First of all I created grid with .space 258 because last two char of an string is \0 and I am trying to create 16x16 grid. Also created bombIndex with .word because I am going to hold integers in that array. Then create single char bomb because I have to hold ’O’ char to change grid indexes.

At the main part firstly I printed "please enter map: " and then read input from user. This input must be 256 character string. which means I take whole grid at the same time. End of this report I will attach 256 character input examples so you can use it easily.

Then I filled registers which I will use at the finding bomb indexes part. I stored addresses of bombIndex array and grid so that I can manipulate it. in the "Loop:" subroute I created a basic loop for moving in bombIndex array. Loop ends when we reached end of the grid. in loop I continue to moving until ’O’ chars which represents bombs. I compare current grid index with ’O’ char and if they are same I store index of a current grid into bombIndex array. Also I increase counter which holds by $s3 to use it at the next parts of a code. When we reache end of the array we jump to "LoopExit:"

At the second part I have to fill every cell of a grid with bombs according to rules. So I created basic loop which storebytes on the current index of grid 256 times. End of this loop I added "jal" to print current grid.

At the third part I manipulate grid indexes according to bombIndex values.In "Looop:" part I increase grid index until grid index becomes equal with current index of a "bombIndex" array. If they are equal we jump to the "make_bomb:" part and starting to manipulate grid according to bombs. At this part I used simple math with complex jump functions. Firstly we make current index of an grid ’.’ because in every case current index must be ’.’. Then I increase grid index counter for checking if we are inside the boundries. İf we are inside the boundries right side of a bomb becomes '.'. But if we are out range "dontAddOne:" activates and we "jal" to "dontAddOne_reverse:" subroute to decrease grid index counter one. Because for the comparison we increased it at the first part. But if we are in the range I dont touch the grid index counter becaus e we moved 1 char indeed. With same logic I implemented subtracting by one which means left side of a bomb , adding by sixteen which means down side of a bomb and subtracting with sixteen for upside of a bomb. If we out range we apply same procedures. We repeat this loop until bombIndex counter becomes equal with bomb number which we stored in the $s3 register at previous part. When loop finished we "jal" to print_grid function which prints current grid (third second).

Last part is print_grid function. In this function we stores address of grid and thanks to nested loop we prints grid char by char. when we reached end of the row which means 16. char of a grid, we print newline for seeing string as a 16x16 table. End of the loops we print another newline for better readibility. End of the program there is Exit: function which ends program.

INPUT EXAMPLES:

.................................O...O...O....O..O...........O..O...O...O....O..O...O...O..........O...O....O..O...O...O....O.
.O...O...O....O..O...O...O....O..O...O...O....O..O...O...O....O..O...O...O....O..O...O...O....O..O...O...O....O..O..
.O...O..

....
.......................O..OO........O

OO..O..O.....O..OO..................O.......O....O..O...........O..O...O...O....O..O...O...O..........O...O....O..O...O...
O....O..O...O...O....O..O...O...O....O..O...O...O....O..O...O...O.......O............O..O...O...O....O..O...O...O....O..
OOO.O.O...O