Assignment 3 - Graph Bacon Number

Goals

The purpose of this assignment is to learn to

1. Use the IMDB Actor Movie graph.
2. Compute BFS on that graph.
3. Highlight a shortest path in the graph.

You will generate a visualization that looks like that!

Programming part

Task

Highlight the shortest path between two actors in a Movie Actor graph.

Getting Started

1. Open your scaffolded code.
2. Plug in your credentials.
3. Change the style of nodes Cate_Blanchett and Kevin_Bacon_(I), directly attached nodes, and directly attached edges.
4. Compile, run, and visualize.

Perform BFS

1. Write a BFS traversal in `getBaconNumber` that keeps track of parent information. Here is the algorithm:

```
BFS(G=(V,E), root)
  forall v in V
    mark[v] = false;
  mark[root] = true;
  queue.push(root);
  while (! queue.empty() )
    v = queue.pop();
    for (u in neighboor(v))
      if (mark[u] == false)
        mark[u] = true;
        parent[u] = v;
```

2. We recommend using a built-in associative array for storing parents, such as Java's `HashMap` or C++'s `std::unordered_map`.
3. We recommend using a built-in queue, such as Java's `ArrayDeque` or C++'s `std::queue`.

Style the BFS path

1. Start from the Cate_Blanchett node.
2. Color the current node red and make it bigger.
3. Style the edge from the current node to its parent. Make it red and bigger.
4. Go to the parent node and go back to 2 until Kevin_Bacon_(I) has been reached.

Help

for Java

ArrayDeque documentation

HashMap documentation

Element documentation

GraphAdjListSimple documentation

ElementVisualizer documentation

LinkVisualizer documentation

ActorMovieIMDB documentation

for C++

std::queue documentation

std::unordered_map documentation

Element documentation

GraphAdjList documentation

ElementVisualizer documentation

LinkVisualizer documentation

ActorMovieIMDB documentation

for Python

Queue documentation

Element documentation

GraphAdjList documentation

ElementVisualizer documentation

LinkVisualizer documentation