PROBLEM OF THE MONTH

October, 2017

2017-2018

MATHEMATICS

5 points: A car drove from point A to point B with average fuel efficiency of 32 miles per gallon. During the last ¼ of the trip, its fuel efficiency was 36 miles per gallon. What was the fuel efficiency during the first ¾ of its trip?

Hint: Assume that the total distance travelled is d. How much fuel is used during the whole trip? How much on each of the segments?

Answer: $30\frac{6}{7}\ mpg$

Solution: Let d be the total distance travelled. The total amount of fuel used in the whole trip was d/32. The amount used during the last ¼ of the trip was d/144 (see the table below).

Distance (in miles)	First 3d/4	Last d/4	Total distance, d
Fuel use (in gal)	?	(d/4)/36=d/144	d/32

This means that the amount of fuel used during the first ¾ of the trip was: $\frac{d}{32} - \frac{d}{144} = \frac{7d}{288}$.
The efficiency can be found as: $\frac{3d/4}{7d/288} = \frac{216}{7} = 30\frac{6}{7}\ mpg$

10 points: A car drove from a town in New York State to a town in New Jersey with an average fuel efficiency of 36 miles per gallon (mpg). While in NY, its fuel efficiency was 30 mpg, and its average speed was 50 miles per hour (mph). In NJ, its fuel efficiency was 40 mpg

and average speed was 60 mph. What was the average speed of the whole trip, if the car did not visit any other state?

Hint: Let *d* be the total distance travelled, out of which *xd* NY, and *(1-x)d* in NJ. How much fuel was used on each segment? You can find x, and then express the total travel time in terms of d.

Answer: $56.25\ mph$

Solution: Let d be the total distance travelled, out of which xd NY, and (1-x)d in NJ. In the table below we put the distance, fuel used and time at each of the segments (in NY, in NJ and total):

Distance travelled	In NY: xd	In NJ : (1-x)d	Total : d
Fuel use (in gal)	xd/30	(1-x)d/40	d/36
Time (in hrs)	xd/50	1-x)d/60	?

By balancing the amount of fuel used in NY+NJ with the total, we obtain: $\frac{xd}{30} + \frac{(1-x)d}{40} = \frac{d}{36}$. This gives: $4x + 3(1-x) = 120/36 = 10/3$. Therefore, $x = 10/3 - 3 = 1/3$. We can now calculate the total time of travel: $xd/50 + (1-x)d/60 = \frac{d}{150} + \frac{d}{90} = \frac{4d}{225}$. The average speed is therefore $225/4 = 56.25\ mph$.

PHYSICS

This month Physics problems are on the free fall and projectile motion. You might find the following links useful.
https://en.wikipedia.org/wiki/Free_fall
https://en.wikipedia.org/wiki/Projectile_motion
http://hyperphysics.phy-astr.gsu.edu/hbase/traj.html#ffall

5 points:

A ball is thrown upwards with initial velocity $12\ m/s$. One second later another ball is thrown upwards. What should be an initial velocity of the second ball so that both balls land at the same moment? Neglect air friction. The free fall acceleration is $g \approx 10\ m/s^2$.

Hint:

What time does it take for the ball thrown upwards to land?

Answer: $V \approx 7\ m/s$

Solution:

The vertical coordinate of the ball $y = v_0 t + \frac{1}{2} g t^2$ (we took initial coordinate as $y = 0$). The ball lands in time t such that $y = v_0 t + \frac{1}{2} g t^2 = 0$. We find $t = 2v_0/g$. For the first ball we have $t = 2v_0/g = 2 * 12/10\ s = 2.4s$. This means that the second ball should land in time $t' = 2v_0'/g = t - 1s = 1.4s$. We find $v_0' = gt'/2 = 10 * 1.4/2\ m/s = 7m/s$.

10 points:

A ball is thrown from the horizontal surface at an angle α to the horizon. What should be the initial velocity V of the ball so that the ball would get into a hole which is at the distance L from the point of the throw? The ball can bounce from the horizontal surface. Assume that all such bounces are absolutely elastic and neglect air friction. The free fall acceleration is g.

Hint:

What is the time between bounces of the ball?

