SORTING ALGORITHMS

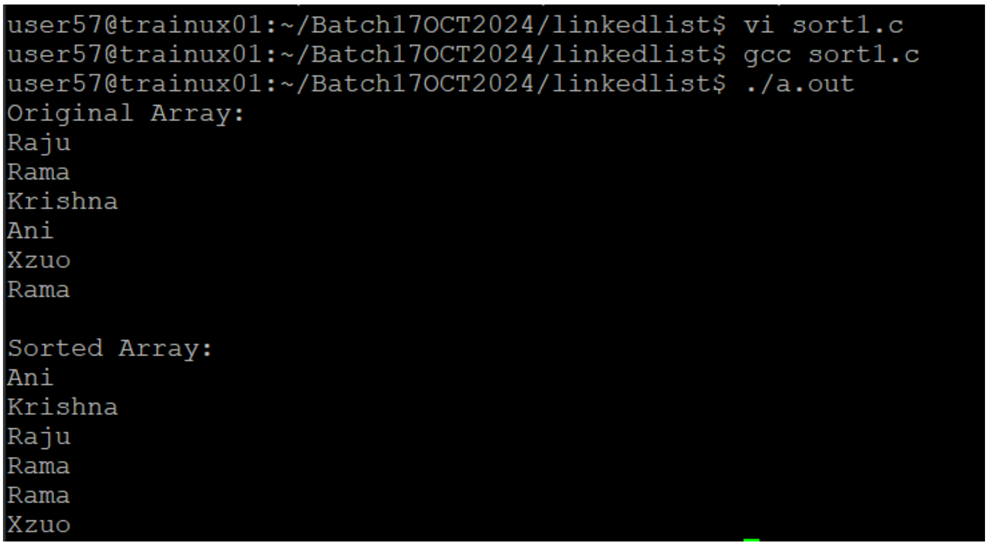
1. Refer the code in “Sorting Techiniques” in socodery folder. Modify the existing implementation of merge sort to use user defined string data.
Use the input dataset below to test the code
Raju, Rama, Krishna, Ani, Xzuo, Rama

Sol:

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

#define MAX_STRINGS 100
#define MAX_LENGTH 100

void merge_sort(char *array[], int BEG, int END);
static void merge(char *array[], int LB1, int UB1, int LB2, int UB2);
void print_array(char *array[], int n);

int main() {
    char *data[] = {"Raju", "Rama", "Krishna", "Ani", "Xzuo", "Rama"};
    int n = 6;

    printf("Original Array:\n");
    print_array(data, n);

    merge_sort(data, 0, n - 1);

    printf("\nSorted Array:\n");
    print_array(data, n);

    return 0;
```

```
}

void merge_sort(char *array[], int BEG, int END) {
    int MID;

    if (BEG < END) {
        MID = (BEG + END) / 2;
        merge_sort(array, BEG, MID);
        merge_sort(array, MID + 1, END);
        merge(array, BEG, MID, MID + 1, END);
    }
}

static void merge(char *array[], int LB1, int UB1, int LB2, int UB2) {
    char *c[MAX_STRINGS];
    int i = LB1, j = LB2, k = 0;

    while (i <= UB1 && j <= UB2) {
        if (strcmp(array[i], array[j]) < 0) {
            c[k] = array[i];
            i++;
        } else {
            c[k] = array[j];
            j++;
        }
        k++;
    }

    while (i <= UB1) {
        c[k] = array[i];
        k++;
        i++;
    }

    while (j <= UB2) {
        c[k] = array[j];
        k++;
        j++;
    }

    i = LB1;
    k = 0;
    while (i <= UB2) {
        array[i] = c[k];
        i++;
        k++;
    }
}

void print_array(char *array[], int n) {
    for (int i = 0; i < n; i++) {
        printf("%s\n", array[i]);
    }
}
```

Output:



```
user57@trainux01:~/Batch17OCT2024/linkedlist$ vi sort1.c
user57@trainux01:~/Batch17OCT2024/linkedlist$ gcc sort1.c
user57@trainux01:~/Batch17OCT2024/linkedlist$ ./a.out
Original Array:
Raju
Rama
Krishna
Ani
Xzuo
Rama

