Merge Sort.

```c
#include < stdio.h>
#include < time.h>
#include < stdlib.h>
void split ( int[], 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 ( "In1: manual enty of N value");
        printf ("\n 2: To display sorting time ");
        printf ("bf ntee your choice: ");
        scanf ("%d", &ch);
        switch (ch)
        {
        case1: printf ("\n Enter the no. of elements :");
            scanf ("%d", &n);
            switch (ch)
            {
            case1: printf ("\n Enter the elements:");
                scanf ("%d",&n) ;
                printf ("\n.Enter array elements: ");
                for( i=0; i<n; i++)
                {
                    scanf ("%d", &a[i]);
                }
```

```c
start = clock();
split (a, 0, n-1);
end = clock();
printf ("Inserted array is :");
for (i=0; i<n; i++),
printf (" %d ", a[i]);
printf ("In Time taken to sort %d number is %f secs', n,
                (((double) (end- start))/CLOCK-PER-SEC));
break;

case 2 : n= 500 ;
        while ( n<=14,500).
        {
        for (i=0; i<n; i++)
        {
        a[i] = n-i ;
        }
        start = clock();
        split (o,0, n-1);
        for (j=0; j<50000; j++) (temp=38/600;)
        end = clock();

        printf ("In time taken to  sort", &(double)(end-start).
                                cLOCK-PER-SEC)
        n = n+1000;
        }
        break;
case 3: exit (0);
    getchar();
    }
}
```

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

Time complexity :-

n = 500	T = 0.001293
n = 1500	T = 0.0013155
n = 2500	T = 0.0014345
n = 3500	T = 0.0015552
n = 4500	T = 0.0016255
n = 12,500	T = 0.0024777
n = 13,500	T = 0.0025566
n = 14500	T = 0.0025643.