Preston Bruce

1. Add a method has Edge() to Graph which takes two int arguments v and w and returns true if the graph has an edge v-w, false otherwise. Assume use of adjacency list.

1. Declare array of length N (N = number of vertices)

2. store vertices = number of vertices, store edges = number of edges

3. Edges are stored in order pair form in edgesPairs example: {(0,1), (0,2)...}

4. Create method Edge() which is described as follows:

5. Create nested for loops to search edgesPairs

for(i=0, i<length(vertices),i++){

for a = 0, a<=length(edges){

if edges[i] == vertices + edges

return True

break

Then

return False

}

}

2. Represent the given map as a graph and answer the following questions

A.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F |
| A | 0 | 1 | 1 | 0 | 1 | 0 |
| B | 1 | 0 | 1 | 1 | 0 | 0 |
| C | 1 | 1 | 0 | 1 | 1 | 1 |
| D | 0 | 1 | 1 | 0 | 0 | 1 |
| E | 1 | 0 | 1 | 0 | 0 | 1 |
| F | 0 | 0 | 1 | 1 | 1 | o |

B. Explain how we can use the graph-coloring problem to color the map so that no two neighboring regions are colored the same. In the graph coloring problem two adjacent vertices will colored differently.  For example, if vertices A and B are adjacent vertex A can be colored 'blue' and B can be colored anything but 'blue'.

1. Create array of size named Vertices

2. In each index of Vertices create a linked list name Edges

Store in Vertices[0] = {B, C, E}

Vertices[1] = {A, C, D}

Vertices[2] = {A, B, D, E, F}

Vertices[3] = {B, C, F}

Vertices[4] = {A, C, E}

Vertices[5] = {C, D, F}

3. Create for Loop

for(x=0; x<length(Vertices); x++){

y=|x-1|

if Vertices[x] != Vertices[y]

Assign Vertices[x] a color

Else

Assign Vertices[x] a color != to Vertices[y]

}

C.

3. For the graph depicted in the figure write (1) the adjacency matrix data structure, and (2) adjacency-list data structure, to store the data.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
| A | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| B | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| E | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| F | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| G | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| H | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| I | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| J | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| K | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| L | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| M | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| N | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| O | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| P | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

|  |  |
| --- | --- |
| A | B E F |
| B | A C F |
| C | B D G |
| D | C G H |
| E | A F I |
| F | A B E |
| G | C D L |
| H | D L |
| I | E F J M N |
| J | G I K |
| K | G J O N |
| L | G H P |
| M | I N |
| N | I K M |
| O | K |
| P | L |

4. Let A be the adjacency matrix of an undirected graph. Explain what property of the matrix indicates that,

1. the graph is complete.
   1. The graph is complete if all elements of the matrix connect are equal to one
2. the graph has a loop, i.e., an edge connecting a vertex to itself.
   1. A graph has a loop if the adjacency matrix only has an element equal to one with a single edge
3. the graph has an isolated vertex, i.e., a vertex with no edges incident to it.
   1. If there are no edges the adjacency matrix is isolated