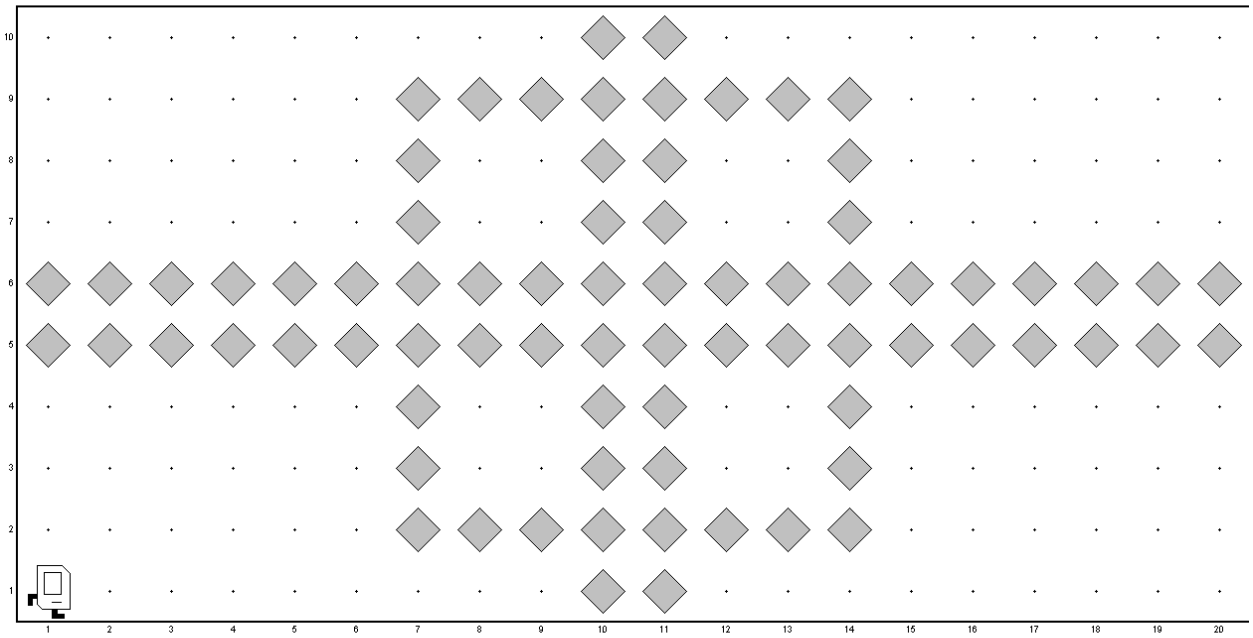


Karel the Robot Assignment

The Report.

By: Ibrahem Jaradat.

Introduction.

In this assignment the goal is to divide a given map into 4 + 4 where the inner chambers should be the biggest possible equals squares, and the outer chambers should be equal in size, and they should be L-shaped. All with using the lowest number of beepers and moves.

Analysis and design.

This program is divided into two main parts:

First part : we calculate the size of the map and then divide the map and find the inner chambers theoretically on a 2D array (Goal Array).

```
[0,0,0,0,1,1,0,0,0,0,]
[0,1,1,1,1,1,1,1,1,0,]
[0,1,0,0,1,1,0,0,1,0,]
[0,1,0,0,1,1,0,0,1,0,]
[1,1,1,1,1,1,1,1,1,1,]
[1,1,1,1,1,1,1,1,1,1,]
[0,1,0,0,1,1,0,0,1,0,]
[0,1,0,0,1,1,0,0,1,0,]
[0,1,1,1,1,1,1,1,1,0,]
[0,0,0,0,1,1,0,0,0,0,]
```

- The ones represent the presence beepers

Second part : Where we make karel move to put the beepers according to the (goal-Array) using the best fitted solution.

Why did I choose this approach?
There may be a lot of ideas of how to move on the map to achieve the goal, but no matter the way you choose to draw the solution, the final goal doesn't change.
So this approach would make understanding and implementing the optimal solution more easier and coordinated.

Creating the solution.

To generate the (Goal-Array) we have to do some initial steps first.

