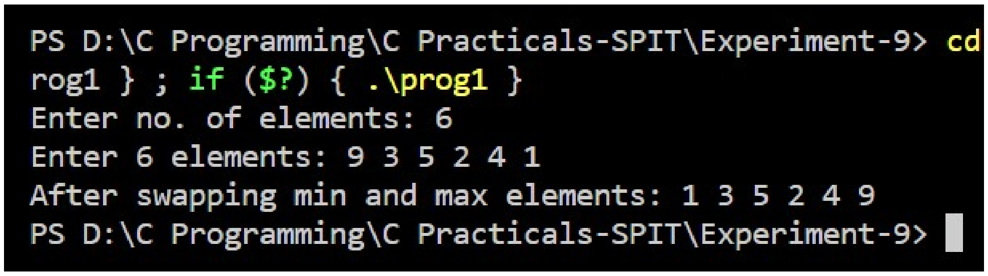
Name	**Hatim Yusuf Sawai**
UID no.	**2021300108**
Experiment No.	9

AIM:	Demonstrate the use of pointers to solve a given problem.
Program 1	
PROBLEM STATEMENT:	Write a program to swap smallest and largest element in an array using pointers
ALGORITHM:	START 2. Define void function swap with a integer n and integer array arr[n] as parameter 3. I=0, min_index=0, max_index = 0 4. Loop from i=0 to n-1 A. If *(arr+i) < *(arr+min index) min index = i B. If *(arr + i) > *(arr + max_index) max_index = i 5. Temp = *(arr + max_index) 6. *(arr + max_index) = *(arr + min_index) 7. *(arr + min_index) = temp 8. Define main function 9. Input number of elements n 10. Input array arr[n] 11. Call function swap(n, arr) 12. Print array arr[n] 13. STOP

PROGRAM:

```
#include<stdio.h>
void swap(int n,int arr[n])
{
    int min=0,max=0,i,temp;
    for(i=0;i<n;i++)
    {
        if(*(arr+i)>*(arr+max))
            max=i;
        if(*(arr+i)<*(arr+min))
            min=i;
    }
    temp = *(arr+max);
    *(arr+max) = *(arr+min);
    *(arr+min) = temp;
}
int main()
{
    int n,i;
    printf("Enter no. of elements: ");
    scanf("%d",&n);
    int arr[n];
    printf("Enter %d elements: ",n);
    for(i=0;i<n;i++)
    {
        scanf("%d",&arr[i]);
    }
    swap(n,arr);
    printf("After swapping min and max elements:");
    for(i=0;i<n;i++)
        printf("%d ",arr[i]);
    return 0;
}
```

RESULT:



```
PS D:\C Programming\C Practicals-SPIT\Experiment-9> cd
rog1 } ; if ($?) { .\prog1 }
Enter no. of elements: 6
Enter 6 elements: 9 3 5 2 4 1
After swapping min and max elements: 1 3 5 2 4 9
PS D:\C Programming\C Practicals-SPIT\Experiment-9>
```

Program 2

PROBLEM STATEMENT:

Write a program to reverse the position of all elements in the array using pointers.

ALGORITHM:

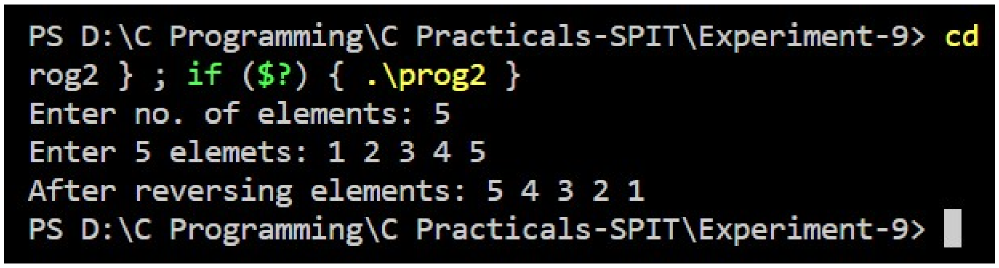
```
START
2. Define void function reverse with integer n and integer array arr[n] as parameters.
3. Loop from I = 0 to n/2-1
A. temp = *(arr + i)
B. *(arr + i) = *(arr + n-1 -i)
C. *(arr + n-1 – i) = temp
4. Define main function
5. Input no of elements of array n
6. Input array arr[n]
7. Call function reverse(n, arr)
8. Print arr[n]
9. STOP
```

PROGRAM:

```
#include<stdio.h>
void reverse(int n,int arr[n])
{
    int i,temp;
    for(i=0;i<n/2;i++)
    {
        temp = *(arr+i);
        *(arr+i) = *(arr+n-1-i);
        *(arr+n-1-i) = temp;
    }
}
```

```
int main()
{
    int n,i;
    printf("Enter no. of elements: ");
    scanf("%d",&n);
    int arr[n];
    printf("Enter %d elemets: ");
    for(i=0;i<n;i++)
        scanf("%d",(arr+i));
    reverse(n,arr);
    printf("After reversing elements: ");
    for(i=0;i<n;i++)
    {
        printf("%d ",*(arr+i));
    }
    return 0;
}
```