Answer: $V = \sqrt{\frac{gL}{n \sin 2\alpha}},\ n = 1, 2, 3, \ldots$

Solution:

The time between bounces of the ball is $t = 2v_{0y}/g$, where $v_{0y} = V \sin \alpha$ is the initial vertical velocity of the ball. The condition that the ball gets into the hole is $(v_{0x}t)n = L$, where $v_{0x} = V \cos \alpha$ is the initial horizontal velocity of the ball and $n = 1, 2, 3, \ldots$ is any integer number (number of bounces before getting into the hole). We have: $V \cos \alpha \frac{2V \sin \alpha}{g} n = L$. We obtain $V^2 = \frac{gL}{2n \sin \alpha \cos \alpha} = \frac{gL}{n \sin 2\alpha}$, where we used trigonometric formula $\sin 2\alpha = 2 \sin \alpha \cos \alpha$. Finally, we obtain $V = \sqrt{\frac{gL}{n \sin 2\alpha}}$, with $n = 1, 2, 3, \ldots$.

CHEMISTRY

5 points:

Four beakers are placed on each scale of the balance. The beakers contain:
on the left scale
- water,
- sodium carbonate (powder,)
- sodium bisulfate (powder)
- potassium acetate (powder)

on the right scale
- water
- phosphoric acid (80%)
- magnesium acetate (powder)
- iron pellets

(each beaker has a volume of 250 mL and contains 50 grams of one of the above listed material)

The scales are at equilibrium.

You have to do some chemical reaction(s) with these chemicals that will shift the equilibrium less than in 1 hr after the process started (the sensitivity of the balance is sufficient to detect the mass change as low as 1 gram). The following rules must be observed:
1. No transfer of matter is allowed between the two scales during your manipulations with the beakers (for example, you can mix the content from the beaker 1 and beaker 2 from the left scale, and put both beakers back; however, you cannot mix the content of the beaker 1 from the left scale and the beaker 2 from the right scale).
2. You can take the beakers from each scale temporarily (to mix their content), but you must put them back after mixing is finished. No spillage of any material is allowed.

In your answer, predict at which direction the equilibrium shifts.

Hint: By mixing a content of some beakers you may start a reaction where gas evolves, so the total mass of the chemicals on one scale decreases. Please, keep in mind that virtually no reaction occurs between phosphoric acid and iron: phosphoric acid is not a strong acid, and iron phosphate that forms initially is insoluble, it precipitates on iron's surface thereby protecting the metal from further reaction.

Solution:

Although iron reacts with strong acids yielding gaseous hydrogen, the reaction between iron and phosphoric acid doesn't go: phosphoric acid is not strong, and iron phosphate that forms in this reaction is insoluble, and it forms a thin and dense layer of iron phosphate that protects iron from further reaction with phosphoric acid. That means by mixing iron shavings with phosphoric acid you would hardly be capable of shifting the balance's position.

What *will* work is mixing water, sodium carbonate and sodium bisulfate in one beaker: sodium bisulfate ($NaHSO_4$) has a dual nature, it is a salt and an acid at the same time, because only one hydrogen is replaced with sodium there. The reaction will go according to the equation:

$$Na_2CO_3 + 2NaHSO_4 \rightarrow Na_2SO_4 + H_2O + CO_2$$

That means the beaker will lose 44 grams (a molecular weight of carbon dioxide) of mass per 103 gram (molecular weight of sodium carbonate). If each beaker contain around 20 gram of each substance (quite a reasonable assumption) the mass change will be easily detectable.

10 points:

Five beakers are placed on each scale of the balance. The beakers contain:

<u>on the left scale</u>

- water,
- sodium hydrophosphate (powder)
- aluminium pellets
- potassium sulfate (powder)
- copper sulfate

<u>on the right scale</u>

- water
- silicic acid (meta, powder)
- iron turnings
- aluminium pellets
- mercuric chloride (powder)

(each beaker has a volume of 250 mL and contains 50 grams of one of the above listed material)

The scales are at equilibrium.

You have to do some chemical reaction(s) with these chemicals that will shift the equilibrium less than in 1 hr after the process started (the sensitivity of the balance is sufficient to detect the mass change as low as 1 gram). The following rules must be observed:

1. No transfer of matter is allowed between the two scales during your manipulations with the beakers (for example, you can mix the content from the beaker 1 and beaker 2 from the left

scale, and put both beakers back; however, you cannot mix the content of the beaker 1 from the left scale and the beaker 2 from the right scale).
2. You can take the beakers from each scale temporarily (to mix their content), but you must put them back after mixing is finished. No spillage of any material is allowed.

