\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Algorithm Comparison**



Session: 2021 – 2025

**Submitted by:**

M Faraz Ali 2021-CS-122

**Supervised by:**

Ms. Maida Shahid

Department of Computer Science

**University of Engineering and Technology Lahore Pakistan**

**Table Of Contents**

Short Description 03

Class Diagram 03

Wireframes 05

Time Analysis 07

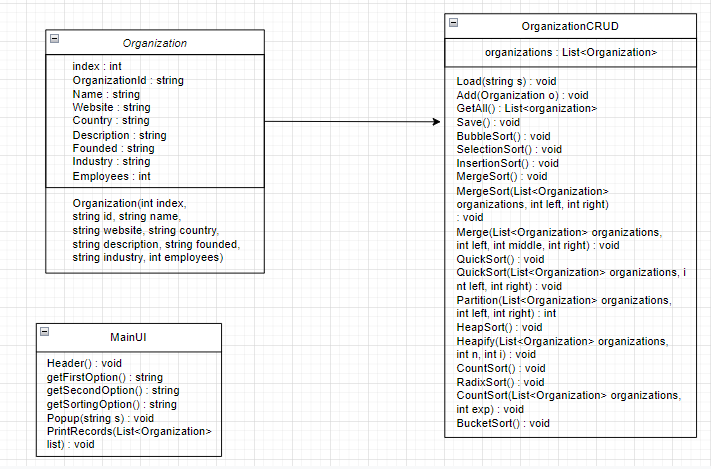
* For Sorted Data 09
* For Unsorted Data 09

Full Code 12

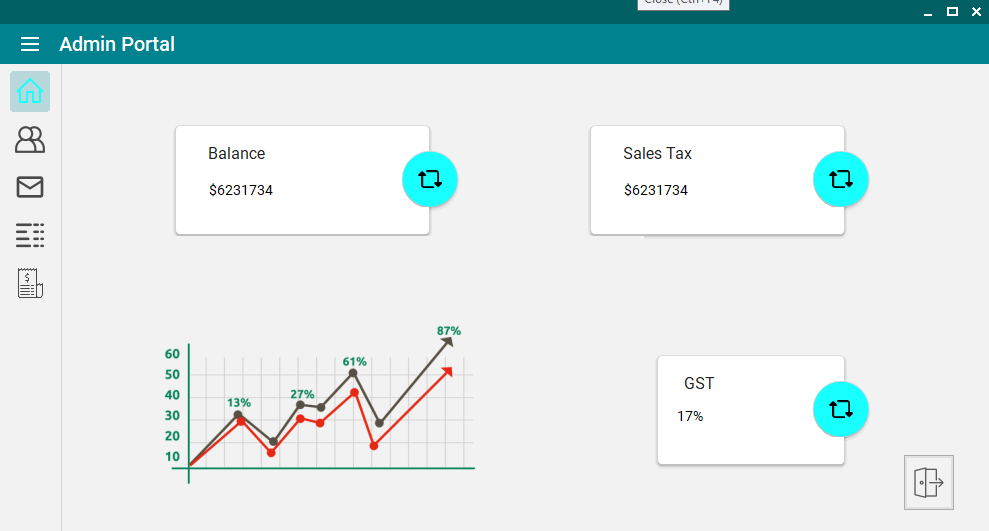
**Short Description**

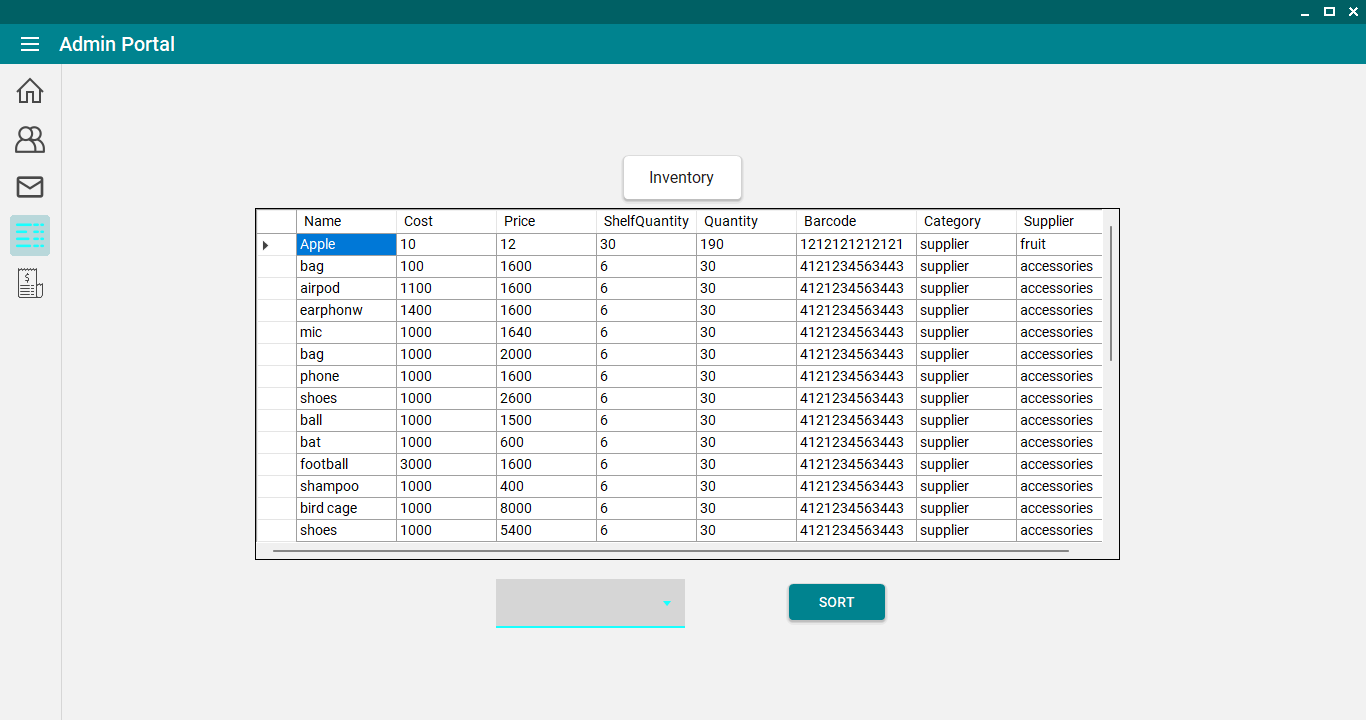
The choice of the algorithm conditions the performance as well as the feasibility of the project. Indeed “sorting a CSV file” of large size is not possible unless you have an exceptional high-performance computer. Give it multiple attempts and experiment and you will find that it will take almost forever to do so. Therefore, it is necessary to select the sorting algorithm carefully.

