frequency of a particular number in an array

Aim

To count how many times a given number appears in the array.

Algorithm

1. Read n

2. Read array
3. Input element to search
4. Loop through array counting matches
5. Print frequency

Pseudocode

```
read n
read array
read key
count=0
for each element:
    if element == key:
        count++
print count
```

Flowchart

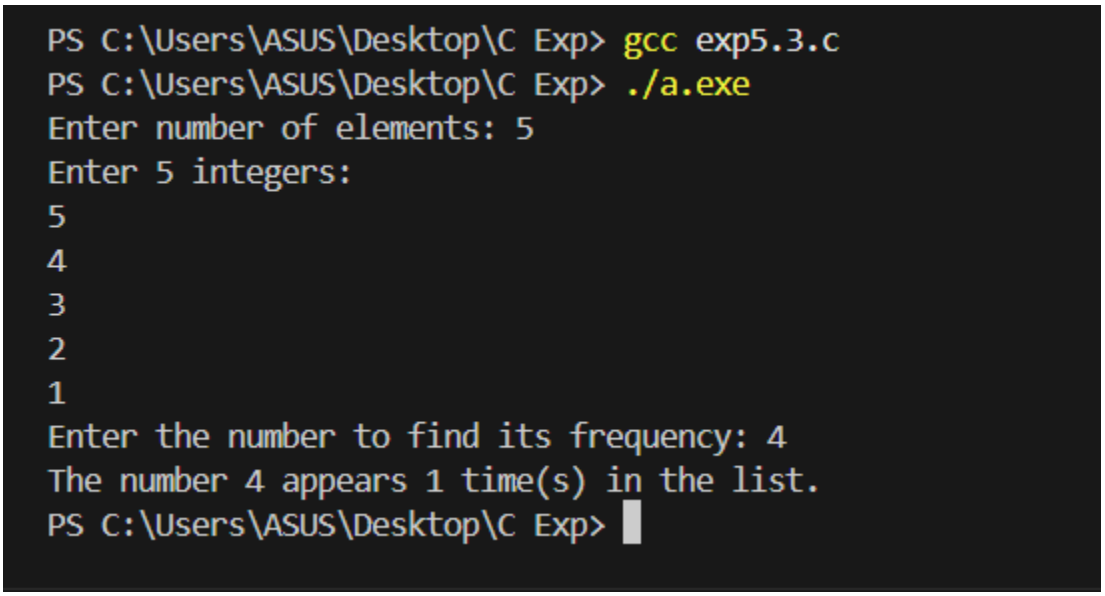
```
START → Read array → Input key
 → Count occurrences → Print frequency → END
```

C Program

```
#include <stdio.h>

int main() {
    int n, i, arr[100], key, count = 0;

    printf("Enter number of integers: ");
    scanf("%d", &n);

    printf("Enter integers:\n");
    for(i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("Enter number to find frequency: ");
```

```
    scanf("%d", &key);

    for(i = 0; i < n; i++)
        if(arr[i] == key)
            count++;

    printf("Frequency of %d is %d\n", key, count);

    return 0;
}
```

Output



Q4. Matrix Multiplication (Check compatibility + Print matrices)

Aim

To multiply two matrices A(m×n) and B(p×q) and check if multiplication is possible.

Algorithm

1. Input rows & columns of A and B
2. If n ≠ p → multiplication not possible
3. Else
    a. Read matrices A and B
    b. Multiply using formula
    c. Print matrices

Pseudocode

```
read m,n
read matrix A
read p,q
if n != p → print incompatible
else
   for i in m
     for j in q
        C[i][j] = sum(A[i][k] * B[k][j])
print C
```

Flowchart

```
START
 ↓
Read size of A and B
 ↓
Check if n == p
 ↓Yes
Compute product
 ↓
Print A, B, C
```

↓
END

(No → Print “Incompatible”)

C Program

```
#include <stdio.h>

int main() {
    int m, n, p, q, i, j, k;
    int A[10][10], B[10][10], C[10][10];

    printf("Enter rows and columns of Matrix A: ");
    scanf("%d %d", &m, &n);

    printf("Enter elements of Matrix A:\n");
    for(i = 0; i < m; i++)
        for(j = 0; j < n; j++)
            scanf("%d", &A[i][j]);

    printf("Enter rows and columns of Matrix B: ");
    scanf("%d %d", &p, &q);

    if(n != p) {
        printf("Matrix multiplication NOT possible!\n");
        return 0;
    }

    printf("Enter elements of Matrix B:\n");
    for(i = 0; i < p; i++)
        for(j = 0; j < q; j++)
            scanf("%d", &B[i][j]);

    for(i = 0; i < m; i++)
        for(j = 0; j < q; j++) {
            C[i][j] = 0;
            for(k = 0; k < n; k++)
```

```
                C[i][j] += A[i][k] * B[k][j];
        }

    printf("\nMatrix A:\n");
    for(i = 0; i < m; i++) {
        for(j = 0; j < n; j++)
            printf("%d ", A[i][j]);
        printf("\n");
    }

    printf("\nMatrix B:\n");
    for(i = 0; i < p; i++) {
        for(j = 0; j < q; j++)
            printf("%d ", B[i][j]);
        printf("\n");
    }

    printf("\nProduct Matrix C = A × B:\n");
    for(i = 0; i < m; i++) {
        for(j = 0; j < q; j++)
            printf("%d ", C[i][j]);
        printf("\n");
    }

    return 0;
}
```

Output

```
PS C:\Users\ASUS\Desktop\Dahadi\class c> ./a.exe
Enter rows and columns of Matrix A: 2
2
Enter elements of Matrix A:
2
2
2
3
Enter rows and columns of Matrix B: 2
2
Enter elements of Matrix B:
2
3
4
5

Matrix A:
2 2
2 3

Matrix B:
2 3
4 5

Product Matrix C = A ├ù B:
12 16
16 21
PS C:\Users\ASUS\Desktop\Dahadi\class c>
```

.