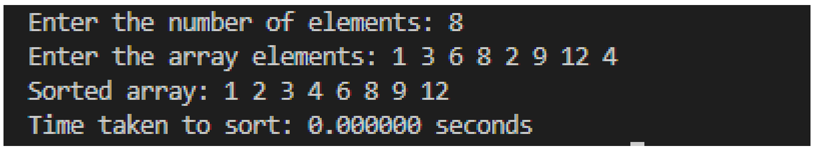
ADA LAB1 : MERGE SORT

```c
#include <stdio.h>

#include <time.h>

void merge(int[], int, int, int);

void merge_sort(int[], int, int);


int main() {

   int n, i;

   clock_t start, end;

   double time_taken;


   printf("Enter the number of elements: ");

   scanf("%d", &n);


   int a[n];

   printf("Enter the array elements: ");

   for (i = 0; i < n; i++) {

      scanf("%d", &a[i]);

   }

   start = clock();

   merge_sort(a, 0, n - 1);

   end = clock();

   time_taken = (double)(end - start) / CLOCKS_PER_SEC;

   printf("Sorted array: ");

   for (i = 0; i < n; i++) {

      printf("%d ", a[i]);

   }

   printf("\n");

   printf("Time taken to sort: %f seconds\n", time_taken);

   return 0;

}
```

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

void merge(int a[], int low, int mid, int high) {
    int i = low, j = mid + 1, k = 0;
    int c[high - low + 1];

    while (i <= mid && j <= high) {
        if (a[i] < a[j]) {
            c[k++] = a[i++];
        } else {
            c[k++] = a[j++];
        }
    }

    while (i <= mid) {
        c[k++] = a[i++];
    }

    while (j <= high) {
        c[k++] = a[j++];
    }
    for (i = low, k = 0; i <= high; i++, k++) {
        a[i] = c[k];
    }
}
```

Output:



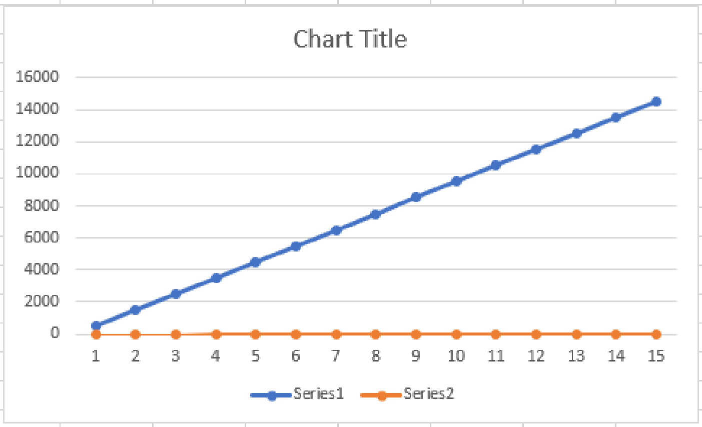
```
Enter the number of elements: 8
Enter the array elements: 1 3 6 8 2 9 12 4
Sorted array: 1 2 3 4 6 8 9 12
Time taken to sort: 0.000000 seconds
```

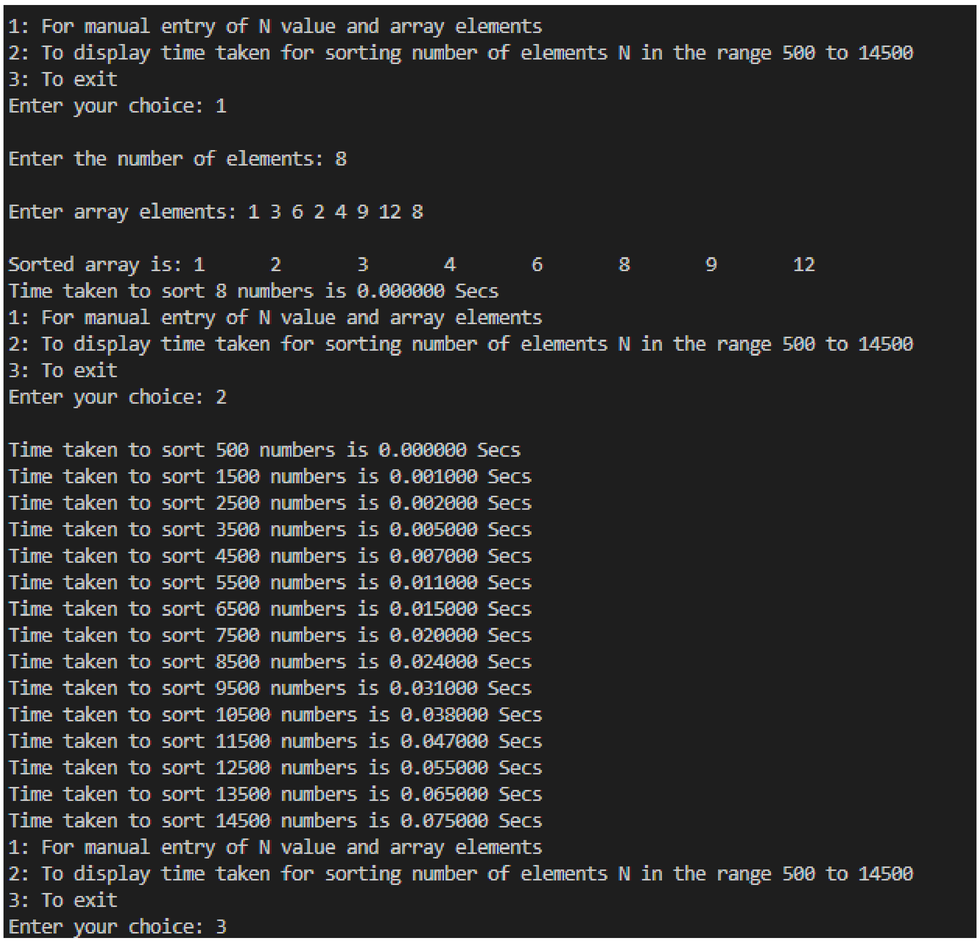
Graph