1. Get Karel's world dimensions.
2. Make a 2-d array the size of map dimensions.
3. Split the map into 4 equal spaces.
4. Draw the inner chambers

After generating the solution map, we use the proper approach to put beepers in place.

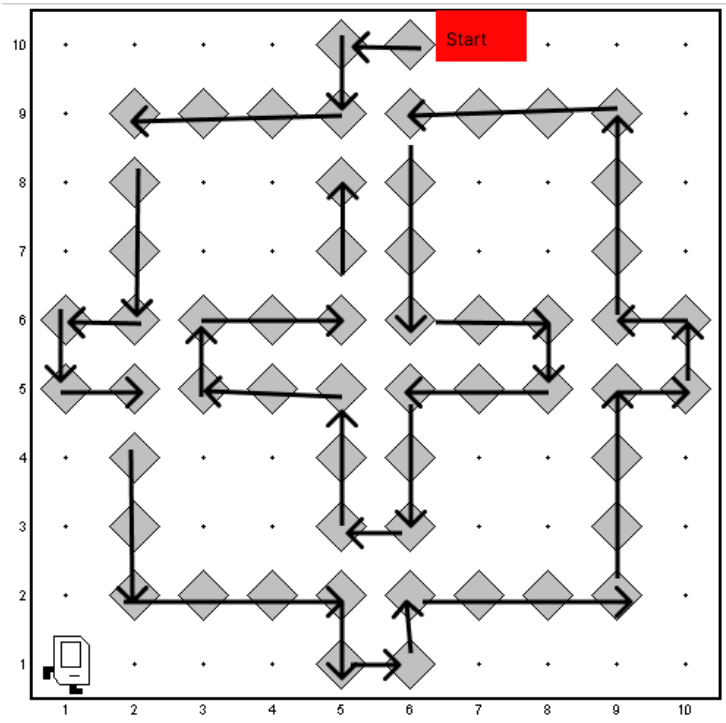
What is the proper approach to put beepers on the map ?
There is mainly four test cases:

If the map dimensions were even*even.
If the map dimensions were odd*odd.
If the map dimensions were even*odd.
If the map dimensions were odd*even.

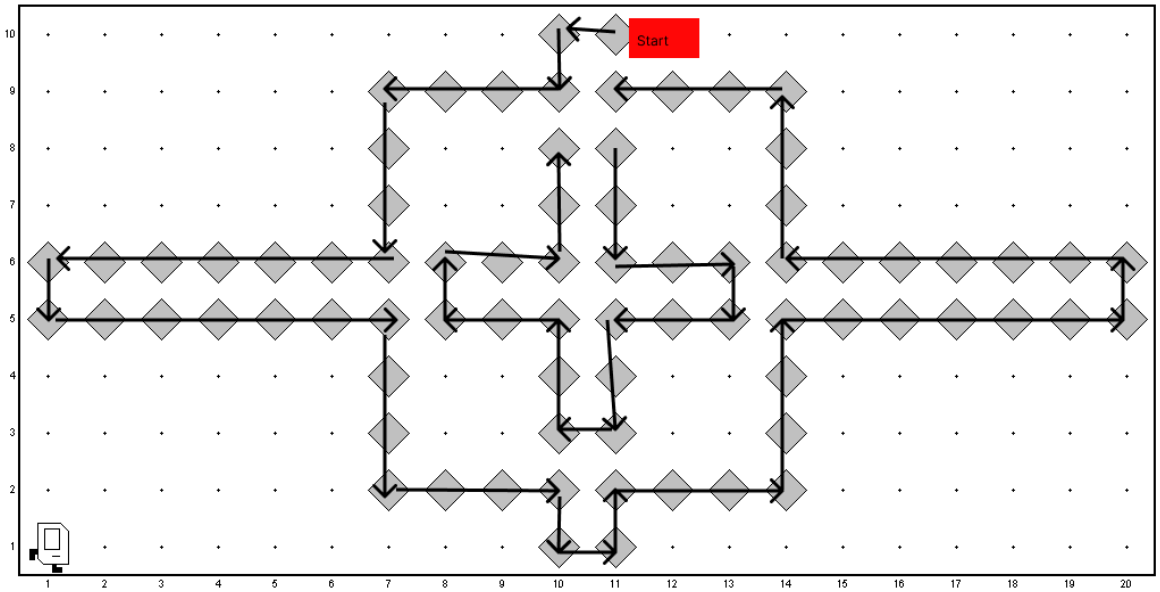
And each has conditions that may differentiate the solution where maybe X > Y or Y > X or X==Y.

