BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
HYDERABAD
CAMPUS,
Data Structures and Algorithms
CS F211
Homework Assignment – 6

1. Students attending course of Machine Learning are categorized in three groups Gp1, Gp2 and Gp3 by some classifier algorithms. Student have attributes like name, branch, cgpa, year, group. You are given list of students and you have to arrange the students in a fashion such that student who belong to Gp1 comes first then comes Gp2 and then Gp3(In case any 2 students s1, s2 belong to same group then the student who is coming first in original list must come before in your output list i.e. stable property). You have to solve this problem in O(n) and O(1) (= constant) space complexity. Define structure student with relevant attributes. And output the list with name along with group to which he belongs.

   Input:

   N

   Following N details of students

2. Game x is played in a country y. There are N states in country y and you are given the number of players from each of them. There is a rule that for any team to participate in that game, there should be exactly M players in a team and no two player should be from same state. How many maximum teams can you form?

   Constraints: $1 <= N <= 50$, $2 <= M <= 100$, $1 <= nop[i] <= 10^9$

   Input:

   N M

   Second line contains the number of players from ith state

   Sample Input-1:

   4 3

   2 4 3 6

   Output:

   4

   Explanation:

   N = 4 and M = 3

   Let 4 states be s1, s2, s3, s4. Given that we have 2 from s1, 4 from s2, 3 from s3 and 6 from s4. Each team should contain 3 players from different states. We can form the

following team: {s1, s2, s3}, {s1, s2, s4}, {s2, s3, s4}, {s2, s3, s4}. Note that even though there are 15 players in total we can't form more than 4 teams.

Sample Input-2:
6 4
10 11 12 13 14 15

Output:
18

3. Krish has come up with a formula of sorting the list of names(max character is 50) because his understanding of world is different. Here is his formula of sorting the names: Take the name and find the sum of ascii value for each character and then convert the final sum it to binary representation and count how many 1s are there in the final sum and then arrange the name in ascending order of the total number of 1s in the final sum. In case 2 names have same total number of 1s then name coming first in the list should come first in the output list i.e. stable property. You all being computer geeks, help Krish to solve this problem in O(n) time. So read list of string and sort it in Krish-way.

4. Given an array of 0s and 1s, find the position of 0 to be replaced with 1 to get longest continuous sequence of 1s. Expected time complexity is O(n) and auxiliary space is O(1).

Example:

Input: arr[] = {1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1}
Output: Index 9

Assuming array index starts from 0, replacing 0 with 1 at index 9 causes the maximum continuous sequence of 1s.

Input: arr[] = {1, 1, 1, 1, 0}
Output: Index 4

5. You are given a dictionary file containing words, each in separate lines. Assume each word is less than 50 letters and each letter is among a, b, c,…, z. So, "apple" upon sorting becomes "aelpp". Many words in the dictionary will result in the same string after sorting. For example, sorted string of both "cat" and "act" will be "act". Similarly, sorted string of both "listen" and "silent" is "eilnst". These actual words are anagrams whose sorted string is same. You have to write a program to read each word from a file wordfile.txt (create it and write some words), sort the word in dictionary order, and print the sorted word and actual word. Use Unix pipes for redirecting outputs. The above print would actually be

piped to sort utility. The output of sort in turn would be piped to the second program you need to write (look at the command at the end).

So, you need to write two programs:

a. "*sortwords.c*" that reads file and sorts each word, and then prints sorted string and the actual word in a line.

printf("%s %s\n", sortedword, actualword);

Call this file sortwords.c. Compile and create executable from this file called sortwords. The above program's output can be fed to *sort* program of Unix. (see the command at the end).

b. The second program *anagrams.c* (its executable called print_anagrams) that will get the input from the *sort* program (*sort* program sorts all the lines of input line by line, so each anagram will become contiguous after *sort*) The output from the *sort* program is taken by program (called) anagrams, and it should print all the anagrams in a separate line.

Save the final output file as *anagrams.txt*.

The **command** that you will run finally will look like:
$ ./sortwords < wordfile.txt | sort | ./print_anagrams > anagrams.txt

An example follows:
wordfile.txt contains
boy
stray
act
tops
apple
cat
stop

The following is the output of sortwords:
act act
apple aelpp
boy boy
act cat
opst stop
arsty stray
opst tops

On sorting (using sort utility in Unix):
act act
act cat

aelpp apple
arsty stray
boy boy
opst stop
opst tops

The above would then be used by ./print_anagrams to print all anagrams in a single line.
The output of ./anagrams would be sent to file anagrams.txt which should now contain:
act cat
apple
stray
boy
stop tops

Note : See that each line contains actual words that are anagrams.

6.
7. Write a menu driven program to create a Singly Linked List by giving option a, b, c for following operation by appropriately reading a number X to be inserted into the list:
   a. Insert at beginning of the List (write a function InsertAtBeg() and call from main)
   b. Insert at end of the List(write a function InsertAtEnd() and call from main)
   c. Read another number Y before which X is to be added.(write a function Insert() and call from main)
   (**Note :** You can't define global variables except your structure.)

8. Write a menu driven program to create a Singly Linked List by inserting in the beginning only i.e. option a and giving option b, c, d for following operation:
   a. Insert at beginning of the List
   b. Delete at beginning of the List.
   c. Deleting from the end of the List.
   d. Read a number X which is to be added.
   (**Note** : In case list is empty and I delete then in that case proper error message(Underflow) should be displayed and You can't define global variables except your structure.)

9. Write a **mergeSortedList**() function that takes two lists, each of which is sorted in increasing order, and merges the two together into one list which is in increasing order. **mergeSortedList** () should return the new list. You can't define global variables except your structure.
   For Example : if the first linked list a is 5->10->15->NULL and the other linked list b is 2->3->20->NULL, then **mergeSortedList** () should return a pointer to the head node of the merged list 2->3->5->10->15->20->NULL
   (**Note :** You need to print the modified list and for creating the two list make function InsertAtBeg to insert at beginning which will very easily create a list)

10. Create a Linked List and reverse the Linked-List(original linked-list should be reversed not a new one should be created) and print the reversed-list. So for creating Linked-List simply

insert at beginning of linked-list and then reverse the Linked-List created and after reversing print the Linked-List. Use proper functions like InsertAtBeg(), reverse() and printList() to solve this problem. You can't define global variables except your structure.

Ex : your List is   4 -> 5 ->1 ->3 -> NULL

then your reversed list is 3->1->5->4->NULL

11. Create a Linked List and reverse the Linked-List(original linked-list should be reversed not a new one should be created) through **Recursion** and print the reversed-list. So for creating Linked-List simply insert at beginning of linked-list and then reverse the Linked-List created and after reversing print the Linked-List. Use proper functions like InsertAtBeg(), reverse() and printList() to solve this problem. You can't define global variables except your structure.

Ex : your List is   4 -> 5 ->1 ->3 -> NULL

then your reversed list is 3->1->5->4->NULL

---------------------------------XXXXXXXXXXXXXXXXXXXXXXXXX---------------------------------