In your answer, predict at which direction the equilibrium shifts.

Hint: By adding a content of some beaker(s) to one of the metals you may activate it, so it will start to absorb atmospheric oxygen. As a result, the total mass of the chemicals on one scale increases.

Solution:

Aluminium is a very active metal, it reacts with atmospheric oxygen very quickly, however, the oxide forms a very thin but very hard and dense protective layer that prevents further reaction and makes aluminium chemically inert. The situation changes when you drop a little bit of a solution of some mercury salt on aluminium's surface. Mercury that forms as a result of a chemical reaction between mercuric chloride and aluminium makes a thin film of mercury *amalgam* (a solution of aluminium in mercury) under the aluminium oxide layer, so the protective layer peels off the metal surface. The problem is that the amalgam film does not protect aluminium from further reaction with atmospheric air, so the metal starts to react with air exothermically, and the reaction continues until the whole piece of the metal is converted into the oxide. For us that means that by mixing aluminium pellets and a little bit of water and mercuric chloride we will see that the metal is converting into a voluminous and loose powder of aluminium oxide and the scale is going down.

BIOLOGY

5 points:

A Museum of Natural History was undergoing an extensive renovation during the summer. Upon completion of all the hard work, the construction crew's last task was to put up labels on the wall for exhibits that the newly reorganized 2nd and 3rd floors will house. They had to finish putting the labels up in the exact locations that exhibits will be located at and do so before the precision handling crew brought the exhibits out of storage rooms. On the last day of their work, during their lunch break, two construction workers discussed what they have been up to in the morning:

worker 1: I have been putting up labels in what looks like latin all morning on the 2nd floor. Now I don't know a word of latin myself, but I can say that my labels had two words in them, and some of the exhibits that would stand next to each other had identical second words in their names.
worker 2: Very strange! On 3rd floor, I had an opposite situation: sometimes labels next to each other only had their first words identical, but not the second. And sometimes, there would be names with 3 words. In that case, both first and second words would be the same for some neighbors, and the third word would be different.
worker 1: Let us come back and see what the curators have in their mind once the whole thing re-opens?
worker 2: Good idea!

Based on the naming of the exhibits and their grouping, what do you think the museum might be housing on the 2nd and 3rd floors in its newly renovated arrangement?

Answer:

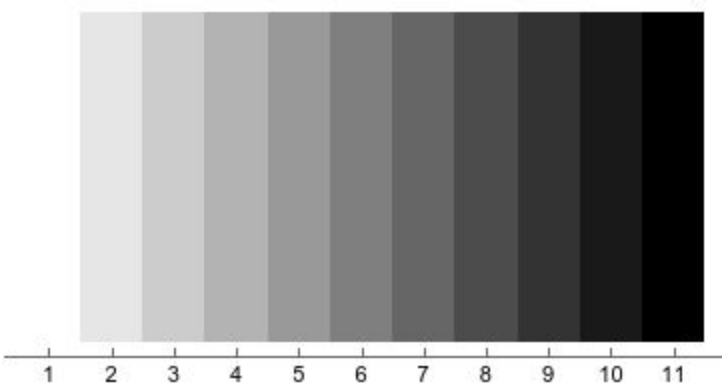
- The convention of naming of a biological organism, 2 and 3 word names (binomens and trinomens) are reserved for species and subspecies. For example: *Lymantria dispar* being the species name for gypsy moth. All class names at levels above species are given as a single word (uninomial naming). For example: *Lepidoptera* being an order that includes thousands of species. Hence it must be that the exhibits on both floors were representatives of various species or their finer subdivisions. https://en.wikipedia.org/wiki/International_Code_of_Zoological_Nomenclature
- The arrangement of exhibits on 2nd floor has 2 main potential explanations: as per convention, the 2nd names can stand for the geography of the species' typical habitat or the discoverer's name. Therefore, the exhibits on this floor might be subdivisioned into several geographical areas or dedicated to prominent individuals and their contributions to the study of natural history. Examples of naming species after an individual: *Homonota darwinii*, *Rhinoderma darwinii*, *Caerostris darwini*, and *Demandasaurus darwini*, etc (i.e. the species named after Charles Darwin). Additionally, in recent years, there is a trend of naming newly discovered species after contemporary individuals of

mostly media and political significance for comical effect (https://en.wikipedia.org/wiki/List_of_organisms_named_after_famous_people).