These cases are represented as follows:

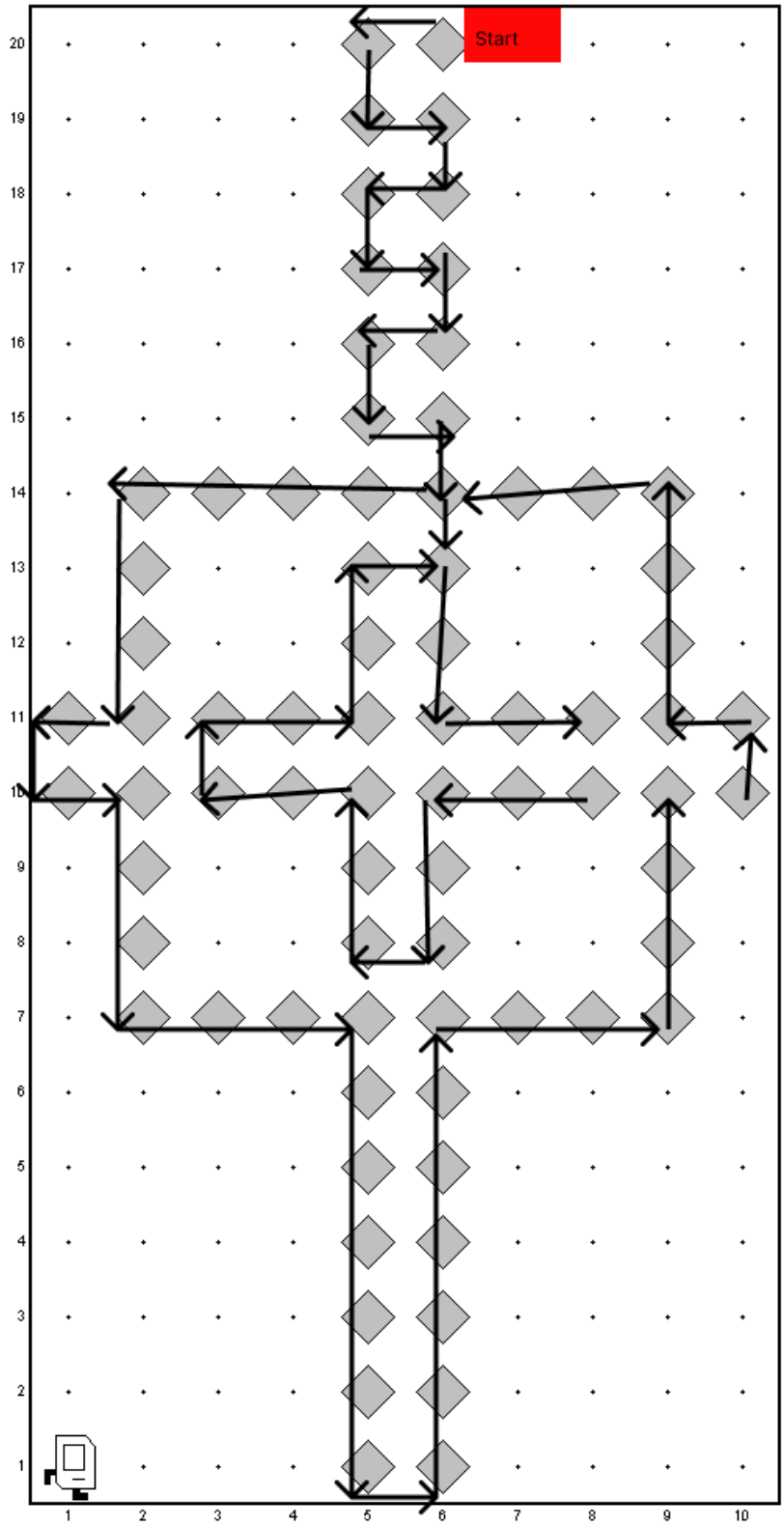
1- A) If the map dimensions were even*even and X==Y:



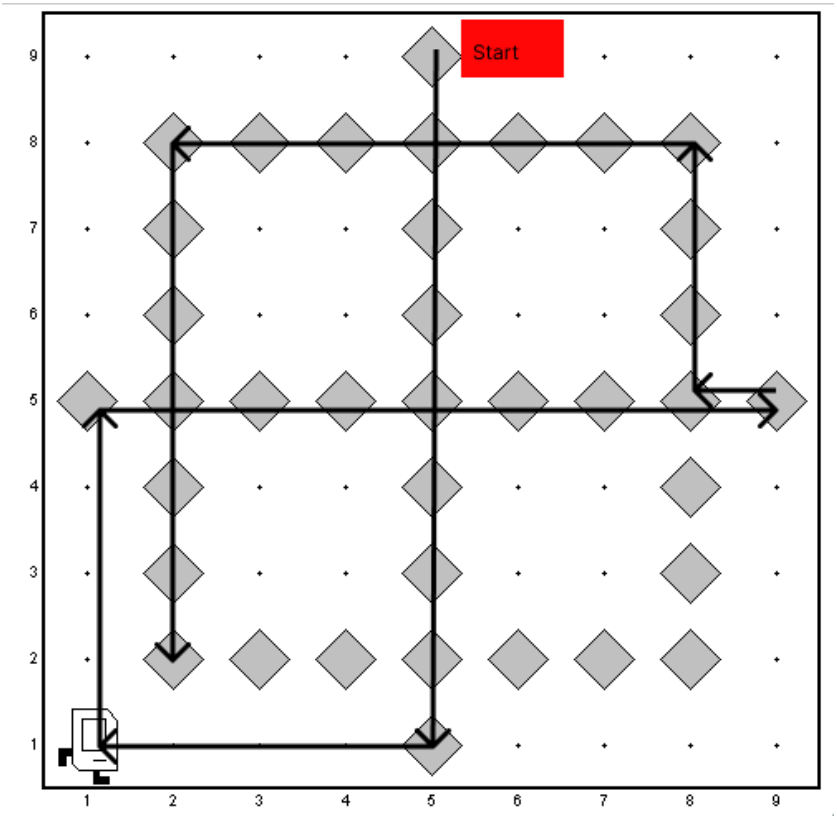
1-B)If the map dimensions were even*even and X > Y:



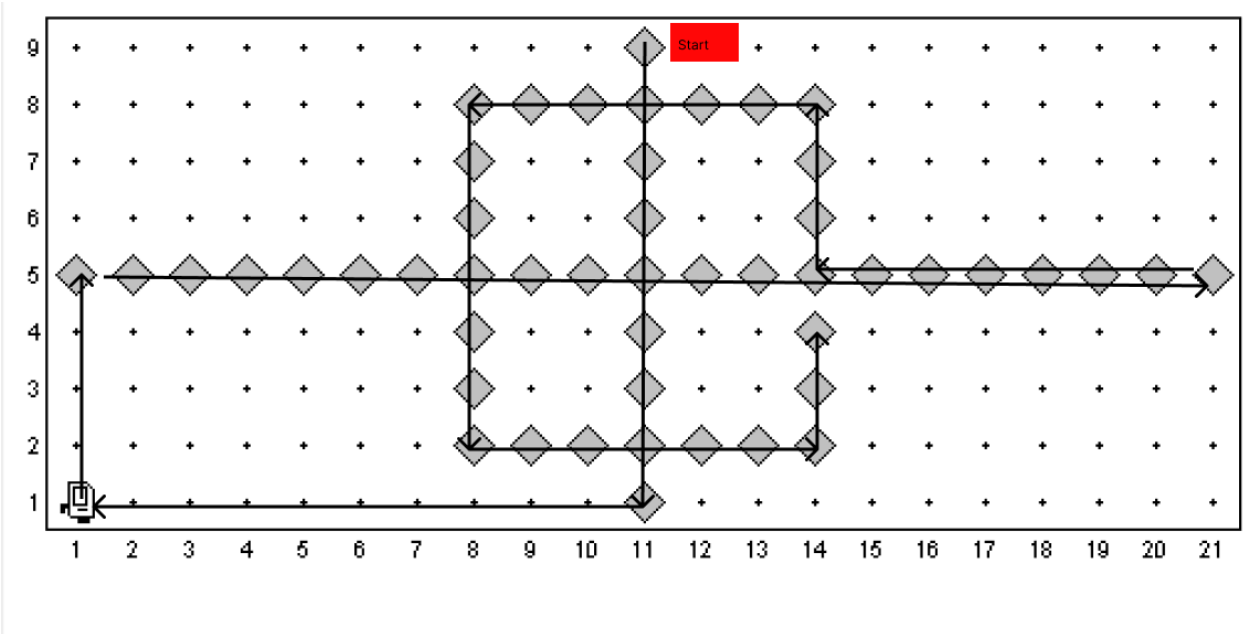
1-C) If the map dimensions were even*even and X < Y



