**By Ahmad Emad**

**YouTube video:** https://youtu.be/z6CUp96oPaM

**Email:** ahmademad995.ae@gmail.com

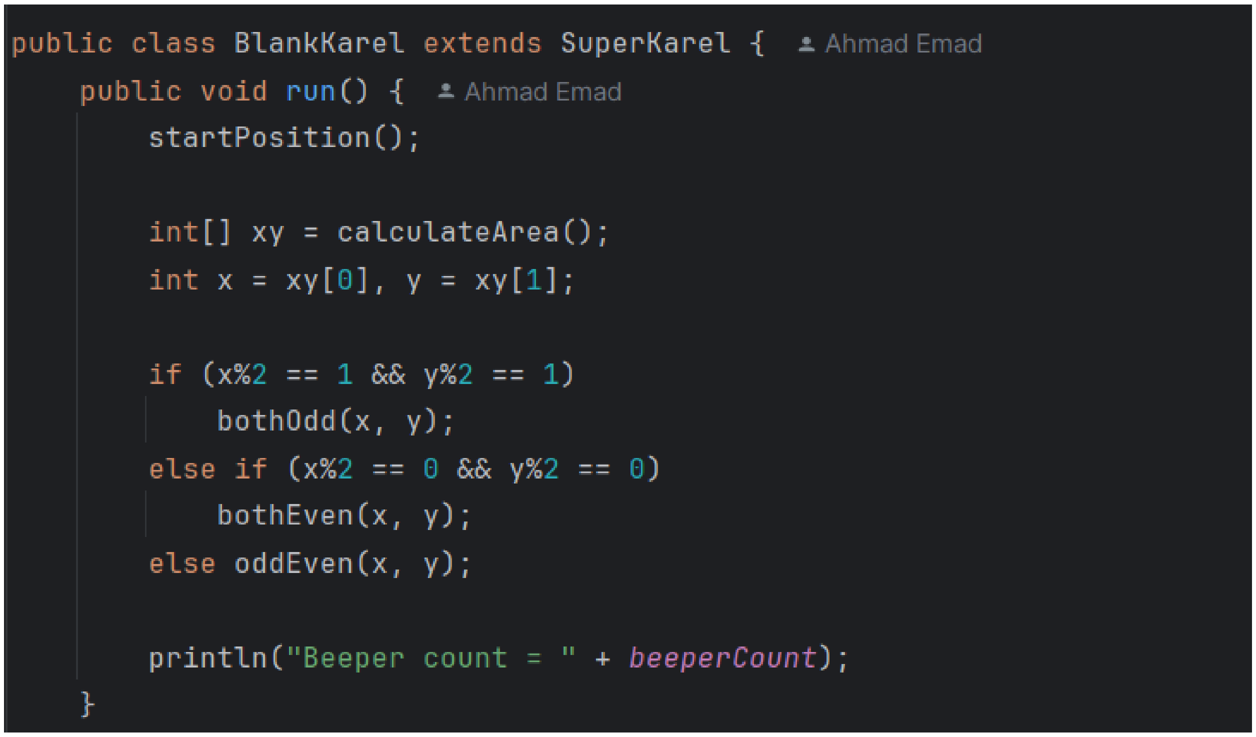
## Objectives:

- Divide any given map into four equal chambers.
- If four chambers are not possible, divide into the highest number possible.
- Minimize the number of code lines.
- minimize the number of moves.
- Use the lowest possible number of beepers.

