Projecte Graph

Jordi
Fran
Abde
1r Daw
Curs 22/23

Índex

1. Donat graf no dirigit, dir si aquest és complet.
    - Entrada: vèrtexs i arestes del graf.
    - Sortida: cert / fals.

- Definició detallada del problema.

  Verificar que donat uns vertexs i arestes, determini (True/False) si un graf és dirigit o no

- Definició detallada de la solució.

  El codi és un programa Java que defineix una classe anomenada Graph, que representa un graf no dirigit utilitzant una llista d'adjacència. Una llista adjacent és una estructura de dades que emmagatzema una llista de vèrtexs i els seus veïns per a cada vèrtex del graf. El codi també defineix una classe anomenada Edge, que representa una aresta entre dos vèrtexs.

  El nostre codi defineix un mètode principal que crea una instància de la classe Graph utilitzant una llista d'arestes i un nombre de vèrtexs. Les vores estan codificades en el codi, però també es poden llegir des d'un fitxer d'entrada. El nombre de vèrtexs es deriva de la mida de la llista d'arestes.
  El codi crida un mètode anomenat isTree a la instància del gràfic, el qual retorna cert si el graf és un arbre i fals en cas contrari. Un arbre és un tipus especial de graf que no té cicles i està connectat. Un cicle és un camí que comença i acaba en el mateix vèrtex, i un graf està connectat si hi ha un camí entre dos vèrtexs qualssevol.

  El codi utilitza DFS (cerca en profunditat) per a comprovar si el gràfic és un arbre o no. DFS és un algorisme transversal que explora el graf anant el més profund possible al llarg de cada branca abans de fer marxa enrere. El nostre codi utilitza un mètode recursiu anomenat DFSUtil per a realitzar DFS a cada vèrtex, començant pel vèrtex 0.
  El nostre codi també utilitza una matriu anomenada “dicovered” per a fer un seguiment dels vèrtexs que s'han visitat durant el recorregut. El codi també utilitza un paràmetre anomenat pare per a fer un seguiment del pare de cada vèrtex a l'arbre DFS.

  El codi comprova els cicles buscant una vora que condueixi a un vèrtex ja visitat que no sigui el pare del vèrtex actual. Si existeix tal aresta, llavors hi ha un cicle en el graf i no és un arbre.

  El codi comprova la connectivitat buscant qualsevol vèrtex no visitat després de fer servir DFS. Si existeix aquest vèrtex, llavors el graf no està connectat i no és un arbre. El codi imprimeix cert o fals depenent de si el gràfic és o no un arbre.

- Definició dels casos de prova.

  Un cas de prova és un conjunt de condicions o variables sota les quals un test determinarà si un sistema sota prova compleix els requisits o funciona correctament. Un cas de prova normalment consisteix en els següents elements:

  - Un identificador únic
  - Una descripció de l'escenari de prova
  - Una llista de condicions prèvies o suposicions
  - Una llista de dades o passos d'entrada
  - Una llista de resultats o resultats esperats
  - Una llista de postcondicions o accions

  Per definir un cas de prova, cal identificar l'escenari de prova, que és una descripció d'alt nivell del qual volem provar i per què. Per exemple, un escenari de prova per al nostre mètode isTree podria ser "comproveu si un graf sense arestes és un arbre".

  A continuació, hem de proporcionar les dades o passos d'entrada, que són les coses que donarem al sistema sota prova o que hi fan. Per exemple, una dada d'entrada per al vostre mètode isTree serà "el nombre de vèrtexs del graf".

  Després d'això, cal indicar els resultats o resultats esperats, que són les coses que s'espera que el sistema sotmès a prova produeixi o faci. Per exemple, un resultat esperat per al vostre mètode isTree podria ser "el mètode retorna True".

  Finalment, cal especificar les accions, que són les coses que han de ser certes o fetes després de la prova. Per exemple, una postcondició per al nostre mètode isTree podria ser "el gràfic no ha canviat".

  Aquest és un exemple d'un cas de prova del nostre mètode ISTree basat en l'escenari de prova anterior:

  - Comprova si un graf sense arestes és un arbre. El graf s'inicialitza amb una llista buida d'arestes i un vèrtex. El nombre de vèrtexs del graf. El mètode retorna cert. El graf no es modifica.

WEBGRAFIA

https://ca.wikipedia.org/wiki/Graf_complet
https://www.grycap.upv.es/gmolto/docs/eda/EDA_Tema_14_Parte_I_gmolto.pdf
https://www.javatpoint.com/java-graph
https://www.baeldung.com/cs/check-if-two-nodes-are-connected