Sorted Array:
Ani
Krishna
Raju
Rama
Rama
Xzuo
```

2. Refer the C++ code for heap sort in link below. Modify the implementation to use user defined string data.

https://www.geeksforgeeks.org/heap-sort/

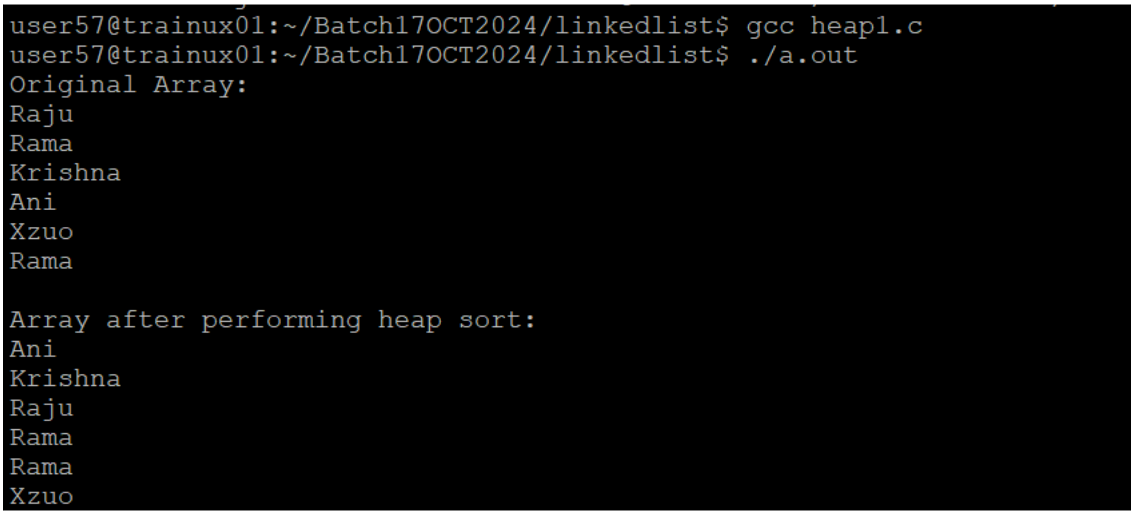
Sol:

```
#include <stdio.h>
#include <string.h>
void heapify(char *arr[], int n, int i)
{
   int maximum, left_index, right_index;
   char *temp;
   maximum = i;
   left_index = 2 * i + 1;
   right_index = 2 * i + 2;
   if (left_index < n && strcmp(arr[left_index], arr[maximum]) > 0)
      maximum = left_index;
   if (right_index < n && strcmp(arr[right_index], arr[maximum]) > 0)
      maximum = right_index;
   if (maximum != i) {
      temp = arr[i];
      arr[i] = arr[maximum];
```

```
        arr[maximum] = temp;
        heapify(arr, n, maximum);
    }
}
void heapsort(char *arr[], int n)
{
    int i;
    char *temp;
    for (i = n / 2 - 1; i >= 0; i--) {
        heapify(arr, n, i);
    }
    for (i = n - 1; i > 0; i--) {
        temp = arr[0];
        arr[0] = arr[i];
        arr[i] = temp;
        heapify(arr, i, 0);
    }
}
int main()
{
    char *arr[] = { "Raju", "Rama", "Krishna", "Ani", "Xzuo", "Rama" };
    int n = 6;
    printf("Original Array: \n");
    for (int i = 0; i < n; i++) {
        printf("%s ", arr[i]);
    }
    printf("\n");
    heapsort(arr, n);
    printf("Array after performing heap sort: \n");
    for (int i = 0; i < n; i++) {
        printf("%s ", arr[i]);
```

```
  }

  return 0;

}
```

Output:



```
user57@trainux01:~/Batch17OCT2024/linkedlist$ gcc heap1.c
user57@trainux01:~/Batch17OCT2024/linkedlist$ ./a.out
Original Array:
Raju
Rama
Krishna
Ani
Xzuo
Rama

Array after performing heap sort:
Ani
Krishna
Raju
Rama
Rama
Xzuo
```