```
class Solution {
    public static boolean[][] check;
    public static char[][] board;
    public void solveSudoku(char[][] b) {

        //Storing Empty values
        board=b;
        boolean flag=false;
        int row=0;
        int column=0;
        check=new boolean[9][9];
        for(int i=0;i<9;i++){

            for(int j=0;j<9;j++){
                if(board[i][j]=='.'){
                    if(flag==false){
                        row=i;
                        column=j;
                       flag=true;
                    }
                    check[i][j]=true;
                }


            }

        }
        // System.out.println(check[0][2]);

        // call all possible values on the first empty index
        ArrayList<Character> valuesOfFirst=new ArrayList<Character>(valuesPossible(row,
column));
        int i=0;
        // for(int i=0;i<valuesOfFirst.size();i++){
            while(valuesOfFirst.size()>0){
            board[row][column]=valuesOfFirst.get(0);
            //System.out.println("at position: board["+row+"]"+"["+column+"]"+" =
"+board[row][column] );
            //System.out.println("at position: board["+row+"]"+"["+column+"]"+" = "+
valuesOfFirst);
            if(solution((column==8)?row+1:row,(column+1)%9)){
               // System.out.println("yo"+board[0][2]);
                b= board;
                break;
            }
//             System.out.println("bktrk at position: board["+row+"]"+"["+column+"]"+" =
"+board[row][column]+" "+valuesOfFirst+board[1][1] );
```

```
//            for(int o=0;o<9;o++){
//                for(int j=0;j<9;j++){
//                    System.out.print(board[o][j]);
//                }
//               // System.out.println();
//            }
            board[row][column]='.';
            valuesOfFirst.remove(0);



        }







        //return board;



}









    public ArrayList<Character> valuesPossible(int row,int column){

        //generates all possible values for a location on board
        //puts that in an arraylist and returns arraylist
        ArrayList<Character> arr1=new ArrayList<>();
        HashSet<Character> arr=new HashSet<>();
        arr.add('1');
        arr.add('2');
        arr.add('3');
        arr.add('4');
        arr.add('5');
        arr.add('6');
        arr.add('7');
        arr.add('8');
        arr.add('9');
        //remove all values already present in he row from the hashset
```

```
for(int i=0;i<9;i++){
    if(arr.isEmpty()){

        return arr1;
        //break;
    }
    arr.remove(board[row][i]);
}
//removes all the values present in the column
for(int j=0;j<9;j++){
    if(arr.isEmpty()){
        return arr1;
       // break;
    }
    arr.remove(board[j][column]);

}

//remove all the values present in the block
//identify the block
int tempRow=row+1;
int tempCol=column+1;
while(!(tempRow%3==0)){
    tempRow+=1;
}
tempRow=3*tempRow/3;
while(!(tempCol%3==0)){
    tempCol+=1;
}
tempCol=3*tempCol/3;

int rowRange=tempRow-3;
int colRange=tempCol-3;

for(int k=0;k<3;k++){
    for(int l=0;l<3;l++){
        if(arr.isEmpty()){
            return arr1;
            // break;
        }
        if(arr.contains(board[k+rowRange][l+colRange])){
            arr.remove(board[k+rowRange][l+colRange]);
        }

    }
}
```

```
        for (Character ch : arr) {

            arr1.add(ch);


        }




        return arr1;
    }




    public boolean solution(int row, int column){
         if(row>=9){
            return true;
            }
        if(check[row][column]==false){

            if(column==8){
                return solution(row+1,(column+1)%9);
            }
            else{
                return solution(row,(column+1)%9);
            }

        }
        else{


            ArrayList<Character> arr=new ArrayList<Character>(valuesPossible(row, column));



            while(arr.size()>0){

                board[row][column]=arr.get(0);
               // System.out.println("at position: board["+row+"]"+"["+column+"]"+" =
"+board[row][column] );
                //System.out.println("at position: board["+row+"]"+"["+column+"]"+" = "+arr);
                if(solution((column==8)?row+1:row,(column+1)%9)){
```

```
                return true;
             }
            else{
               //.out.println("at position: board["+row+"]"+"["+column+"]"+" =
"+board[row][column] );
             //System.out.println("at position: board["+row+"]"+"["+column+"]"+" = "+arr);
               board[row][column]='.';
               arr.remove(0);
              // System.out.println("at position: board["+row+"]"+"["+column+"]"+" =
"+board[row][column] );
            // System.out.println("at position: board["+row+"]"+"["+column+"]"+" = "+arr);

               if(arr.isEmpty()){
                  // System.out.println("at position: board["+row+"]"+"["+column+"]"+"arraylist is
empty"+arr);
                   return false;
               }



           }



        }

   }
      return false;



}
}
```