

# **Knight's Tour**

## **(Companion to the Tour Guide Activity)**

<b>Age group:</b>	8 – adult
<b>Abilities assumed:</b>	None
<b>Time:</b>	50-60 minutes
<b>Size of group:</b>	1 upwards

### **Focus**

Graphs, Data Representation, Generalisation, Computational Thinking

### **Syllabus Links**

This activity can be used (for example)

- as a general introduction to data representation from KS2 up.
- as a general introduction to graphs from KS2 up.
- to develop computational thinking problem solving from KS2 up.

### **Summary**

*Solve a puzzle where you must find a way for a knight to visit every square on a board exactly once.* This activity involves trying to solve a puzzle using two different representations. To start with it is fairly hard, but when the representation of the board and moves is changed it becomes really easy. Create graphs to represent the problem. See the power of using abstraction and how the choice of representation can make a problem much easier.

This activity follows directly from the Tour Guide Activity – read that first.

### **Technical Terms**

Graph, node, edge, abstraction, data representation, generalisation, pattern matching.

### **Materials**

Per person:

- Materials for the Tour Guide Activity (which is done first)
- Knight's Tour Rule sheet
- Knight's Tour Board
- Single Knight piece/counter
- Blank Solution sheet
- Completed solution sheet
- Solution graph
- Pencil

## What to do

Introduce the class as a puzzle solving session where you will see how the power of computational thinking can sometimes make puzzles easier to solve.

### Step 1: Solving a simple puzzle: the Tour Guide

First have the class do the Tour guide activity (available from the resources section of [www.teachinglondoncomputing.org](http://www.teachinglondoncomputing.org)). This involves solving a fairly simple problem of finding a route round a series of tourist sites that visits them all. Once everyone has had a chance to come up with a solution, show everyone a route that does work. For a shorter version of this part just get the class to draw the route rather than fill out the steps as a written algorithm.

### Step 2: Trying to solve the Knight's Tour puzzle

Next give everyone a copy of the Knight's Tour rules, board, a Knight piece and blank solution sheet. Explain that the knight moves as in chess – 1 square along and 2 up in any direction (see rules sheet). Challenge everyone to find a tour of the board for the knight – a sequence of moves, starting from Square 1 that visits every square exactly once and finishes where it started.

Emphasise that they must give you an algorithmic solution. They must record the sequence of moves. Point out you won't believe they have solved it unless you can follow their instructions and correctly get the knight to complete the tour of the board.

Give them time to solve the problem. If anyone has a solution then check it and if correct congratulate them. If not then show them how it goes wrong by stepping through the algorithm. It doesn't matter if most don't solve it. The next step helps. If any do work out the later steps themselves even better – encourage them to do so.

### Step 3: Generalising the two puzzles

Discuss whether the class think that this is easier or harder than the tour guide problem. They are likely to have found the tour guide easier.

Explain that the Knight's tour puzzle can be solved really easily if you use some computational thinking tricks. Encourage them to see how the two problems are similar – in fact we can **generalise** the statement of the two problems to be exactly the same – the aim of both is to come up with a route that visits everywhere once and returns to the starting place. Discuss with the class why the tour guide problem seems easier, encouraging them to work it out for themselves – the underground map shows the information that matters clearly, ignoring detail that doesn't matter.

### Step 4: Represent it as a graph

Once we have seen that the problems can be generalized so that they can be stated the same way, the next step is to see if we can make them actually look the same: i.e., can we change what the board and moves look like to make the things that matter in the puzzle as clear as they were for the Tour Guide problem?

We will make a map (computer scientists actually call this kind of map-like diagram a **graph**) of the knight's puzzle just like our map of the underground. This involves two changes.

First of all, it doesn't actually matter how the board is laid out – we don't care that the

squares of the board are squares for example – they could be any shape and size. Let's draw each square as a spot instead, just as the tourist attractions were spots on the underground map.

Secondly, which squares are actually next to one another actually doesn't matter for the puzzle either. What matters is which ones you can jump between with a Knight's move. So let's draw lines between any two dots when you can use a single Knight's move to jump between them. That is just like the way the underground map shows which attractions can be jumped between using the underground.

To create the map move from square to square drawing lines and spots. Start with square 1 – draw a spot and label it 1. Now from square 1 you can move to square 9 so draw another spot and label it 9, drawing a line between them.

