3.13 Merge Sort



This section will guide you to:

* Create a Windows Console project in Visual Studio to demonstrate the use of Merge Sort
* Create a method runApp() that will use Merge Sort to sort marks of subjects in an ascending order through mergeSort() and merge() methods

**Development Environment**

* Visual Studio 2019 Community Version
* Windows 10

This guide has five subsections, namely:

* + 1. Creating a Windows Console project in Visual Studio to demonstrate the use of Merge Sort
    2. Adding a method runApp() in Program class that will use Merge Sort to sort marks of subjects in an ascending order through mergeSort() and merge() methods
    3. Building the project
    4. Publishing and running the project
    5. Pushing the code to your GitHub repositories

**Step 3.13.1:** Creating a Windows Console project in Visual Studio to demonstrate the use of Merge Sort

* Open Visual Studio.
* From the top menu, select **File->New->Project.**
* In **Create A New Project** screen, select **Console app (.NET Core)** from the list of available project types and click on **Next.**
* Enter **Project Name** as **Phase1Section4.31** and click on **Create.**
* This will create the files for a Windows Console project.

**Step 3.13.2:** Adding a method runApp() in Program class that will use Merge Sort to sort marks of subjects in an ascending order through mergeSort() and merge() methods

* Select **Program.cs** as the current Code tab.
* Enter the following code:

**using** System;

**namespace** Phase1Section4.\_31

{

**class** Program

{

**static** **void** Main(**string**[] args)

{

runApp();

}

**public** **static** **void** runApp()

{

**int**[] marks = **new** **int**[10];

marks[0] = 80;

marks[1] = 90;

marks[2] = 93;

marks[3] = 76;

marks[4] = 77;

marks[5] = 92;

marks[6] = 89;

marks[7] = 78;

marks[8] = 80;

marks[9] = 56;

mergeSort(marks, 0, 10 - 1);

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

{

Console.WriteLine(marks[i]);

}

}

**public** **static** **void** mergeSort(**int**[] arr, **int** p, **int** r)

{

**if** (p < r)

{

**int** q = (p + r) / 2;

mergeSort(arr, p, q);

mergeSort(arr, q + 1, r);

merge(arr, p, q, r);

}

}

**public** **static** **void** merge(**int**[] arr, **int** p, **int** q, **int** r)

{

**int** i, j, k;

**int** n1 = q - p + 1;

**int** n2 = r - q;

**int**[] L = **new** **int**[n1];

**int**[] R = **new** **int**[n2];

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

{

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

}

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

{

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

}

i = 0;

j = 0;

k = p;

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

{

**if** (L[i] <= R[j])

{

arr[k] = L[i];

i++;

}

**else**

{

arr[k] = R[j];

j++;

}

k++;

}

**while** (i < n1)

{

arr[k] = L[i];

i++;

k++;

}

**while** (j < n2)

{

arr[k] = R[j];

j++;

k++;

}

}

}

}

**Step 3.13.3:** Building the project

* From the top menu, choose **Build->Build Solution.**
* If any compile errors are shown, fix them as required.

**Step 3.13.4:** Publishing and running the project

* From the top menu, select **Debug->Start Without Debugging.**
* This will execute the program in a console window.

**Step 3.13.5:** Pushing the code to your GitHub repositories

Open your command prompt and navigate to the folder where you have created your files.

cd <folder path>

Initialize your repository using the following command:

git init

Add all the files to your git repository using the following command:

git add .

Commit the changes using the following command:

git commit -m “Changes have been committed.”

Push the files to the folder you created initially using the following command:

git push -u origin master