Tour Guide
(Companion to the Knight’s Tour Activity)

Age group: 8 – adult

Abilities assumed: None

Time: 15 minutes for basic activity, longer with extended activities or younger groups

Size of group: 1 upwards

Focus
Algorithms, Sequences of instructions, Graphs, Requirements
Abstraction, Data representation, Computational Thinking

Syllabus Links
This activity can be used (for example)
• as a general introduction to what an algorithm is from KS2 up.
• to introduce algorithmic thinking from KS 2 up.

Summary
You are a tour guide and must come up with a tour that gets a tourist from their hotel to all the city sights and back to their hotel at the end of the day.

This activity is an example of creating an algorithm that is a simple sequence of instructions to do in order. It shows that if we have written down a solution to the problem in the form of an algorithm then we are able to do tours in future just by following the steps, without having to work it out from scratch again. Also if we write down the algorithm we can check that it definitely works by following it step by step on paper.

It also introduces graphs and with them the computational thinking idea that by representing a problem (i.e., the data we have about it) in a way that removes unnecessary detail (i.e., using abstraction) we can make the problem easier to solve.

Technical Terms
Algorithm, algorithmic thinking, data representation, evaluation, graph, node, edge, dry run, trace, requirements.

Materials
Per person:
- Copy of Tourist Map Board per person
- Single Smiley tourist piece/counter per person
- Route recording sheet
- Pencil
What to do

Setup
Give everyone a copy of the city Tube map below, a cut out piece and a route recording sheet to record their answer on.

The Problem
You are a hotel tour guide. Tourists staying in your hotel expect to be taken on a tour visiting all the city’s attractions. You have been given a special tube map that shows all the locations of the attractions and how they are linked by the tube lines.

Solving the problem
The class must work out a route that starts from the hotel and visits every tourist site. The ideal route will visit every site exactly once. They should try and work it out by placing the smiley piece on the hotel and then moving it from location to location following the lines. Looking for a route that works.

Record the algorithm
Once they have worked out a route they should write it down as a series of instructions on the route-recording sheet. Their solution is an algorithm for solving the problem.

Evaluating the algorithm
They should check that they definitely do have a correct solution. We don’t want to actually do a tour and find at the end of the day that we missed something important out and have to deal with angry tourists!
To do this they should dry run (or ‘trace’) their algorithm. That just means follow the steps of the algorithm on paper before they do it for real. One way to do this is to draw the route as they follow each instruction, ticking each location.
They need to check the following properties of (ie requirements for) their answer:

- It starts at the hotel.
- It visits every location.
- It does not pass through a location already visited.
- It ends at the hotel.

Of course as a real tour guide you would not just rely on checking the route on paper. You would go out and test it for real too. Programmers do the same thing. They check their program works on paper but then also check it for real – that is called testing.

Computational Thinking
The route they come up with is a sequence of instructions that can be followed to visit every tourist attraction and get back to the start. It is a simple algorithm for doing a tour of the city. Notice that there are several different routes you could take – so several different algorithms are solutions to the same problem. If we don’t write the route down then we may forget the steps we took and have to work it out again. Once we write the algorithm down we can follow it as many times as we want (give tours over and over again) or even give it to someone else to follow who does not have to work a route out for themselves.
Once we have written down an algorithm it is important that we **evaluate** it. We must check that it works. In particular that means that we must check that the algorithm meets a set of properties. These are known as **requirements**.

This problem was made easier to solve because of the way we represented the data as a diagram. The tube map is a special kind of diagram called a **graph**. A graph to a computer scientist is a series of dots (we call those the **nodes** of the graph) and lines that join them (we call those the **edges** of the graph).

The nodes of a graph **represent** something about the data that we are interested in. The edges show which nodes are linked in some way that we are interested in.

In our case we are interested in the tourist attractions (our nodes) and which ones are linked to each other by the underground connections (our edges). We aren’t interested in anything else about the places so we ignore everything else. We hide the exact locations and how far they each are apart as that doesn’t matter to our problem of finding a route that visits them all. The graph is an **abstraction** of the real city. We have hidden all the extra detail that we don’t need in creating the graph. That makes the information we are interested in much clearer.

Graphs are often used to represent information about the connections between things. You will often find them on the signs at bus stops, showing the routes, on train and underground maps. They are a very good **representation** in situations where you want to find routes from place to place as we did here. The simplified graph makes it easier to find a route than if we had a fully detailed map as then the information that mattered would be hard to see amongst all the detail.

