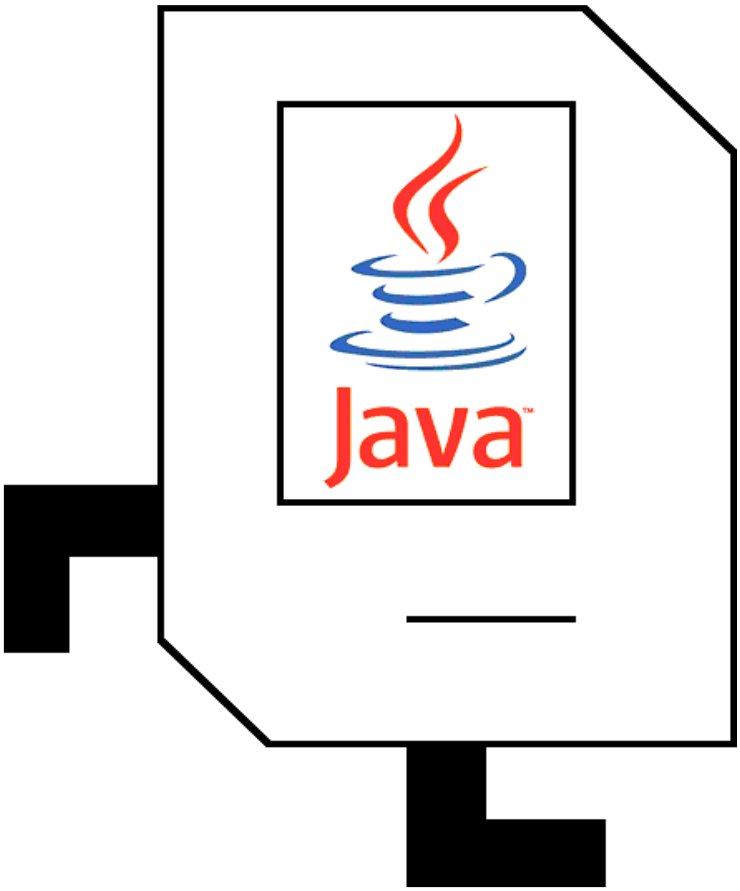



# Report

## Assignment (Karel the Robot)

**Name:** Zain Alabdeen Adelelah Aburayya

# • Introduction

In this assignment of (Karel the robot), I have reached satisfactory results for me, and I expect that I have come up with many simple and somewhat complex solutions, and most of the solutions are in the best cases and using the least possible number of steps and using the least possible number of devices Whistles to divide the map into the largest equal number of chambers , I will explain each method how I did it and explain all cases.

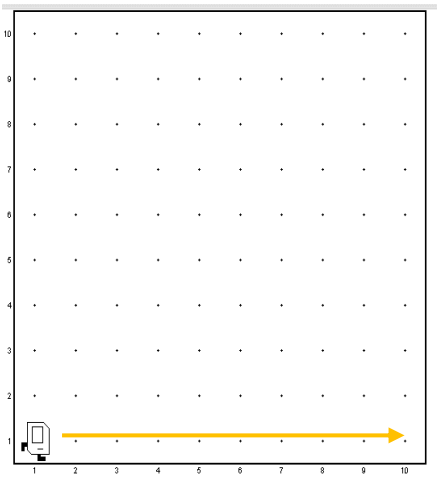
# • Explanation before start

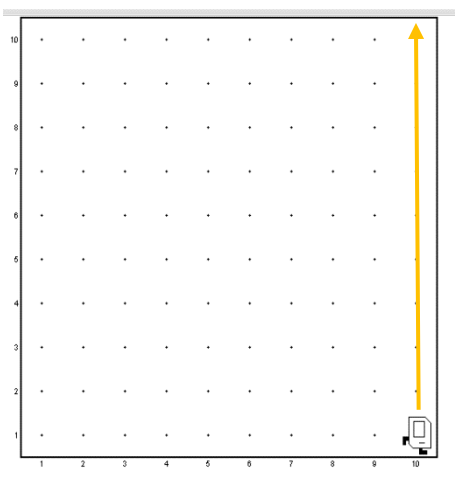
In the beginning, to divide any map into four equal parts that can be divided into many parts, but according to what is required in this assignment, which is to find the largest number of chambers divided, whether it is 4 chambers, 3, 2, or 1, to have an accurate answer, I will say that I have to experiment All possible cases I have and when the output will be four chambers and so on, so I will divide each chamber into several conditions that I will explain in detail.