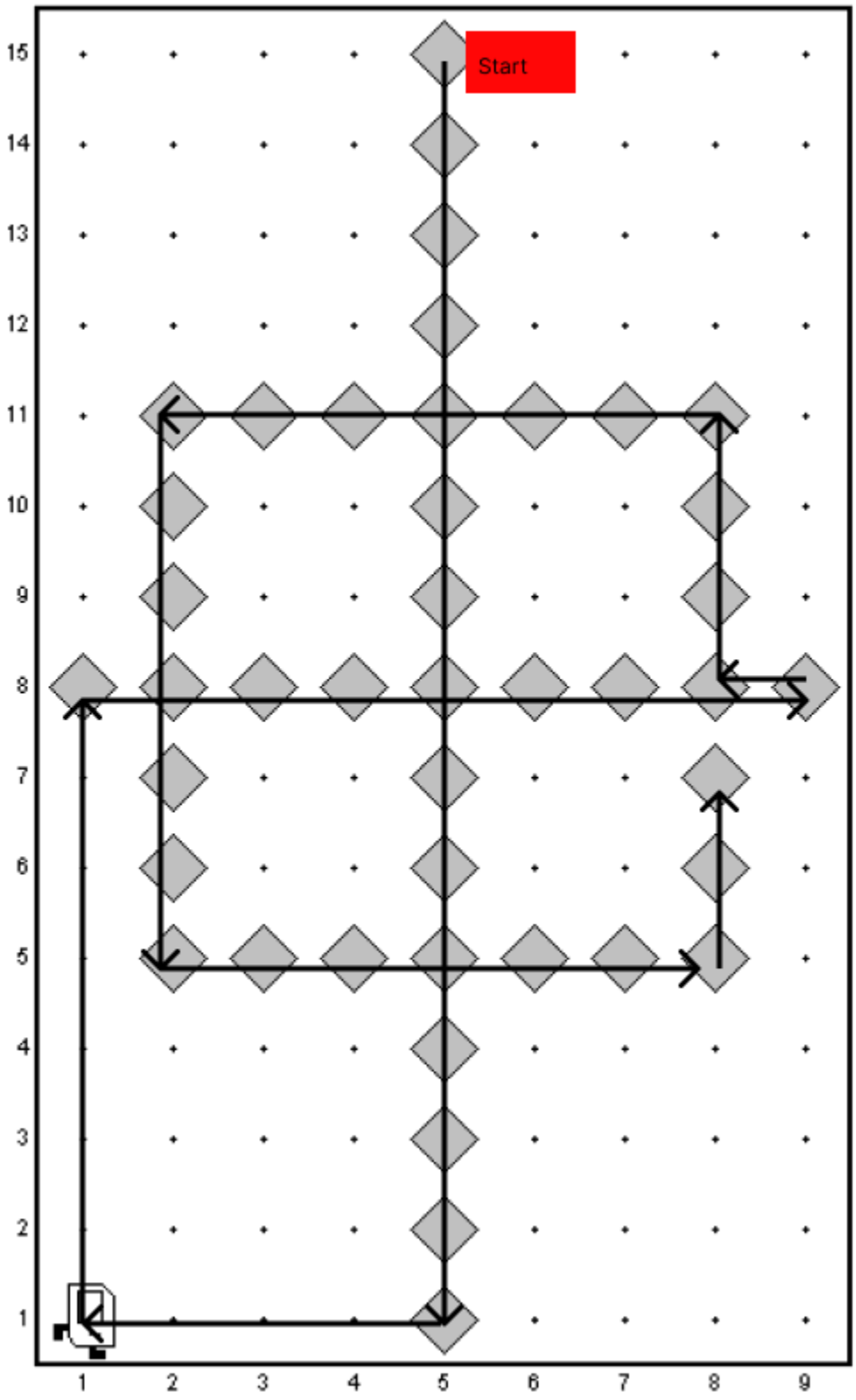
2- A) If the map dimensions were odd*odd and X==Y