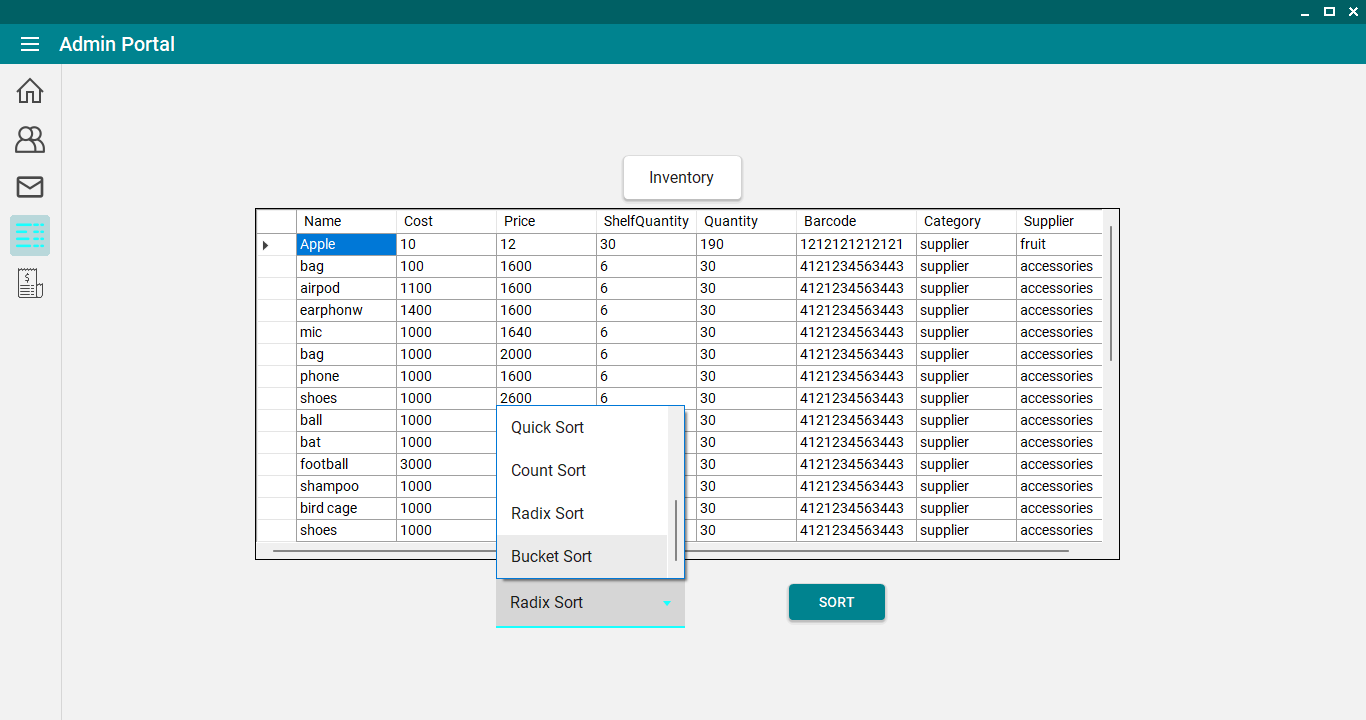
**Class Diagram**

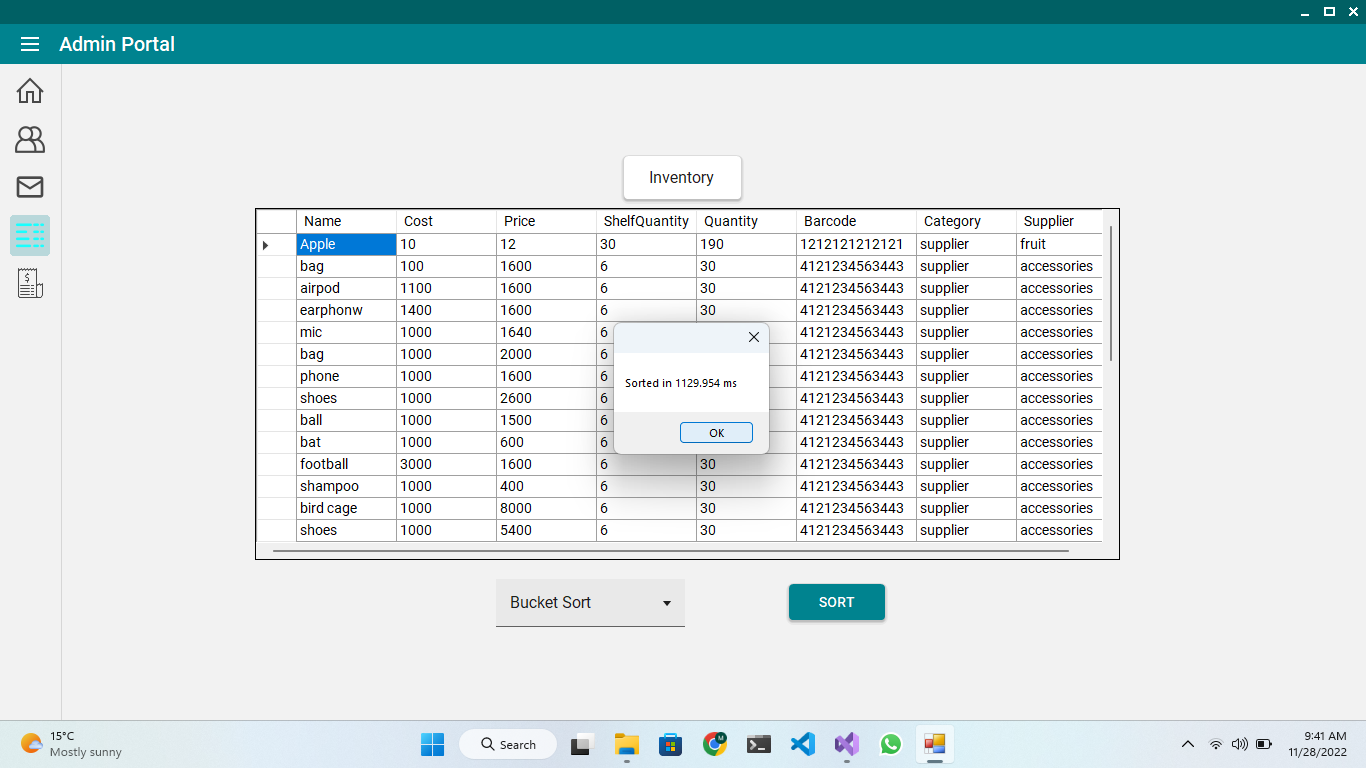
****

**Wireframes**

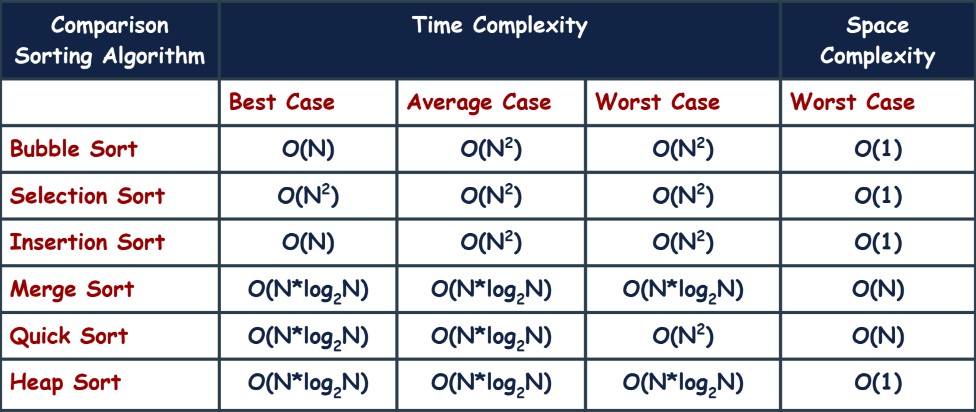


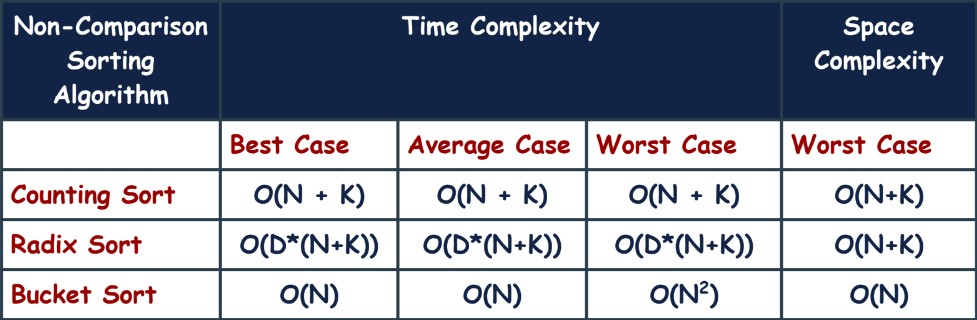






**General Time and space of Every Algorithm**





**Time Analysis (Sorted Data)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Bubble**  **Sort** | **Selection**  **Sort** | **Insertion**  **Sort** | **Merge**  **Sort** | **Quick**  **Sort** |
| **100** | 2 | 2 | 1 | 1 | 15 |
| **1000** | 46 | 18 | 0 | 0 | 38 |
| **10000** | 8606 | 3595 | 1 | 28 | 6284 |
| **100000** | 1236352 | 461903 | 13 | 190 | 724561 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Heap**  **Sort** | **Counting**  **Sort** | **Radix**  **Sort** | **Bucket**  **Sort** |
| **100** | 1 | 0 | 2 | 1 |
| **1000** | 1 | 1 | 1 | 1 |
| **10000** | 25 | 4 | 16 | 8 |
| **100000** | 187 | 37 | 155 | 37 |

**Discussion**

Bubble Sort:

Bubble sort has to do all comparisons irrespective of the fact that data is sorted or not. It has the highest time complexity that’s why it is the slowest in all data sets.

Selection Sort:

Selection sort also does all comparisons and is not affected by difference of sorted and unsorted data.

Insertion Sort:

Insertion sort is dependent on the sequence of the data. So, in case of sorted data, it takes least time of all.

Merge Sort:

Merge sort is independent of the sequence of data. But it still utilizes reasonable amount of time as its time complexity id low.

Quick Sort:

In case of quick sort, the corner values are selected as pivot and that’s why it behaves slow in sorted data. So, it is slowest after bubble and selection sort.

Heap Sort:

This sort works fine for sorted data. It is also independent of sequence and has low time complexity.

Count Sort:

This sort works fine for sorted data. It is also independent of sequence and has low time complexity.

Radix Sort:

This sort works fine for sorted data. It is also independent of sequence and has low time complexity.