From square 9 you can only move back to 1 or on to square 3 so put a new spot marked 3 and draw a line to it from spot 9.

Keep doing this until you get back to a spot you have already drawn. Then go back to square 1 and follow another trail. For example from square 1 you can also move to square 7, so if you have a spot 7 already, then draw a line to it. If not, draw a new spot marked 7 and again draw a line to it, then continue the trail. Once you have followed all trails from spot 1. Move to spot 2 and follow all trails from it (adding spot 2 if it's not already there) in the same way. Then move on to trails from spot 3. Keep doing this until you have covered all trails from all spots.

**HINT:** there are only 2 moves possible from each of the inner three squares so their spots will each have 2 lines out of them in the finished map. There are 3 moves possible from all the other squares so their spots will have 3 lines out of them.

If the drawing is a bit messy you may want to redraw it neatly with no lines crossing. It can be done very neatly as two linked hexagons one inside the other (see solution graph).

### **Step 5: Solve the Knight's tour puzzle easily**

Once the graph is drawn try and solve the puzzle again. Start at spot 1 and follow the lines, noting the spots you pass through. It should be fairly easy to come up with a solution.

### **Step 6: Spotting the similarities**

If no one has pointed this out already get them to look at the similarities between the two graphs – the one for the Underground map and the one for the knight's tour. Up to the labels they are identical. If you have a solution to one you can use it as the solution to the other just by labeling the spots in a similar way (see diagram). You don't actually have to solve it anew. (Note as there are many solutions possible, the actual solutions they came up with might be different).

Perhaps surprisingly, the two apparently different problems are actually exactly the same problem with exactly the same solutions (once generalized). Once you have solved one, you have solved both.

### **Computational thinking**

The diagram that we drew is what a computer scientist calls a **graph**. A graph is a series of spots (we call those the **nodes** of the graph) and lines that join them (we call

those the **edges** of the graph). For the Knight's puzzle we have a node for each square of the board. We then add an edge for each possible knight move. It is this change of **representation** of the data that makes the puzzle seem easier.

To create the graph we had to do an '**abstraction**'. Abstraction is just the hiding of information, or to think of it another way you must change the **representation** of the puzzle – change the way that it is presented, change what the board looks like and how the moves are done – to make clearer the things that matter in the puzzle. The positions of the board and how you can move between them are the only things that matter, so that is the abstraction we use - we hide all other information like the shape, size and position of the 'squares' of the board.

We also saw two examples of **generalization**. We saw that we could generalize both problem statements and see that they are really the same kind of problem of finding a series of moves that visit every point exactly once and return to the start. A graph is a general representation – lots of apparently different problems can be represented by graphs. We also saw that in this case we could generalize the solutions – the solutions to both problems turn into exactly the same series of steps of the graph up to the words/numbers we have used to label the nodes. That of course isn't always the case, the graphs and solutions of two problems could be very different.

Another aspect of computational thinking that we have used here is **pattern matching** – when you see a problem or puzzle that involves moving from place to place, whatever the places are, consider representing it as a graph. Another way of saying that is if you can **match** a problem to the **pattern** of moving from place to place then use a graph to represent it. In this case we could go a step further. Once we see that the two graphs are the same (i.e., we have pattern matched the solutions) we realise that we already have the answer. All we have to do is transform the graphs to be the same thing by swapping the labels over and we can transform the answer of one into the answer of the other. We then just read off an answer to both problems from the general solution.

As an aside, the way we explored all the possible moves to draw the graph is a variation of what is called **depth first search**: we explored paths to their end, following the trail 1 – 9 – 3 – 11 to the end, before backing up and trying different paths. An alternative (**breadth first search**) would involve drawing all the edges from a node before moving on to a new node, i.e., drawing all the edges from node 1, then drawing all the edges from node 9, then all the edges from node 6. These are two different algorithms for exploring graphs exhaustively: two different **graph traversal algorithms**.

## Variations and Extensions

### Shorter version

For a shorter version, or with younger students develop the graph step by step yourself with the class rather than leaving them to create it.

### Links to other activities

The following activities are also available via [teachinglondoncomputing.org](http://teachinglondoncomputing.org)

### Tour Guide

*Devise a tour that gets a tourist from their hotel to all the city sights and back to their hotel.*

Show how the representation of a problem can alter how easy it is to do and how apparently different problems can be generalized to reveal that they are the same. Show how graphs can be used as a powerful representation.

