Submissions

Leaderboard

Discussions

Editorial

- an array of strings : *arr*
- length of string array: *count*
- pointer to the string comparison function: *cmp_func*

You also need to implement the following four string comparison functions:

1. **`int lexicographic_sort(char*, char*)`** to sort the strings in lexicographically non-decreasing order.
2. **`int lexicographic_sort_reverse(char*, c`** to sort the strings in lexicographically non-increasing order.
3. **`int sort_by_number_of_distinct_characte`** to sort the strings in non-decreasing order of the number of distinct characters present in them. If two strings have the same number of distinct characters present in them, then the lexicographically smaller string should appear first.
4. **`int sort_by_length(char*, char*)`** to sort the strings in non-decreasing order of their lengths. If two strings have the same length, then the lexicographically smaller string should appear first.

Change Theme Language: C

```

    char** arr;
    arr = (char**)malloc(n * sizeof(char*));

    for(int i = 0; i < n; i++){
        *(arr + i) = malloc(1024 * sizeof(char));
        scanf("%s", *(arr + i));
        *(arr + i) = realloc(*(arr + i), strlen(*(
    }

    string_sort(arr, n, lexicographic_sort);
    for(int i = 0; i < n; i++)
        printf("%s\n", arr[i]);
    printf("\n");

    string_sort(arr, n, lexicographic_sort_reverse
    for(int i = 0; i < n; i++)
        printf("%s\n", arr[i]);
    printf("\n");

    string_sort(arr, n, sort_by_length);
    for(int i = 0; i < n; i++)
        printf("%s\n", arr[i]);
    printf("\n");

    string_sort(arr, n, sort_by_number_of_distinct
    for(int i = 0; i < n; i++)
        printf("%s\n", arr[i]);
    printf("\n");
}
```