Bucket Sort:

This sort works fine for sorted data. It is also independent of sequence and has low time complexity.

**Time Analysis (Unsorted Data)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Bubble**  **Sort** | **Selection**  **Sort** | **Insertion**  **Sort** | **Merge**  **Sort** | **Quick**  **Sort** |
| **100** | **2** | **1** | **1** | **2** | **19** |
| **1000** | **240** | **229** | **126** | **14** | **12** |
| **10000** | **50641** | **49028** | **10979** | **81** | **88** |

**.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Heap**  **Sort** | **Counting**  **Sort** | **Radix**  **Sort** | **Bucket**  **Sort** |
| **100** | **2** | **0** | **2** | **1** |
| **1000** | **13** | **5** | **9** | **2** |
| **10000** | **128** | **30** | **127** | **41** |

**Discussion**

Bubble Sort:

Bubble sort has to do all comparisons irrespective of the fact that data is sorted or not. It has the highest time complexity that’s why it is the slowest in all data sets.

Selection Sort:

Selection sort also does all comparisons and is not affected by difference of sorted and unsorted data.

Insertion Sort:

Insertion sort is dependent on the sequence of the data. So, in case of sorted data, it takes least time of all but in sorted data, it takes more time.

Merge Sort:

Merge sort is independent of the sequence of data. But it still utilizes reasonable amount of time as its time complexity is low.

Quick Sort:

In case of quick sort, the corner values are selected as pivot and that’s why it behaves slow in sorted data. So, it is slowest after bubble and selection sort in sorted data but in case of unsorted data, its performance is better.

Heap Sort:

Its best time complexity is O(n\*logN).

Count Sort:

Its best time complexity occurs when the elements are placed in jumbled order.

Radix Sort:

Its time consumption is just fine but not the best choice.

Bucket Sort:

It is among one of the fastest sorting algorithms.