### Spit-Not-So

*How good at you at this simple word game?*

This provides an introduction to how the organisation of data can make it easier for people to process and why graphical user interfaces (eg windows) are often easier to use than command-line interfaces (typing commands).

## Live demonstration of this activity

Teaching London Computing give live sessions for teachers demonstrating this and our other activities. See <http://teachinglondoncomputing.org/> for details. Videos of some activities are also available or in preparation.

## Use of this material



[Attribution NonCommercial ShareAlike](https://creativecommons.org/licenses/by-nc-sa/4.0/) - "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.



Department  
for Education

SUPPORTED BY

**MAYOR OF LONDON**

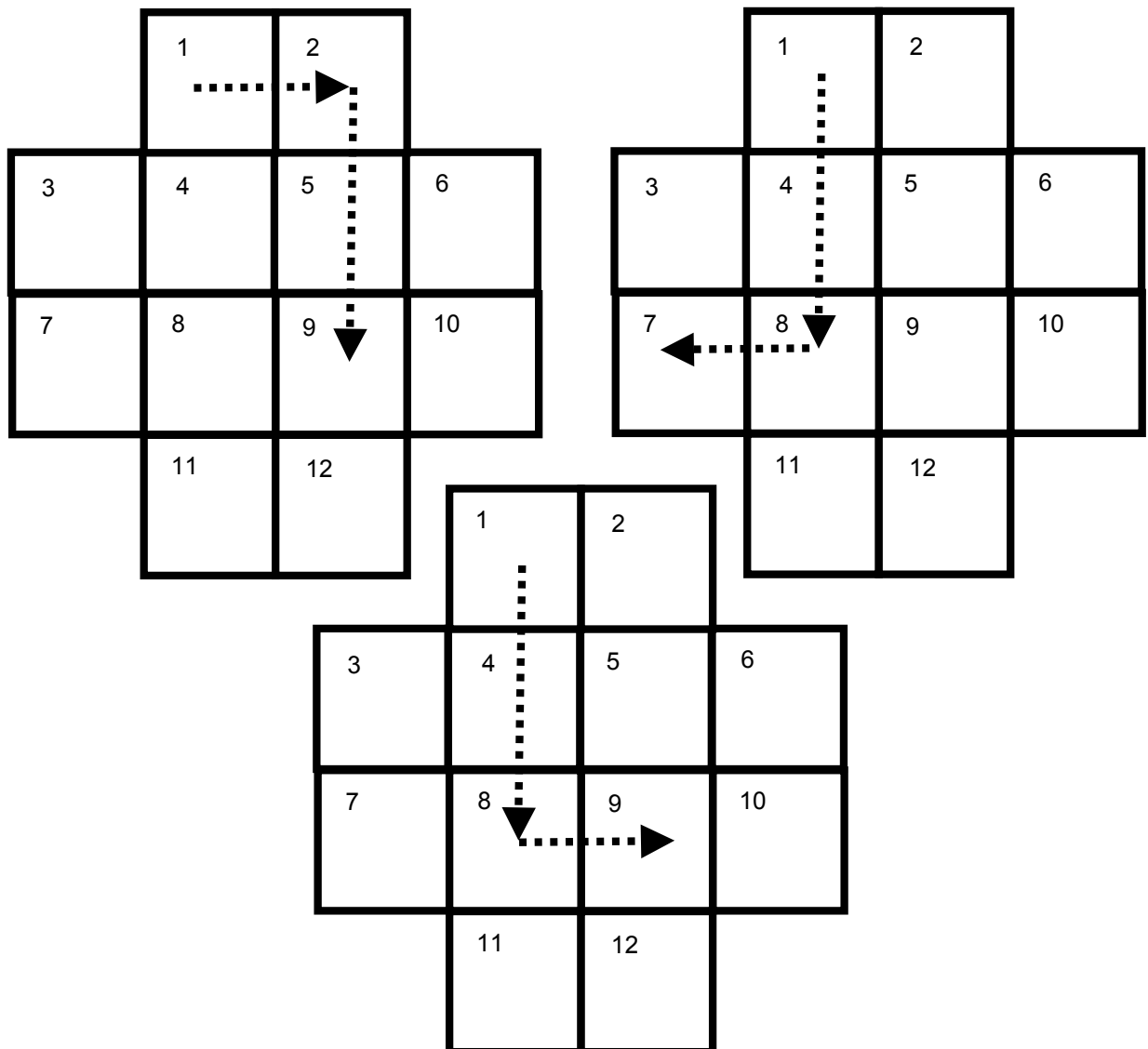
**COMPUTING AT SCHOOL**  
EDUCATE · ENGAGE · ENCOURAGE



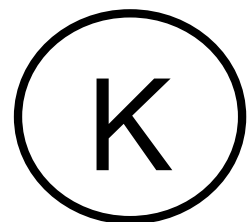
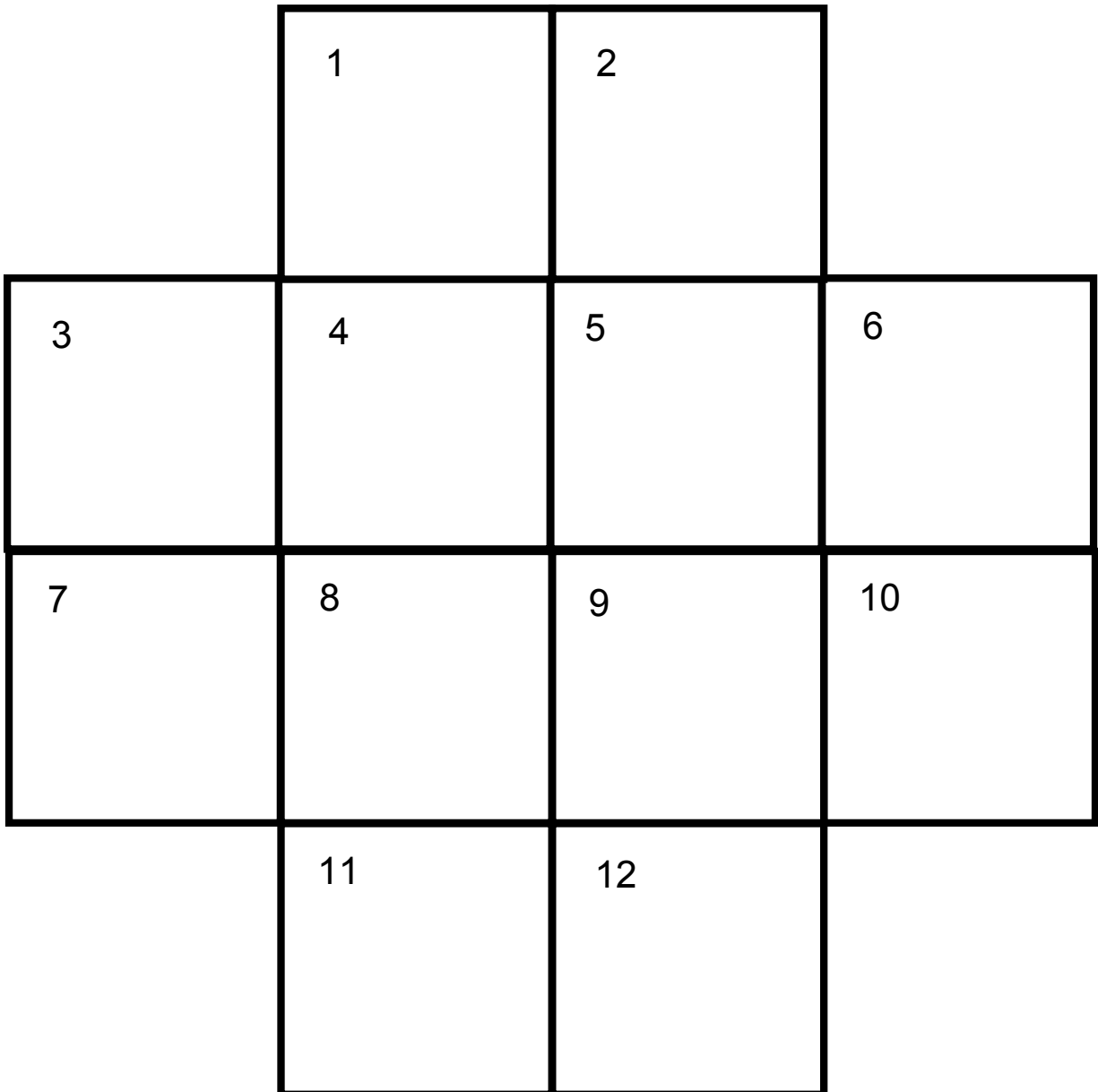
# The Knight's Tour Rules

Place the piece on square 1. By making only “knight” moves as in chess (see below), find a series of moves so that the knight visits every square exactly once, returning to where it started.

