Followed steps to find the optimal requirement blocks.

Description:

I have implemented the code with a function and using two “for” loops one for to iterate the input blocks and another for to manipulate and fetch the data by using “if” conditions within the same loop.

Steps:

* Applied try and exception cases for the total code
* Used enumerate to find both index and index data within the same loop
* Approached with two use cases one for to find shortest path and another for optimal index (if requirement has more than one in s single block).
* If we find more than one requirement in a single block it returns the block index along with requirement keys.
* For each use cases returns different output.

Key details:

* **global blocks, choices** 🡪 to use the variables outside of the loop
* **if requirement in choices and block[1][requirement]** == 'true' 🡪 this condition helps to find both cases whether the loop element is present in the choices and the element value is set to true.
* **farthest\_keys = []** 🡪 this helps to fetch all the keys within a same block which is set to true and the same keys present in the choices with comma separated values.