Assignment #1

KAREL THE ROBOT

Student Name : Diya Momani

The Report

* Assignment purposes and requirements

This assignment focuses on dividing Karel's world into four or less equal parts. Using beepers to divide the shape. The solution must be reached with the least number of steps, the least number of beepers, and the largest number of parts less than or equal to four.

* Introduction

In this assignment, I tried to follow all the required conditions by reaching as many chambers as possible, less than or equal to four, with the least number of steps and beepers possible, through mathematical equations using only the functions given in Karel reference card. I tried to make my code reusable and structured with the fewest possible lines by using functions. It has four basic functions : CountRowsAndColumns(), OneColOrRow(int GreaterThanOne), TwoColOrRow(int GreaterThanTwo) and ThreeOrMore(),which I will explain in detail later.

* The Global Variables:

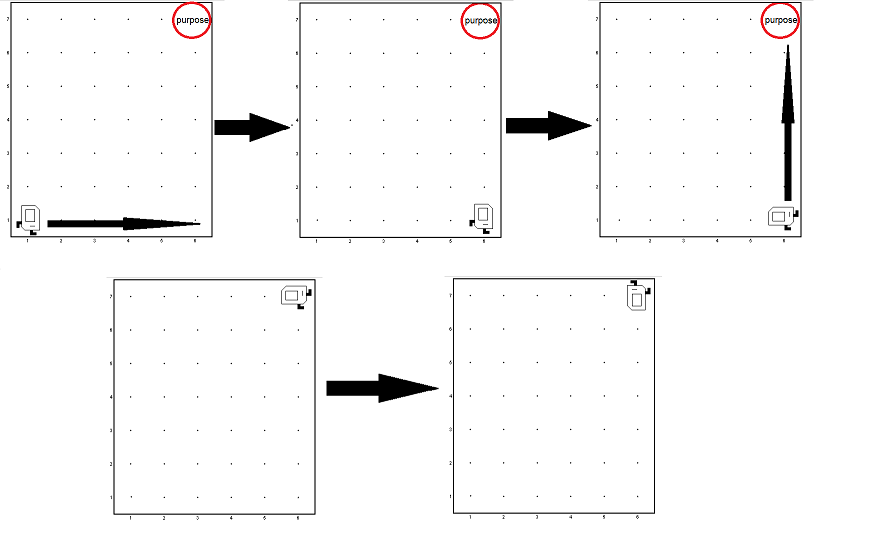
1. Col : number of columns in Karel world.
2. Row : number of rows in Karel world.
3. Steps : number of steps to achieve four or less chambers.

* The Functions:

1. LeftThenRight() turn left and put beeper and move and put beeper and turn right.
2. RightThenLeft() turn right and put beeper and move and put beeper and turn left.
3. CountRowsAndColumns()

* **Overview:**

This function calculates the number of rows and columns in Karel's world. This is by moving to the upper right corner. It also calculates the number of steps needed to reach this corner. as follows:



* **The Algorithm:**

1. Declare 2 boolean variables and assign them to false (initialization) : completeRow and completeCol
2. While completeRow is false do the following:
3. If frontIsClear (there is no wall in front of Karel) , then move Karel.

* If completeCol is still false (Karel hasn't finished counting the columns yet) , then increase global variable col by 1 and increase global variable steps by 1.
* If completeRow is still false (Karel hasn't finished counting the rows yet) , then increase global variable row by 1 and increase global variable steps by 1.