As an aside computer scientists actually call this kind of tour where you visit every node in a graph exactly once returning back to the start a **Hamiltonian cycle**.
Variations and Extensions

The Knight's Tour
This activity is intended to be used as an introduction to graphs before moving on to the Knight’s Tour activity. It focuses on the power of using abstraction and how the choice of representation can make a problem easier.

Make your own graph
Come up with graphs for students to draw for things in their lives. Draw a graph of places in your school – eg, the reception, your classroom, the assembly hall, the playground. Draw a node for each place and put edges between any place that there is a way to get directly between them without having to go via any of the others. Now suppose you were asked to give a group of prospective students a tour of the school. Work out a route to take them from the reception to all the interesting places and back to the entrance.

Links to other activities
The following activities are also available via teachinglondoncomputing.org

The Knight’s Tour
Find a way for a knight to visit every square on a board exactly once. Create graphs to represent a problem. See the power of using abstraction and how the choice of representation can make a problem easier. This follows directly from the Tour Guide Activity.

The Swap Puzzle
Solve a puzzle, coming up with an algorithm that your team can follow faster than anyone else. Introduce the idea of the solution to a problem being a set of instructions that allow others to ‘solve’ it with no understanding. Explore how different algorithms can solve the same problem but may not be equally good – some may be faster.

Live demonstration of this activity
Teaching London Computing give live sessions for teachers demonstrating many of our activities. See http://teachinglondoncomputing.org/ for details. Videos of some activities are also available or in preparation.

Use of this material

Attribution NonCommercial ShareAlike - "CC BY-NC-SA"

This license lets others remix, tweak, and build upon a work non-commercially, as long as they credit the original author and license their new creations under the identical terms. Others can download and redistribute this work just like the by-nc-nd license, but they can also translate, make remixes, and produce new stories based on the work. All new work based on the original will carry the same license, so any derivatives will also be non-commercial in nature.
Guided Tour
City Tube Map

Starting at the hotel, plan a route so that tourists can visit every tourist attraction just once ending up back at the hotel.
## Route Recording Sheet

<table>
<thead>
<tr>
<th>STEP</th>
<th>PLACE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Start at the Hotel</td>
</tr>
<tr>
<td>2</td>
<td>Then go to the</td>
</tr>
<tr>
<td>3</td>
<td>Next go to the</td>
</tr>
<tr>
<td>4</td>
<td>Next go to the</td>
</tr>
<tr>
<td>5</td>
<td>Next go to the</td>
</tr>
<tr>
<td>6</td>
<td>Next go to the</td>
</tr>
<tr>
<td>7</td>
<td>Next go to the</td>
</tr>
<tr>
<td>8</td>
<td>Next go to the</td>
</tr>
<tr>
<td>9</td>
<td>Next go to the</td>
</tr>
<tr>
<td>10</td>
<td>Next go to the</td>
</tr>
<tr>
<td>11</td>
<td>Next go to the</td>
</tr>
<tr>
<td>12</td>
<td>Next go to the</td>
</tr>
<tr>
<td>13</td>
<td>Finally go to the Hotel</td>
</tr>
</tbody>
</table>
One possible solution route to the tour guide problem
## Route Recording Sheet

<table>
<thead>
<tr>
<th>STEP</th>
<th>PLACE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Start at the Hotel</td>
</tr>
<tr>
<td>2</td>
<td>Then go to the Science Museum</td>
</tr>
<tr>
<td>3</td>
<td>Next go to the Toy Shop</td>
</tr>
<tr>
<td>4</td>
<td>Next go to the Big Wheel</td>
</tr>
<tr>
<td>5</td>
<td>Next go to the Park</td>
</tr>
<tr>
<td>6</td>
<td>Next go to the Zoo</td>
</tr>
<tr>
<td>7</td>
<td>Next go to the Aquarium</td>
</tr>
<tr>
<td>8</td>
<td>Next go to the Art Gallery</td>
</tr>
<tr>
<td>9</td>
<td>Next go to the Waxworks</td>
</tr>
<tr>
<td>10</td>
<td>Next go to the War ship</td>
</tr>
<tr>
<td>11</td>
<td>Next go to the Castle</td>
</tr>
<tr>
<td>12</td>
<td>Next go to the Cathedral</td>
</tr>
<tr>
<td>13</td>
<td>Finally go to the Hotel</td>
</tr>
</tbody>
</table>