Bahria University,

Karachi Campus

A picture containing text, room

Description automatically generated

LAB EXPERIMENT NO.

**07**

LIST OF TASKS

|  |  |
| --- | --- |
| TASK NO | OBJECTIVE |
|  | MERGE SORT AND QUICK SORT |
| 1 | Implement recursive method of merge sort algorithm to sort an array of 10 characters |
| 2 | value as Implement Quick Sort Algorithm on string array using left value as first pivoting value. |

Submitted On:

**- 12 - 2022**

(Date: DD/MM/YY)

**Task # 1:** Implement recursive method of merge sort algorithm to sort an array of 10 characters.

**Solution**

using System;

namespace LAb07

{

class MergeSort

{

static void merge(char[] arr, int low, int mid, int high)

{

int i, j, k;

char[] temp = new char[low+high+1];

i = low;

j = mid + 1;

k = low;

while (i <= mid && j <= high)

{

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

{

temp[k] = arr[i];

i++;

}

else

{

temp[k] = arr[j];

j++;

}

k++;

}

while (i<=mid)

{

temp[k] = arr[i];

i++;

k++;

}

while (j<=high)

{

temp[k] = arr[j];

j++;

k++;

}

for (int m = low; m <= high; m++)

{

arr[m] = temp[m];

}

}

static void mergesort(char []arr,int low,int high)

{

int mid;

if (low<high)

{

mid = (low + high) / 2;

mergesort(arr,low,mid);

mergesort(arr,mid+1,high);

merge(arr,low,mid,high);

}

}

static void printarrr(char []arr)

{

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

{

Console.Write(arr[i]);

Console.Write(" ");

}

}

public static void Main(String[] args)

{

int size;

Console.Write("Enter Size of character you want to enter in array : ");

size = int.Parse(Console.ReadLine());

char[] character = new char[size];

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

{

Console.Write($"Enter Character {i+1} : ");

character[i] = char.Parse(Console.ReadLine());

}

Console.WriteLine("Before Sorting ");

printarrr(character);

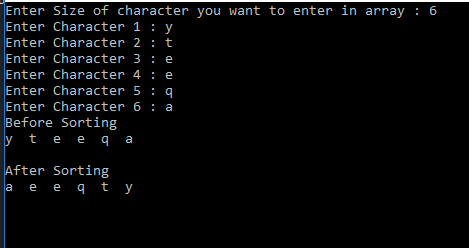
Console.WriteLine("\n\nAfter Sorting ");

mergesort(character, 0, character.Length - 1);

printarrr(character);

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

**OUTPUT:**



**Task # 2:** value as Implement Quick Sort Algorithm on string array using left value as first pivoting value.

**Solution**

using System;

namespace LAb07

{

class MergeSort

{

static void quicksort(string[] arr, int low, int high)

{

if (low < high)

{

int partitionIndex = partition(arr, low, high);

quicksort(arr, low, partitionIndex - 1);

quicksort(arr, partitionIndex + 1, high);

}

}

static int partition(string[] arr, int low, int high)

{

string temp;

string pivot = Convert.ToString(arr[low].ToLower()[0]);

int i = low + 1;

int j = high;

do

{

while (arr[i].ToLower()[0] <= pivot[0] && i < j)

{

i++;

}

while (arr[j].ToLower()[0] > pivot[0])

{

j--;

}

if (i < j)

{

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

} while (i < j);

//temp = pivot;

temp = arr[low];

arr[low] = arr[j];

arr[j] = temp;

return j;

}

static void print\_array(string[] arr)

{

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

{

Console.Write(arr[i].ToLower() + " ");

}

Console.WriteLine(" ");

}

public static void Main(String[] args)

{

string[] arr = { "DSA", "is", "Logic", "Building", "Subject", "Created", "By", "SABIR" };

Console.WriteLine("\nBefore Sorting\n");

Console.Write("\t> \"");

print\_array(arr);

Console.WriteLine("\n\nAfter Sorting\n");

Console.Write("\t> ");

quicksort(arr, 0, arr.Length - 1);

print\_array(arr);

Console.WriteLine("\n");

Console.ReadLine();

}

}

}

**OUTPUT:**

