Selection sort

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

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

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

   while(1) {
      printf(" 1:For manual entry of N value 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("\nEnter your choice: ");
      scanf("%d", &ch);

      switch(ch) {
         case 1:
            printf("Enter the number of elements: ");
            scanf("%d", &n);
            printf("Enter array elements: ");
            for(i = 0; i < n; i++) {
               scanf("%d", &a[i]);
            }
            start = clock();
            selsort(a, n);
            end = clock();
            printf("Sorted array is: ");
            for(i = 0; i < n; i++) {
               printf("%d\t", a[i]);
```

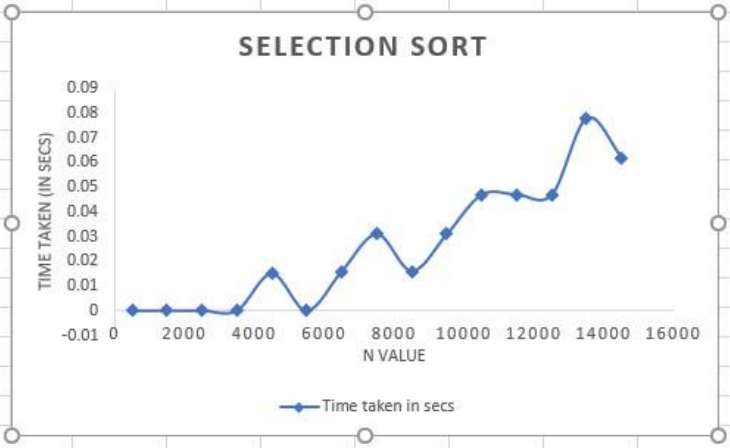
```
            }
            printf("Time taken to sort %d numbers is %f Secs\n", n, ((double)(end - start)) /
CLOCKS_PER_SEC);
            break;

         case 2:
            n = 500;
            while(n <= 14500) {
               for(i = 0; i < n; i++) {
                  a[i] = n - i;
               }
               start = clock();
               selsort(a, n);
               // Dummy loop to create delay
               for(j = 0; j < 500000; j++) {
                  temp = 38 / 600;
               }
               end = clock();
               printf("Time taken to sort %d numbers is %f Secs\n", n, ((double)(end - start)) /
CLOCKS_PER_SEC);
               n += 1000;
            }
            break;

         case 3:
            exit(0);

         default:
            printf("\nInvalid choice! Please try again.\n");
      }
   }
   return 0;
}
```

Output:

```
 1:For manual entry of N value 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: 1
Enter the number of elements: 4
Enter array elements: 44 33 22 11
Sorted array is: 11     22      33      44      Time taken to sort 4 numbers is 0.000000 Secs
 1:For manual entry of N value 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 0.000000 Secs
Time taken to sort 1500 numbers is 0.000000 Secs
Time taken to sort 2500 numbers is 0.000000 Secs
Time taken to sort 3500 numbers is 0.000000 Secs
Time taken to sort 4500 numbers is 0.015000 Secs
Time taken to sort 5500 numbers is 0.000000 Secs
Time taken to sort 6500 numbers is 0.016000 Secs
Time taken to sort 7500 numbers is 0.031000 Secs
Time taken to sort 8500 numbers is 0.016000 Secs
Time taken to sort 9500 numbers is 0.031000 Secs
Time taken to sort 10500 numbers is 0.047000 Secs
Time taken to sort 11500 numbers is 0.047000 Secs
Time taken to sort 12500 numbers is 0.047000 Secs
Time taken to sort 13500 numbers is 0.078000 Secs
Time taken to sort 14500 numbers is 0.062000 Secs
 1:For manual entry of N value 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: 3
```

Graph:

N value	Time taken in secs
500	0
1500	0
2500	0
3500	0
4500	0.015
5500	0
6500	0.016
7500	0.031
8500	0.016
9500	0.031
10500	0.047
11500	0.047
12500	0.047
13500	0.078
14500	0.062



MERGE SORT

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

void split(int[], int, int);
void combine(int[], int, int, int);

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

   while (1) {
      printf("1: For manual entry of N value 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("\nEnter your choice: ");
      scanf("%d", &ch);

      switch (ch) {
         case 1:
            printf("Enter the 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 numbers is %f Secs\n", 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);
               // Dummy loop to create delay
               for (j = 0; j < 50000000; j++) { temp = 38 / 600; }
```

```
                end = clock();
                printf("Time taken to sort %d numbers is %f Secs\n", n, ((double)(end - start)) /
CLOCKS_PER_SEC);
                n += 1000;
            }
            break;

        case 3:
            exit(0);

        default:
            printf("\nInvalid choice! Please try again.\n");
      }
   }
   return 0;
}

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:

```
1: For manual entry of N value 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: 1
Enter the number of elements: 4
Enter array elements: 44 33 22 11
Sorted array is: 11     22      33      44      Time taken to sort 4 numbers is 0.000000 Secs
1: For manual entry of N value 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 0.000000 Secs
Time taken to sort 1500 numbers is 0.000000 Secs
Time taken to sort 2500 numbers is 0.000000 Secs
Time taken to sort 3500 numbers is 0.000000 Secs
Time taken to sort 4500 numbers is 0.000000 Secs
Time taken to sort 5500 numbers is 0.000000 Secs
Time taken to sort 6500 numbers is 0.000000 Secs
Time taken to sort 7500 numbers is 0.000000 Secs
Time taken to sort 8500 numbers is 0.000000 Secs
Time taken to sort 9500 numbers is 0.000000 Secs
Time taken to sort 10500 numbers is 0.000000 Secs
Time taken to sort 11500 numbers is 0.000000 Secs
Time taken to sort 12500 numbers is 0.000000 Secs
Time taken to sort 13500 numbers is 0.016000 Secs
Time taken to sort 14500 numbers is 0.000000 Secs
1: For manual entry of N value 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: 3
```

Graph:

N value	SS time(secs)	MS time(secs)
500	0	0
1500	0	0
2500	0	0
3500	0	0
4500	0.015	0
5500	0	0
6500	0.016	0
7500	0.031	0
8500	0.016	0
9500	0.031	0
10500	0.047	0
11500	0.047	0
12500	0.047	0
13500	0.078	0.016
14500	0.062	0