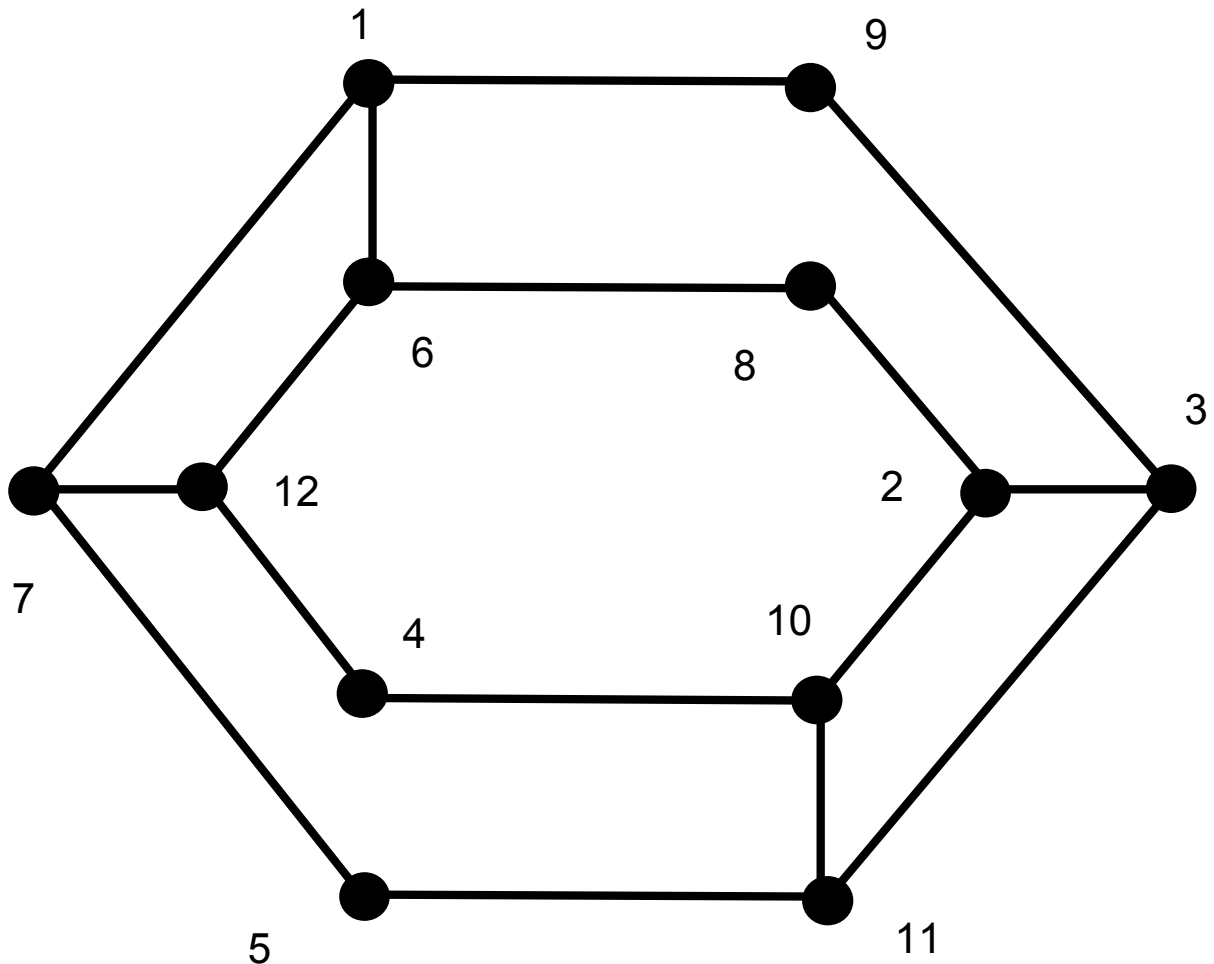
A knight moves in an L-shape: one square along and two squares up (in any direction. For example a piece on square 1 can move to square 6, 7 or 9.



# The Knight's Tour Board



# The Knight's Tour Board as a Graph



Each spot is a square on the original board.

Each line show a possible square a knight could jump to from a square.



