Descripción del Juego de Pruebas de la clase 'branchBound.java'

• **Objeto de la prueba:** Este juego de pruebas engloba la clase branchBound.
Dicha clase representa el algoritmo Branch & Bound. Su uso se basa en tener una cola de prioridad, un mejor node y mejor coste que aplicando un conjunto de metodos hacemos que el mejor node sea el node con la solución mas optimizada.

• **Otros elementos integrados en la prueba:** Se ha crado el archivo de compilación y ejecución 'Makefile'. Con sólo teclear 'make' se compilará y ejecutará la clase con el juego de pruebas.

• **Drivers:** Se ha creado el driver “driverBranchBound.java” para probar el correcto funcionamiento de la clase. En él se ha creado la función 'main' para que se prueben todas sus funcionalidades. Cuando se ejecuta el driver, salen enumeradas todos los métodos que te permite hacer la clase.

• **Stubs:** No se han creado stubs para esta clase.

• **Fichero de datos necesario:** En el archivo “driverBranchBound.in” están los datos necesarios para probar la clase desde un archivo. En éste se prueban que todas las funcionalidades de la clase funcionen correctamente. En el código se han puesto excepciones de posibles errores, que informaran de ellos si es debido el caso.

• **Valores estudiados:** Básicamente se han puesto valores aleatorios para hacer la prueba de esta clase.

- La primera funcionalidad es la de crear un controlador de persistencia , por lo que sólo se ejecuta ese método sin ningun otro parámetro de entrada.
- La segunda funcionalidades la de devolver el tamaño de las matrices, por lo que sólo se ejecuta ese método sin ningun otro parámetro de entrada
- La tercera funonalidad es la de devolver la matriz de afinidad, por lo que sólo se ejecuta ese método sin ningun otro parámetro de entrada
- La cuarta funcionalidad es la de devolver la matriz de distancias, por lo que sólo se ejecuta ese método sin ningun otro parámetro de entrada
- La quinta funcionalid es la de leer el juego de pruebas a partir de un 'String' con nombre del archivo donde esta el juego de pruebas. En nuestro juego de pruebas es '4', que es un juego de prueba con una n = 4.
- La sexta funcionalidad es la de devolver la solución del algoritmo, a partir de un juego de pruebas que previamente habremos leido. Saldra el vector de la mejor solución y su coste.
- La septima funcionalidad es la de guardar la solución del algoritmo, que previamente habremos calculado, para este metodo se necesita el nombre del fichero donde se guardara la solución. En nuestro juego de pruebas es '4' y dara como resultado el fichero 4SOL.dat

• **Operativa:** El funcionamiento del ejecutable se basa en la repetida introducción de números y seguidamente de los datos con los que se quieran operar. Cuando se ejecuta, aparecen por pantalla todos los métodos de la clase:

Elige una opicion:

1) ctrlPersistencia()");
2) getTam()");
3) getAfin()");
4) getDist()");
5) leerJuegoDePrueba(String ruta)");
6) Ejecutar BranchBound");
7) guardarResultado(int[] res, String ruta)");