1. Write a program to sort the array elements using Quick Sort Technique.

#include <stdio.h>

void swap(int \*a, int \*b) {

int t = \*a;

\*a = \*b;

\*b = t;

}

int partition(int arr[], int low, int high) {

int pivot = arr[high], i = low - 1,j;

for (j = low; j < high; j++) {

if (arr[j] < pivot) {

i++;

swap(&arr[i], &arr[j]);

}

}

swap(&arr[i + 1], &arr[high]);

return i + 1;

}

void quickSort(int arr[], int low, int high) {

if (low < high) {

int pi = partition(arr, low, high);

quickSort(arr, low, pi - 1);

quickSort(arr, pi + 1, high);

}

}

int main() {

int n, i;

printf("Enter the number of elements: ");

scanf("%d", &n);

int arr[n];

printf("Enter %d elements:\n", n);

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

scanf("%d", &arr[i]);

printf("Original array:\n");

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

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

printf("\n");

quickSort(arr, 0, n - 1);

printf("Sorted array:\n");

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

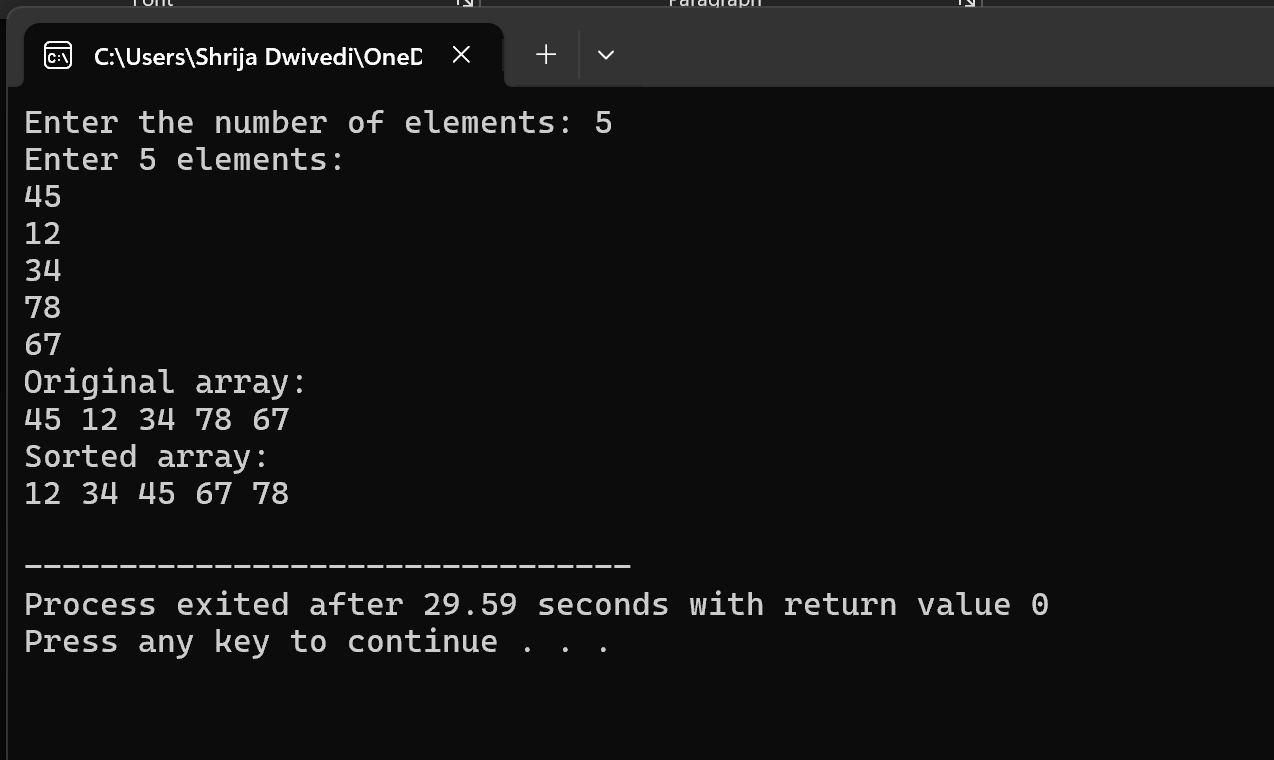
printf("%d ", arr[i]);

printf("\n");

return 0;

}

Output:



2. Divide and conquer: Implementation of Strassen’s algorithm for matrix multiplication.

Also analyze how this approach is advantage when compared to normal multiplication

#include <stdio.h>

void strassenMultiply(int A[2][2], int B[2][2], int result[2][2]) {

int M1 = (A[0][0] + A[1][1]) \* (B[0][0] + B[1][1]);

int M2 = (A[1][0] + A[1][1]) \* B[0][0];

int M3 = A[0][0] \* (B[0][1] - B[1][1]);

int M4 = A[1][1] \* (B[1][0] - B[0][0]);

int M5 = (A[0][0] + A[0][1]) \* B[1][1];

int M6 = (A[1][0] - A[0][0]) \* (B[0][0] + B[0][1]);

int M7 = (A[0][1] - A[1][1]) \* (B[1][0] + B[1][1]);

result[0][0] = M1 + M4 - M5 + M7;

result[0][1] = M3 + M5;

result[1][0] = M2 + M4;

result[1][1] = M1 - M2 + M3 + M6;

}

int main() {

int A[2][2], B[2][2], result[2][2];

printf("Enter elements of 2x2 matrix A:\n");

for(int i=0; i<2; i++)

for(int j=0; j<2; j++)

scanf("%d", &A[i][j]);

printf("Enter elements of 2x2 matrix B:\n");

for(int i=0; i<2; i++)

for(int j=0; j<2; j++)

scanf("%d", &B[i][j]);

strassenMultiply(A, B, result);

printf("Resultant matrix after multiplication:\n");

for(int i=0; i<2; i++) {

for(int j=0; j<2; j++)

printf("%d ", result[i][j]);

printf("\n");

}

return 0;

}

Output:

