Task 2
Model Answer – Video 1
Video Transcript

I will show you how you can use the UMPIRE method to go through the steps of solving this challenge.

First, we have 'U' - understand the problem.

We want to make sure we are solving the right problem. My understanding of the problem is that we need to determine whether or not a given n by n matrix is valid. In order to be valid, every row of the matrix in every column of the matrix must have exactly one of each integer from 1 to n.

If there are any duplicate integers in the same row or column, the matrix is invalid and if there is an integer outside the range from 1 to n, the matrix is also invalid. n is the length of the matrix, which is the same as the length of any row or column, and we are only dealing with integers from 1 to 100 inclusive.

If it is determined that the matrix is valid, we return true.

If it's determined that the matrix is invalid, we return false.

Now that we think we understand the problem, we should move to the next step in the UMPIRE method.

'M' - matching the problem with the interviewer.

Here, we just want to confirm with the interviewer that our understanding is correct and we are not trying to solve a problem that is different from the one they are actually asking.

Once we are confident that we know what the problem is asking, we will go to 'P' - plan your approach and solution.

My plan is to go through the matrix, checking to make sure the integers are inbounds. If they are, I will add them to a row set and a column set and then I will check to see if the sets are equal to n.

Sets cannot add duplicates, so if there is more than one of an integer, the set will be smaller than n, and I will return false.

If the sets are the correct size, I will return true.

Now it's time for step 'I' - implementing your solution.

I will be writing a method in Java to solve this problem.

First I'm creating a method called valid matrix, which takes in a 2d array. I'll call it matrix as a parameter and returns a Boolean.

True or false?

I'm setting n to n to be the length of the matrix. I'm creating a set of rows and a set for columns.

I will now make my way through the rows and columns of the matrix from i=0
to i is less than n and j is 0 to j is less than n.

For each integer in the matrix, I am checking to see if it is within bounds, if not just return false here.

Otherwise the integer is between 1 and n.

So add it to the row set and the column set.

Notice the row is ij and the columns is ji.

Now I can check if the row set and the column set are the right size, if not, return false.

Otherwise, return true.

And that's the solution.

I can test my solution against the examples given, and I find that it works.

Time for 'R' – review your solution.

In this problem we were given an n by n matrix. I store the length of this matrix as n to use later.

Then I go through the rows and columns of the matrix from 0 to n minus 1.

As I go through, I check each integer in the matrix is between 1 and n, if not I will immediately return false because that means that the matrix is invalid.

To make sure that each integer from 1 to n appears once in each row and column of the matrix, I'm using sets.

Sets do not allow duplicates, so I add each integer in a row or column to a set, the size of the set at the end will only be the same size as n if there are no duplicates.

Therefore, if the size of the row set is less than n or the size of the column set is less than n, I return false.

If both sets are equal to n, I return true.

Finally, we are at 'E' - evaluating your solution.

I feel pretty confident that my solution is correct.

In the future, I might want to do some more tests to make sure my solution works in all cases.