**Full Code**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace CustomerManagementSystem.BL

{

public class Organization

{

public int index { get; set; }

public string OrganizationId { get; set; }

public string Name { get; set; }

public string Website { get; set; }

public string Country { get; set; }

public string Description { get; set; }

public string Founded { get; set; }

public string Industry { get; set; }

public int Employees { get; set; }

//parametrized constructor

public Organization(int index, string id, string name, string website, string country, string description, string founded, string industry, int employees)

{

this.index = index;

this.OrganizationId = id;

this.Name = name;

this.Website = website;

this.Country = country;

this.Description = description;

this.Founded = founded;

this.Industry = industry;

this.Employees = employees;

}

}

}

namespace CustomerManagementSystem.DL

{

public class OrganizationCRUD

{

static List<Organization> organizations = new List<Organization>();

public static void Load(string s)

{

DateTime start = DateTime.Now;

StreamReader sr = new StreamReader(s);

string str;

int count = 0;

while ((str = sr.ReadLine()) != null)

{

if (count == 0)

{

count++;

continue;

}

count++;

string[] arr = str.Split(',');

int index = Convert.ToInt32(arr[0]);

string id = arr[1];

string site = arr[2];

string country = arr[3];

string description = arr[4];

string founded = arr[5];

string industry = arr[6];

int employees = int.Parse(arr[7]);

string name = arr[8];

Organization o = new Organization(index, id, name, site, country, description, founded, industry, employees);

Add(o);

//Console.WriteLine(index);

}

sr.Close();

//get system time

DateTime end = DateTime.Now;

TimeSpan ts = end - start;

Console.WriteLine("Time taken to load {0} records: {1}", count, ts);

Console.ReadKey();

}

public static void Add(Organization o)

{

organizations.Add(o);

}

public static List<Organization> GetAll()

{

return organizations;

}

public static void Save()

{

StreamWriter sw = new StreamWriter("organizations.csv");

sw.WriteLine("Index,Id,Site,Country,Description,Founded,Industry,Employees,Name");

foreach (Organization o in organizations)

{

sw.WriteLine(o.index + "," + o.OrganizationId + "," + o.Website + "," + o.Country + "," + o.Description + "," + o.Founded + "," + o.Industry + "," + o.Employees + "," + o.Name);

}

sw.Close();

}

public static void BubbleSort()

{

for (int i = 0; i < organizations.Count; i++)

{

for (int j = 0; j < organizations.Count - 1; j++)

{

if (organizations[j].Employees > organizations[j + 1].Employees)

{

Organization temp = organizations[j];

organizations[j] = organizations[j + 1];

organizations[j + 1] = temp;

}

}

}

}

public static void SelectionSort()

{

for (int i = 0; i < organizations.Count; i++)

{

int min = i;

for (int j = i + 1; j < organizations.Count; j++)

{

if (organizations[j].Employees < organizations[min].Employees)

{

min = j;

}

}

Organization temp = organizations[i];

organizations[i] = organizations[min];

organizations[min] = temp;

}

}

public static void InsertionSort()

{

for (int i = 1; i < organizations.Count; i++)

{

Organization temp = organizations[i];

int j = i - 1;

while (j >= 0 && organizations[j].Employees > temp.Employees)

{

organizations[j + 1] = organizations[j];

j--;

}

organizations[j + 1] = temp;

}

}

public static void MergeSort()

{

MergeSort(organizations, 0, organizations.Count - 1);

}

public static void MergeSort(List<Organization> organizations, int left, int right)

{

if (left < right)

{

int middle = (left + right) / 2;

MergeSort(organizations, left, middle);

MergeSort(organizations, middle + 1, right);

Merge(organizations, left, middle, right);

}

}

public static void Merge(List<Organization> organizations, int left, int middle, int right)

{

int leftLength = middle - left + 1;

int rightLength = right - middle;

List<Organization> leftList = new List<Organization>();

List<Organization> rightList = new List<Organization>();

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

{

leftList.Add(organizations[left + i]);

}

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

{

rightList.Add(organizations[middle + 1 + j]);

}

int i1 = 0;

int j1 = 0;

int k = left;

while (i1 < leftLength && j1 < rightLength)

{

if (leftList[i1].Employees <= rightList[j1].Employees)

{

organizations[k] = leftList[i1];

i1++;

}

else

{

organizations[k] = rightList[j1];

j1++;

}

k++;

}

while (i1 < leftLength)

{

organizations[k] = leftList[i1];

i1++;

k++;

}

while (j1 < rightLength)

{

organizations[k] = rightList[j1];

j1++;

k++;

}

}

//quick sort iterative

public static void QuickSort()

{

QuickSort(organizations, 0, organizations.Count - 1);

}

public static void QuickSort(List<Organization> organizations, int left, int right)

{

Stack<int> stack = new Stack<int>();

stack.Push(left);

stack.Push(right);

while (stack.Count > 0)

{

int end = stack.Pop();

int start = stack.Pop();

int p = Partition(organizations, start, end);

if (p - 1 > start)

{

stack.Push(start);

stack.Push(p - 1);

}

if (p + 1 < end)

{

stack.Push(p + 1);

stack.Push(end);

}

}

}

public static int Partition(List<Organization> organizations, int left, int right)

{

int pivot = organizations[right].Employees;

int i = left - 1;

for (int j = left; j < right; j++)

{

if (organizations[j].Employees <= pivot)

{

i++;

Organization temp = organizations[i];

organizations[i] = organizations[j];

organizations[j] = temp;

}

}

Organization temp1 = organizations[i + 1];

organizations[i + 1] = organizations[right];

organizations[right] = temp1;

return i + 1;

}

//heap sort

public static void HeapSort()

{

int n = organizations.Count;

for (int i = n / 2 - 1; i >= 0; i--)

{

Heapify(organizations, n, i);

}

for (int i = n - 1; i >= 0; i--)

{

Organization temp = organizations[0];

organizations[0] = organizations[i];

organizations[i] = temp;

Heapify(organizations, i, 0);

}

}

public static void Heapify(List<Organization> organizations, int n, int i)

{

int largest = i;

int left = 2 \* i + 1;

int right = 2 \* i + 2;

if (left < n && organizations[left].Employees > organizations[largest].Employees)

{

largest = left;

}

if (right < n && organizations[right].Employees > organizations[largest].Employees)

{

largest = right;

}

if (largest != i)

{

Organization temp = organizations[i];

organizations[i] = organizations[largest];

organizations[largest] = temp;

Heapify(organizations, n, largest);

}

}

//count sort

public static void CountSort()

{

int max = 0;

for (int i = 0; i < organizations.Count; i++)

{

if (organizations[i].Employees > max)

{

max = organizations[i].Employees;

}

}

int[] count = new int[max + 1];

for (int i = 0; i < organizations.Count; i++)

{

count[organizations[i].Employees]++;