- Exhibits on 3rd floor must've been species that are closely related to each other in the sense of genetic distance. By the same zoological naming convention, in a bi-/trinomial name, the 1st word stands for the genus name, 2nd word for species name and if necessary, 3rd word may stand for the subspecies name. Therefore some of the neighboring exhibits with 3-word names with first and second words identical might simply be closely related say moths: *Lymantria dispar dispar*, *Lymantria dispar japonica*, and *Lymantria dispar asiatica*.

10 points:

Our visual system is known to play tricks on us quite frequently: we constantly encounter with or without our knowledge multitude of optical illusions that make us "see things" not exactly the way they actually are. One such example can be seen in the figure below. In a visual scene where multiple bands of brightness are packed closely together, our visual machinery tricks us into seeing each individual band of brightness (exactly 11 bands shown: from white to black) as a gradually changing pattern in itself: each band between 2nd and 10th appear to us as darker



on the left and brighter on the right. But we know that each band is in fact a single uniform level of brightness, hence exactly the same color. It is no doubt you have noticed this before.

A) Describe how the human visual system accomplishes this deception and what purpose this mechanism serves, commenting on whether seeing a gradually varying brightness where there really is a single level of brightness is advantageous to our survival or not. Make sure you define all terms used.

B) Imagine an obscure and rare primate species whose eyes and brain are anatomically similar to humans', but it isn't yet clear to the scientific community whether they also possess a visual system capable of the same trickery. Describe an experiment that would help resolve this ambiguity and establish whether their vision is as tricky as ours.

Answer:

- The mechanism responsible for the Mach band illusion is known as lateral inhibition (https://en.wikipedia.org/wiki/Lateral_inhibition). Lateral inhibition describes any process where a small part of sensory organ/system being stimulated inhibits and distorts the signaling activity of its immediate neighbor. In the visual system, it is a process of a patch of neurons receiving light from a bright source inhibiting the activity of their neighbors that are receiving signals from neighboring darker sources. This makes the left-most edge of each band in our figure appear darker due to the signals in our visual system corresponding to their brightness being inhibited (under-represented) by the activity of the neurons corresponding to the brighter areas immediately to the left of them. This inhibition of darker patches by neighboring brighter patches serves among others the purpose of contrast and edge enhancement, which enables earlier detection of the surrounding stimuli. This in turn speeds up further decisions on whether to flee (dangerous stimulus) or pursue (desirable stimulus) by the specimen.
- The most direct (albeit most invasive) experiment for detecting whether the newly discovered species displays lateral inhibition would be done by the way of direct electrode measurements simultaneously from multiple neurons of visual system that receive signals from neighboring patches in the retina. By positioning the electrodes in close proximity to neurons of the center of the visual field of view of the primates, the electrodes will be tuned to "listening" to neurons being activated as well as their immediate neighbors. Now by presenting the visual system with stimulus containing Mach band-like gradients at specific times and seeing if activations of neurons receiving "brighter signals" are accompanied by deactivations of neurons receiving "darker signal" next to them at the time of stimulation, we can detect if lateral inhibition is present.

COMPUTER SCIENCE

- You can write and compile your code here: http://www.tutorialspoint.com/codingground.htm
- Your program should be written in Java or Python
- No GUI should be used in your program: eg., easygui in Python. All problems in POM require only text input and output. GUI usage complicates solution validation, for which we are also using *codingground* site. Solutions with GUI will have points deducted or won't receive any points at all.
- Please make sure that the code compiles and runs on http://www.tutorialspoint.com/codingground.htm before submitting it.
- Any input data specified in the problem should be supplied as user input, not hard-coded into the text of the program.
- Submit the problem in a plain text file, such as .txt, .dat, etc. **No .pdf, .doc, .docx, etc!**