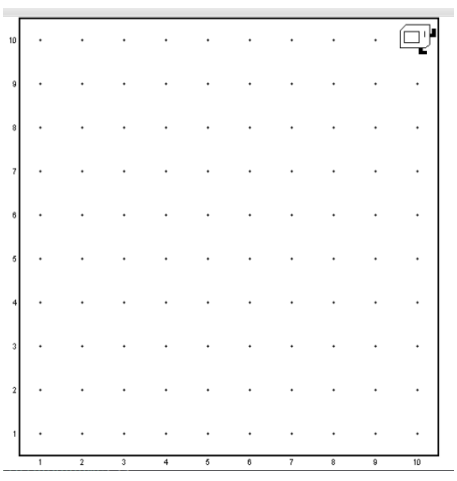
First, before it is calculated whether it is divided into the largest possible number, I had to calculate the number of rows and columns, because the solutions depend on this, so my method was as follows:

- I made Karel walk to the far right and then made her go to the highest point, which allowed me to calculate the length and width without using the existing methods. I defined the length and width variables as global variables, so I could use them anywhere I needed them, that cost me [ ( width + height ) – 2 ]  step's.

I used the `frontIsClear()` function to help me bypass any error or problem in the program.



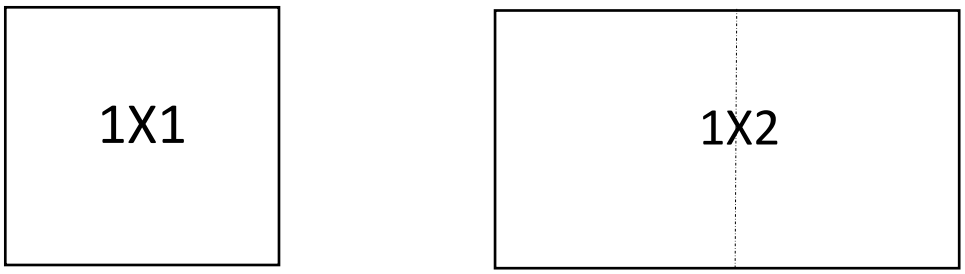




- # Divided to one chamber

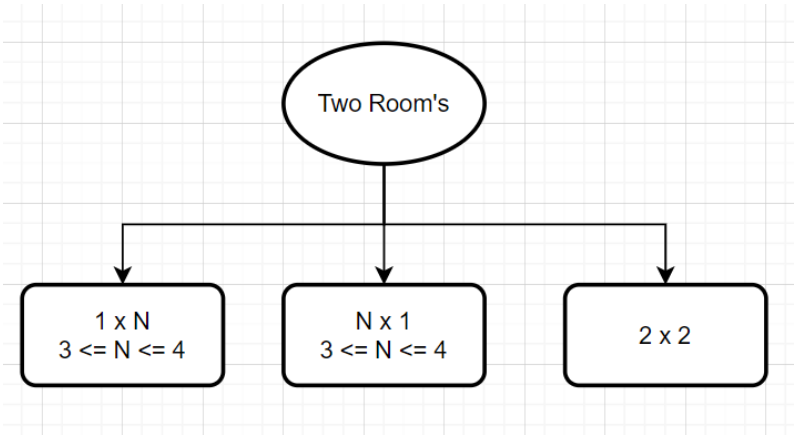
Any map can be divided into one part, but in this assignment, I have to avoid as much as possible that it is divided into one chamber, to divide any map into one equal part, I only have two cases to be divided into one part ,If the map contains one row and one column, then this will certainly not be divided into four, three, or two parts, so it will be one chamber, and there is the other case that if the row is equal to one and the column is equal to two or vice versa, these are the only cases to divide it into one chamber, either If it is 1x3 or 2x2, it is better to divide it into two. I will explain these cases in the next part.

