#include <stdio.h>

#include <stdlib.h>

#include <time.h>

// Function to swap two elements

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

int t = \*a;

\*a = \*b;

\*b = t;

}

// Function to partition the array on the basis of pivot

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

int pivot = arr[high];

int i = (low - 1);

for (int j = low; j <= high - 1; j++) {

if (arr[j] <= pivot) {

i++;

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

}

}

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

return (i + 1);

}

// Function to implement Quick sort

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;

clock\_t start, end;

double cpu\_time\_used;

// Sorting for different values of n

int sizes[] = {10, 100, 1000};

int num\_sizes = sizeof(sizes) / sizeof(sizes[0]);

for (int i = 0; i < num\_sizes; i++) {

n = sizes[i];

int arr[n];

// Generate random array elements

for (int j = 0; j < n; j++) {

arr[j] = rand() % 1000;

}

// Measure the execution time

start = clock();

quickSort(arr, 0, n - 1);

end = clock();

cpu\_time\_used = ((double)(end - start)) / CLOCKS\_PER\_SEC;

printf("Time taken to sort %d elements: %lf seconds\n", n, cpu\_time\_used);

}

return 0;

}