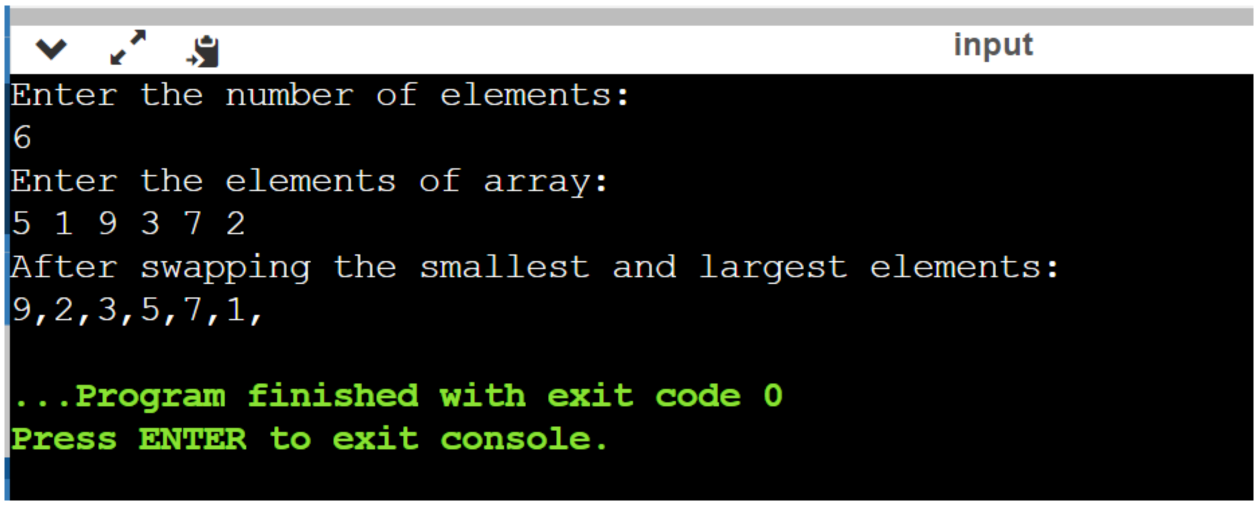
Name	Adwait S Purao
UID no.	2021300101
Experiment No.	9

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:	1) START 2) Define function void swap with integer pointer parameters a and b 3) Define a temporary variable temp 4) Store the value of *a in temp 5) Store the value of *b in *a 6) Store the value of temp in *b 7) Define function void sort with integer parameters array and size 8) for(int i=0;i<size;i++) for(int j=i+1;j<size;j++) if(*(arr+i)>*(arr+j)) 9) swap(&arr[i],&arr[j]) 10) Define function int main 11) Define integer parameters n,ar[100],h 12) Take input of the number of elements 13) Take input of the elements of the array 14) Call function sort 15) Call function swap to swap the first and last element of the sorted array 16) Print the array after swapping 17) STOP

PROGRAM:	#include<stdio.h>
	void sort(int arr[],int size);
	void swap(int*a,int*b);
	int main(){
	int n,ar[100],h;
	printf("Enter the number of elements:\n");
	scanf("%d",&n);
	printf("Enter the elements of array:\n");
	for(h=0;h<n;h++){
	scanf("%d",(ar+h));
	}
	sort(ar,n);
	swap(&ar[0],&ar[n-1]);
	printf("After swapping the smallest and largest elements:\n");
	for(int s=0;s<n;s++){
	printf("%d,",*(ar+s));
	}
	}
	void sort(int arr[],int size){
	for(int i=0;i<size;i++){
	for(int j=i+1;j<size;j++){
	if(*(arr+i)>*(arr+j)){
	swap(&arr[i],&arr[j]);
	}
	}
	}
	}
	void swap(int*a,int*b){
	int temp;
	temp=*a;
	*a=*b;
	*b=temp;
	}

RESULT:	



```
Enter the number of elements:
6
Enter the elements of array:
5 1 9 3 7 2
After swapping the smallest and largest elements:
9,2,3,5,7,1,

...Program finished with exit code 0
Press ENTER to exit console.
```

Program 2

PROBLEM STATEMENT :	Write a program to reverse the position of all elements in the array using pointers.
ALGORITHM:	1) START 2) Define function void ar_reverse with integer parameters arr,start,end 3) if(start<end) 4) Declare a integer variable temp temp=*(arr+start); *(arr+start)=*(arr+end); *(arr+end)=temp; 5) Call the function recursively ar_reverse(arr,start+1,end-1) 6) In function main declare integer variables ar[100],k,st=0,ed 7) Take the number of elements as input 8) Take the elements of the array as input Call function ar_reverse(ar,st,ed) 9) Print Reversed array 10) STOP

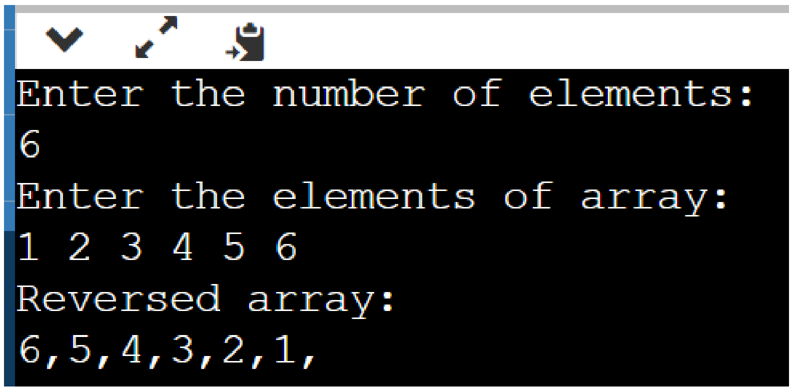
PROGRAM:

```
#include<stdio.h>
void ar_reverse(int arr[],int start,int end);
int main(){
int ar[100],k,st=0,ed;
printf("Enter the number of elements:\n");
scanf("%d",&k);
ed=k-1;
printf("Enter the elements of array:\n");
for(int r=0;r<k;r++){
scanf("%d",(ar+r));
}
ar_reverse(ar,st,ed);
printf("Reversed array:\n");
for(int g=0;g<k;g++){
printf("%d,",*(ar+g));
}



Return 0;
}
void ar_reverse(int arr[],int start,int end){
if(start<end){
int temp;
temp=*(arr+start);
*(arr+start)=*(arr+end);
*(arr+end)=temp;
ar_reverse(arr,start+1,end-1);
}
}
```