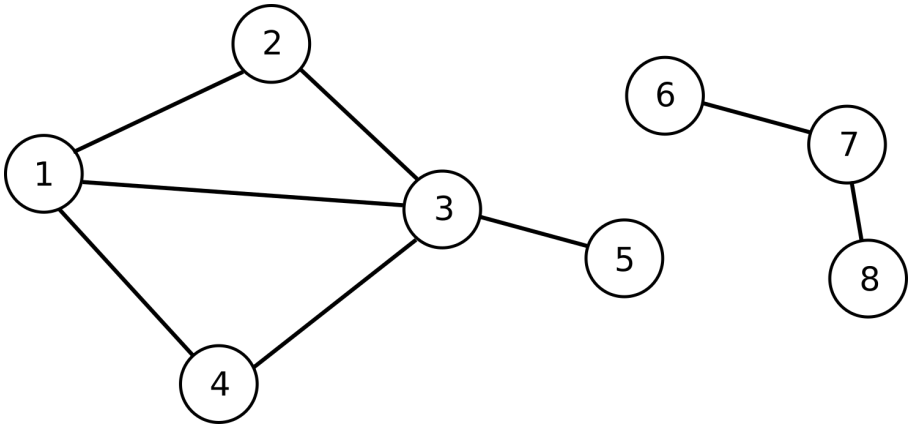
5 points:

Given a graph (https://en.wikipedia.org/wiki/Graph_(discrete_mathematics)*), your program must determine which node has the greatest number of neighbors.* The input is a 2xN array that lists each pair of connected nodes. The nodes are labeled by numbers:

For example, given the input

```
[[1,    2],
[2,    3],
[3,    4],
[4,    1],
[1,    3],
[3,    5],
[6,    7],
[7,    8]]
```

The output should be "3". The graph that corresponds to this example is:



Solution:

Python:

```
#!/usr/bin/python3

"""
idea: just count how many edges (connections) exist for a given vertex (node)

We assume that duplicate edges, e.g. "1 - 2" and "2 - 1" are valid case in this problem.
Moreover, even "1 - 2" can be valid, e.g. if there are 2 roads from town 1 to town 2.
If it were not, we could create a set that holds processed items. As we read them one by one,
e.g. "1 - 2" we can form literally a string "1-2" and check if it exists in the set. If not
then we process it, and add both "1-2" and "2-1" to the set (our graph is non-directional).
"""

import operator

edges = [[1,  2],
         [2,  3],
         [3,  4],
         [4,  1],
         [1,  3],
         [3,  5],
         [6,  7],
         [7,  8]]

counts = {} # vertex number -> edges count to/from it

# parse edges and count vertices
for i in range(len(edges)):
  edge = edges[i]
  vertex1 = edge[0]
  vertex2 = edge[1]
  if vertex1 in counts:
    counts[vertex1] += 1
  else:
    counts[vertex1] = 1
```

```
  if vertex2 in counts:
    counts[vertex2] += 1
  else:
    counts[vertex2] = 1

# select the largest count(s)
sorted_counts = sorted(counts.items(), key=operator.itemgetter(1)) # sort by counts
for i in range(-1, -len(sorted_counts)-1, -1): # we may have > 1 vertex with the max count
  if sorted_counts[i][1] < sorted_counts[-1][1]:
    break
  else:
          print("vertex/node %d has the most edges (%d)" % (sorted_counts[i][0],
sorted_counts[i][1]))

print("end.")
```

Java:

```
/*
idea: just count how many edges (connections) exist for a given vertex (node)

We assume that duplicate edges, e.g. "1 - 2" and "2 - 1" are valid case in this
problem.
Moreover, even "1 - 2" can be valid, e.g. if there are 2 roads from town 1 to town 2.
If it were not, we could create a set that holds processed items. As we read them one
by one,
e.g. "1 - 2" we can form literally a string "1-2" and check if it exists in the set. If
not
then we process it, and add both "1-2" and "2-1" to the set (our graph is
non-directional).
*/

import java.util.*;
import java.util.stream.Stream;

public class Graph5 {
  public static void main(String[] args) {
    int[][] edges = {{1, 2},
        {2, 3},
        {3, 4},
        {4, 1},
        {1, 3},
        {3, 5},
        {6, 7},
        {7, 8}};
    Map<Integer, Integer> counts = new HashMap<>(); // vertex number -> edges count
to/from it

    // parse edges and count vertices
    for(int[] edge : edges) {
      int vertex1 = edge[0];
      int vertex2 = edge[1];
      Integer count = counts.get(vertex1);
```

```
      if(count != null)
        counts.put(vertex1, count + 1);
      else
        counts.put(vertex1, 1);
      count = counts.get(vertex2);
      if(count != null)
        counts.put(vertex2, count + 1);
      else
        counts.put(vertex2, 1);
    }

    // select the largest count(s) by sorting in descending order by value, i.e. by
count
    Stream<Map.Entry<Integer, Integer>> sortedCounts =
counts.entrySet().stream().sorted(Collections.reverseOrder(Map.Entry.comparingByValue()
));
    Iterator<Map.Entry<Integer, Integer>> it = sortedCounts.iterator();
    Map.Entry<Integer, Integer> maxNode = it.next();
    System.out.printf("vertex/node %d has the most edges (%d)\n", maxNode.getKey(),
maxNode.getValue());
    while(it.hasNext()) { // we may have > 1 vertex with the max count
      Map.Entry<Integer, Integer> entry = it.next();
      if(entry.getValue() < maxNode.getValue())
        break;
      else
        System.out.printf("vertex/node %d has the most edges (%d)\n", entry.getKey(),
entry.getValue());
    }

    System.out.println("end.");
  }
}
```

