



UNIVERSITY OF AGRICULTURE, FAISALABAD

ASSIGNMENT

Data Structure and Algorithm

CS-410

SUBMITTED TO:

Dr. Nayyar Iqbal SB

SUBMITTED BY:

Muhammad Hamza (2020-ag-6492)

Insertion Sort

```
1 #include<iostream>
2 using namespace std;
3 int main ()
4 {
5     int myarray[8] = { 77,33,44,11,88,22,66,55};
6
7     cout<<"Given array: \n";
8     for(int i=0;i<8;i++)
9     {
10         cout <<myarray[i]<<"\t";
11     }
12     for(int k=1; k<8; k++)
13     {
14         int temp = myarray[k];
15         int j= k-1;
16         while(j>=0 && temp <= myarray[j])
17         {
18             myarray[j+1] = myarray[j];
19             j = j-1;
20         }
21         myarray[j+1] = temp;
22     }
23     cout<<"\nSorted array: \n";
24     for(int i=0;i<8;i++)
25     {
26         cout <<myarray[i]<<"\t";
27     }
28 }
```

```
Given array:
77    33    44    11    88    22    66    55
Sorted array:
11    22    33    44    55    66    77    88
-----
Process exited after 0.08201 seconds with return value 0
Press any key to continue . . .
```

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;
}
```

Ex\Menhiran Arshad\Selection Output.exe

```
Given array;
77, 33, 44, 11, 88, 22, 66, 55
Sorted array:
11, 22, 33, 44, 55, 66, 77, 88,
-----
Process exited after 0.07889 seconds with return value 0
Press any key to continue . . . _
```

Merge Sort

```
1 #include <iostream>
2 using namespace std;
3 void merge(int arr[], int p, int q, int r) {
4     int n1 = q - p + 1;
5     int n2 = r - q;
6
7     int L[n1], M[n2];
8
9     for (int i = 0; i < n1; i++)
10         L[i] = arr[p + i];
11     for (int j = 0; j < n2; j++)
12         M[j] = arr[q + 1 + j];
13     int i, j, k;
14     i = 0;
15     j = 0;
16     k = p;
17     while (i < n1 && j < n2) {
18         if (L[i] <= M[j]) {
19             arr[k] = L[i];
20             i++;
21         } else {
22             arr[k] = M[j];
23             j++;
24         }
25         k++;
26     }
27     while (i < n1) {
28         arr[k] = L[i];
29         i++;
30         k++;
31     }
32     while (j < n2) {
33         arr[k] = M[j];
34         j++;
35         k++;
36     }
37 }
38 }
```

```
26 }
27 while (i < n1) {
28     arr[k] = L[i];
29     i++;
30     k++;
31 }
32
33 while (j < n2) {
34     arr[k] = M[j];
35     j++;
36     k++;
37 }
38 }
39 void mergeSort(int arr[], int l, int r) {
40     if (l < r) {
41         int m = l + (r - l) / 2;
42
43         mergeSort(arr, l, m);
44         mergeSort(arr, m + 1, r);
45         merge(arr, l, m, r);
46     }
47 }
48 void printArray(int arr[], int size) {
49     for (int i = 0; i < size; i++)
50         cout << arr[i] << " ";
51     cout << endl;
52 }
53 int main() {
54     cout << "Given array: \n66, 33, 40, 22, 55, 80, 20, 50, 44, 77, 30 \n";
55     int arr[] = {66, 33, 40, 22, 55, 80, 20, 50, 44, 77, 30};
56     int size = sizeof(arr) / sizeof(arr[0]);
57
58     mergeSort(arr, 0, size - 1);
59
60     cout << "Sorted array: \n";
61     printArray(arr, size);
62     return 0;
63 }
```

2: Merge Sort

Given array:

66, 33, 40, 22, 55, 80, 20, 50, 44, 77, 30

Sorted array:

20, 22, 30, 33, 40, 44, 50, 55, 66, 77, 80,

Process exited after 0.08405 seconds with return value 0

Press any key to continue . . .

Radix Sort