But there is something I must mention if I know that it will only be divided into one chamber, I will not put any beepers, because as it was said in the assignment, I must use the least number of beepers and steps, so far, we will not touch on calculating the steps until now.




- # Divided to Two chambers

Now we will come to the case that the map is divided into two equal parts, and there are a few cases that will allow us to divide it into two equal parts, the following cases:



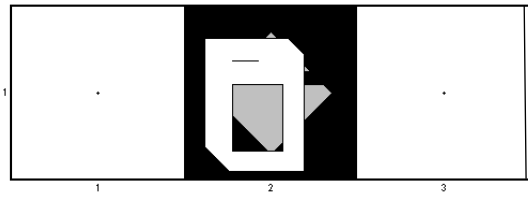
- # Divided to Two chambers (Cont.'s)

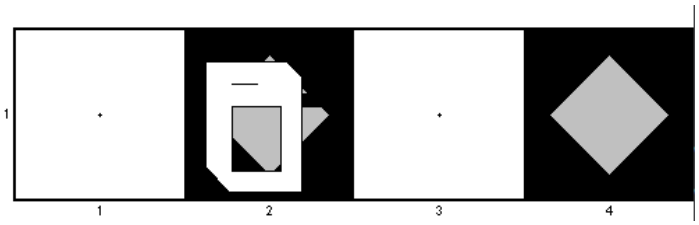
As we can see, those three cases are the only ones in which we can divide the map into two equal parts, but as I said before *(I need the largest number of equal sections)*, in the past cases it is only the special cases of the two chambers.

➔ In the following two cases ( 1 x N and N x 1 , 3 <= N <= 4 ) :

- In this case, I wrote a simple code and not the algorithm for any case, it was as follows that if it found that **N** is equal to **four**, it puts two beepers, in order to achieve the condition of reducing the steps, I had to put in the last place where I stood and then walk two steps and put the other beeper It costs me only two beepers and five steps, that's the minimum number of steps and beepers in this case.

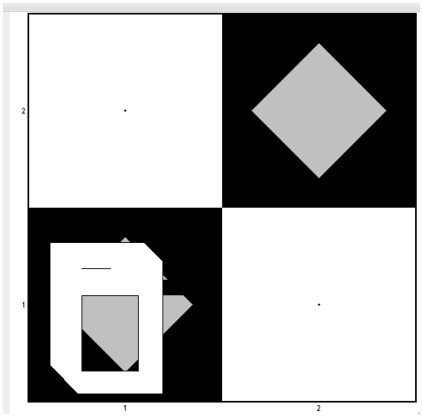
- If **N** is equal to **three**, then he will put only one beeper in the middle, and it is also a very simple code, only he must walk a step from where he is standing and put a beeper, in this case it costs me one beeper and three steps, that's the minimum number of steps and beepers in this case.



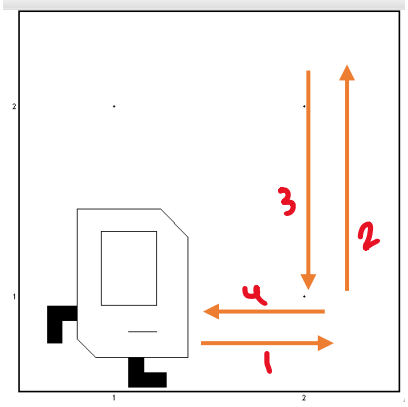


➔ In the following case ( 2 x 2 ) :

- In this case, I had to write code for Karel to walk a zigzag, which means he puts a beeper in the last place he stands and then walks to the square in his corner and puts in another beeper, it cost me two whistles and four steps, that's the minimum number of steps and beepers in this case.