## Step-by-Step Solution:

1. Ensure Karel is at the starting position.
2. Calculate x-axis and y-axis length.
3. Divide the map into four chambers according to
    a. Odd-odd, axes
    b. Even-even, axes
    c. Odd-even, axes

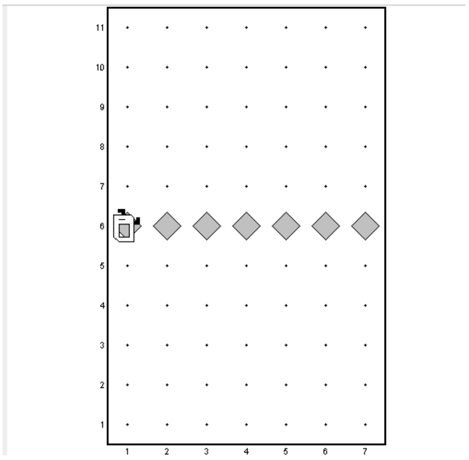
## Main:



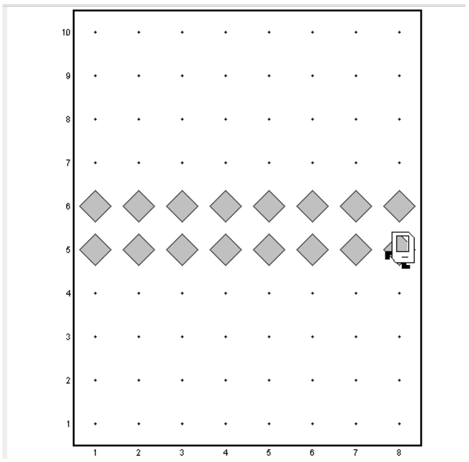
```java
public class BlankKarel extends SuperKarel {  Ahmad Emad
    public void run() {  Ahmad Emad
        startPosition();

        int[] xy = calculateArea();
        int x = xy[0], y = xy[1];

        if (x%2 == 1 && y%2 == 1)
            bothOdd(x, y);
        else if (x%2 == 0 && y%2 == 0)
            bothEven(x, y);
        else oddEven(x, y);

        println("Beeper count = " + beeperCount);
    }
```

**Public Methods:**

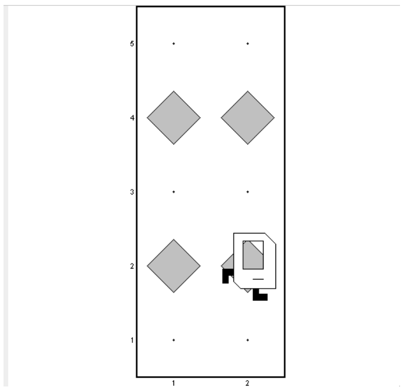
- DivideY(int)
  - Takes the y axis length as a parameter and divide the y axis in half using one wall of beepers.



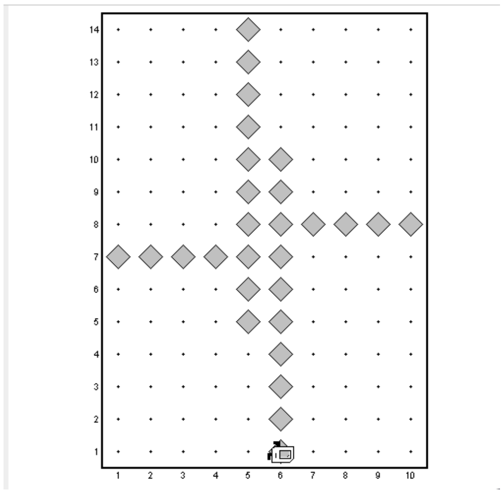
- DoubleDivideY(int)
  - Takes the y axis length as a parameter and divide the y axis in half using two walls of beepers.



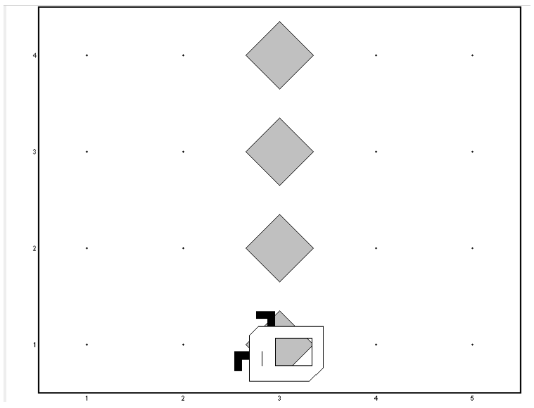
- ThreeChambersY(int)
  - Takes the y axis length as a parameter and divides the y axis into three using two walls of beepers.



