**Experiment-No.3**

**Objective**: Implement the **Merge sort** algorithm to sort the given list of N numbers and plot graph.

|  |  |  |
| --- | --- | --- |
| Scheduled Date: | Compiled Date: | Submitted Date: |
| 23-8-2020 | 24-8-2020 | 30-8-2020 |

**Algorithm:**

MERGE-SORT(A, p, r)

1. If p < r

2. q = [ ( p + q ) /2 ]

3. MERGE-SORT(A, p, q)

4. MERGE-SORT(A, q+1, r)

5. MERGE(A, p, q, r)

MERGE (A, p, q, r)

1. n1 = q – p +1

2. n2 = r – q

3. let L [1.. n1 + 1 ] and L [1..n2 + 1 ] be new arrays

4. for i=1 to n1

5. L[ i ] = A [ p + i -1]

6. for j=1 to n2

7. R[ j ]= A[ q + j]

8. L [n1 + 1 ] = ∞

9. R [n2 + 1 ] = ∞

10. i = 1

11. j = 1

12. for k = p to r

13. if L[ i ] < R [ j ]

14. A[ k ] = L[ i ]

15. i = i + 1

16. else A[ k ] = R [ j]

17. j = j + 1

#include<stdio.h>

int count=0;

int main()

{

void mergeSort(int [], int , int );

void put\_data(int [], int);

void get\_data(int [], int );

int arr[100];

int n;

printf("Enter the array size\n");

scanf("%d",&n);

printf("Enter the array elements\n");

get\_data(arr, n);

printf("unsorted array is \n");

put\_data(arr, n);

mergeSort(arr, 0, n-1);

printf("\nSorted array is \n");

put\_data(arr, n);

printf("For n=%d and count=%d",n,count);

return 0;

}

void merge(int arr[], int l, int m, int r)

{

count++;

int i, j, k;

int n1 = m - l + 1;

int n2 = r - m;

count++;

int L[n1+1],R[n2+1];

count++;

for (i = 1; i <= n1; i++)

{

count++;

count++;

L[i] = arr[l + i-1];

count++;

}

for (j = 1; j <= n2; j++)

{

count++;

count++;

R[j] = arr[m + j];

count++;

}

L[n1+1]=999;

R[n2+1]=999;

i = 1;

j = 1;

count++;

for(k=l;k<=r;k++)

{

count++;

count++;

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

{

count++;

arr[k]=L[i];

count++;

i=i+1;

count++;

}

else

{

arr[k]=R[j];

count++;

j=j+1;

count++;

}

count++;

}

}

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

{

if (l < r)

{

count++;

int m = (l + r) / 2;

count++;

mergeSort(arr, l, m);

count++;

mergeSort(arr, m + 1, r);

count++;

merge(arr, l, m, r);

count++;

}

}

void get\_data(int arr[],int n)

{

int i;

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

{

scanf("%d",&arr[i]);

}

}

void put\_data(int A[], int size)

{

int i;

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

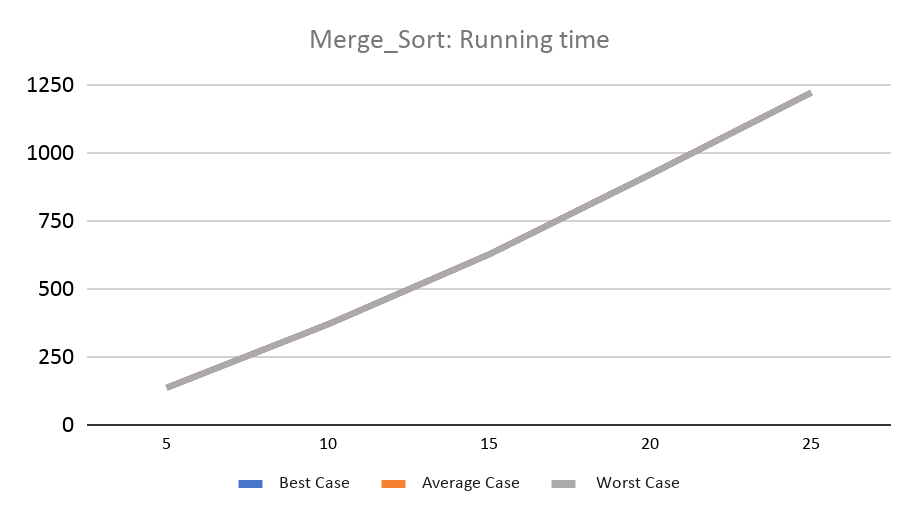
printf("%d ", A[i]);

printf("\n");

}

**Output:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Input Size** | **Best Case** | **Average Case** | **Worst Case** |
| 5 | 139 | 139 | 139 |
| 10 | 372 | 372 | 372 |
| 15 | 629 | 629 | 629 |
| 20 | 923 | 923 | 923 |
| 25 | 1224 | 1224 | 1224 |

**Graph:**

**Conclusion:**

|  |  |  |
| --- | --- | --- |
| **Case** | **Running Time : Growth of Function mathematically** | **Running Time : Growth of Function after observing graph** |
| **Best Case** | O(nlogn) | O()nlogn |
| **Average Case** | O(nlogn) | O(nlogn) |
| **Worst Case** | O()nlogn | O(nlogn) |