1. If frontIsBlocked (there is a wall in front of Karel) and completeCol still false, then turn left and assign completeCol to true.
2. If frontIsBlocked (there is a wall in front of Karel) and completeRow still false, then turn left and assign completeRow to true.
3. Stop when completeRow variable becomes true.
4. OneColOrRow(int GreaterThanOne)

* **Overview:**

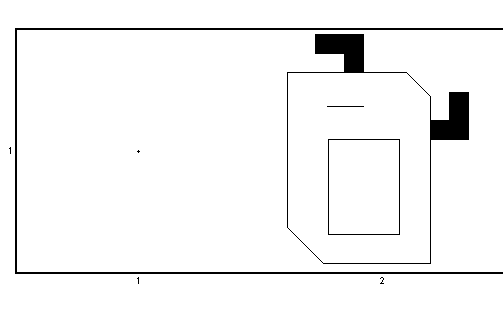
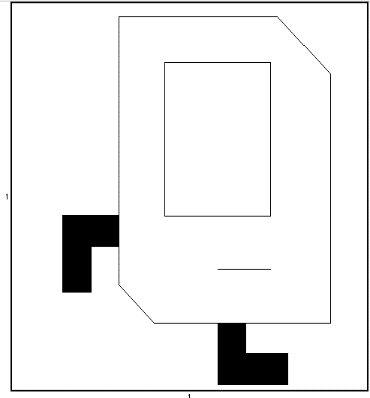
This function calculates the number of steps needed to divide Karel's world in case the minimum number of rows or columns is equal to 1, this function receives a parameter of type Integer indicating the number of rows or columns that is not equal to 1 (greater than 1), and there are five main cases in this function :

1. When GreaterThanOne variable is 1 or 2.
2. When GreaterThanOne variable is 3.
3. When GreaterThanOne variable is 4.
4. When GreaterThanOne variable is 5 or 6.
5. When GreaterThanOne variable is 7 or greater.

* **The Algorithm:**

1. When GreaterThanOne variable is 1 or 2 :

In this case, Karel does not put any beeper ,because the Karel's world cannot be divided to more than 1 chamber, so I did not include this case in the code to shorten the number of lines (The movement that Karel moves in this case is the result of counting the rows and columns in CountRowsAndColumns function).

Shape

Description automatically generated

Case 2X1

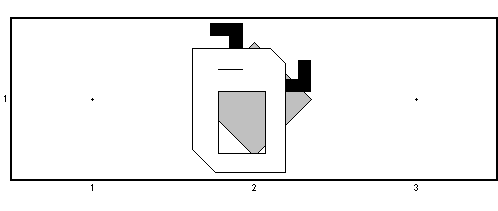
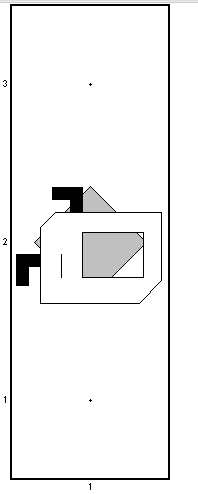
Case 1X1

Case 1X2

1. When GreaterThanOne variable is 3 :

In this case, Karel put one beeper in the middle of the world, if the col variable equals 1 (the columns numbers is 1), Karel will turn left first ,and if the col variable equals 1 or the row variable equals 1, it will do the following:

1. Move one step
2. Increase steps variable by 1
3. Put one Beeper



Case 3X1

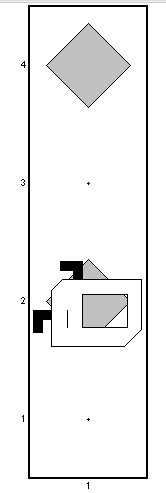
Case 1X3

1. When GreaterThanOne variable is 4:

In this case, Karel put 2 beepers in the world in the first and third cell, if the col variable equals 1 (the columns numbers is 1), Karel will turn left first ,and if the col variable equals 1 or the row variable equals 1, it will do the following:

1. Put beeper and move and increase steps variable by 1
2. Move and increase steps variable by 1
3. Put beeper

A picture containing box and whisker chart

Description automatically generated

Case 4X1

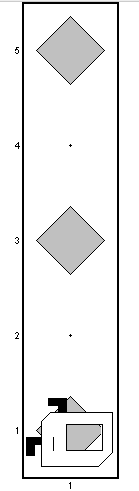
Case 1X4

1. When GreaterThanOne variable is 5 or 6:

In this case, Karel put 3 beepers in the world, if the col variable equals 1 (the columns numbers is 1), Karel will turn left first ,and if the col variable equals 1 or the row variable equals 1, it will do the following:

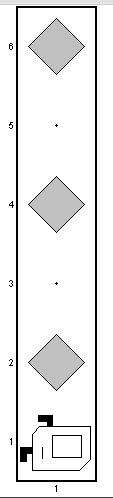
1. Enter a for loop from the number 0 to the number 4
2. move and increase the value of steps each time inside the for loop if there is no wall in front of Karel.
3. puts beeper if i is even number

Shape, polygon

Description automatically generated

Case 5X1

Case 1X5



Shape

Description automatically generated

Case 6X1

Case 1X6

1. When GreaterThanOne variable is 7 or greater:

In this case, Karel put 3 or more beepers in the world, if the col variable equals 1 (the columns numbers is 1), Karel will turn left first ,and if the col variable equals 1 or the row variable equals 1, it will do the following:

1. Initialize 2 variables :

* div : which calculate (GreaterThanOne-3)/4 , and indicates to the number of cells between every 2 beepers
* mod : which calculate (GreaterThanOne-3)%4, and indicates to the number of beepers at the end of world before starting the division.

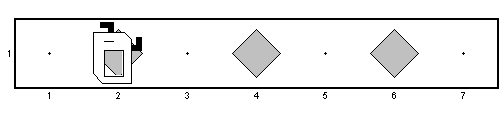
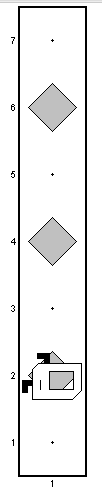
e.g : GreaterThanOne=7 , then:

div=1 : the numbers of cells between every 2 beepers is 1.

mod=0 : there is no beepers at the end of Karel's world before starting the division.

b. Enter for loop from 0 to 2 (because there is no more 3 beepers will be placed), and inside the for loop there is 4 main operations:

* while loop to put beepers at the end of Karel's world
* for loop to move between every 2 beepers
* put beeper
* if statement Indicates that Karel does not move after putting on the last beeper



Case 7X1

Shape

Description automatically generated with medium confidence

Case 10X1

Case 7X1

1. TwoColOrRow(int GreaterThanOne)

* **Overview:**

This function calculates the number of steps needed to divide Karel's world in case the minimum number of rows or columns is equal to 2, this function receives a parameter of type Integer indicating the number of rows or columns that is not equal to 2 (greater than 2), and there are two main cases in this function :

1. When GreaterThanTwo variable is less than or equals 6.
2. When GreaterThanTwo variable is more than or equals 7.

* **The Algorithm:**

1. **When GreaterThanTwo variable is less than or equals 6.**

First of all, Karel put beeper on the first cell, and initializing Boolean variable (blocked) which indicates if in front of Karel there is wall or not, and assign it to false.

Then, there are two cases:

1. For loop from 0 to 2 to put beepers in zigzag order when GreaterThanTwo variable less than or equals to 4.

In this case ,there are 4 main operations:

1. If statement that move Karel and turn left and move(While maintaining an increase in the value of the steps) if :

* The rows number is 2 and is i value is odd
* The columns number is 2 and I value is even

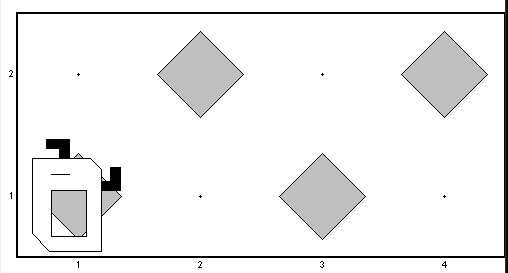
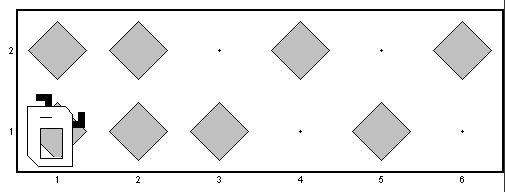
Note: if columns number is 2 and i value is more than 0, or row number is 2 , then turn right before doing the previous operations.

