Name: Thompson Pham

Check Columns to see if they contain 1-9 w/o a duplicate
Parameters - char array(Sudoku Input)
Return - string (character representing which column is solved, valid, or invalid)

```
Set variables to count each time a certain number is used from 1 - 9(Ex: int num1 = 0)
Set string colD = “”
For(i = 1 < 10){
        For(j = 1 < 10){
                Access pointer
                Check the character at pointer
                If character is 1, add 1 to counter for its variable
                If character is 2, add 1 to counter for its variable
                If character is 3, add 1 to counter for its variable
                If character is 4, add 1 to counter for its variable
                If character is 5, add 1 to counter for its variable
                If character is 6, add 1 to counter for its variable
                If character is 7, add 1 to counter for its variable
                If character is 8, add 1 to counter for its variable
                If character is 9, add 1 to counter for its variable
                Move pointer forward 10 (next number in column)
                If counter for number 1 is above 1, cat colD with 1 then break
                Else if counter for number 2 is above 1, cat colD with 2 to string then break
                Else if counter for number 3 is above 1, cat colD with 3 to string then break
                Else if counter for number 4 is above 1, cat colD with 4 to string then break
                Else if counter for number 5 is above 1, cat colD with 5 to string then break
                Else if counter for number 6 is above 1, cat colD with 6 to string then break
                Else if counter for number 7 is above 1, cat colD with 7 to string then break
                Else if counter for number 8 is above 1, cat colD with 8 to string then break
                Else if counter for number 9 is above 1, cat colD with 9 to string then break
                Else if counter for space is above 1, cat colD with v to string then break
                Else cat colD with “s”
                }
        Move pointer backwards 90 (to beginning)
        Move pointer forward 1 (So it will go through next column)
        Reset counters
}
        Return string colD
```

```
/* Comments: The position the character is on the string represents the column, while the
character represents if its valid, solved, or has multiples of a certain number from 1-9
(Ex: ssss5ss7v
Column 1, 2, 3, 4, 6, 7 are solved
Column 5 is invalid with multiples of 5
Column 8 is invalid with multiples of 7
Column 9 is valid
)*/
```

Check Rows to see if they contain 1-9 w/o a duplicate
Parameters - char array(Sudoku Input)
Return - int (character representing which row is solved, valid, or invalid)

```
Set variables to count each time a certain number is used from 1 - 9(Ex: int num1 = 0)
Set string rowD  = “”
For(i = 1 < 10){
    For(j = 1 < 10){
        Access pointer
        Check the integer at pointer
        If character is 1, add 1 to counter for its variable
        If character is 2, add 1 to counter for its variable
        If character is 3, add 1 to counter for its variable
        If character is 4, add 1 to counter for its variable
        If character is 5, add 1 to counter for its variable
        If character is 6, add 1 to counter for its variable
        If character is 7, add 1 to counter for its variable
        If character is 8, add 1 to counter for its variable
        If character is 9, add 1 to counter for its variable
        Move pointer forward 1 (next number in row)
        If counter for number 1 is above 1, cat rowD with 1 to string then break
        Else if counter for number 2 is above 1, cat rowD with 2 to string then break
        Else if counter for number 3 is above 1, cat rowD with 3 to string then break
        Else if counter for number 4 is above 1, cat rowD with 4 to string then break
        Else if counter for number 5 is above 1, cat rowD with 5 to string then break
        Else if counter for number 6 is above 1, cat rowD with 6 to string then break
        Else if counter for number 7 is above 1, cat rowD with 7 to string then break
        Else if counter for number 8 is above 1, cat rowD with 8 to string then break
        Else if counter for number 9 is above 1, cat rowD with 9 to string then break
        Else if counter for space is above 1, cat rowD with v to string then break
        Else cat rowD with “s”
        }
    Move pointer forward 1 (So it will go through next row)
    Reset counters
}
    Return string rowD
```

Check 3 x 3 grid if they contain 1-9 w/o a duplicate
Parameters - char array(Sudoku Input)
Return - int character representing which grid is solved, valid, or invalid)

```
Set variables to count each time a certain character is used from 1 - 9(Ex: int num1 = 0)
Set string check = “”
For(i = 1 < 10){
    For(j = 1 < 4){
        For(k = 1 < 4)){
            Access pointer
            Check the integer at pointer
            If character is 1, add 1 to counter for its variable
            If character is 2, add 1 to counter for its variable
            If character is 3, add 1 to counter for its variable
            If character is 4, add 1 to counter for its variable
            If character is 5, add 1 to counter for its variable
            If character is 6, add 1 to counter for its variable
            If character is 7, add 1 to counter for its variable
            If character is 8, add 1 to counter for its variable
            If character is 9, add 1 to counter for its variable
            If character is space, add 1 to counter for its variable
            Move pointer forward 1 (next number in grid)
            If counter for number 1 is above 1, cat check with 1 to string then break
            Else if counter for number 2 is above 1, cat check with 2 to string then break
            Else if counter for number 3 is above 1, cat check with 3 to string then break
            Else if counter for number 4 is above 1, cat check with 4 to string then break
            Else if counter for number 5 is above 1, cat check with 5 to string then break
            Else if counter for number 6 is above 1, cat check with 6 to string then break
            Else if counter for number 7 is above 1, cat check with 7 to string then break
            Else if counter for number 8 is above 1, cat check with 8 to string then break
            Else if counter for number 9 is above 1, cat check with 9 to string then break
            Else if counter for space is above 1, cat check with v to string then break
            Else cat check with “s”
            }
    Move pointer forward 7 (So it will go through next row in grid)
    Reset counters
    }
    Move pointer backwards 20(So it will look at next grid)
}
    Return string rowD
```

Main

```
Open file in binary
If file opens properly
Create char array
Do{
    Getline, read nine lines
    Use pointer to input the characters in the char array
    Call check columns
        Check string
        If string contains “s” tell user solved
        If string contains “s” and “v” or just “v” tell user user puzzle is valid
        If string contains numbers, tell user which column is invalid with which duplicate number it has
        Check string
        If string contains “s” tell user solved
        If string contains “s” and “v” or just “v” tell user user puzzle is valid
        If string contains numbers, tell user which row is invalid with which duplicate number it has
    Call Check 3x3
        Check string
        If string contains “s” tell user solved
        If string contains “s” and “v” or just “v” tell user user puzzle is valid
        If string contains numbers, tell user which grid is invalid with which duplicate
```

```
                number it has
        }While if not EOF(
                Check first character of line (peek)
                If character = newline)

Close file
```

Test Cases

- Check row 1 with duplicate number
- Check row 9 with duplicate number
- Check row 1 with the entire row containing spaces
- Check column 1 with duplicate number
- Check column 9 with duplicate number
- Check column 1 with the entire column containing spaces
- Check grid 1 with duplicate number
- Check grid 9 with duplicate number
- Check grid 1 with no duplicate and some spaces if it outputs valid
- Check grid 1 with no duplicates and no spaces if it would output solved