Valid Sudoku

```
class Solution {
public:
    bool isValidSudoku(vector<vector<char>>& board) {
       int i;
       int j;
       int k;
       unordered_map<int,int> map;
       unordered_map<int,int> map1;
       unordered_map<int,int> map2;

       for (i = 0;i<board.size();i++){
        for (j = 0; j<board[i].size();j++){
          map[board[i][j]] += 1;
          map1[board[j][i]] += 1;
          if ((i == 0 || i ==3 || i == 6) && (j == 0 || j == 3 || j == 6))
{
            map2[board[i][j]] += 1;
            map2[board[i+1][j]] += 1;
            map2[board[i+2][j]] += 1;
            map2[board[i][j+1]] += 1;
            map2[board[i][j+2]] += 1;
            map2[board[i+1][j+1]] += 1;
            map2[board[i+1][j+2]] += 1;
            map2[board[i+2][j+1]] += 1;
            map2[board[i+2][j+2]] += 1;
            if (!validate(map2)) return false;
            map2.clear();
          }
        }
        if (!validate(map)) return false;
        map.clear();
        if (!validate(map1)) return false;
        map1.clear();
       }
```

```
       return true;
    }

    bool validate(unordered_map<int,int> map){

      for (auto i: map){
          if (i.second > 1 && i.first != 46 ) return false;
      }

      return true;
    }
};
```

- Working solution
  - checks rows
  - checks columns
  - checks 3x3 square
  - all in same nested loop to maximize efficiency and speed