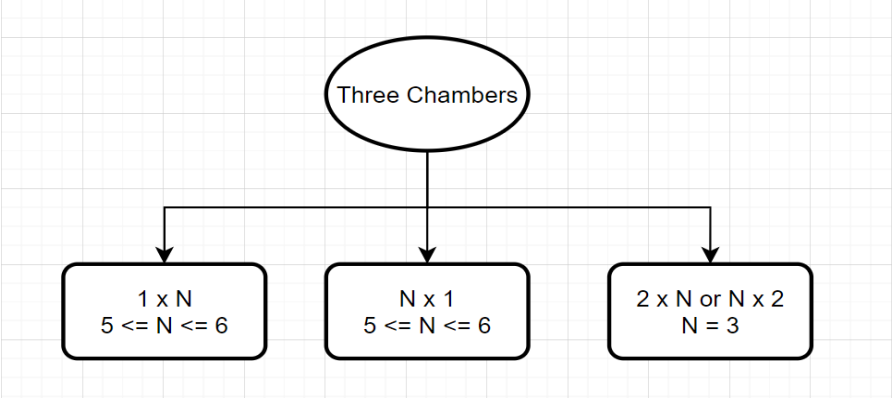


How did karel move?



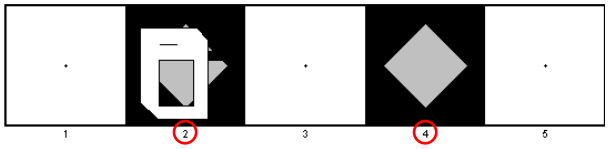
# • Divided to Three chambers

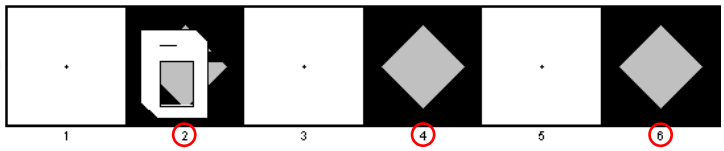
As we have seen in the previous cases, the cases of dividing chambers into one or two were completely covered. Now, at the time of dividing the map into three chambers, the chambers were divided in similar cases to divide them into two, with a difference of some N value , To divide the map into three equal parts, I have only three cases, (1 x **N** | **N** x 1) which are that **N** is equal to 5 or 6, or that it is ( 2 x **N** | **N** x 2) which are that **N** is equal to 3 .




➔ In the following two cases ( 1 x N and N x 1 , 5 <= N <= 6 ) :

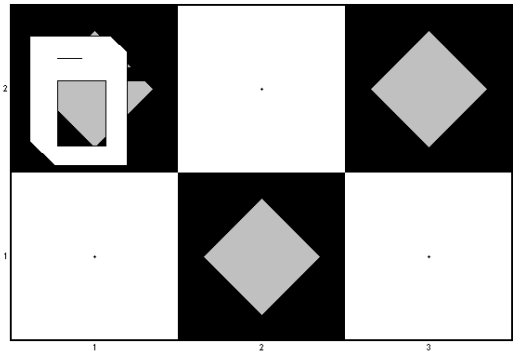
- In this case, I wrote a simple code to solve the problem and put the beepers in their place and the least number of steps. I could have written it as a general algorithm, but I did not find any other cases that could be related to the same rule, so I wrote a code that must put the beeper in the squares that have an even index, As shown in the pictures , In the case of N equal to 5, it cost me seven steps and two beepers, but in the case of N equal to 6, it cost me nine steps and three beepers that's the minimum number of steps and beepers in this case.