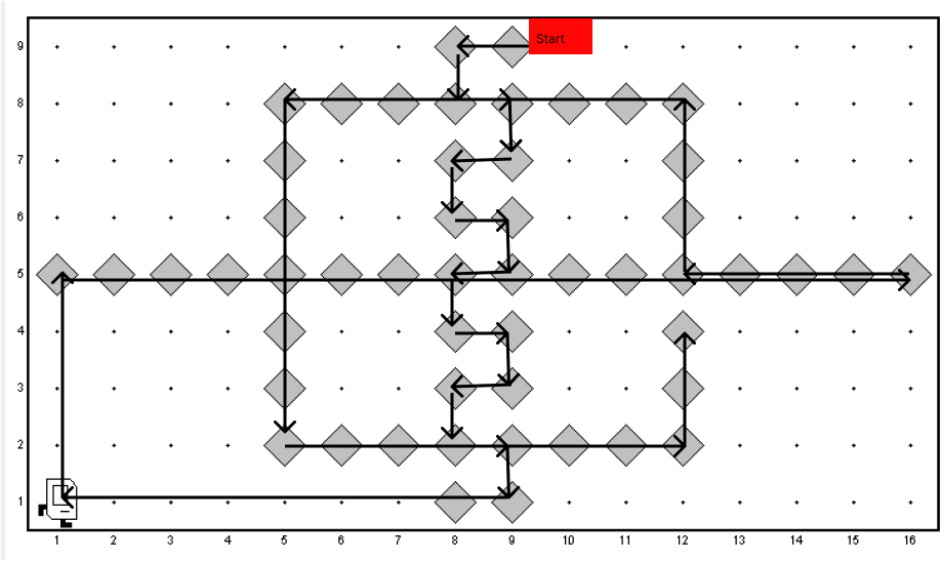
2- B) If the map dimensions were odd*odd and X>Y



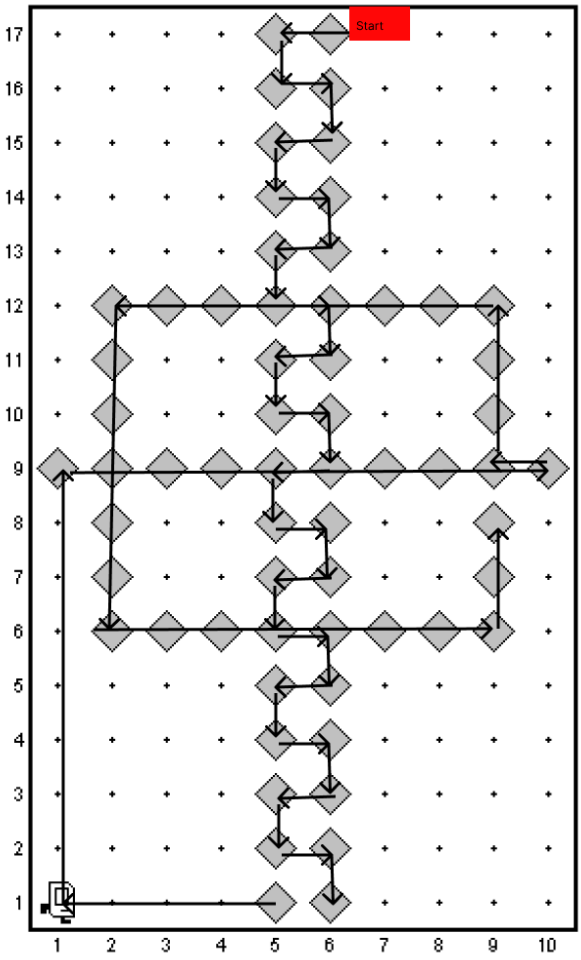
2-C) If the map dimensions were odd*odd and X>Y



