

4.Sort a given set of N integer elements using Merge Sort technique and compute its time taken. Run the program for different values of N and record the time taken to sort.

Code:

```
#include <stdio.h>
#include <stdlib.h>
```

```
void merge(int low,int mid,int high,int a[])
```

```
{
    int c[50];
    int i,j,k;
    i=low;
    j=mid+1;
    k=low;
    while(i<=mid&& j<=high)
    {
        if(a[i]<a[j])
        {
            c[k]=a[i];
            i++;
            k++;
        }
        else{
            c[k]=a[j];
            j++;
            k++;
        }
    }
    while(i<=mid)
    {
        c[k]=a[i];
        k++;
        i++;
    }
    while(j<=high)
    {
        c[k]=a[j];
        k++;
        j++;
    }
    for(i=low;i<=high;i++)
    {
```

```

        a[i]=c[i];

    }
}

void mergeSort(int low, int high,int a[])
{
    if(low<high)
    {
        int mid=(low+high)/2;
        mergeSort(low, mid, a);
        mergeSort(mid+1,high,a);
        merge(low, mid, high, a);

    }
}

void main()
{
    int n;
    printf("\nEnter the size");
    scanf("%d",&n);
    int i, a[50], low=0, high=n-1;

    printf("Enter the elements to be sorted: ");
    for(i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }
    mergeSort(low, high, a);
    for(i=0;i<n;i++)
    {
        printf("%d\t",a[i]);
    }

}

```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS D:\VS Code\ADA> cd "d:\VS Code\ADA\" ; if ($?) { gcc monotonic.c -o monotonic } ; if ($?) { .\monotonic }

Enter the size7
Enter the elements to be sorted: 99 88 77 66 55 44 33
33 44 55 66 77 88 99
PS D:\VS Code\ADA> 
```

Observation:

13-07-23

Sort a given set of N integer elements using merge sort technique & compute its time taken

#include <stdio.h>

void merge (int low, int mid, int high, int a[])

```

{
    int c[100];
    int i, j, k;
    i = low;
    j = mid + 1;
    k = low;
    while (i <= mid & j <= high) {
        if (a[i] < a[j]) {
            c[k] = a[i];
            i++;
            k++;
        }
        else {
            c[k] = a[j];
            j++;
            k++;
        }
    }
    while (i <= mid) {
        c[k] = a[i];
        k++;
        i++;
    }
    for (i = low; i <= high; i++) {
        a[i] = c[i];
    }
}

```

void merge_sort (int low, int high, int a[])

```

{
    if (low < high) {
        int mid = (low + high) / 2;
        merge_sort (low, mid, a);
        merge_sort (mid + 1, high, a);
        merge (low, mid, high, a);
    }
}

```

void main ()

```

{
    int n;
    printf ("Enter the size n");
    scanf ("%d", &n);
    int i, a[100], low = 0, high = n - 1;
    printf ("Enter the elements to be sorted n");
    for (i = 0; i < n; i++) {
        scanf ("%d", &a[i]);
    }
    merge_sort (low, high, a);
    for (i = 0; i < n; i++) {
        printf ("%d ", a[i]);
    }
}

```

Output:

Enter size.

5

Enter elements to

Sorted is

3 4 5 10 12

merge sort technique

Output:


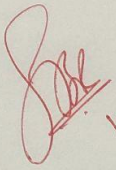
Enter size.

5

Enter elements to be sorted: 10 12 3 5 4

Sorted is

3 4 5 10 12.



13/7/23