Selection Sort And Merge Sort

① Selection sort

→ Algorithm : sel_sort (a[0...n-1]

// Sort a given array by selection sort
// Input : An array a[0..n-1] of orderable elements
// Output : Array a[0..n-1] sort in ascending order

for i ← 0 to n-2 do
small_pos ← i
for j ← i+1 to n-1 do
if a[j] < a[small_pos]
small[pos] > j
end if
end for
swap a[i] and a[small-pos]
end for

→ Program

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

void selsort (int n, int a[]);

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

    while (1)
    {
```

```
printf("\n1: For manual entry of N values
        and array elements");
printf("\n2: To display time taken for sorting
        number of elements N in the
        range 500 to 14500");
printf("\n3 Enter your choice:");
scanf("%d", &ch);

switch(ch)
{
    case 1:
        printf("\n Enter no. of elements:");
        scanf("%d", &n);
        printf("\n Enter array elements:");
        for(i=0; i<n; i++)
        {
            scanf("%d", &a[i]);
        }
        start = clock();
        selsort(a, n);
        end = clock();
        printf("\n Sorted array is:");
        for(i=0; i<n; i++)
        {
            printf("%d\t", a[i]);
        }
        printf("\n Time taken to sort
                %d numbers is %f sec\n",
                n, ((double)(end-start))/
                CLOCKS_PER_SEC));
        break;
```

```
case 2:
        n = 500;
        while ( n <= 14500)
        {
            for (i=0; i<n; i++)
            {
                a[i] = rand() %1000;
            }
            start = clock();
            select (n, a);
            for (j=0; j<50000; j++)
            {
                temp = 38/600;
            }
            end = clock();
            printf ("\n time taken to sort
                %d numbers is %f \n", n,
                (((double)(end - start))/CLOCK_PER_SEC
            n += 1000;
        }
        break;

case 3:
        exit(0);
        break;

        default:
            printf ("Invalid choice. Please try again\n");

    }
    getchar();
}
    return 0;
}
```

```
void selsort(int n, int a[])
{
    int i, j, t, small, pos;
    for (i=0; i<n-1; i++)
    {
        pos = i;
        small = a[i];
        for (j=i+1; j<n; j++)
        {
            if (a[j] < small)
            {
                small = a[j];
                pos = j;
            }
        }
        t = a[i];
        a[i] = a[pos];
        a[pos] = t;
    }
}
```

30/8/24

(2) Merge Sort

```
Algorithm : combine (a[0..n-1], low, mid, high)
        i ← low
        j ← mid+1
        k ← low
        while i<=mid and j <= high
        do
        if a[i] < a[j]
            c[k] ← a[i]
            k ← k+1
            i ← i+1
        else
            c[k] ← a[j]
            k ← k+1
            j ← j+1
        end if
    end while
    if  i > mid
            while j <= high
            do
            c[k] ← a[j]
            k ← k+1
            j ← j+1
        end while
    end if
    if j > high
        while i <= mid do
            c[k] ← a[i]
            k ← k+1
            i ← i+1
        end while
    end if
```

if j > high
while i <= mid do
c[k] ← a[i]
k ← k+1
i ← i+1
end while
end if
if j > high
while i <= mid do
c[k] ← a[i]
k ← k+1
i ← i+1
end while
end if
for i ← low to high
do
a[i] ← c[i]
end for

Algorithm : split (a[0... n-1], low, high)
if low < high
mid ← (low + high)/2
split (a, low, mid)
split (a, mid+1, high)
combine (a, low, mid, high)
end if

Program

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

void split (int [], int, int);
void combine (int [], int, int, int);
void main()
{
    int a[150000], n, i, j, ch, temp;
    clock_t start, end;

    while (1)
    {
    printf ("\n1: For manual of N values and
                array elements");
    printf ("\n2: To display time taken for
                sorting number of elements
                N in the range 500 to
                14500");
    printf ("\n3 To exit");
        printf ("\n Enter your choice:");
        scanf ("%d", &ch);
        switch (ch)
        {
        case 1: printf ("\n Enter the no.
            no. of elements: ");
            scanf ("%d", &n);
            printf ("\n Enter array elements");
            for (i=0; i<n; i++)
            {
```

scanf ("%d", &a[i]);
}
start = clock();
split (a, 0, n-1);
end = clock();
printf ("\n sorted array is:");
for (i=0; i<n; i++)
printf ("%d\t", a[i]);
printf ("\n Time taken to sort %d numbers is
%f secs", n, ((double)(end-start)/
CLOCKS_PER_SEC));
break;

case 2:
n = 500;
while (n<= 14700)
{
for (i=0; i<n; i++)
{
a[i] = rand() %1000;
}
start = clock();
selsort (n, a)
for (j=0; j< 500000; j++)
{
temp = 38/600;
}
end = clock();
printf ("\n Time taken to sort %d
numbers is %f secs \n", n,
((double)(end-start))/
CLOCKS_PER_SEC));
n+=1000;

```
case 3:
        exit(0);
        break;

default:
        printf("Invalid choice. Please try again.");

}
getchar();
}
return 0;

void displaysplit(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];
}
}
```