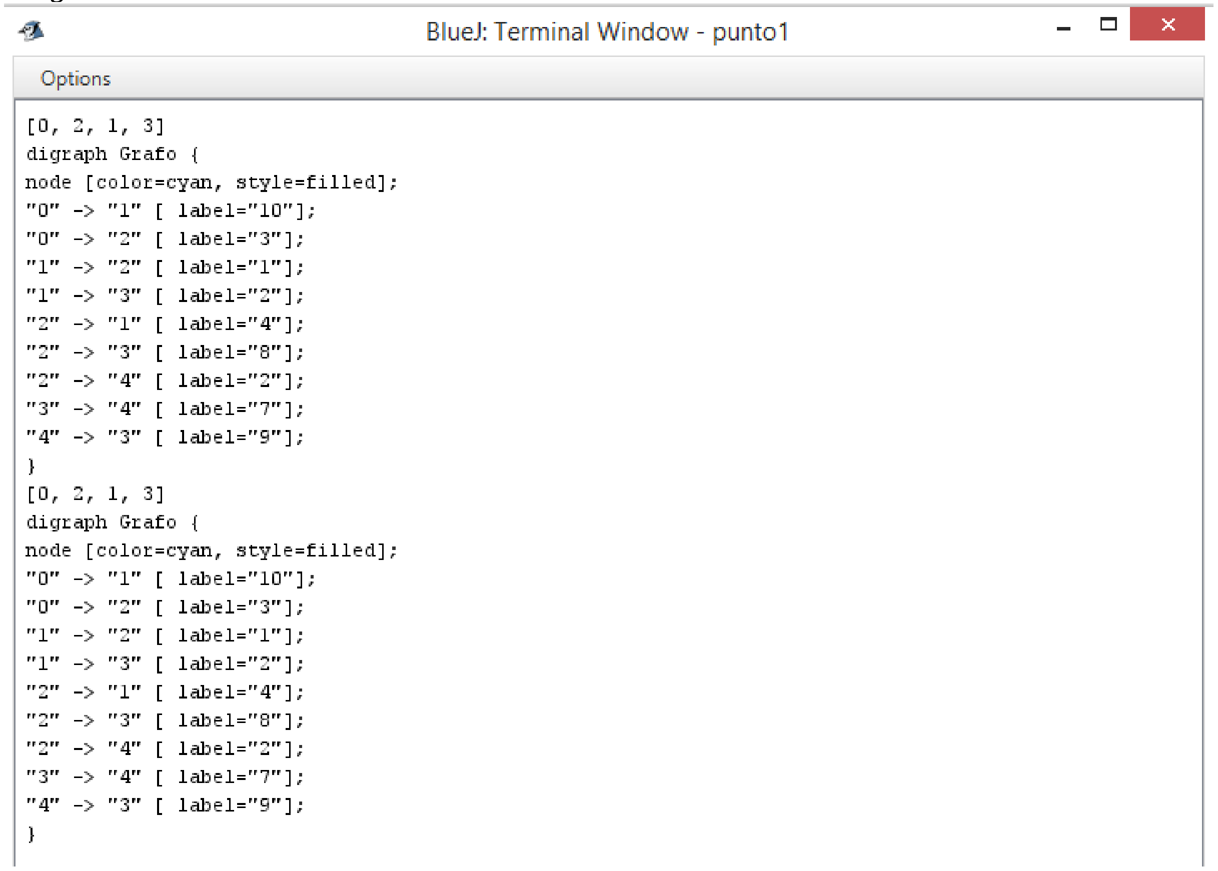
Laboratory practice No. 5: Implementation of a Graph

Manuel Garcia
Universidad Eafit
Medellín, Colombia
megarciaj@eafit.edu.co

Isaias Labrador
Universidad Eafit
Medellín, Colombia
ilabradors@eafit.edu.co

3) Practice for final project defense presentation

1. **Image:**



```
[0, 2, 1, 3]
digraph Grafo {
node [color=cyan, style=filled];
"0" -> "1" [ label="10"];
"0" -> "2" [ label="3"];
"1" -> "2" [ label="1"];
"1" -> "3" [ label="2"];
"2" -> "1" [ label="4"];
"2" -> "3" [ label="8"];
"2" -> "4" [ label="2"];
"3" -> "4" [ label="7"];
"4" -> "3" [ label="9"];
}
[0, 2, 1, 3]
digraph Grafo {
node [color=cyan, style=filled];
"0" -> "1" [ label="10"];
"0" -> "2" [ label="3"];
"1" -> "2" [ label="1"];
"1" -> "3" [ label="2"];
"2" -> "1" [ label="4"];
"2" -> "3" [ label="8"];
"2" -> "4" [ label="2"];
"3" -> "4" [ label="7"];
"4" -> "3" [ label="9"];
}
```

2. **With matrices the graph is just a square matrix were the nodes implicitly are the “x” coordinates of the first row. To express that a node is related to another a weight is designated to the spot on the matrix where “i” is the source node and “j” is the destination of the arrow.**

With lists what was done was a LinkedList full of LinkedLists that had key and value pairs as their type. This allowed for the lists to have relations be unordered yet traceable through the use of their key.

3. **It's best to use the matrix implementation when we will be asked a lot whether there are arcs between two nodes or not. On the other hand, if we instead want to know who all of the neighbors of a particular node very oftenly then it is better to use the list implementation.**
4. **Exercise 1.3 is better done with matrices because they make Dijsktra's algorithm O(n^2) while lists make it O(n^2logn).**
5. **In social networks it's always best to work with lists because a matrix would cause there to be too many unused memory slots taken up and eventually storage would run out.**
6. **Matrices are best in this scenario. This is because we know that all of the slots would be in use and because we want to get the weights inside every slot, which is O(1) for matrices.**
7. **Complexity of exercise 2.1= n*nn*nr**
8. **n is the number of graphs that are going to be described, nn is the number of nodes each graph has, and nr is the amount of relations within each graph.**

4)

1.

	0	1	2	3	4	5	6	7
0				1	1			
1	1		1			1		
2					1		1	
3								1
4			1					
5								
6			1					
7								

2. 0 -->[3,4]
   1 -->[0,2,5]
   2 -->[4,6]
   3-->[7]
   4 -->[2]
   5 -->[]
   6 -->[2]
   7 -->[]

3. *O(n^2)*