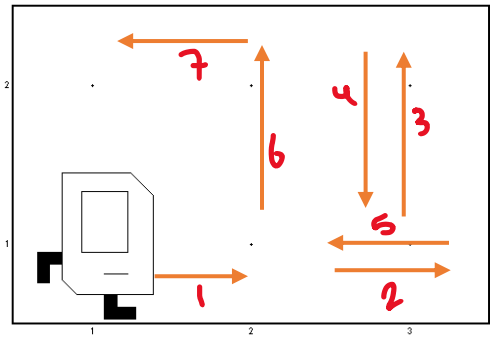


➔ In the following case ( 2 x N and N x 2 , N = 3 ) :

- In this case, I had to use the function that I used previously in the division into two chambers, and this means that I used a unified, fixed method, which was called zigzag, and it cost me seven steps and three beepers to divide it into three chambers, that's the minimum number of steps and beepers in this case.

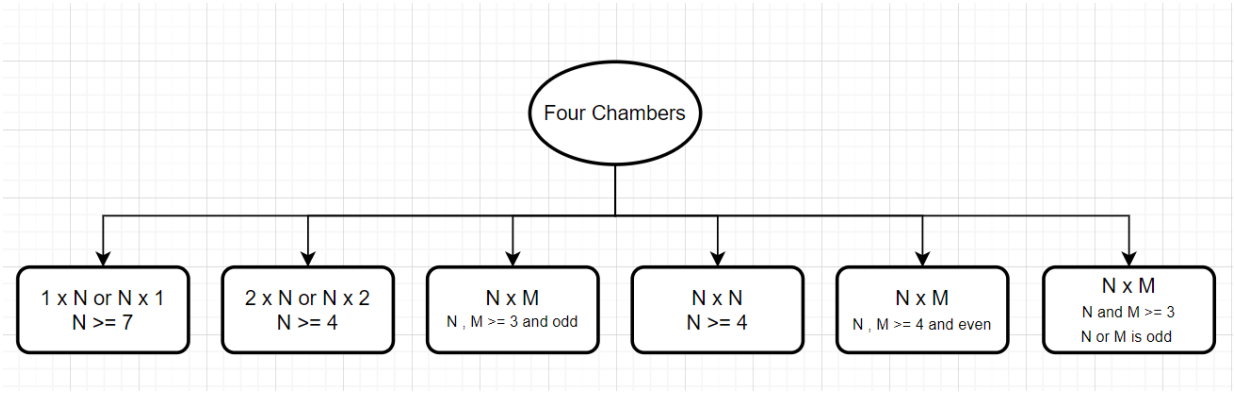


How did karel move?



# • Divided to Four chambers

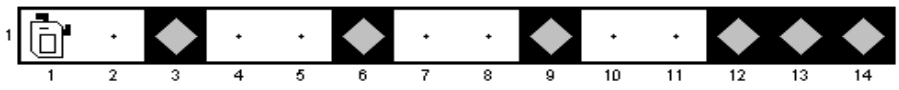
This is the most comprehensive case, which has many cases, and that required a lot of thinking from me to reach the best suitable solution for dividing the chambers, in some cases I did not reach a more appropriate solution, and in others I found a suitable solution that matches the conditions required in the duty, so after counting the cases I divided them into several cases, namely As shown in the picture:




➔ In the following two cases ( 1 x N and N x 1 , N >= 7 ) :

- In this case, my idea was that if I have the row or column equal to one and the other is greater than or equal to seven, then surely the map will be divided into four equal chambers. You will have a question that what will be the division of chambers if each of the row equals one and the column equals 14 For example, in this case, I will calculate how much empty space I can put in each chamber. First, I will calculate whether there are remaining spaces that cannot be distributed among the rest of the chambers. It is calculated by the following formula: ***div = (a+1) / 4***, here I am I will insert it in the loop statement until I check whether the div multiplied by four is less than the number of columns. If it is correct, it will put a beeper in its place and advance. It will remain so until it exits the loop statement, after that the number of empty spaces is skipped and the empty spaces are calculated using the following equation: ***freq = div - 1***, it will remain so until it ends up in a closed wall.