1. Else (If the previous conditions are not met), then turn Karel left and move and turn right and move.
2. Put beeper
3. If frontIsBlocked (there is a wall in front of Karel ), assign blocked to true.
4. If blocked variable is still false, then Karel will put beeper in every cell it enter , so this case is represented by entering for loop from 0 to 3 while frontIsClear is true (there is no wall in front of Karel),and there is 3 main operation in this case :
5. Move Karel and put beeper while entering the for loop.
6. If columns number is 2, then:

* If i value is 0 or 3 Karel turn left
* Else turn right

1. If rows number is 2, then:

* If i value is 0 or 3 Karel turn right
* Else turn left



Case 4X2

Case 6X2

1. **When GreaterThanTwo variable is more than or equals 7.**

**Note: this case is similar to OneColOrRows -> when GreaterThanOne variable is 7 or more case, the difference is that Karel will put beeper pairs instead of one, but I will Explain this case too.**

In this case, Karel put 3 or more beeper pairs in the world, if the col variable equals 2 (the columns numbers is 2) and mod variable is 0, Karel will move and turn left first ,and if the col variable equals 2 or the row variable equals 2, it will do the following:

1. Initialize 3 variables :

* div : which calculate (GreaterThanOne-3)/4 , and indicates to the number of cell pairs between every 2 beeper pairs
* mod : which calculate (GreaterThanOne-3)%4, and indicates to the number of beeper pairs at the end of world before starting the division.

e.g : GreaterThanOne=7 , then:

div=1 : the numbers of cells between every 2 beepers is 1.

mod=0 : there is no beeper pairs at the end of Karel's world before starting the division.

* EvenOrOdd : assign it to mod value to maintain its value.

1. Enter for loop from 0 to (mod-1), to put beeper pairs at the end of world before starting the division:

In this case ,there are 3 main operations:

* If statement that put beeper and move Karel and put beeper and turn left (While maintaining an increase in the value of the steps) if :
* The rows number is 2 and j value is odd
* The columns number is 2 and j value is even

Note: if columns number is 2 and j value is more than 0, or row number is 2 , then turn right before doing the previous operations.

* Else (If the previous conditions are not met), then turn Karel left and put beeper and move and put beeper and turn right by using LeftThenRight function .
* Move

Shape, square, polygon

Description automatically generated

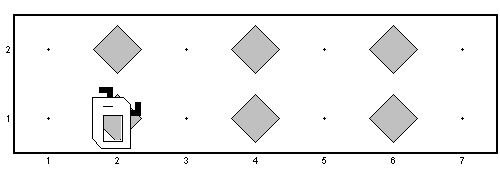
1. Enter for loop from 0 to 2,to put beeper pairs that divide Karel's world into 4 chambers, there are 2 main operations inside this for loop:
2. If i value more than 0 or equal 0, this if statement make Karel move between two beeper pairs:

* If i value value more than zero then move by div value +1
* Else move by div value

1. If number of columns equals 2, or number of rows equals 2:

* If number of columns equals 2
* If mod value equals 0
* If EvenOrOdd value is even , then turn left and put beeper and move and put beeper and turn right by using LeftThenRight function.
* Else , turn right and put beeper and move and put beeper and turn left by using RightThenLeft function.If mod value greater than 0
* If EvenOrOdd value is even , then use RightThenLeft function.
* Else then use LeftThenRight function.
* If number of rows equals 2
* If EvenOrOdd value is even , then use LeftThenRight function.
* Else then use RightThenLeft function.