}

for (int i = 1; i < count.Length; i++)

{

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

}

Organization[] output = new Organization[organizations.Count];

for (int i = organizations.Count - 1; i >= 0; i--)

{

output[count[organizations[i].Employees] - 1] = organizations[i];

count[organizations[i].Employees]--;

}

for (int i = 0; i < organizations.Count; i++)

{

organizations[i] = output[i];

}

}

//radix sort

public static void RadixSort()

{

int max = 0;

for (int i = 0; i < organizations.Count; i++)

{

if (organizations[i].Employees > max)

{

max = organizations[i].Employees;

}

}

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

{

CountSort(organizations, exp);

}

}

public static void CountSort(List<Organization> organizations, int exp)

{

int[] count = new int[10];

for (int i = 0; i < organizations.Count; i++)

{

count[(organizations[i].Employees / exp) % 10]++;

}

for (int i = 1; i < count.Length; i++)

{

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

}

Organization[] output = new Organization[organizations.Count];

for (int i = organizations.Count - 1; i >= 0; i--)

{

output[count[(organizations[i].Employees / exp) % 10] - 1] = organizations[i];

count[(organizations[i].Employees / exp) % 10]--;

}

for (int i = 0; i < organizations.Count; i++)

{

organizations[i] = output[i];

}

}

//bucket sort

public static void BucketSort()

{

int max = 0;

for (int i = 0; i < organizations.Count; i++)

{

if (organizations[i].Employees > max)

{

max = organizations[i].Employees;

}

}

List<Organization>[] buckets = new List<Organization>[max + 1];

for (int i = 0; i < buckets.Length; i++)

{

buckets[i] = new List<Organization>();

}

for (int i = 0; i < organizations.Count; i++)

{

buckets[organizations[i].Employees].Add(organizations[i]);

}

int k = 0;

for (int i = 0; i < buckets.Length; i++)

{

for (int j = 0; j < buckets[i].Count; j++)

{

organizations[k] = buckets[i][j];

k++;

}

}

}

}

}

namespace CustomerManagementSystem.UI

{

internal class MainUI

{

public static void Header()

{

Console.Clear();

Console.WriteLine("================================================");

Console.WriteLine(" Sorting Algorithms Comparison ");

Console.WriteLine("================================================");

}

public static string getFirstOption()

{

Header();

Console.WriteLine("1. Load 100 Records");

Console.WriteLine("2. Load 1000 Records");

Console.WriteLine("3. Load 10000 Records");

Console.WriteLine("4. Load 100000 Records");

Console.WriteLine("5. Load 500000 Records");

Console.WriteLine("6. Go Back");

Console.Write("Enter your option: ");

string opt = Console.ReadLine();

return opt;

}

public static string getSecondOption()

{

Header();

Console.WriteLine("1. Sort Data");

Console.WriteLine("2. View Data");

Console.WriteLine("3. Save Records");

Console.WriteLine("4. Just Give Me CSV file for Sorted Records Time");

Console.WriteLine("5. Just Give Me CSV file for UnSorted Records Time");

Console.WriteLine("6. Go Back");

Console.Write("Enter your option: ");

string opt = Console.ReadLine();

return opt;

}

public static string getSortingOption()

{

Header();

Console.WriteLine("1. Bubble Sort" );

Console.WriteLine("2. Selection Sort");

Console.WriteLine("3. Insertion Sort");

Console.WriteLine("4. Merge Sort");

Console.WriteLine("5. Quick Sort");

Console.WriteLine("6. Heap Sort");

Console.WriteLine("7. Counting Sort");

Console.WriteLine("8. Radix Sort");

Console.WriteLine("9. Bucket Sort");

Console.WriteLine("10. Go Back");

Console.Write("Enter your option: ");

string opt = Console.ReadLine();

return opt;

}

public static void Popup(string s)

{

Console.WriteLine(s);

Console.ReadKey();

}

public static void PrintRecords(List<Organization> list)

{

Console.WriteLine("-----" + "\t\t" + "----------------" + "\t" + "--" + "\t\t\t" + "-----------------");

Console.WriteLine("Index" + "\t\t" + "No. of Employees" + "\t" + "ID" + "\t\t\t" + "Organization Name");

Console.WriteLine("-----" + "\t\t" + "----------------" + "\t" + "--" + "\t\t\t" + "-----------------");

foreach (Organization t in list)

{

Console.WriteLine(t.index + "\t\t" + t.Employees + "\t\t\t" + t.OrganizationId + "\t\t" + t.Name);

}

Console.ReadKey();

}

}

}

namespace CustomerManagementSystem

{

internal class Program

{

static void Main(string[] args)

{

firstMenu();

}

static void firstMenu()

{

string path="";

string opt = MainUI.getFirstOption();

switch (opt)

{

case "1":

path = "organizations-100.csv";

break;

case "2":

path = "organizations-1000.csv";

break;

case "3":

path = "organizations-10000.csv";

break;

case "4":

path = "organizations-100000.csv";

break;

case "5":

path = "organizations-500000.csv";

break;

case "6":

return;

default:

MainUI.Popup("Invalid Option");

firstMenu();

break;

}

OrganizationCRUD.Load(path);

SecondMenu();

}

static void SecondMenu()

{

string opt = UI.MainUI.getSecondOption();

switch (opt)

{

case "1":

ThirdMenu();

break;

case "2":

List<Organization> lst = OrganizationCRUD.GetAll();

MainUI.PrintRecords(lst);

break;

case "3":

OrganizationCRUD.Save();

break;

case "4":

SortedTime();

break;

case "5":

UnsortedTime();

break;

case "6":

firstMenu();

break;

default:

MainUI.Popup("Invalid Option");

SecondMenu();

break;

}

SecondMenu();

}

static void ThirdMenu()

{

string opt = MainUI.getSortingOption();

switch (opt)

{

case "1":

DateTime start = DateTime.Now;

OrganizationCRUD.BubbleSort();

DateTime end = DateTime.Now;

TimeSpan ts = end - start;

MainUI.Popup("Time taken: " + ts.TotalMilliseconds + " ms");

break;

case "2":

start = DateTime.Now;

OrganizationCRUD.SelectionSort();

end = DateTime.Now;

ts = end - start;

MainUI.Popup("Time taken: " + ts.TotalMilliseconds + " ms");

break;

case "3":

start = DateTime.Now;

OrganizationCRUD.InsertionSort();

end = DateTime.Now;

ts = end - start;

MainUI.Popup("Time taken: " + ts.TotalMilliseconds + " ms");

break;

case "4":

start = DateTime.Now;