- Example: ***1 x 14***



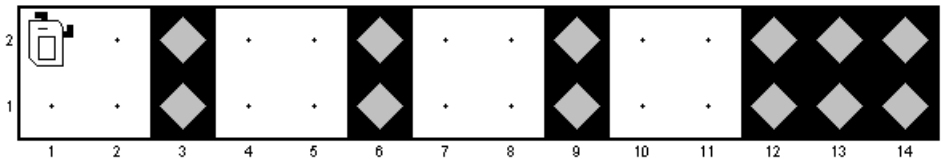

✓ In this example, we had a = 14, and according to the first equation, the result of the div will be equal to (14 + 1)/4, which equals 3.75, but according to the variable integer it will be 3, after that, you will enter the loop statement and check if 3 * 4 <= 14? The answer will be correct, so he will put a beeperand take a step forward and reduce the number of columns and remain like this until the case ends, after which he will enter another loop sentence, which is to skip the empty spaces freq = 3 - 1, which means that each chamber will contain two squares.

# • Divided to Four chambers (Cont.'s)

➔ In the following two cases ( 2 x N and N x 2 , N >= 4 ) :

- It is the same as the previous case, but the only difference is that it will fill every two lines or two columns together. I used some scripts written only to reduce the lines in the code, but the logic is the same as the previous case, it cost me 33 steps and 12 beepers.

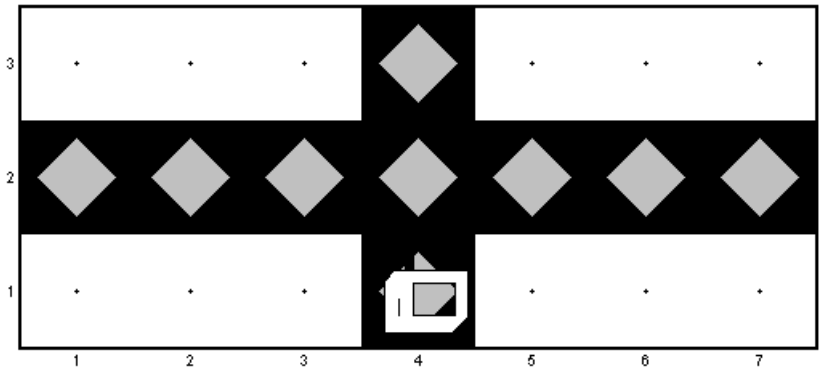
- Example: ***2 x 14***




➔ In the following two cases ( N x M and M x N , N, M >= 3 , ODDS ) :

- In case the column and row are greater than or equal to three, we will have the simplest case I encountered, which is to just divide the map for half of the column and half of the row, or in other words I put all the beeps on the longitude of column/2 + 1 and also the same along the line row/2 + 1 .

- Example: ***3 x 7***



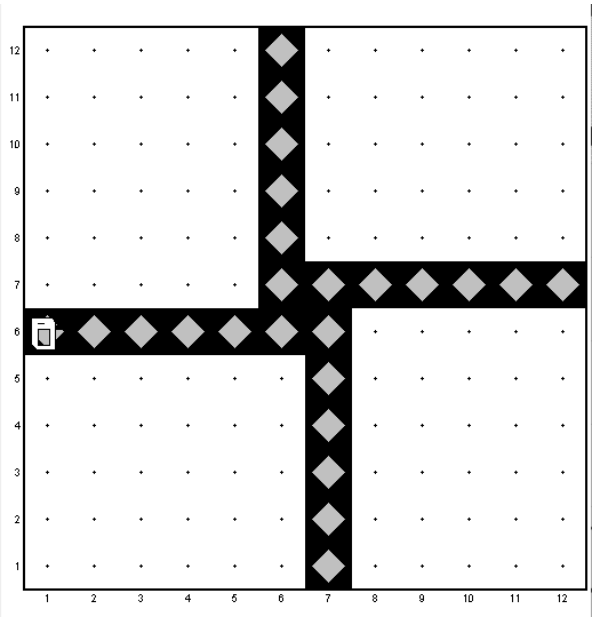
# • Divided to Four chambers (Cont.’s)

➔ In the following case ( N x N , N >= 4 , EVENS) :

- If the number of rows and columns is equal, this allows me to move called the fan. It can only be used if the rows and columns are equal and even. The way it works is as follows:

  It puts beeps along ***column/2*** and extends to ***row/2***, and then Karel goes to the opposite corner and fills the remaining column and goes back and does the same for the rows. I came up with the idea of this solution after trying many cases, but I tried to apply it to all even cases, but it did not work There is another way.