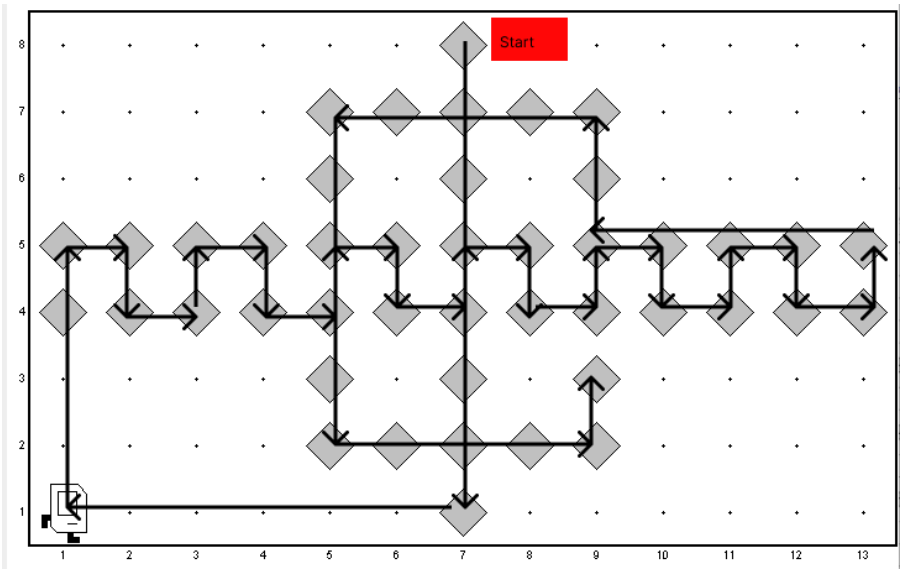
3-A) If the map dimensions were even*odd and X > Y.



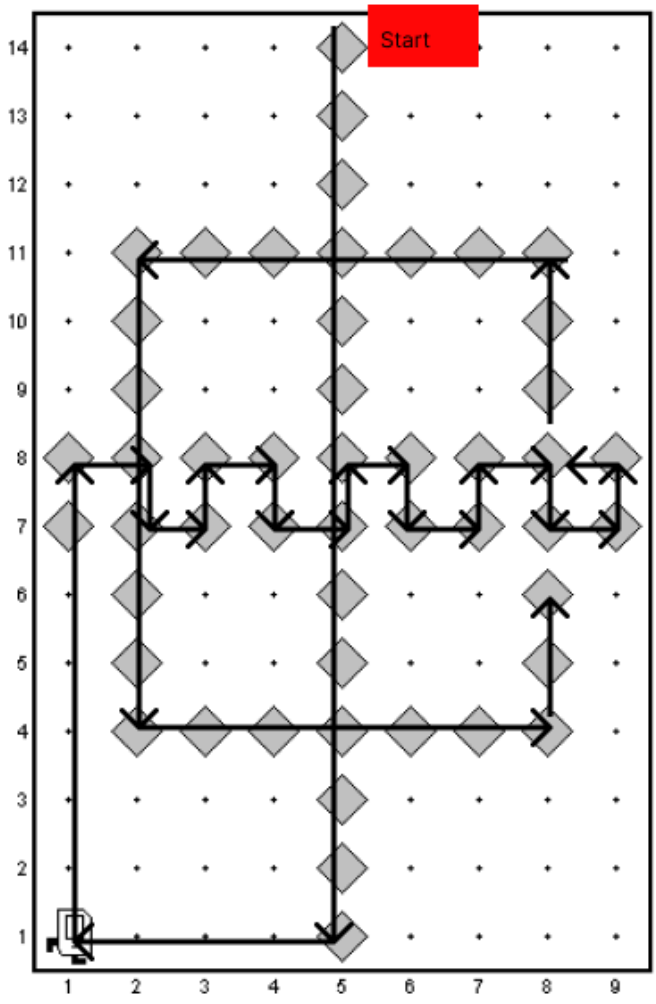
3-B) If the map dimensions were even*odd and X < Y.



