CS19001: Programming and Data Structures Laboratory

Soumyajit Dey, Aritra Hazra;
CSE, IIT Kharagpur

http://cse.iitkgp.ac.in/~aritrah/course/lab/PDS/Autumn2018/CS19101_PDS-Lab_Autumn2018.html

08-Oct-2018

Soumyajit Dey, Aritra Hazra; CSE, IIT Kharagpur

Programming Assignments

Complete and submit during lab

Assignment 1 [Text-Stats]

Write a C-program to perform the following:

- Ask the user to input some text/string (may contain anything that can be entered via keyboard including spaces, tabs, new-lines etc.). The end of entry will be determined by pressing $\langle Ctrl + D \rangle$ keys (together).
- Computes and Displays the following statistics:
  1. the number of lines present in the text;
  2. the number of words presnt in the text;
  3. the number of total characters present in the text;
  4. the number of lower-case alphabets, upper-case alphabets and numeric digits present in the text (report these three statistics additionally);
  5. the number of spaces entered in the text; and
  6. the number of tabs entered in the text.

Assignment 2 [Code-Word]

Assuming that the fixed codes for the alphabets [a - z] are given as, 'a'=1, 'b'=2, ..., 'y'=25 and 'z'=26, write a recursive function to generate all possible alphabatic words from a given code string.

Example:

Let the given code string be, ''1123''.
Then, all the possible alphabetic words are:

```
aabc // a =  1, a =  1, b =  2, c =  3
aaw  // a =  1, a =  1, w = 23
alc  // a =  1, l = 12, c =  3
kbc  // k = 11, b =  2, c =  3
kw   // k = 11, w = 23
```

Write a (C-program) main function that takes a code string from the user and displays all the possible alphabetic words.

Assignment 3 [Rotation-Equivalence]

Definitions

k-**rotation:** For any string *str* of length *n*, the *k*-rotation of *str* from index *i* ($0 < i + k \leq n$) creates a new string, where *only* the *k*-character substring of *str* (from index *i*), i.e. $str[i..(i + k - 1)]$, is reversed/rotated and all other characters remain intact.

k-**rotation equivalence:** A string *str*1 is said to be *k*-rotation equivalent with another string *str*2, if the *k*-rotation from any index *i* ($0 < i + k \leq n$) of *str*1 can produce identical *str*2 (both *str*1 and *str*2 are of equal length *n*).

Example

Let, *str*1 = ``abacus'', *str*2 = ``abucas'' and *str*3 = ``baacsu''. Suppose, $k = 3$.
Then, *str*1 is 3-rotation equivalent with *str*2, because *str*1 can be 3-rotated from index 2 to get *str*2.
However, *str*1 is NOT 3-rotation equivalent with *str*3.

Assignment 3 [Rotation-Equivalence]

Recursive k-rotation function: `rotateStr(char str[], int idx, int k);`

Write a **recursive** function `rotateStr` which produces rotations in the k-length substring of *str* starting from index, *idx*.

Write a C-Program (main) that,

- prompts the user to enter two strings, `String1` and `String2`[a];
- asks the user to enter a rotation length[b], say k;
- uses the `rotateStr` function and finds out whether `String2` is a *k-rotation equivalent* of `String1`,
- if yes, reports the index of `String1` from which k-rotation creates `String2`. Otherwise, reports the *non-equivalence* as a result.

---

[a] Remember, the first criteria of equivalence between any two strings is that they must be of equal size – *so make appropriate checks for that*!

[b] If the rotation length is more than the string length, automatically *it will be set to string length*, which is the maximum applicable part!

Thank You