OrganizationCRUD.MergeSort();

end = DateTime.Now;

ts = end - start;

MainUI.Popup("Time taken: " + ts.TotalMilliseconds + " ms");

break;

case "5":

start = DateTime.Now;

OrganizationCRUD.QuickSort();

end = DateTime.Now;

ts = end - start;

MainUI.Popup("Time taken: " + ts.TotalMilliseconds + " ms");

break;

case "6":

start = DateTime.Now;

OrganizationCRUD.HeapSort();

end = DateTime.Now;

ts = end - start;

MainUI.Popup("Time taken: " + ts.TotalMilliseconds + " ms");

break;

case "7":

start = DateTime.Now;

OrganizationCRUD.CountSort();

end = DateTime.Now;

ts = end - start;

MainUI.Popup("Time taken: " + ts.TotalMilliseconds + " ms");

break;

case "8":

start = DateTime.Now;

OrganizationCRUD.RadixSort();

end = DateTime.Now;

ts = end - start;

MainUI.Popup("Time taken: " + ts.TotalMilliseconds + " ms");

break;

case "9":

start = DateTime.Now;

OrganizationCRUD.BucketSort();

end = DateTime.Now;

ts = end - start;

MainUI.Popup("Time taken: " + ts.TotalMilliseconds + " ms");

break;

case "10":

SecondMenu();

break;

default:

MainUI.Popup("Invalid Option");

ThirdMenu();

break;

}

ThirdMenu();

}

static void SortedTime()

{

List<int> TimeSpan = new List<int>();

OrganizationCRUD.Load("organizations-100.csv");

OrganizationCRUD.MergeSort();

Console.WriteLine("100 Records Loaded and Sorted\n");

Console.WriteLine("Doing Bubble Sort...");

DateTime start = DateTime.Now;

OrganizationCRUD.BubbleSort();

DateTime end = DateTime.Now;

TimeSpan ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bubble Sort Completed!\n");

Console.WriteLine("Doing Selection Sort...");

start = DateTime.Now;

OrganizationCRUD.SelectionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Selection Sort Completed!\n");

Console.WriteLine("Doing Insertion Sort...");

start = DateTime.Now;

OrganizationCRUD.InsertionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Insertion Sort Completed!\n");

Console.WriteLine("Doing Merge Sort...");

start = DateTime.Now;

OrganizationCRUD.MergeSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Merge Sort Completed!\n");

Console.WriteLine("Doing Quick Sort...");

start = DateTime.Now;

OrganizationCRUD.QuickSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Quick Sort Completed!\n");

Console.WriteLine("Doing Heap Sort...");

start = DateTime.Now;

OrganizationCRUD.HeapSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Heap Sort Completed!\n");

Console.WriteLine("Doing Count Sort...");

start = DateTime.Now;

OrganizationCRUD.CountSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Count Sort Completed!\n");

Console.WriteLine("Doing Radix Sort...");

start = DateTime.Now;

OrganizationCRUD.RadixSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Radix Sort Completed!\n");

Console.WriteLine("Doing Bucket Sort...");

start = DateTime.Now;

OrganizationCRUD.BucketSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bucket Sort Completed!\n");

Console.WriteLine("=======================================\n\n");

//1000 records

OrganizationCRUD.Load("organizations-1000.csv");

OrganizationCRUD.RadixSort();

Console.WriteLine("1000 Records Loaded and Sorted!\n");

Console.WriteLine("Doing Bubble Sort...");

start = DateTime.Now;

OrganizationCRUD.BubbleSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bubble Sort Completed!\n");

Console.WriteLine("Doing Selection Sort...");

start = DateTime.Now;

OrganizationCRUD.SelectionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Selection Sort Completed!\n");

Console.WriteLine("Doing Insertion Sort...");

start = DateTime.Now;

OrganizationCRUD.InsertionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Insertion Sort Completed!\n");

Console.WriteLine("Doing Merge Sort...");

start = DateTime.Now;

OrganizationCRUD.MergeSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Merge Sort Completed!\n");

Console.WriteLine("Doing Quick Sort...");

start = DateTime.Now;

OrganizationCRUD.QuickSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Quick Sort Completed!\n");

Console.WriteLine("Doing Heap Sort...");

start = DateTime.Now;

OrganizationCRUD.HeapSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Heap Sort Completed!\n");

Console.WriteLine("Doing Count Sort...");

start = DateTime.Now;

OrganizationCRUD.CountSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Count Sort Completed!\n");

Console.WriteLine("Doing Radix Sort...");

start = DateTime.Now;

OrganizationCRUD.RadixSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Radix Sort Completed!\n");

Console.WriteLine("Doing Bucket Sort...");

start = DateTime.Now;

OrganizationCRUD.BucketSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bucket Sort Completed!\n");

Console.WriteLine("=======================================\n\n");

//10000 records

OrganizationCRUD.Load("organizations-10000.csv");

OrganizationCRUD.BucketSort();

Console.WriteLine("10000 Records Loaded and Sorted\n");

Console.WriteLine("Doing Bubble Sort...");

start = DateTime.Now;

OrganizationCRUD.BubbleSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bubble Sort Completed!\n");

Console.WriteLine("Doing Selection Sort...");

start = DateTime.Now;

OrganizationCRUD.SelectionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Selection Sort Completed!\n");

Console.WriteLine("Doing Insertion Sort...");

start = DateTime.Now;

OrganizationCRUD.InsertionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Insertion Sort Completed!\n");

Console.WriteLine("Doing Merge Sort...");

start = DateTime.Now;

OrganizationCRUD.MergeSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Merge Sort Completed!\n");

Console.WriteLine("Doing Quick Sort...");

start = DateTime.Now;

OrganizationCRUD.QuickSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Quick Sort Completed!\n");

Console.WriteLine("Doing Heap Sort...");

start = DateTime.Now;

OrganizationCRUD.HeapSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Heap Sort Completed!\n");

Console.WriteLine("Doing Count Sort...");

start = DateTime.Now;

OrganizationCRUD.CountSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Count Sort Completed!\n");

Console.WriteLine("Doing Radix Sort...");

start = DateTime.Now;

OrganizationCRUD.RadixSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Radix Sort Completed!\n");

Console.WriteLine("Doing Bucket Sort...");

start = DateTime.Now;

OrganizationCRUD.BucketSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bucket Sort Completed!\n");

Console.WriteLine("=======================================\n\n");

//100000 records

OrganizationCRUD.Load("organizations-100000.csv");

OrganizationCRUD.BucketSort();

Console.WriteLine("100000 Records Loaded and Sorted\n");

