06/06/24.

// Quick sort

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

void quicksort(int arr[], int low, int high);
int partition(int arr[], int low, int high);

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

    while(1){
        printf("\n1: For manual entry of n array elements\n");
        printf("2: To display time taken for sorting number of elements N in the range 500 to 14500\n");
        printf("3: To exit\n");
        printf("Enter your choice: ");
        scanf("%d", &ch);

        switch(ch){
            case 1:
                printf("Enter the no. of elements: ");
                scanf("%d", &n);
                printf("enter array elements: ");
                for(i=0; i<n; i++)
                    scanf("%d", &a[i]);
                start = clock();
                quicksort(a, 0, n-1);
                end = clock();
```

```
printf(" Sorted array is: \n");
for(i=0; i<n; i++){
    printf("%d \t", a[i]);
}
printf("\n Time taken to sort %d numbers is %f secs \n", n, ((double)(end-start)) / CLOCKS_PER_SEC);
break;
case 2:
    n=800;
    while (n<=16000){
        for(i=0; i<n; i++){
            a[i]=n-i;
        }
        start=clock();
        quickSort(a, 0, n-1);

        for(j=0; j<800000; 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=n+1000;
    }
    break;
case 3:
    exit(0);
}
getchar();
}
return 0;
}
```

```
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 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++;
            int temp = arr[i];
            arr[i] = arr[j];
            arr[j] = temp;
        }
    }
    int temp = arr[i+1];
    arr[i+1] = arr[high];
    arr[high] = temp;
    return (i+1);
}
```

output:

1: For manual entry of n array elements
2: To display time taken for sorting number of elements in range 500 to 4500
3: To exit

Enter your choice: 2

Time taken to sort 500 numbers is 0.000000

Time taken to sort 1500 numbers is 0.000000

Time taken to sort 2500 numbers is 0.015000 sec
Time taken to sort 3500 numbers is 0.000000 sec
Time taken to sort 4500 numbers is 0.016000 sec
Time taken to sort 5500 numbers is 0.015000 sec
Time taken to sort 6500 numbers is 0.016000 sec
Time taken to sort 7500 numbers is 0.031000 sec
Time taken to sort 8500 numbers is 0.031000 sec
Time taken to sort 9500 numbers is 0.047000 sec
Time taken to sort 10500 numbers is 0.063000 sec
Time taken to sort 11500 numbers is 0.063000 sec
Time taken to sort 12500 numbers is 0.094000 sec
Time taken to sort 13500 numbers is 0.094000 sec
Time taken to sort 14500 numbers is 0.093000 sec
Time taken to sort