```
9 }
10 void countingSort(int array[], int size, int place) {
11     const int max = 10;
12     int output[size];
13     int count[max];
14
15     for (int i = 0; i < max; ++i)
16         count[i] = 0;
17     for (int i = 0; i < size; i++)
18         count[(array[i] / place) % 10]++;
19     for (int i = 1; i < max; i++)
20         count[i] += count[i - 1];
21     for (int i = size - 1; i >= 0; i--) {
22         output[count[(array[i] / place) % 10] - 1] = array[i];
23         count[(array[i] / place) % 10]--;
24     }
25
26     for (int i = 0; i < size; i++)
27         array[i] = output[i];
28 }
29 void radixsort(int array[], int size) {
30     int max = getMax(array, size);
31     for (int place = 1; max / place > 0; place *= 10)
32         countingSort(array, size, place);
33 }
34 void printArray(int array[], int size) {
35     int i;
36     for (i = 0; i < size; i++)
37         cout << array[i] << " ";
38     cout << endl;
39 }
40 int main() {
41     cout << "Given array:\n121 432 94 873 564 23 1 45 788\nSorted array:\n";
42     int array[] = {121, 432, 94, 873, 564, 23, 1, 45, 788};
43     int n = sizeof(array) / sizeof(array[0]);
44     radixsort(array, n);
45     printArray(array, n);
46 }
```

```

1 #include <iostream>
2 using namespace std;
3 int getMax(int array[], int n) {
4     int max = array[0];
5     for (int i = 1; i < n; i++)
6         if (array[i] > max)
7             max = array[i];
8     return max;
9 }
10 void countingSort(int array[], int size, int place) {
11     const int max = 10;
12     int output[size];
13     int count[max];
14
15     for (int i = 0; i < max; ++i)
16         count[i] = 0;
17     for (int i = 0; i < size; i++)
18         count[(array[i] / place) % 10]++;
19     for (int i = 1; i < max; i++)
20         count[i] += count[i - 1];
21     for (int i = size - 1; i >= 0; i--) {
22         output[count[(array[i] / place) % 10] - 1] = array[i];
23         count[(array[i] / place) % 10]--;
24     }
25
26     for (int i = 0; i < size; i++)
27         array[i] = output[i];
28 }
29 void radixsort(int array[], int size) {
30     int max = getMax(array, size);
31     for (int place = 1; max / place > 0; place *= 10)
32         countingSort(array, size, place);
33 }
34 void printArray(int array[], int size) {
35     int i;
36     for (i = 0; i < size; i++)
37         cout << array[i] << " ";
38     cout << endl;
39 }

```

E:\Mehran Arshad\Radix output.exe

```

Given array:
121 432 94 873 564 23 1 45 788
Sorted array:
1 23 45 94 121 432 564 788 873

-----
Process exited after 0.07637 seconds with return value 0
Press any key to continue . . .

```

Brute Force

```
brute force.cpp
1  #include<iostream>
2  using namespace std;
3  main(){
4      //enter x-coordinates respectively
5      int px[12]={2,13,4,5,4,7,7,9,11,12,14,15};
6      //enter y-coordinates respectively
7      int py[12]={5,3,4,1,11,7,13,10,5,12,10,7};
8      int i,j;
9      for(i=0;i<=11;i++){
10         bool maxima=true;
11         for(j=0;j<=11;j++){
12             if(!i=j)&(px[i]<=px[j])&&(py[i]<=py[j])){
13                 maxima = false;
14                 break;
15             }
16         }
17         if(maxima=true){
18             cout<<"("<<px[i]<<","<<py[i]<<")  ";
19         }
20     }
21 }
22 }
```

```
C:\Users\Lenovo\Documents\plane sweep.exe
(13,3) (12,12) (14,10) (15,7)
-----
Process exited after 2.011 seconds with return value 0
Press any key to continue . . .
```

Plane Sweep

```
brute force.cpp  plane sweep.cpp
1  #include<iostream>
2  #include<stack>
3  using namespace std;
4  main(){
5      //enter x-coordinates respectively
6      int px[12]={2,13,4,5,4,7,7,9,11,12,14,15};
7      //enter y-coordinates respectively
8      int py[12]={5,3,4,1,11,7,13,10,5,12,10,7};
9      int i,j;
10     stack<int> stx;
11     stack<int> sty;
12     for(i=0;i<=11;i++){
13         bool maxima=true;
14         for(j=0;j<=11;j++){
15             if((!(i=j))&&(px[i]<=px[j])&&(py[i]<=py[j])){
16                 bool maxima = false;
17                 break;
18             }
19         }
20         if(maxima=true){
21             stx.push(px[i]);
22             sty.push(py[i]);
23         }
24     }
25     while ((!stx.empty())&&(!sty.empty())){
26         cout<<" "<<stx.top()<<" "<<sty.top()<<" ";
27         stx.pop();
28         sty.pop();
29     }
30 }
31
32
```

C:\Users\Lenovo\Documents\plane sweep.exe

(13,3) (12,12) (14,10) (15,7)

Process exited after 2.011 seconds with return value 0
Press any key to continue . . .