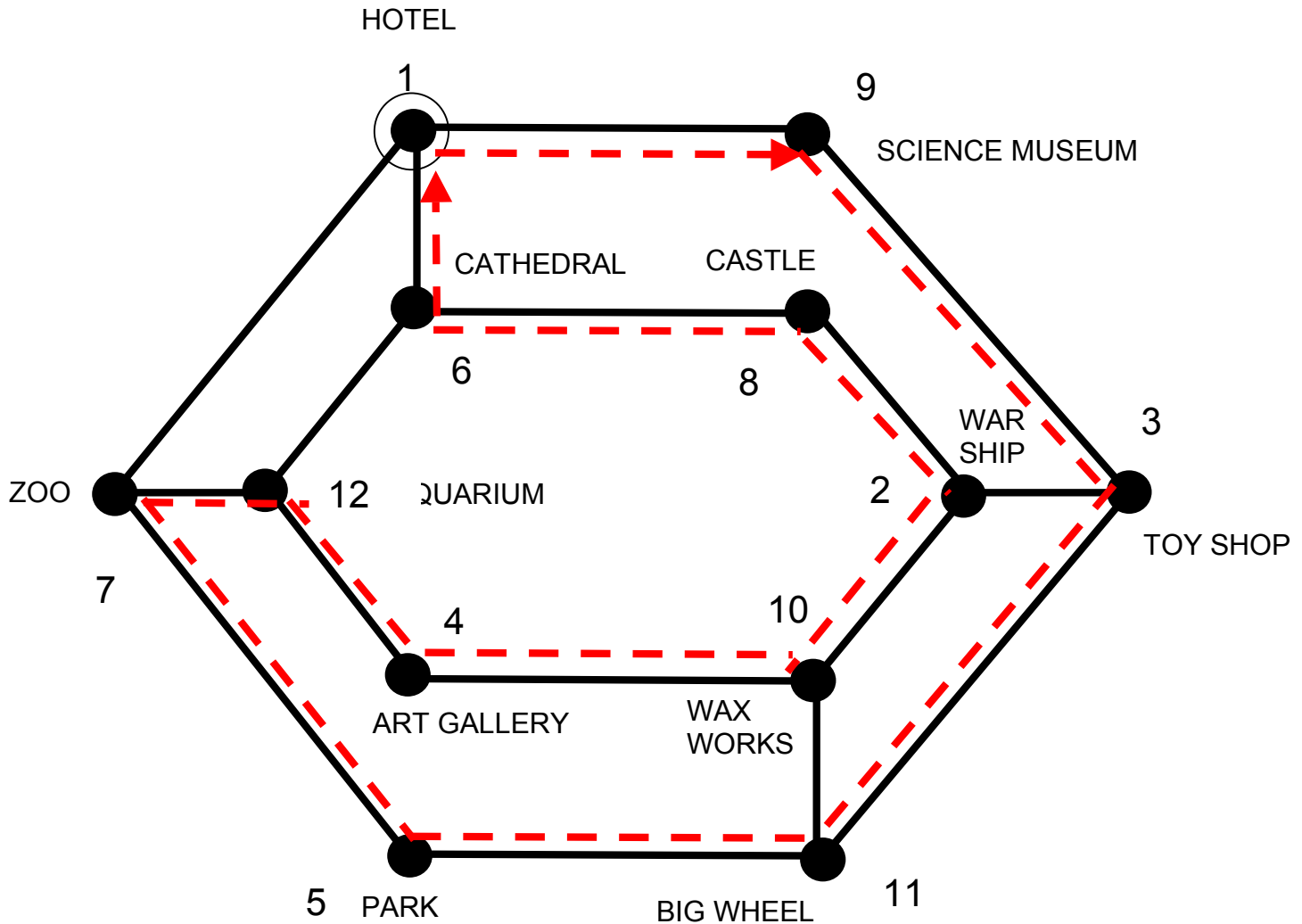
## Solution Sheet for Knight's Tour

Move	Jump from Square	To Square
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		

## One Solution to Knight's Tour (there are many)

Move	Jump from Square	To Square
1	1	9
2	9	3
3	3	11
4	11	5
5	5	7
6	7	12
7	12	4
8	4	10
9	10	2
10	2	8
11	8	6
12	6	1

# Identical problems, Identical solutions



The Graph of both the Tourist's Guided Tour and Knight's Tour shows that the two problems are identical and have the same solution