Console.WriteLine("Doing Bubble Sort...");

start = DateTime.Now;

OrganizationCRUD.BubbleSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bubble Sort Completed!\n");

Console.WriteLine("Doing Selection Sort...");

start = DateTime.Now;

OrganizationCRUD.SelectionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Selection Sort Completed!\n");

Console.WriteLine("Doing Insertion Sort...");

start = DateTime.Now;

OrganizationCRUD.InsertionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Insertion Sort Completed!\n");

Console.WriteLine("Doing Merge Sort...");

start = DateTime.Now;

OrganizationCRUD.MergeSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Merge Sort Completed!\n");

Console.WriteLine("Doing Quick Sort...");

start = DateTime.Now;

OrganizationCRUD.QuickSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Quick Sort Completed!\n");

Console.WriteLine("Doing Heap Sort...");

start = DateTime.Now;

OrganizationCRUD.HeapSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Heap Sort Completed!\n");

Console.WriteLine("Doing Count Sort...");

start = DateTime.Now;

OrganizationCRUD.CountSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Count Sort Completed!\n");

Console.WriteLine("Doing Radix Sort...");

start = DateTime.Now;

OrganizationCRUD.RadixSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Radix Sort Completed!\n");

Console.WriteLine("Doing Bucket Sort...");

start = DateTime.Now;

OrganizationCRUD.BucketSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bucket Sort Completed!\n");

Console.WriteLine("=======================================\n\n");

//saving timeSpan to file

StreamWriter sw = new StreamWriter("SortedTimeSpan.csv");

sw.WriteLine(" ,Bubble Sort, Selection Sort, Insertion Sort, Merge Sort, Quick Sort, Heap Sort, Counting Sort, Radix Sort, Bucket Sort");

sw.WriteLine("100 Records," + TimeSpan[0] + "," + TimeSpan[1] + "," + TimeSpan[2] + "," + TimeSpan[3] + "," + TimeSpan[4] + "," + TimeSpan[5] + "," + TimeSpan[6] + "," + TimeSpan[7] + "," + TimeSpan[8]);

sw.WriteLine("1000 Records," + TimeSpan[9] + "," + TimeSpan[10] + "," + TimeSpan[11] + "," + TimeSpan[12] + "," + TimeSpan[13] + "," + TimeSpan[14] + "," + TimeSpan[15] + "," + TimeSpan[16] + "," + TimeSpan[17]);

sw.WriteLine("10000 Records," + TimeSpan[18] + "," + TimeSpan[19] + "," + TimeSpan[20] + "," + TimeSpan[21] + "," + TimeSpan[22] + "," + TimeSpan[23] + "," + TimeSpan[24] + "," + TimeSpan[25] + "," + TimeSpan[26]);

sw.WriteLine("100000 Records," + TimeSpan[27] + "," + TimeSpan[28] + "," + TimeSpan[29] + "," + TimeSpan[30] + "," + TimeSpan[31] + "," + TimeSpan[32] + "," + TimeSpan[33] + "," + TimeSpan[34] + "," + TimeSpan[35]);

sw.WriteLine("500000");

sw.Close();

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine("Congratulations!");

Console.WriteLine("TimeSpan Saved to SortedTimeSpan.csv");

Console.ForegroundColor = ConsoleColor.White;

Console.ReadKey();

}

static void UnsortedTime()

{

List<int> TimeSpan = new List<int>();

OrganizationCRUD.Load("organizations-100.csv");

Console.WriteLine("100 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Bubble Sort...");

DateTime start = DateTime.Now;

OrganizationCRUD.BubbleSort();

DateTime end = DateTime.Now;

TimeSpan ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bubble Sort Completed!\n");

OrganizationCRUD.Load("organizations-100.csv");

Console.WriteLine("100 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Selection Sort...");

start = DateTime.Now;

OrganizationCRUD.SelectionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Selection Sort Completed!\n");

OrganizationCRUD.Load("organizations-100.csv");

Console.WriteLine("100 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Insertion Sort...");

start = DateTime.Now;

OrganizationCRUD.InsertionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Insertion Sort Completed!\n");

OrganizationCRUD.Load("organizations-100.csv");

Console.WriteLine("100 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Merge Sort...");

start = DateTime.Now;

OrganizationCRUD.MergeSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Merge Sort Completed!\n");

OrganizationCRUD.Load("organizations-100.csv");

Console.WriteLine("1000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Quick Sort...");

start = DateTime.Now;

OrganizationCRUD.QuickSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Quick Sort Completed!\n");

OrganizationCRUD.Load("organizations-100.csv");

Console.WriteLine("100 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Heap Sort...");

start = DateTime.Now;

OrganizationCRUD.HeapSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Heap Sort Completed!\n");

OrganizationCRUD.Load("organizations-100.csv");

Console.WriteLine("100 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Counting Sort...");

start = DateTime.Now;

OrganizationCRUD.CountSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Counting Sort Completed!\n");

OrganizationCRUD.Load("organizations-100.csv");

Console.WriteLine("100 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Radix Sort...");

start = DateTime.Now;

OrganizationCRUD.RadixSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Radix Sort Completed!\n");

OrganizationCRUD.Load("organizations-100.csv");

Console.WriteLine("100 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Bucket Sort...");

start = DateTime.Now;

OrganizationCRUD.BucketSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bucket Sort Completed!\n");

Console.WriteLine("=======================================\n\n");

//1000 records

OrganizationCRUD.Load("organizations-1000.csv");

Console.WriteLine("1000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Bubble Sort...");

start = DateTime.Now;

OrganizationCRUD.BubbleSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bubble Sort Completed!\n");

OrganizationCRUD.Load("organizations-1000.csv");

Console.WriteLine("1000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Selection Sort...");

start = DateTime.Now;

OrganizationCRUD.SelectionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Selection Sort Completed!\n");

OrganizationCRUD.Load("organizations-1000.csv");

Console.WriteLine("1000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Insertion Sort...");

start = DateTime.Now;

OrganizationCRUD.InsertionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Insertion Sort Completed!\n");

OrganizationCRUD.Load("organizations-1000.csv");

Console.WriteLine("1000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Merge Sort...");

start = DateTime.Now;

OrganizationCRUD.MergeSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Merge Sort Completed!\n");

OrganizationCRUD.Load("organizations-1000.csv");

Console.WriteLine("1000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Quick Sort...");

start = DateTime.Now;

OrganizationCRUD.QuickSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Quick Sort Completed!\n");

OrganizationCRUD.Load("organizations-1000.csv");

Console.WriteLine("1000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Heap Sort...");

start = DateTime.Now;

