**#include<bits/stdc++.h>**

int **gcd**(int x, int y){

int val;

for(int i = max(x,y); i!=0; i--){

if(x%i==0 && y%i==0){

val = i;

break;

}

}

return val;

}

}

void **arrprint** (int arr[], int size){

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

cout << arr[i] << " | ";

}

cout << endl;

}

void **sortarrayplus** (int arr[], int size){

int temp;

for(int i = 0; i < size-1; i++){

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

if(arr[i] > arr[j]){

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

}

}

void **sortarrayminus**(int arr[], int size){

int temp;

for(int i = 0; i < size-1; i++){

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

if(arr[i] < arr[j]){

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

}

}

void **FindElement (**int arr[], int size, int element){

int max,min,avg;

min = 0;

max = size-1;

while(min <= max){

avg=min+max / 2;

if(arr[avg] == element){

cout << avg << " " <<arr[avg] << endl;

break;

}

if(arr[avg] < element){

min = avg+1;

}

if(arr[avg] > element){

max = avg-1;

}

}

}

void **largest** (int arr[], int size){

int biggest = arr[0];

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

if(arr[i] > biggest){

biggest = arr[i];

}

}

cout << biggest << endl;

}

void **smallest** (int arr[], int size){

int smallest = arr[0];

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

if(arr[i] < smallest){

smallest = arr[i];

}

}

cout << smallest << endl;

}

void **secondlargest** (int arr[], int size){

int biggest = arr[0];

int secondbiggest;

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

if(arr[i] > biggest){

secondbiggest = biggest;

biggest = arr[i];

}

}

cout << secondbiggest << endl;

}

void **secondsmallest** (int arr[], int size){

int smallest = arr[0];

int secondsmallest;

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

if(arr[i] < smallest){

secondsmallest = smallest;

smallest = arr[i];

}

}

cout << secondsmallest << endl;

}

unique(a, a + size) - a

int **Part**(int \*arr, int start, int end){

int pivot = arr[end];

int partitionIndex = start;

for(int i = start; i < end; i++){

if (arr[i] <= pivot){

swap(arr[i],arr[partitionIndex]);

partitionIndex++;

}

}

swap(arr[partitionIndex],arr[end]);

return partitionIndex;

}

void **sortarr**(int \*arr,int start,int end){

if(start < end){

int partitionIndex = Part(arr,start,end);

sortarr(arr,start,partitionIndex-1);

sortarr(arr,partitionIndex+1,end);

}

}

void **InsertionSort**(int\* A, int size){

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

int hole = i;

int value = A[i];

while(hole > 0 && A[hole-1] > value){

A[hole] = A[hole-1];

hole-=1;

}

A[hole] = value;

} }

void PrintChar(char\* C){

int i = 0;

while(C[i] != '\0'){

cout << C[i];

i++;

}

cout << endl;

}