RESULT:



```
Enter the number of elements:
6
Enter the elements of array:
1 2 3 4 5 6
Reversed array:
6,5,4,3,2,1,
```

Program 3

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

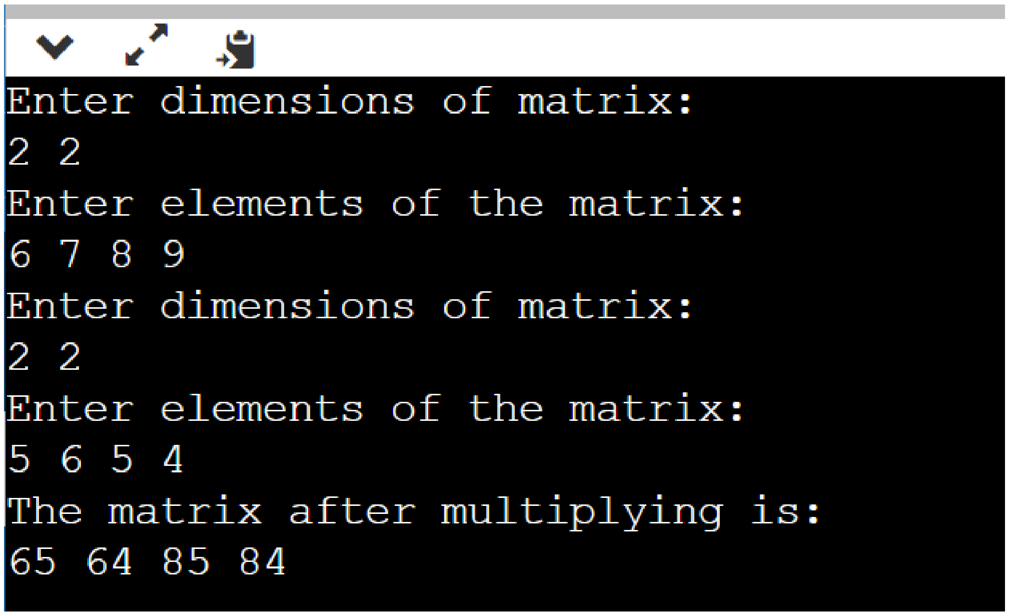
ALGORITHM:

```
1. START
2. Define void function multiply with 4 integers m, n, a, b and three 2-D integer arrays arr1[m][n], arr2[a][b], arr3[m][b]
3. If n is equal to a
    A. Loop from I = 0 to m-1
        I. Loop from J = 0 to b-1
            a. sum = 0
            b. Loop from k = 1 to n-1
                sum += (*(*(arr1 + i) + k)) * (*(*(arr2 + k) + j))
4. Define main function
5. Input dimensions of first matrix m and n
6. Input first matrix arr1[m][n]
7. Input dimensions of second matrix a and b
8. Input second matrix arr2[a][b]
9. Define arr3[m][b]
10. Call function multiply(m, n, a, b, arr1, arr2, arr3)
11. Print 2-D array arr3
12. STOP
```

PROGRAM:

```
#include<stdio.h>
void multiply(int m,int n ,int arr1[m][n],int a, int b, int arr2[a][b],int arr3[m][b])
{
 if(n==a)
 for(int i=0;i<m;i++)
 for(int j=0;j<b;j++)
 {
 int sum = 0;
 for(int k=0;k<n;k++)
 sum += (*(*(arr1 + i) + k)) * (*(*(arr2 + k) + j));
 *(*(arr3 + i) + j) = sum;
 }
}
int main()
{
 int m,n,a,b;
 printf("Enter dimensions of matrix:\n");
 scanf("%d %d",&m,&n);
 printf("Enter elements of the matrix:\n");
 int arr1[m][n];
 for(int i=0;i<m;i++)
 for(int j=0;j<n;j++)
 scanf("%d",(*(arr1 + i) + j));
 printf("Enter dimensions of matrix:\n");
 scanf("%d %d",&a,&b);
 printf("Enter elements of the matrix:\n");
 int arr2[a][b];
 for(int i=0;i<a;i++)
 for(int j=0;j<b;j++)
 scanf("%d",(*(arr2 + i) + j));
 int arr3[m][b];
 multiply(m,n,arr1,a,b,arr2,arr3);
 printf("The matrix after multiplying is:\n");
 for(int i=0;i<m;i++)
 for(int j=0;j<b;j++)
 printf("%d ", *(*(arr3 + i) + j));
 return 0;
}
```

RESULT:



```
Enter dimensions of matrix:
2 2
Enter elements of the matrix:
6 7 8 9
Enter dimensions of matrix:
2 2
Enter elements of the matrix:
5 6 5 4
The matrix after multiplying is:
65 64 85 84
```

CONCLUSION:	In the above experiment we learnt how to take inputs and print outputs using pointers and we also learned how to access the elements of the array using pointers.