OrganizationCRUD.HeapSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Heap Sort Completed!\n");

OrganizationCRUD.Load("organizations-1000.csv");

Console.WriteLine("1000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Counting Sort...");

start = DateTime.Now;

OrganizationCRUD.CountSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Counting Sort Completed!\n");

OrganizationCRUD.Load("organizations-1000.csv");

Console.WriteLine("1000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Radix Sort...");

start = DateTime.Now;

OrganizationCRUD.RadixSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Radix Sort Completed!\n");

OrganizationCRUD.Load("organizations-1000.csv");

Console.WriteLine("1000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Bucket Sort...");

start = DateTime.Now;

OrganizationCRUD.BucketSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bucket Sort Completed!\n");

Console.WriteLine("=======================================\n\n");

//10000 records

OrganizationCRUD.Load("organizations-10000.csv");

Console.WriteLine("10000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Bubble Sort...");

start = DateTime.Now;

OrganizationCRUD.BubbleSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bubble Sort Completed!\n");

OrganizationCRUD.Load("organizations-10000.csv");

Console.WriteLine("10000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Selection Sort...");

start = DateTime.Now;

OrganizationCRUD.SelectionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Selection Sort Completed!\n");

OrganizationCRUD.Load("organizations-10000.csv");

Console.WriteLine("10000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Insertion Sort...");

start = DateTime.Now;

OrganizationCRUD.InsertionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Insertion Sort Completed!\n");

OrganizationCRUD.Load("organizations-10000.csv");

Console.WriteLine("10000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Merge Sort...");

start = DateTime.Now;

OrganizationCRUD.MergeSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Merge Sort Completed!\n");

OrganizationCRUD.Load("organizations-10000.csv");

Console.WriteLine("10000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Quick Sort...");

start = DateTime.Now;

OrganizationCRUD.QuickSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Quick Sort Completed!\n");

OrganizationCRUD.Load("organizations-10000.csv");

Console.WriteLine("10000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Heap Sort...");

start = DateTime.Now;

OrganizationCRUD.HeapSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Heap Sort Completed!\n");

OrganizationCRUD.Load("organizations-10000.csv");

Console.WriteLine("10000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Counting Sort...");

start = DateTime.Now;

OrganizationCRUD.CountSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Counting Sort Completed!\n");

OrganizationCRUD.Load("organizations-10000.csv");

Console.WriteLine("10000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Radix Sort...");

start = DateTime.Now;

OrganizationCRUD.RadixSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Radix Sort Completed!\n");

OrganizationCRUD.Load("organizations-10000.csv");

Console.WriteLine("10000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Bucket Sort...");

start = DateTime.Now;

OrganizationCRUD.BucketSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bucket Sort Completed!\n");

Console.WriteLine("=======================================\n\n");

//100000 records

OrganizationCRUD.Load("organizations-100000.csv");

Console.WriteLine("100000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Bubble Sort...");

start = DateTime.Now;

OrganizationCRUD.BubbleSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bubble Sort Completed!\n");

OrganizationCRUD.Load("organizations-100000.csv");

Console.WriteLine("100000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Selection Sort...");

start = DateTime.Now;

OrganizationCRUD.SelectionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Selection Sort Completed!\n");

OrganizationCRUD.Load("organizations-100000.csv");

Console.WriteLine("100000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Insertion Sort...");

start = DateTime.Now;

OrganizationCRUD.InsertionSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Insertion Sort Completed!\n");

OrganizationCRUD.Load("organizations-100000.csv");

Console.WriteLine("100000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Merge Sort...");

start = DateTime.Now;

OrganizationCRUD.MergeSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Merge Sort Completed!\n");

OrganizationCRUD.Load("organizations-100000.csv");

Console.WriteLine("100000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Quick Sort...");

start = DateTime.Now;

OrganizationCRUD.QuickSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Quick Sort Completed!\n");

OrganizationCRUD.Load("organizations-100000.csv");

Console.WriteLine("100000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Heap Sort...");

start = DateTime.Now;

OrganizationCRUD.HeapSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Heap Sort Completed!\n");

OrganizationCRUD.Load("organizations-100000.csv");

Console.WriteLine("100000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Counting Sort...");

start = DateTime.Now;

OrganizationCRUD.CountSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Counting Sort Completed!\n");

OrganizationCRUD.Load("organizations-100000.csv");

Console.WriteLine("100000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Radix Sort...");

start = DateTime.Now;

OrganizationCRUD.RadixSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Radix Sort Completed!\n");

OrganizationCRUD.Load("organizations-100000.csv");

Console.WriteLine("100000 Unsorted Records Loaded!\n");

Console.WriteLine("Doing Bucket Sort...");

start = DateTime.Now;

OrganizationCRUD.BucketSort();

end = DateTime.Now;

ts = end - start;

TimeSpan.Add(Convert.ToInt32(ts.TotalMilliseconds));

Console.WriteLine("Time taken: " + ts.TotalMilliseconds + " ms\n");

Console.WriteLine("Bucket Sort Completed!\n");

Console.WriteLine("=======================================\n\n");

//saving timeSpan to file

StreamWriter sw = new StreamWriter("UnSortedTimeSpan.csv");

sw.WriteLine(" ,Bubble Sort, Selection Sort, Insertion Sort, Merge Sort, Quick Sort, Heap Sort, Counting Sort, Radix Sort, Bucket Sort");

sw.WriteLine("1000," + TimeSpan[0] + "," + TimeSpan[1] + "," + TimeSpan[2] + "," + TimeSpan[3] + "," + TimeSpan[4] + "," + TimeSpan[5] + "," + TimeSpan[6] + "," + TimeSpan[7] + "," + TimeSpan[8]);

sw.WriteLine("10000," + TimeSpan[9] + "," + TimeSpan[10] + "," + TimeSpan[11] + "," + TimeSpan[12] + "," + TimeSpan[13] + "," + TimeSpan[14] + "," + TimeSpan[15] + "," + TimeSpan[16] + "," + TimeSpan[17]);

sw.WriteLine("100000," + TimeSpan[18] + "," + TimeSpan[19] + "," + TimeSpan[20] + "," + TimeSpan[21] + "," + TimeSpan[22] + "," + TimeSpan[23] + "," + TimeSpan[24] + "," + TimeSpan[25] + "," + TimeSpan[26]);

sw.WriteLine("500000");

sw.Close();

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine("Congratulations!");

Console.WriteLine("TimeSpan Saved to UnSortedTimeSpan.csv");

Console.ForegroundColor = ConsoleColor.White;

Console.ReadKey();

}

}

}