Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación
Segundo Semestre de 2014

IIC 1103 - Introducción a la Programación

Laboratorio 12

Objetivo General

En este laboratorio reforzaremos los contenidos vistos en clases relacionados con recursión. Para llevar a cabo los ejercicios debes ser capaz de pensar recursivamente e implementar algoritmos recursivos.

Enunciado

1. En esta pregunta deberás usar recursión para computar todas las permutaciones de un string, es decir, encontrar todos los strings posibles utilizando los caracteres del string original. Para esto, recibirás como input una línea que contiene un string sin espacios. Luego como output debes imprimir todas las permutaciones ordenadas lexicográficamente, en que un caracter será menor que otro si aparece primero en el string original. Por ejemplo si el string original es `zad`, la permutación `zda` debe imprimirse primero que la `azd`, puesto que `z` es menor que `a`. También la permutación `azd` se debe imprimir primero que la `adz`, ya que su primer caracter es igual, pero en el segundo caracter tenemos que el caracter`z` es menor que el caracter `d`.
   Puedes suponer que no habrán caracteres repetidos. A continuación se muestran dos ejemplos de diálogo del programa:

caso	input	output
1	abc	abc acb bac bca cab cba
2	zad	zad zda azd adz dza daz

2. Estás cargo de la organización de una conferencia internacional, y debes localizar a los invitados en las habitaciones de un hotel, el problema es que existen algunos invitados que se odian entre sí, por lo que tienes que hacer un programa que permita saber si conociendo cuales invitados se odian entre sí, es posible organizar a los invitados en las habitaciones disponibles.

   La primera línea del input tiene 3 enteros H,P,R separados por un espacio que indican la cantidad de habitaciones, de personas que participan en la conferencia y de relaciones de odio respectivamente, las

siguientes H líneas tienen cada una un entero L que indica la capacidad de la h-ésima habitación,las siguientes P líneas tienen cada una el nombre de un participante en la conferencia y las siguientes R líneas tienen cada una dos nombres escritos con letras minúsculas y separados por un espacio que indican que esas dos personas se odian.

Debes responder "True" en caso de que se pueda organizar a los invitados en las habitaciones sin que queden 2 personas que se odien juntas, o "False" en el caso contrario.

Por ejemplo en el primer caso se puede poner a pedro con juan en una habitación y a mario con josé en otra habitación, pero en el segundo ya no es posible.

caso	input	output
1	2 4 3 2 3 pedro jose mario juan mario pedro pedro jose jose juan	True
2	2 5 5 2 3 pedro jose mario juan ignacio mario pedro pedro jose jose juan ignacio pedro ignacio jose	False

3. Para ayudarte a lidiar con el problema anterior, se te ha dado la opción de llevar a algunos de los invitados a una terápia de grupo para eliminar el odio entre ellos, sin embargo, un requisito para que el tratamiento funcione es que todos los que vayan se odien entre sí, por que tienes que escribir un programa que dados la cantidad de cupos, los invitados y las relaciones de odio, te permita saber si existe un grupo lo suficientemente grande de invitados donde todos se odien entre sí.

   El input tiene en la primera línea 3 enteros K,P, R separados por un espacio, que indican los cupos para el tratamiento, la cantidad de personas y la cantidad de relaciones de odio respectivamente, las siguientes P líneas tienen cada una el nombre de un participante en la conferencia y las siguientes R líneas tienen cada una dos nombres escritos con letras minúsculas y separados por un espacio que indican que esas dos personas se odian.

   Debes imprimir "True" en caso de que exista un grupo de tamaño K (o mayor) que cumpla la condición pedida y "False" en caso contrario.

caso	input	output
1	3 5 5 pedro jose mario juan ignacio mario pedro pedro jose jose juan ignacio pedro ignacio jose	True