- Example: ***12 x 12***




- As shown in the previous example, first walk to ***column /2*** and start from there with the position of the beeps and extend down by ***row/2***, skip to the bottom right corner and repeat the process until it reaches the wall and then returns to the middle and will do as it did in the column the first time and so on, It cost me 63 plans and 24 beepers, I expect that's the minimum number of steps and beepers possible if I'm not sure.

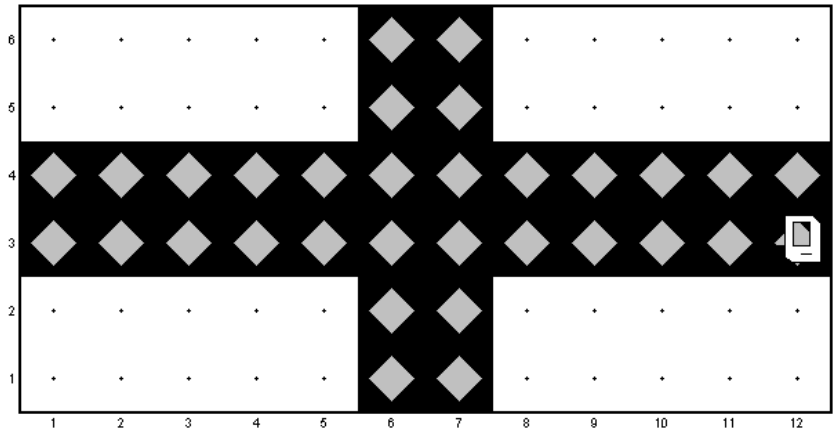
# • Divided to Four chambers (Cont.'s)

➔ In the following two cases ( N x M and M x N , N, M >= 4 , EVENS ) :

- I can say that this case is one of the longest steps written in the code, that is, it depends on drawing two lines for each of the row and column if the two are even numbers and the two are greater than or equal to four, so I had to write a long code to draw these lines, the underlying reason Behind that I will draw two lines instead of a line, as we did in the case of the odd N and M, because in the even numbers, if I ask you for an average number, you will not be able to answer, because it actually consists of two numbers or two columns, for example that you had 9 when you divide it into two in the table, you will see The column on 5 is the one that divides the middle, but in the even numbers, the one that cuts the middle is ***a/2*** and ***a/2 + 1***.

- How does Karel walk on it? Karel walks starting with the smaller number of rows or columns, of course, but he will not fill row by row, he will fill them and walk like a circle, he will only pass through all the middle once without going back and forth with the same steps, I have done everything I can to combine the two cases together, whether the number of rows is less or the number of columns is less.

