









#include <stdio.h>

//const

#define MAX 10

// khai báo hàm

void display(int arr[], int size);

int getMax(int arr[], int size);

void radixSort(int arr[], int size);

int main() {

//unsorted elements

int arr[] = {10,15,1,60,5,100,25,50};

// kích thước của mảng

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

//output unsorted elements

/ / ra các yếu tố chưa được phân loại

display(arr, n);

//sort the elements

/ / hiển thị các yếu tố sắp xếp

radixSort(arr, n);

//display sorted elements

display(arr, n);

return 0;

}

void display(int arr[], int size) {

int i;

for(i = 0; i < size; i++) {

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

}

printf("\n");

}

int getMax(int arr[], int size) {

int i, max = arr[0];

for(i = 1; i < size; i++) {

if(arr[i] > max) {

max = arr[i];

}

}

return max;

}

void radixSort(int arr[], int size) {

int i, max, bucket[MAX], count[10], expo = 1;

//get the max value element in the unsorted array

// lấy phần tử giá trị tối đa trong dãy không được sắp xếp

max = getMax(arr, size);

while(max / expo > 0) {

//reset count

for(i = 0; i < 10; i++) {

count[i] = 0;

}

//save count of the occurrence

// lưu lại số lần xuất hiện

for(i = 0; i < size; i++) {

count[ (arr[i] / expo) % 10 ]++;

}

//set count to contain the actual position of the digits

/ / set count để chứa vị trí thực tế của các chữ số

for(i = 1; i < size; i++) {

count[i] += count [i - 1];

}

//build the bucket

/ / xây dựng xô

for(i = size - 1; i >= 0; i--) {

bucket[ count[ (arr[i]/expo) % 10 ] - 1] = arr[i];

count[ (arr[i]/expo) % 10 ]--;

}

//copy the result to arr

/ / sao chép kết quả để arr

for(i = 0; i < size; i++) {

arr[i] = bucket[i];

}

// tăng expo, 10 ^ x

//increase expo, 10^x

expo \*= 10;

}

}