10 points:

Given a graph (same input format as in the 5pt problem), and two nodes, determine if there is a path between the two nodes.

For example, given as input the same graph as above and nodes 1 and 5, your program should return “true”. Given input 5 and 6, your program should return “false”.

Solution:

Python:

```
#!/usr/bin/python3

"""
create a map/dictionary of each node/vertex connections/edges for a quick access
```

```
start with "start" adding it to "todo" list
move processed vertices to "done"
check "done" to detect loops
repeat until either found "end" or no more links
"""

from collections import deque

edges = [[1,  2],
         [2,  3],
         [3,  4],
         [4,  1],
         [1,  3],
         [3,  5],
         [6,  7],
         [7,  8]]

start = 1
end = 5
# start = 5
# end = 6

def bfs(vmap, start, end):
  todo = deque([start])
  done = deque([])
  while len(todo) > 0:
    item = todo.popleft()
    done.append(item)
    children = vmap[item]
    for child in children:
      if child == end:
        print("reached the final node")
        return True
      elif child in done:
        pass # ignore
      else:
        todo.append(child)
  print("we searched all nodes in this (sub)graph and did not find the target node")
  return False

vmap = {}
for i in range(len(edges)):
  if edges[i][0] in vmap:
    vmap[edges[i][0]].append(edges[i][1])
  else:
    vmap[edges[i][0]] = [edges[i][1]]
  # our graph is non-directional, let's add both ways
  if edges[i][1] in vmap:
    vmap[edges[i][1]].append(edges[i][0])
  else:
```

```
    vmap[edges[i][1]] = [edges[i][0]]

res = bfs(vmap, start, end)
print(res)

print("end.")
```

Java:

```
/*
create a map/dictionary of each node/vertex connections/edges for a quick access
start with "start" adding it to "todo" list
move processed vertices to "done"
check "done" to detect loops
repeat until either found "end" or no more links
*/

import java.util.*;

public class Graph10 {
  static boolean bfs(Map<Integer, List<Integer>> vmap, int start, int end) {
    Queue<Integer> todo = new LinkedList<>();
    Queue<Integer> done = new LinkedList<>();
    todo.add(start);
    while(todo.size() > 0) {
      Integer item = todo.remove();
      done.add(item);
      List<Integer> children = vmap.get(item);
      for(Integer child : children) {
        if(child == end) {
          System.out.println("reached the final node");
          return true;
        }
        else if(done.contains(child))
          ; // ignore
        else
          todo.add(child);
      }
    }
     System.out.println("we searched all nodes in this (sub)graph and did not find the
target node");
    return false;
  }

  public static void main(String[] args) {
    int[][] edges = {{1, 2},
                     {2, 3},
                     {3, 4},
                     {4, 1},
                     {1, 3},
                     {3, 5},
```

```
                    {6, 7},
                    {7, 8}};
    int start = 1;
    int end = 5;
    //int start = 5;
    //int end = 6;

    Map<Integer, List<Integer>> vmap = new HashMap<>();
    for(int i=0; i<edges.length; i++) {
      List<Integer> items = vmap.get(edges[i][0]);
      if(items != null)
        items.add(edges[i][1]);
      else {
        Integer item = edges[i][1];
        items = new LinkedList<>();
        items.add(item);
        vmap.put(edges[i][0], items);
      }
      // our graph is non-directional, let's add both ways
      items = vmap.get(edges[i][1]);
      if(items != null)
        items.add(edges[i][0]);
      else {
        Integer item = edges[i][0];
        items = new LinkedList<>();
        items.add(item);
        vmap.put(edges[i][1], items);
      }
    }

    boolean res = bfs(vmap, start, end);
    System.out.println(res);

    System.out.println("end.");
  }
}
```