d. decrease the value of EvenOrOdd by 1 while entering the loop



Case 10X2

Case 7X2

1. ThreeOrMore()

* **Overview:**

This function calculates the number of steps needed to divide Karel's world in case the minimum number of rows or columns is equal to 3 or more, and there are three main cases in this function :

1. **When number of columns and number of rows is odd**

In this case ,Karel will draw a cross shape with beepers to divide the world into 4 equals chambers in shape and area , Karel starts dividing world from the lowest between columns and rows , for instance: if rows number is lower than columns, then Karel will start dividing rows .

* **The Algorithm:**

There are 3 main operations in this case:

1. Initialize 3 variables:

* Low: indicates to the lowest value between columns/2 and rows/2.
* High: indicates to the highest value between columns/2 and rows/2.

1. If statement do the following:

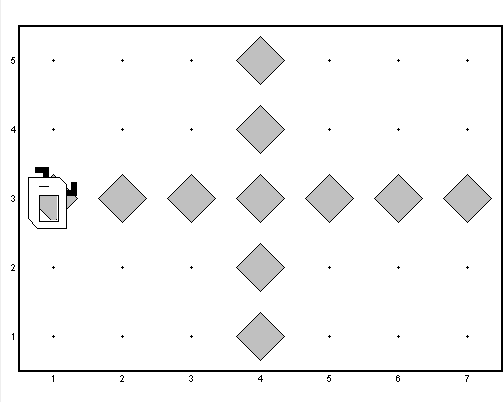
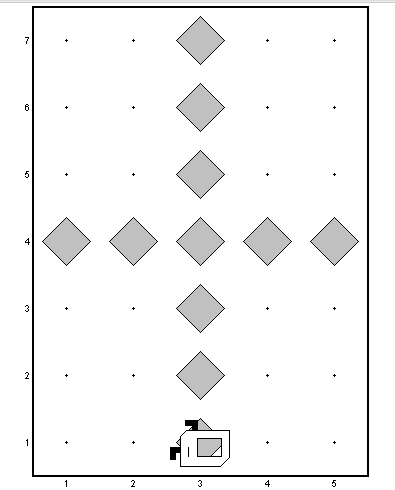
* If col variable less or equals row variable, then move by low value and then turn left.
* Else , turn left and move by low value and turn right

1. Complete the moves as follow:

* Move by high value and put beeper each turn
* Turn left
* Move by low value and turn around
* If statement:

1. If col value is less than or equals row value, then move by col-1 value and put beeper each turn.
2. Else , move by row-1 value and put beeper each turn.

* Put beeper and turn around
* Move by low value
* Turn right
* Move by high value and put beeper each turn



**Case 5X7**

**Case 7X5**

1. **When one of the rows or columns is even, and the other is odd :**

Since it’s not possible to split an even length into two equal parts the main idea here is to move part of the line as much as needed to make the small square equals the big one, and I avoid the cross method by cross two columns with one row or vice versa to reduce the number of steps and beepers.

. In this case Karel starts dividing world from the odd variable between number of columns or rows, for instance: if the rows variable is odd then it will start dividing rows.

* **The Algorithm:**

There are 3 main operations in this case:

1. Initialize two variables:

* Odd : the value of the odd variable between col variable and row variable.
* Even : the value of the even variable between col variable and row variable.
* Moves: indicates to the number of steps Karel will moves before and after the displacement. Note: you will notice that Moves value will change by increasing or decreasing the value by 1, and this is to avoid duplicate the beeper at the same cell.

1. If statement:

* If col variable is odd , then move by Odd/2 value and turn left.
* Else , turn left and move by Odd/2 value and turn right.

