```
oobar:~/dont-get-volunteered cihan.goksu.88$ cat readme.txt
Don't Get Volunteered!
======================

As a henchman on Commander Lambda's space station, you're expected to
be resourceful, smart, and a quick thinker. It's not easy building a
doomsday device and ordering the bunnies around at the same time,
after all! In order to make sure that everyone is sufficiently quick-
witted, Commander Lambda has installed new flooring outside the
henchman dormitories. It looks like a chessboard, and every morning
and evening you have to solve a new movement puzzle in order to cross
the floor. That would be fine if you got to be the rook or the queen,
but instead, you have to be the knight. Worse, if you take too much
time solving the puzzle, you get "volunteered" as a test subject for
the LAMBCHOP doomsday device!

To help yourself get to and from your bunk every day, write a
function called solution(src, dest) which takes in two parameters:
the source square, on which you start, and the destination square,
which is where you need to land to solve the puzzle.  The function
should return an integer representing the smallest number of moves it
will take for you to travel from the source square to the destination
square using a chess knight's moves (that is, two squares in any
direction immediately followed by one square perpendicular to that
direction, or vice versa, in an "L" shape).  Both the source and
destination squares will be an integer between 0 and 63, inclusive,
and are numbered like the example chessboard below:

-------------------------
| 0| 1| 2| 3| 4| 5| 6| 7|
-------------------------
| 8| 9|10|11|12|13|14|15|
-------------------------
|16|17|18|19|20|21|22|23|
-------------------------
|24|25|26|27|28|29|30|31|
-------------------------
|32|33|34|35|36|37|38|39|
-------------------------
|40|41|42|43|44|45|46|47|
-------------------------
|48|49|50|51|52|53|54|55|
-------------------------
|56|57|58|59|60|61|62|63|
-------------------------
```

```
Languages
=========

To provide a Python solution, edit solution.py
To provide a Java solution, edit Solution.java

Test cases
==========
Your code should pass the following test cases.
Note that it may also be run against hidden test cases not shown
here.

-- Python cases --
Input:
solution.solution(0, 1)
Output:
    3

Input:
solution.solution(19, 36)
Output:
    1

-- Java cases --
Input:
Solution.solution(19, 36)
Output:
    1

Input:
Solution.solution(0, 1)
Output:
    3

Use verify [file] to test your solution and see how it does. When you
are finished editing your code, use submit [file] to submit your
answer. If your solution passes the test cases, it will be removed
from your home folder.

foobar:~/dont-get-volunteered cihan.goksu.88$ cat constraints.txt

Java
====
Your code will be compiled using standard Java 8. All tests will be
run by calling the solution() method inside the Solution class
```

```
Execution time is limited.

Wildcard imports and some specific classes are restricted (e.g.
java.lang.ClassLoader). You will receive an error when you verify
your solution if you have used a blacklisted class.

Third-party libraries, input/output operations, spawning threads or
processes and changes to the execution environment are not allowed.

Your solution must be under 32000 characters in length including new
lines and and other non-printing characters.

Python
======
Your code will run inside a Python 2.7.13 sandbox. All tests will be
run by calling the solution() function.

Standard libraries are supported except for bz2, crypt, fcntl, mmap,
pwd, pyexpat, select, signal, termios, thread, time, unicodedata,
zipimport, zlib.

Input/output operations are not allowed.

Your solution must be under 32000 characters in length including new
lines and and other non-printing characters.
```