Quick sort:

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

void swap(int* a, int* b)
{
   int t = *a;
   *a = *b;
   *b = t;
}
int partition (int arr[], int low, int high)
{
   int pivot = arr[high];
   int i = (low - 1);

   for (int j = low; j <= high- 1; j++)
   {
      if (arr[j] < pivot)
      {
         i++;
         swap(&arr[i], &arr[j]);
      }
   }
   swap(&arr[i + 1], &arr[high]);
   return (i + 1);
```

```
}
void quickSort(int arr[], int low, int high)
{
    if (low < high)
    {
        int pi = partition(arr, low, high);
        quickSort(arr, low, pi - 1);
        quickSort(arr, pi + 1, high);
    }
}
int main()
{
    int arr[100000],size,i;
    clock_t timereq;
    float cpu_time;

   //printf("Enter size\n");
   //scanf("%d",&size);
  for(size=25000;size<=100000;size=size+10000){
          i=0;

   srand(time(NULL));
          for(int i=0;i<size;i++){
          arr[i]=rand()%100;
          // printf("%d\t",arr[i]);
          }
```

```
        printf("\n");

   timereq= clock();
    quickSort(arr, 0, size-1);
    timereq=clock()-timereq;
    cpu_time=((float)(timereq))/CLK_TCK;
    printf("\nNote: ");
    //printArray(arr, size);
    for(i=0;arr[i]<=arr[i+1] && i<size ;i++);
    if(size-1 == i)
        printf("All the elements are Sorted\n");
    else
        printf("Elements are NOT  Sorted\n");

    printf("\n Time taken in seconds for arr %d: %f\n",size,cpu_time);

        }
    return 0;
}
```

D:\codes\quicksort.exe

```
Note: All the elements are Sorted

 Time taken in seconds for arr 25000: 0.007000


Note: All the elements are Sorted

 Time taken in seconds for arr 35000: 0.013000


Note: All the elements are Sorted

 Time taken in seconds for arr 45000: 0.021000


Note: All the elements are Sorted

 Time taken in seconds for arr 55000: 0.031000


Note: All the elements are Sorted

 Time taken in seconds for arr 65000: 0.042000


Note: All the elements are Sorted

 Time taken in seconds for arr 75000: 0.055000


Note: All the elements are Sorted

 Time taken in seconds for arr 85000: 0.071000


Note: All the elements are Sorted

 Time taken in seconds for arr 95000: 0.087000

--------------------------------
Process exited after 3.772 seconds with return value 0
Press any key to continue . . .
```

25000	0.007
35000	0.013
45000	0.021
55000	0.031
65000	0.042
75000	0.055
85000	0.071
95000	0.087