# Lab-6

Write a program to do quick sort

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

int main() {
    int a[15000], n, i, j, ch;
    float 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 is range 500 to 14500 :");
        printf("\n3 : Exit");
        printf("\n Enter your choice :");
        scanf("%d", &ch);
        switch (ch) {
            case 1:-
            printf("Enter number of elements :");
            scanf("%d", &n);
            printf("Enter array elements :");
            for (int i=0; i<n; i++){
            scanf("%d", &a[i]);
            }
            start = clock();
            quicksort(a, 0, n-1);
            end = clock();
            printf("sorted array is :");
            for(int i=0; i<n; i++)
                printf("%d\t", a[i]);
            printf("Time 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();
        quicksort(a, 0, n-1);
        for (j = 0; j < 50000000; j++)
            temp = 38/600;
        end = clock();
        printf("Time taken to sort %d numbers is %f secs",
            (double)(end - start)) / CLOCKS_PER_SEC);
        n += 1000.0;
    }
    break;
case 3:
    exit(0);
    break;
}
getchar();
}
return 0;
}

void quicksort (int a[], int low, int high)
{
    if (low < high){
        int split_point = partition (a, low, high);
        quicksort (a, low, split_point - 1);
        quicksort (a, split_point + 1, high);
    }
}
```

```
int partition (int a[], int low, int high)
{
    int pivot = a[low];
    int i = low;
    int j = high+1;
    while (i<=j){
      while (true) {
        i+=1;
        if (a[i]>=pivot)
          break;
      }
      while(true) {
        j-=1;
        if (a[j]<=pivot)
          break;
      }
      int temp = a[i];
      a[i]=a[j];
      a[j]=temp;
    }
    int temp=a[i];
    a[i]=a[j];
    a[j]=temp;

    int temp1 = a[j];
    a[j]=a[low];
    a[low]=temp1;
    return j;
}
```

Output:

1. For manual entry of N value and array elements
2. To display time taken for sorting number of elements N in range of 500 to 14500
3. To exit

Enter your choice : 2
Enter the number of elements : 5
Enter array elements : 2 4 3 1 5
Sorted array is : 1 2 3 4 5

Enter your choice : 2
Time taken to sort 500 numbers is 0.031000 sec

| | |
|---|---|
| 1500 | 0.047000 |
| 2500 | 0.031000 |
| 3500 | 0.031000 |
| 4500 | 0.047000 |
| 5500 | 0.078000 |
| 6500 | 0.031000 |
| 7500 | 0.078000 |
| 8500 | 0.047000 |
| 9500 | 0.047000 |
| 10500 | 0.093000 |
| 11500 | 0.094000 |
| 12500 | 0.141000 |
| 13500 | 0.094000 |
| 14500 | 0.078000 |

NLP
13/6/24