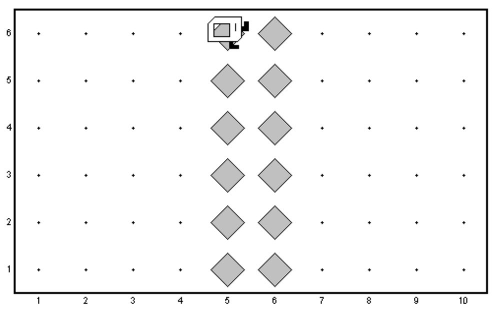
- yAxisCurve(int, int)
    - Takes the x axis and y axis lengths as parameters and divides the whole map into four using a curved vertical wall of beepers.



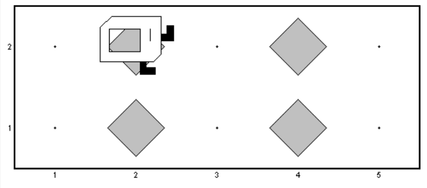
- DivideX(int)
    - Takes the x axis length as a parameter and divides the x axis in half using one wall of beepers.



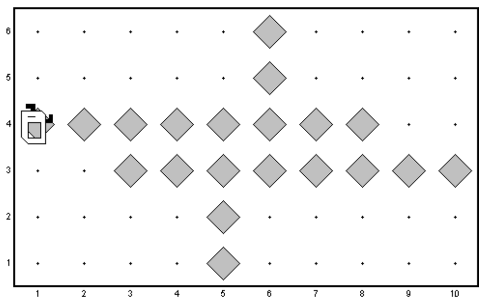
- DoubleDivideX(int)
    - Takes the x axis length as a parameter and divides the x axis in half using two walls of beepers.



- ThreeChambersX(int)
  - Takes the x axis length as a parameter and divides the x axis into three using two walls of beepers.



- XAxisCurve(int, int)
  - Takes the x axis and y axis lengths as parameters and divides the whole map into four using a curved horizontal wall of beepers.



**Helper methods:**

- bothOdd(int, int)
    - Takes the odd x and y parameters and decides the best way to divide the map then calls the appropriate public method e.g.(calling ThreeChambersX() if the y axis length is not divisible, and if the x axis length is divisible by three chambers)
- BothEven(int, int)
    - Takes the even x and y parameters and decides the best way to divide the map then calls the appropriate public method e.g.(calling yAxisCurve() if the y axis length is larger than the x axis length, and both axes are larger than two)
- Oddeven(int, int)
    - Takes the x and y parameters and decides the best way to divide the map then calls the appropriate public method e.g.(calling doubleDivideX() and divideY() if the x axis length is larger than two and even, and the y axis length is larger than one and odd)
- calculateArea()
    - Calculate the x and y axes lengths.
- moveWhileFrontClearPutBeeper()
    - A reusable piece of code to move while front is clear and put beeper.

**Overridden Methods:**

- move()
    - Added a move counter that is incremented every time the method is called to track number of moves.
- putBeeper()
    - added a condition to check if there is no beeper present then calls superclass method to prevent beeper overlap.
    - Added a beeper counter that is incremented when method is invoked and passes the condition check to track number of beepers.

## Conclusion

The given solution involves a top-down approach, where we start by categorizing the axes into three main categories, even-even, odd-odd and odd-even axes and then we start analyzing what appropriate methods to use on the given scenario.

We override methods to exclude repetitive condition checks and to track beeper count which is printed at the end of the run while move counter is printed while Karel is moving. We used reusable methods to utilize clean code principles. moveWhileFrontClearPutBeeper() method has 13 usages divideX() and divideY() has 3 usages each.

We used yAxisCurve() and xAxisCurve() methods to minimize beeper usage by curving the walls and avoid using double walls of beepers.

Our top-down approach and axes analyzing ensured dividing the map into 4 equal chambers or the next possible equal number of chambers.

At the end, the program just used 300 lines of code!