# ASSIGNMET

**DATA STRUCTURES AND ALGORITHMS**

# CS-410

## SUBMITTED TO:

Prof. Nayyar Iqbal

## SUBMITTED BY:

Umer Shafiq (2020-ag-6482)

SELECTION SORT

#include <bits/stdc++.h>

using namespace std;

void swap(int \*xp, int \*yp)

{

int temp = \*xp;

\*xp = \*yp;

\*yp = temp;

}

void selectionSort(int arr[], int n)

{

int i, j, min\_idx;

for (i = 0; i < n-1; i++)

{

min\_idx = i;

for (j = i+1; j < n; j++)

if (arr[j] < arr[min\_idx])

min\_idx = j;

swap(&arr[min\_idx], &arr[i]);

}

}

void printArray(int arr[], int size)

{

int i;

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

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

cout << endl;

}

int main()

{

cout<<"Given array;\n77, 33, 44, 11, 88, 22, 66, 55\n";

int arr[] = {77,33,44,11,88,22,66,55};

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

selectionSort(arr, n);

cout << "Sorted array: \n";

printArray(arr, n);

return 0;

}

INSERTION SORT

#include<iostream>

using namespace std;

int main ()

{

int myarray[8] = { 77,33,44,11,88,22,66,55};

cout<<"Given array: \n";

for(int i=0;i<8;i++)

{

cout <<myarray[i]<<"\t";

}

for(int k=1; k<8; k++)

{

int temp = myarray[k];

int j= k-1;

while(j>=0 && temp <= myarray[j])

{

myarray[j+1] = myarray[j];

j = j-1;

}

myarray[j+1] = temp;

}

cout<<"\nSorted array: \n";

for(int i=0;i<8;i++)

{

cout <<myarray[i]<<"\t";

}

}

MERGE SORT

#include <iostream>

using namespace std;

void merge(int arr[], int p, int q, int r) {

int n1 = q - p + 1;

int n2 = r - q;

int L[n1], M[n2];

for (int i = 0; i < n1; i++)

L[i] = arr[p + i];

for (int j = 0; j < n2; j++)

M[j] = arr[q + 1 + j];

int i, j, k;

i = 0;

j = 0;

k = p;

while (i < n1 && j < n2) {

if (L[i] <= M[j]) {

arr[k] = L[i];

i++;

} else {

arr[k] = M[j];

j++;

}

k++;

}

while (i < n1) {

arr[k] = L[i];

i++;

k++;

}

while (j < n2) {

arr[k] = M[j];

j++;

k++;

}

}

void mergeSort(int arr[], int l, int r) {

if (l < r) {

int m = l + (r - l) / 2;

mergeSort(arr, l, m);

mergeSort(arr, m + 1, r);

merge(arr, l, m, r);

}

}

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

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

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

cout << endl;

}

int main() {

cout<<"Given array: \n66, 33, 40, 22, 55, 80, 20, 50, 44, 77, 30 \n";

int arr[] = {66,33,40,22,55,80,20,50,44,77,30};

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

mergeSort(arr, 0, size - 1);

cout << "Sorted array: \n";

printArray(arr, size);

return 0;

}

RADIX SORT

#include <iostream>

using namespace std;

int getMax(int array[], int n) {

int max = array[0];

for (int i = 1; i < n; i++)

if (array[i] > max)

max = array[i];

return max;

}

void countingSort(int array[], int size, int place) {

const int max = 10;

int output[size];

int count[max];

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

count[i] = 0;

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

count[(array[i] / place) % 10]++;

for (int i = 1; i < max; i++)

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

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

output[count[(array[i] / place) % 10] - 1] = array[i];

count[(array[i] / place) % 10]--;

}

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

array[i] = output[i];

}

void radixsort(int array[], int size) {

int max = getMax(array, size);

for (int place = 1; max / place > 0; place \*= 10)

countingSort(array, size, place);

}

void printArray(int array[], int size) {

int i;

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

cout << array[i] << " ";

cout << endl;

}

int main() {

cout<<"Given array:\n121 432 94 873 564 23 1 45 788\nSorted array:\n";

int array[] = {121, 432, 94, 873, 564, 23, 1, 45, 788};

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

radixsort(array, n);

printArray(array, n);

}

BRUTE-FORCE

#include<iostream>

using namespace std;

main(){

//enter x-cordinates respectively

int px[12]={2,13,4,5,4,7,7,9,11,12,14,15};

//enter y-cordinates respectively

int py[12]={5,3,4,1,11,7,13,10,5,12,10,7};

int i,j;

for(i=0;i<=11;i++){

bool maxima=true;

for(j=0;j<=11;j++){

if(!(i=j)&(px[i]<=px[j])&&(py[i]<=py[j])){

maxima = false;

}

}

if(maxima=true){

cout<<"("<<px[i]<<","<<py[i]<<") ";

}

}

}

PLANE-SWEEP

#include<iostream>

#include<stack>

using namespace std;

main(){

//enter x-cordinates respectively

int px[12]={2,13,4,5,4,7,7,9,11,12,14,15};

//enter y-cordinates respectively

int py[12]={5,3,4,1,11,7,13,10,5,12,10,7};

int i,j;

stack<int> stx;

stack<int> sty;

for(i=0;i<=11;i++){

bool maxima=true;

for(j=0;j<=11;j++){

if((!(i=j))&&(px[i]<=px[j])&&(py[i]<=py[j])){

bool maxima = false;

}

}

if(maxima=true){

stx.push(px[i]);

sty.push(py[i]);

}

}

while ((!stx.empty())&&(!sty.empty())){

cout<<"("<<stx.top()<<","<<sty.top()<<") ";

stx.pop();

sty.pop();

}

}