EDA: ficha con la descripción de las pruebas para el corrector de la práctica 3 (2019)

p01	Se crean distintas localidades y su correspondiente Localidad2, y se comparan entre ellas. Se muestra por pantalla el resultado de cada acción.	0.5
p02	Se crea una Coleccion, se invoca su método lectura. Se crea un GrafLoc y se invoca su método insertaLocalidades con la colección creada. Se muestra el grafo por pantalla.	0.5
p03	Se crea una Coleccion, se invoca su método lectura. Se crea un TNear y se invoca su método insertaLocalidades con la colección creada. Se muestra el árbol por pantalla.	0.5
p04	Se crea un GrafLoc y se invoca su método insertaArista con distintos valores. Se muestra por pantalla el resultado de cada acción y el grafo creado.	0.5
p05	Se invoca la aplicación Prim con un fichero con localidades y partiendo de una vértice existente.	0.5
p06	Se crea un GrafLoc, se invoca su método insertaArista con diversos valores y se muestra el grafo resultante por pantalla. Se crea una Coleccion, se invoca su método lectura, y el método insertaLocalidades del grafo. Se muestra por el árbol devuelto y el grafo resultante.	0.5
p07	Se crea un GrafLoc, se invoca su método insertaArista con diversos valores y se muestra el grafo resultante por pantalla. Se invoca su método recuperaArista con diferentes valores, mostrando por pantalla el resultado de cada acción.	0.5
p08	Se crea una Coleccion, se invoca su método lectura. Se crea un TNear y se invoca su método insertaLocalidades con la colección creada. Se muestra el árbol por pantalla. Se invoca su método getLocalidades para todos los nombres de localidades, mostrando el resultado de cada acción.	0.5
p09	Se crea un GrafLoc, se invoca su método insertaArista con diversos valores y se muestra el grafo resultante por pantalla. Se invoca su método borraArista con diferentes valores. Se invoca escribeDFS para todos los vértices.	0.5
p10	Se crea una Coleccion, se invoca su método lectura. Se crea un TNear y se invoca su método insertaLocalidades con la colección creada. Se muestra el árbol por pantalla. Se invoca su método borraLocalidad con distintas cadenas (aparecen como máximo una vez en el árbol), mostrando por pantalla el resultado de cada acción.	0.5

p11	Se crea una Coleccion, se invoca su método lectura. Se crea un GrafLoc y se invoca su método insertaLocalidades con la colección creada. Se invoca su método escribeDFS desde cada vértice. Se invoca su método borraVertice con diferentes valores, invocando cada vez a escribeDFS para todos los vértices.	0.5
p12	Se crea una Coleccion, se invoca su método lectura. Se crea un TNear y se invoca su método insertaLocalidades con la colección creada. Se muestra el árbol por pantalla. Se invoca su método borraLocalidad con distintas cadenas (pueden aparecer varias veces en el árbol), hasta vaciar el árbol, mostrando por pantalla el resultado de cada acción.	0.5
p13	Se crea una Coleccion, se invoca su método lectura. Se crea un TNear y se invoca su método insertaLocalidades con la colección creada. Se muestra el árbol por pantalla. Se invoca su método getTop con distintos valores de manera que siempre encuentra localidades que cumplan la condición.	0.5
p14	Se crea una Coleccion, se invoca su método lectura. Se crea un TNear y se invoca su método insertaLocalidades con la colección creada. Se muestra por pantalla el árbol y se invocan sus métodos borraLocalidad, setDn con un valor inferior al que tenía, insertaLocalidad2 y setDn con un valor superior al que tenía. Se muestra por pantalla el resultado de cada acción.	0.5
p15	Se crea una Coleccion, se invoca su método lectura. Se crea un TNear y se invoca su método insertaLocalidades con la colección creada. Se muestra el árbol por pantalla. Se invoca su método getTop con distintos valores de manera que unas veces encuentra localidades que cumplan la condición, y otras no.	0.5
p16	Se crea una Coleccion, se invoca su método lectura. Se crea un TNear y se invoca su método insertaLocalidades con la colección creada. Se muestra el árbol por pantalla. Se invoca su método insertaLocalidad2 con las localidades de otra colección, en la que algunas localidades se repiten. Se vuelve a mostrar el árbol. Se invoca su método getLocalidades con diferentes cadenas, que unas veces están en el árbol y otras no, mostrando el resultado de cada acción.	0.5
p17	Se invoca la aplicación Prim con un fichero con más de 40 localidades y partiendo de un vértice existente.	0.5
p18	Se invoca la aplicación Prim con un fichero con localidades y partiendo de un vértice existente.	0.5
p19	Se invoca la aplicación Prim con el mismo fichero con localidades de la prueba anterior y partiendo de un vértice diferente.	0.5
p20	Se invoca la aplicación Prim con un fichero con localidades y partiendo de cada uno de los vértices.	0.5