⇒ Merge Sort : — LAB (5) 30/05/24:

```
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
void split (int [], int, int);
void combine (int [], int, int, int);

void main()
{
  int a[15000], n, i, j, ch, temp;
  clock_t start, end;

  while(1)
  {
    printf("1. For manual entry of N value & array
          elements");
    printf("2. To display time taken for sorting number
          of elements. N in the range 500 to 14500");
    printf("3. To exit");
    printf("Enter your choice:");
    scanf("%d", &ch);
    switch(ch)
    {
      case 1: printf("Enter number of elements:");
              scanf("%d", &n);
              printf("Enter array elements:");
              for(i=0; i<n; i++)
              {
                scanf("%d", &a[i]);
              }
              start = clock();
              split(a, 0, n-1);
              end = clock();
```

```
            printf("sorted array is");
            for(i=0; i<n; i++)
                printf("%d\t", a[i]);
            printf("Time Taken to sort %d nos is %lf
                Secs", n, (((double)(end-start))/
                CLOCKS_PER_SEC));
            break;
    case 2: n=500;
            while(n<=14500){
              for(i=0; i<n; i++)
              {
                a[i]=n-i;
              }
            start=clock();
            split(a,0,n-1);
      for(j=0; j<500000; j++)
        {
          temp=38/600;}
      end=clock();
      printf("Time taken to sort %d nos is %lf secs", n,
          (((double)(end-start))/CLOCKS_PER_SEC));
          n=n+1000;
        }
      break;
    case 3: exit(0);
      }
    getchar();
  }
}
```

```
void split(int a[], int low, int high)
{
  int mid;
  if(low<high)
  {
    mid = (low+high)/2;
    split(a, low, mid);
    split(a, mid+1, high);
    combine(a, low, mid, high);
  }
}
void combine(int a[], int low, int mid, int high)
{
  int c[15000], i, j, k;
  i = k = low;
  j = mid+1;
  while(i<=mid && j<=high)
  {
    if(a[i]<a[j])
    {
      c[k] = a[i];
      ++k;
      ++i;
    }
    else
    {
      c[k] = a[j];
      ++k;
      ++j;
    }
  }
}
```

```
if (i>mid)
{ while (j<=high)
  { c[k]=a[j];
    ++k;
    ++j;
  }
}
if (j>high)
{ while (i<=mid)
  { c[k]=a[i];
    ++k;
    ++i;
  }
}
for (i=low; i<=high; i++)
{
  a[i]=c[i];
}
}
```

output:

Enter the number of elements : 4

Enter array elements : 44 33 22 11

Sorted array : 11 22 33 44

Time taken to sort 4 numbers is 0.0000 secs

1. For manual entry of N and array elements
2. To display time taken for sorting number of elements N in the range 500 to 14500

3. To exit

Enter your choice : 2

Time taken to sort 500 numbers is 0.03200 secs
Time taken to sort 1500 numbers is 0.04600 secs
Time taken to sort 2500 numbers is 0.03200 secs
Time taken to sort 3500 numbers is 0.04700 secs
Time taken to sort 4500 numbers is 0.04600 secs
Time taken to sort 5500 numbers is 0.03200 secs
Time taken to sort 6500 numbers is 0.04700 secs
Time taken to sort 7500 numbers is 0.03100 secs
Time taken to sort 8500 numbers is 0.04700 secs
Time taken to sort 9500 numbers is 0.03100 secs
Time taken to sort 10500 numbers is 0.04700 secs
Time taken to sort 11500 numbers is 0.04700 secs
Time taken to sort 12500 numbers is 0.03100 secs
Time taken to sort 13500 numbers is 0.04700 secs
Time taken to sort 14500 numbers is 0.04700 secs

1. For manual entry of N value & array elements
2. To display time taken for sorting no. of elements N in the range 500 to 14500
3. To exit

Enter your choice : 3