מספר רכיבי קשירות

1. רכיב קשירות אחד (גרף אחד)
2. **שלושה רכיבי קשירות** (גרף אחד)

**איך למצוא את** מספר רכיבי הקשירות?

**איך למצוא את כל הקודקודים של כל אחד מ**רכיב הקשירות ?

**ארבעה רכיבי קשירות** (גרף אחד)

1. **בניית מטריצה קשירות ממטריצה שכנות** **algorithm Floyd-Warshall**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | index |
|  |  |  |  | **T** |  |  |  |  | 0 |
|  |  | **T** |  |  |  |  |  |  | 1 |
|  | **T** |  | **T** |  |  |  |  |  | 2 |
|  |  |  |  |  |  |  |  |  | 3 |
| **T** |  |  |  |  |  |  |  | **T** | 4 |
|  | **T** |  |  |  |  | **T** |  |  | 5 |
|  |  |  |  |  |  |  | **T** |  | 6 |
|  |  |  | **T** |  |  | **T** |  |  | 7 |
|  |  |  |  | **T** |  |  |  |  | 8 |

**מטריצה שכנות**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | index |
| **T** |  |  |  | **T** |  |  |  | **T** | 0 |
|  |  | **T** |  |  |  |  | **T** |  | 1 |
|  | **T** |  | **T** |  |  | **T** |  |  | 2 |
|  |  |  |  |  | **T** |  |  |  | 3 |
| **T** |  |  |  | **T** |  |  |  | **T** | 4 |
|  | **T** |  | **T** |  |  | **T** |  |  | 5 |
|  |  | **T** |  |  |  |  | **T** |  | 6 |
|  | **T** |  | **T** |  |  | **T** |  |  | 7 |
| **T** |  |  |  | **T** |  |  |  | **T** | 8 |

**מטריצה קשירות**

1. חישוב מספר רכיבי קשירות

**Loop2**

**Loop1**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | index |
| **T** |  |  |  | **T** |  |  |  | **T** | 0 |
|  |  | **T** |  |  |  |  | **T** |  | 1 |
|  | **T** |  | **1** |  |  | **T** |  |  | 2 |
|  |  |  |  |  | **T** |  |  |  | 3 |
| **T** |  |  |  | **T** |  |  |  |  | 4 |
|  | **T** |  | **T** |  |  |  |  |  | 5 |
|  |  | **T** |  |  |  |  |  |  | 6 |
|  | **T** |  |  |  |  |  |  |  | 7 |
| **T** |  |  |  |  |  |  |  |  | 8 |

**int numComp = 0 number of connected**

**components**

**int conComp[i] connected components array**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| data | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**Loop1 i = 0 to n**

**if conComp[i] == 0**

**numComp++**

**conComp[i] = numComp**

**endif**

**Loop2 j = i+1 to n**

**if (conComp[j] == 0 && mat[i][j])  
 conCompA[j] = numComp**

**endif**

**endloop2**

**endloop1**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |  |  |
| data | **1** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Loop1 = 0 | **numComp = 1** |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |  |  |
| data | **1** | 0 | 0 | 0 | **1** | 0 | 0 | 0 | **1** | Loop2 | **numComp = 1** |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |  |  |
| data | **1** | **2** | 0 | 0 | **1** | 0 | 0 | 0 | **1** | Loop1 = 1 | **numComp = 2** |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |  |  |
| data | **1** | **2** | 0 | 0 | **1** | 0 | **2** | 0 | **1** | Loop2 | **numComp = 3** |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |  |  |
| data | **1** | **2** | **3** | 0 | **1** | 0 | **2** | 0 | **1** | Loop1 = 2 | **numComp = 3** |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |  |  |
| data | **1** | **2** | **3** | **0** | **1** | **3** | **2** | **3** | **1** | Loop2 | **numComp = 3** |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |  |  |
| data | **1** | **2** | **3** | **4** | **1** | **3** | **2** | **3** | **1** | Loop1 = 3 | **numComp = 4** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| data | **1** | **2** | **3** | **4** | **1** | **3** | **2** | **3** | **1** |

number of components = 4

component number 1, vertices: [0,4,8]

component number 2, vertices: [1,6]

component number 3, vertices: [2,5,7]

component number 4, vertices: [3]

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**public** **static** **void** printBoolMatrix(**boolean**[][] mat){

**for**(**int** i=0; i<mat.length; i++){

**for**(**int** j=0; j<mat[0].length; j++){

System.***out***.print(mat[i][j]+", ");

}

System.***out***.println();

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**public** **static** **boolean**[][] initB(){

**boolean** [][] mat = {

{**false**,**false**,**false**,**false**,**true**,**false**,**false**,**false**,**false**}, // 0

{**false**,**false**,**false**,**false**,**false**,**false**,**true**,**false**,**false**}, // 1

{**false**,**false**,**false**,**false**,**false**,**true**,**false**,**true**,**false**}, // 2

{**false**,**false**,**false**,**false**,**false**,**false**,**false**,**false**,**false**}, // 3

{**true**,**false**,**false**,**false**,**false**,**false**,**false**,**false**,**true**}, // 4

{**false**,**false**,**true**,**false**,**false**,**false**,**false**,**true**,**false**}, // 5

{**false**,**true**,**false**,**false**,**false**,**false**,**false**,**false**,**false**}, // 6

{**false**,**false**,**true**,**false**,**false**,**true**,**false**,**false**,**false**}, // 7

{**false**,**false**,**false**,**false**,**true**,**false**,**false**,**false**,**false**}}; // 8

**return** mat;

}

**משימה:**

ליצור פונקציה שמחשבת ומדפיסה אתמספר רכיבי הקשירות וגם מדפיסה את כל הקודקודים של כל אחד מרכיב הקשירות

//Problem 3: Number connected component + List connected component

**public** **static** **void** connectComponentsOfGraphBoolean(**boolean** [][] mat)