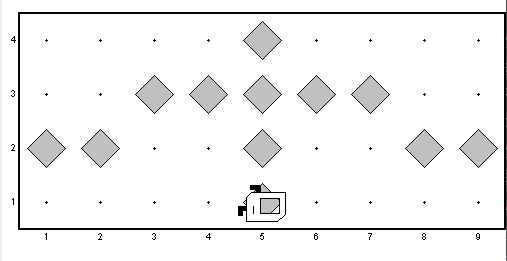
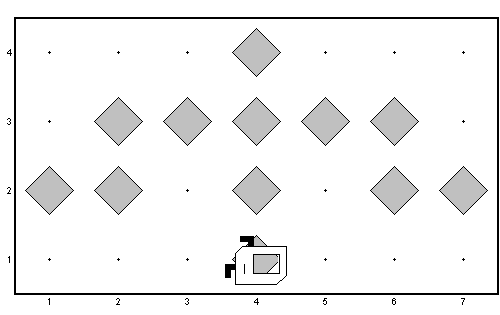
1. Complete the moves as follow:
2. Move by Even/2 value and put beeper each turn.
3. Turn left
4. Move by Odd/2 value
5. Turn around
6. If the half of Odd variable is even then decrease Moves variable by 1
7. Move by Moves value and put beeper each turn
8. If statement, indicates to putting two beeper or one at the displacement:

* If the half of Odd variable is even , then turn right, put beeper, move, put beeper and turn left using RightThenLeft function (put two beepers at the displacement).
* Else turn right, put beeper, move and turn left(put one beeper at the displacement).

1. If the half of Odd variable is even then increase Moves variable by 1
2. Move by Moves value and put beeper each turn
3. Move two times
4. If the half of Odd variable is even then decrease Moves variable by 1
5. Move by Moves value and put beeper each turn
6. If statement, indicates to putting two beeper or one at the displacement:

* If the half of Odd variable is even , then turn left, put beeper, move, put beeper and turn right using LeftThenRight function (put two beepers at the displacement).
* Else turn left, put beeper, move and turn right (put one beeper at the displacement).

1. If the half of Odd variable is even then increase Moves variable by 1
2. Move by Moves value and put beeper each turn
3. Turn around
4. Move by Odd/2 value
5. Turn right
6. Move by Even/2 value and put beeper each turn.



**Case 9X4**

**Case 7X4**

1. **When number of columns and number of rows are even:**

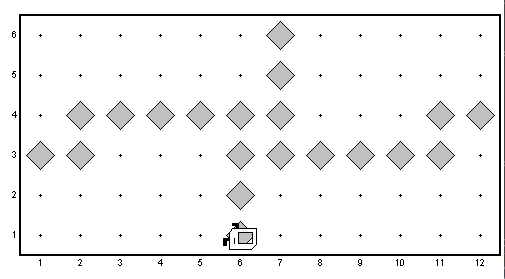
* **Overview:**

Since it’s not possible to split an even length into two equal parts the main idea here is to move part of the line as much as needed to make the small square equals the big one . so also here I used moving part of the line method ,but here I used it for 2 lines not one like previous case, and I avoid the cross method by cross two columns with two rows to reduce the number of steps and beepers.

* **The Algorithm:**
* the maximum between columns and row called high and the minimum called low
* The squares dimensions are high/2 \* low/2 -1 and high/2-1 \* low/2 .
* First of all calculate the area of each rectangle to find out the different in area between them , let’s call diff .
* If the rows are more than columns Karel will turn right and move high/2 steps then turn right
* otherwise just move high/2 steps and turn left Move low/2-1 steps and put beeper on each one
* turn right and move high/2 - diff/2 steps and put beeper on each one then turn left move one step and put a beeper only if the diff is not divisible by 2 , then turn right and another one step and continue diff/2 steps and put beeper on each then turn around and move high/2-1 steps and repeat the same operation with the opposite directions

A picture containing scatter chart

Description automatically generated

Shape

Description automatically generated

**Case 12X8**

**Case 12X12**

**Case 12X6**