1. Binary search

#include <stdio.h>

int binarysearch(int size, int arr[], int key)

{

int start = 0, end = size – 1;

while (start <= end)

{

int mid = (start + end) / 2;

if (arr[mid] == key)

{

return mid;

}

else if (arr[mid] < key)

{

start = mid + 1;

}

else

{

end = mid – 1;

}

}

return -1;

}

int main()

{

Int arr[] = {2, 5, 7, 9, 14, 15, 29};

int size = sizeof(arr) / sizeof(arr[0]);

int key;

printf(“Enter the element to search: “);

scanf(“%d”, &key);

int result = binarysearch(size, arr, key);

if (result != -1)

{

printf(“Element found at index %d\n”, result);

}

else

{

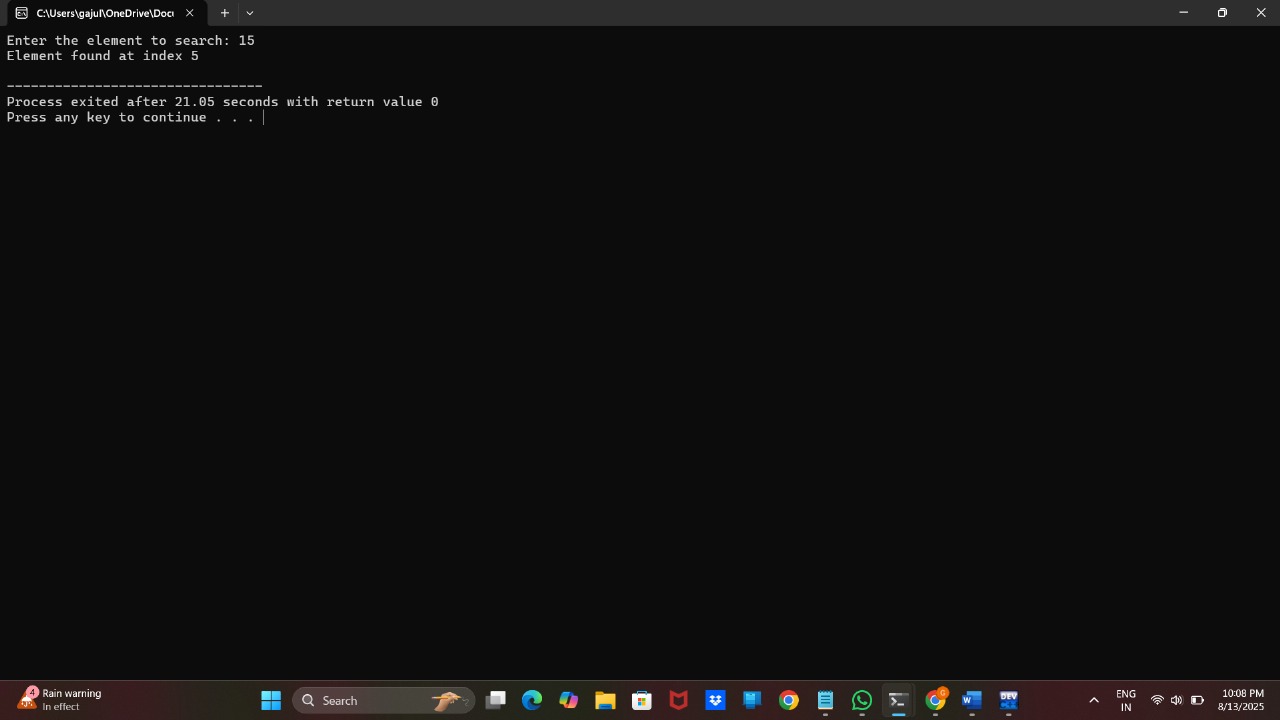
printf(“Element not found\n”);

}

return 0;

}

Output:



1. Quick sort

#include <stdio.h>

void quicksort(int a[], int left, int right)

{

if (left >= right) return;

int pivot = a[right];

int i = left;

int j;

for (j = left; j < right; j++)

{

if (a[j] < pivot) {

int temp = a[i];

a[i] = a[j];

a[j] = temp;

i++;

}

}

int temp = a[i];

a[i] = a[right];

a[right] = temp;

quicksort(a, left, i - 1);

quicksort(a, i + 1, right);

}

int main()

{

int a[] = {2, 37, 4, 29, 16, 49, 13}; // Your array

int n = sizeof(a) / sizeof(a[0]);

int i;

quicksort(a, 0, n - 1);

printf(“Sorted array: “);

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

{

printf(“%d “, a[i]);

}

printf(“\n”);

return 0;

}

Output:

