Integrantes:

Rondeau, Matías.
Delle Vedove, Mauricio.

Ejercicio 1

Técnica de desarrollo implementada:

La técnica utilizada para resolver el ejercicio fue *fuerza bruta*, buscando todas las permutaciones de los datos de entrada. Cada palabra del conjunto se la permuta con las demás, la forma de hacerlo se basa en permutar cada caracter de una palabra con el resto en la posición de dicho caracter. Luego de obtener todas las permutaciones, calculamos la distancia de hamming entre cada permutación y cada palabra, formando una lista con las distancias de cada comparación, después se compara el *k* con todas las distancias.

Uso:

- Ejecutar el archivo *hamming.hs* de la siguiente forma “hugs hamming.hs”
- Invocar método Main: “hammingExistMain *k [String]”*

Donde k = Integer y [String] = Lista de strings

Ejemplo: hammingExistMain 2 [“hola”, “rima”] retorna “true”

Ejercicio 2

Técnica de desarrollo implementada:

La técnica utilizada para la resolución fue *Decrease and Conquer (decremento por una constante).* El algoritmo consiste en calcular las partidas a partir del jugador que tiene menos horas disponibles con el resto de los jugadores, del el que tiene menos horas disponibles al que tiene mas, luego ir decrementando los horarios disponibles de cada jugador que jugó.

Se considera que el torneo tiene 6 jugadores(0..5), los cuales tienen que dar, al menos, 7 horarios(0..23) en los que podrán realizar las partidas.

Uso:

- Compilar la clase Torneo.java “javac Torneo.java”
-Ejecutarla “java Torneo”
-Colocar los horarios de los jugadores (se considera que los horarios son representados como enteros (int) y que son bien ingresados ya la finalidad del algoritmo no es controlar los datos de entrada sino centrarse en la resolución bajo la técnica de *Decrease and Conquer*)

* En el codigo posee ejemplos de prueba, descomentar para usarlos.