RESULT:



```
PS D:\C Programming\C Practicals-SPIT\Experiment-9> cd
rog2 } ; if ($?) { .\prog2 }
Enter no. of elements: 5
Enter 5 elemets: 1 2 3 4 5
After reversing elements: 5 4 3 2 1
PS D:\C Programming\C Practicals-SPIT\Experiment-9>
```

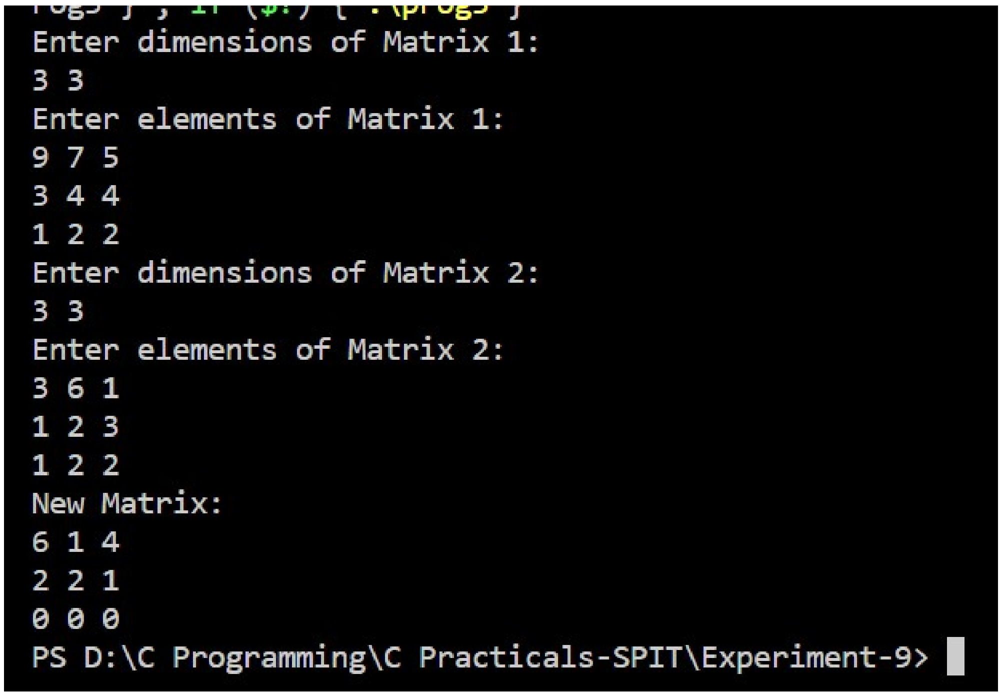
Program 3

PROBLEM STATEMENT: Write a program to calculate the subtraction of matrices using pointers. Dimensions of the matrix will be decided by the user.

PROGRAM:

```
#include<stdio.h>
int main()
{
    int m, n, a, b, i, j;
    printf("Enter dimensions of Matrix 1:\n");
    scanf("%d %d", &m, &n);
    int mat1[m][n];
    printf("Enter elements of Matrix 1:\n");
    for (i = 0; i < m; i++)
    {
        for (j = 0; j < n; j++)
            scanf("%d",&mat1[i][j]);
    }
    printf("Enter dimensions of Matrix 2:\n");
    scanf("%d %d", &a, &b);
    int mat2[a][b];
    printf("Enter elements of Matrix 2:\n");
    for (i = 0; i < a; i++)
    {
        for (j = 0; j < b; j++)
            scanf("%d",&mat2[i][j]);
    }
    if(m==a && n==b)
    {
        printf("New Matrix:\n");
        for (i = 0; i < m; i++)
        {
            for (j = 0; j < n; j++)
            {
                printf("%d ", *(*(mat1 + i) + j) - *(*(mat2 + i) + j));
            }
            printf("\n");
        }
    }
    else
        printf("Matrices cannot be subtracted\n");
    return 0;
}
```

RESULT:



```
rogs } ; if ($?) { .\progs }
Enter dimensions of Matrix 1:
3 3
Enter elements of Matrix 1:
9 7 5
3 4 4
1 2 2
Enter dimensions of Matrix 2:
3 3
Enter elements of Matrix 2:
3 6 1
1 2 3
1 2 2
New Matrix:
6 1 4
2 2 1
0 0 0
PS D:\C Programming\C Practicals-SPIT\Experiment-9>
```

CONCLUSION:	In this experiment, we learnt how to use pointers in 2D arrays and 1D arrays and write basic functions of swapping, sorting and operations.