```
#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 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("\nEnter the number of elements: ");

            scanf("%d", &n);

            printf("\nEnter array elements: ");

            for (i = 0; i < n; i++) {
```

```
            scanf("%d", &a[i]);
          }
          start = clock();
          selsort(n, a);
          end = clock();
          printf("\nSorted array is: ");
          for (i = 0; i < n; i++)
            printf("%d\t", a[i]);
          printf("\nTime taken to sort %d numbers is %f 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();
            selsort(n, a);
            for (j = 0; j < 500000; j++) {
              temp = 38 / 600;
            }
            end = clock();
            printf("\nTime taken to sort %d numbers is %f Secs", n, ((double)(end - start)) /
CLOCKS_PER_SEC);
            n = n + 1000;
          }
          break;

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

array size	time
500	0
1500	0.001
2500	0.002
3500	0.005
4500	0.007
5500	0.011
6500	0.015
7500	0.02
8500	0.024
9500	0.031
10500	0.038
11500	0.047
12500	0.055
13500	0.065
14500	0.075





```
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: 8

Enter array elements: 1 3 6 2 4 9 12 8

Sorted array is: 1      2       3       4       6       8       9       12
Time taken to sort 8 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 0.001000 Secs
Time taken to sort 2500 numbers is 0.002000 Secs
Time taken to sort 3500 numbers is 0.005000 Secs
Time taken to sort 4500 numbers is 0.007000 Secs
Time taken to sort 5500 numbers is 0.011000 Secs
Time taken to sort 6500 numbers is 0.015000 Secs
Time taken to sort 7500 numbers is 0.020000 Secs
Time taken to sort 8500 numbers is 0.024000 Secs
Time taken to sort 9500 numbers is 0.031000 Secs
Time taken to sort 10500 numbers is 0.038000 Secs
Time taken to sort 11500 numbers is 0.047000 Secs
Time taken to sort 12500 numbers is 0.055000 Secs
Time taken to sort 13500 numbers is 0.065000 Secs
Time taken to sort 14500 numbers is 0.075000 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
```