CLASS Graf:

PROPERTIES:

int jumlahKota O(1)

Map<Integer, Set<Integer>> aliansiKota

Map<String, List<String>> graf

CONSTRUCTOR Graf(int jumlahKota):

SET this.jumlahKota TO jumlahKota O(1)

INIT aliansiKota AS new HashMap O(1)

INIT graf AS new HashMap O(1)

METHOD tambahAliansi(int warna, List<Integer> kota):

FOR each k IN kota: O(n)

IF k NOT IN aliansiKota: O(1)

PUT new HashSet() TO aliansiKota WITH key k O(1)

ADD warna TO aliansiKota[k] O(1)

FOR i FROM 0 TO kota.size() - 1: O(n)

FOR j FROM i + 1 TO kota.size(): O(n)

SET kunci1 TO kota.get(i) + "-" + warna O(1)

SET kunci2 TO kota.get(j) + "-" + warna O(1)

IF kunci1 NOT IN graf: O(1)

PUT new ArrayList() TO graf WITH key kunci1 O(1)

IF kunci2 NOT IN graf: O(1)

PUT new ArrayList() TO graf WITH key kunci2 O(1)

ADD kunci2 TO graf[kunci1] O(1)

ADD kunci1 TO graf[kunci2] O(1)

METHOD boolean dapatMengunjungiSemuaKotaDari(int kotaMulai):

SET tiketAwal TO aliansiKota.get(kotaMulai) O(1)

IF tiketAwal IS null: O(1)

RETURN false O(1)

FOR each tiket IN tiketAwal: O(n)

INIT dikunjungi AS new HashSet O(1)

IF bfs(kotaMulai, tiket, dikunjungi) == jumlahKota: O(n+1)

RETURN true O(1)

RETURN false O(1)

METHOD int bfs(int kotaMulai, int aliansiMulai, Set<Integer> dikunjungi):

INIT antrian AS new LinkedList O(1)

OFFER new Node(kotaMulai, aliansiMulai, new HashSet(Arrays.asList(kotaMulai))) TO antrian O(1)

WHILE antrian IS NOT empty: O(n)

POLL node FROM antrian O(1)

SET kotaSaatIni TO node.kota O(1)

SET aliansiSaatIni TO node.aliansi O(1)

SET dikunjungiSaatIni TO node.dikunjungi O(1)

IF dikunjungiSaatIni.size() == jumlahKota: O(1)

RETURN dikunjungiSaatIni.size() O(1)

INIT aliansiBerikutnyaList AS new ArrayList(aliansiKota.getOrDefault(kotaSaatIni, new HashSet())) O(1)

FOR each aliansiBerikutnya IN aliansiBerikutnyaList: O(n)

IF aliansiBerikutnya != aliansiSaatIni OR aliansiBerikutnyaList.size() == 1: O(1)

SET kunci TO kotaSaatIni + "-" + aliansiBerikutnya O(1)

SET tetangga TO graf.getOrDefault(kunci, new ArrayList()) O(1)

FOR each tetanggaStr IN tetangga: # O(n)

SET kotaTetangga TO Integer.parseInt(tetanggaStr.split("-")[0]) O(1)

SET aliansiTetangga TO Integer.parseInt(tetanggaStr.split("-")[1]) O(1)

IF kotaTetangga NOT IN dikunjungiSaatIni: O(1)

INIT baruDikunjungi AS new HashSet(dikunjungiSaatIni) O(1)

ADD kotaTetangga TO baruDikunjungi O(1)

OFFER new Node(kotaTetangga, aliansiTetangga, baruDikunjungi) TO antrian O(1)

RETURN dikunjungi.size() O(1)

CLASS Node:

PROPERTIES:

int kota O(1)

int aliansi O(1)

Set<Integer> dikunjungi O(1)

CONSTRUCTOR Node(int kota, int aliansi, Set<Integer> dikunjungi):

SET this.kota TO kota O(1)

SET this.aliansi TO aliansi O(1)

SET this.dikunjungi TO dikunjungi O(1)

MAIN:

INIT pemindai AS new Scanner(System.in) O(1)

INIT hasil AS new ArrayList O(1)

WHILE true:

READ jumlahKota FROM pemindai O(1)

READ jumlahAliansi FROM pemindai O(1)

IF jumlahKota == 0 AND jumlahAliansi == 0: O(1)

BREAK O(1)

INIT graf AS new Graf(jumlahKota) O(1)

FOR i FROM 0 TO jumlahAliansi - 1: O(n)

READ jumlahKotaDalamAliansi FROM pemindai O(1)

INIT kota AS new ArrayList O(1)

FOR j FROM 0 TO jumlahKotaDalamAliansi - 1: O(n-1)

READ k FROM pemindai O(1)

ADD k TO kota O(1)

CALL graf.tambahAliansi(i, kota) O(n2)

SET kotaMulai TO -1 O(1)

FOR i FROM 0 TO jumlahKota - 1: O(n)

IF graf.dapatMengunjungiSemuaKotaDari(i): O(n+1)

SET kotaMulai TO i O(1)

BREAK O(1)

ADD kotaMulai TO hasil O(1)

FOR each result IN hasil: O(n)

PRINT result O(1)

CALL pemindai.close() O(1)

**Sehingga notasi big O adalah n2**