- Example: ***6 x 12***



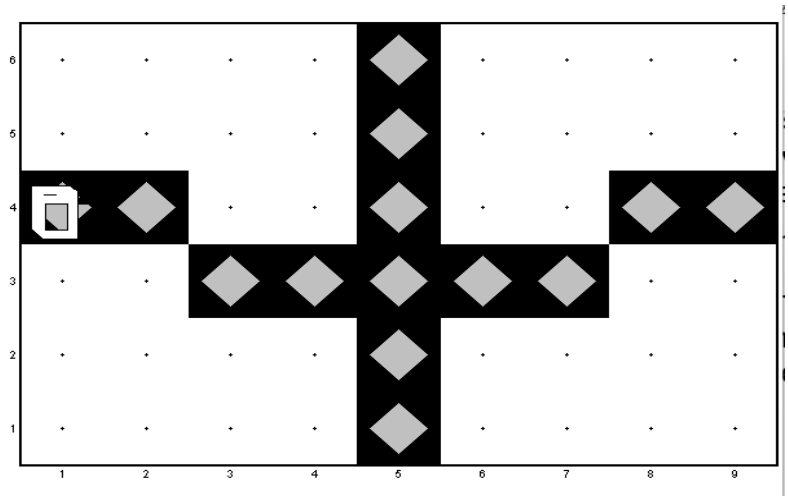

- Especially, in this case, I am sure that there is a better solution, but I could not get to it correctly or with errors in equations, etc. It is one of the most used cases for steps and whistles.

- # Divided to Four chambers (Cont.'s)

➔ In the following two cases ( N x M  and  M x N ,  N, M >= 3 , ODD and EVEN ) :

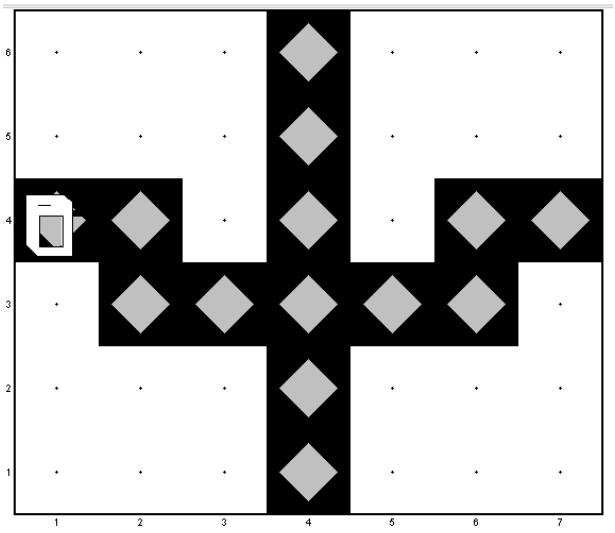
- This is one of the cases that I tried as much as possible to optimize it. I had a preliminary solution, which was to go towards the odd number, the row or column, and start filling the beepers from ***N/2***, where ***N is the odd*** number, and complete the rest of the lines as I did in the case of the even rows and columns, so I started Thinking about how I can reach a suitable and optimize solution so that the number of steps is as few as possible and the number of beepers is as few as possible, so I started experimenting on many different cases, which I have a general perspective on the thing, which is the following:

- When I divide the map in half on the odd side, I have two equal parts, so when I tried many cases, I noticed that I might have a somewhat fork-like shape or a kind of zigzag, but there were two impediments. The first is when the curve will be completed, and how much it will walk until it reaches the curve, so the rule was as follows: first he will walk a certain distance, then he will stop, and I will come to the turn of the curve. I calculated this distance by means that since the map is divided in half, this means that the remaining distance from the right and left is ***N/2***, to see how you walk and whether the curve is also full of a beeper or not, you have to see if the result of the division is not a multiple of the two. ا This means that you have to fill in the angle or curve. Things will become clear in the examples.




- In this example, divide the map into two halves first, then it will come up with the remaining distance, which can be calculated without moving the robot, as follows: the length is divided by 2, and the distance will be found (only when it is odd), when calculating the distance here, the result was floor(9/2)= 4 which means that the distance between the far right and the middle is 4, now we will come to see if we can put a beeper or not, I will check if the 4%2?=0 the answer is yes then I will not put any beeper at the curve, so he will walk two steps and then crouch Up or down doesn't make a difference and finally goes to the wall and does the same to the opposite side.

- **Divided to Four chambers (Cont.’s)**




  - In this example, when we calculated the distance here, the result was floor(7/2)=3, which means that the distance between the far right and the middle is 3. Now we will come to see if we can put the beeper or not. I will check if the 3%2? =0 The answer is no. So I will put a beeper at the curve, so he will walk one step + one because he did not fulfill the condition, then crouch up or down and finally turn to the wall and do the same to the opposite side.

***Video link: https://youtu.be/7M4MO291cyY***