4-A) If the map dimensions were odd*even and X > Y.



4-A) If the map dimensions were odd*even and X < Y.



Methods used to solve the problem.

1- `moveAndCount()`

This method will move Karel forward while counting the steps and tracking his location.

```
public void moveAndCount(){  // this method moves carl around but all with tracki
    if(facingEast()) currentX++;
    else if(facingWest()) currentX--;
    else if(facingNorth()) currentY++;
    else currentY--;
    move();
    count++;
    System.out.println("step count = "+count);
    System.out.println("Current Location: x = "+ currentX + " y = "+ currentY);
}
```

2- `putBeeperAndCount()`

This method will put a beeper while counting how much was used so far.

```
public void putBeeperAndCount(){  // this method pu
 putBeeper();
 beeperCount++;
 System.out.println("Beeper count = "+beeperCount);
}
```

3- `turn(char c)`

This method is used to turn Karel to a certain direction.

-it can be made using if statements but using (while) is cleaner for me .

```
public void turn(char d){
    if(d=='N')while (notFacingNorth())turnLeft();
    else if(d=='S')while (notFacingSouth())turnLeft();
    else if(d=='E')while (notFacingEast())turnLeft();
    else if(d=='W')while (notFacingWest())turnLeft();
}
```

4- goTo(int nextX , int nextY)

This method is used to move Karel to a certain position, it takes 2 integers representing the destination location (x,y) .

```
public void goTo(int nextX,int nextY ){
    if(nextX> currentX)turn( d: 'E');
    else turn( d: 'W');
    while (nextX!= currentX){
        moveAndCount();}
    if(nextY> currentY)turn( d: 'N');
    else turn( d: 'S');
    while (nextY!= currentY){
        moveAndCount();}
}
```

5- go(char c)

This method is used to move Karel one step to a certain direction; it takes one character representing the direction .
'W'= West / 'E' = East / 'N'=North/ 'S' = South.

```
public void go(char c){
    if(c == 'W' && 0 < currentX-1  && currentX-1 <= x  ){turn( d: 'W'); moveAndCount();}
    else if(c == 'E' && 0 < currentX+1  && currentX  < x  ){turn( d: 'E'); moveAndCount();}
    else if(c == 'N' && 0 < currentY+1  && currentY < y  ){turn( d: 'N'); moveAndCount();}
    else if(c == 'S' && 0 < currentY-1  && currentY-1 < y  ){turn( d: 'S'); moveAndCount();}
}
```

6- `boolean beepersOn(char c)`

This method is used to check if there should be a beeper on the next step in a certain direction based on the solution map ; it takes one character representing the direction and if the condition is true , it deletes this beeper from the solution map and returns true.

‘W’= West / ‘E’ = East / ‘N’=North/ ‘S’ = South.

```
public boolean beeperOn(char c) {
    boolean R = false;
    if      (c=='W' && 0 <= currentX-2  && currentX-2 <= x  && map[currentX - 2][currentY - 1] == 1){map[currentX - 2][currentY - 1] =0;R=  true;}
    else if (c=='E' && 0 <= currentX    && currentX   < x   && map[currentX    ][currentY - 1] == 1){map[currentX    ][currentY - 1] =0;R=  true;}
    else if (c=='S' && 0 <= currentY -2 && currentY-2  <= y && map[currentX - 1][currentY - 2] == 1){map[currentX - 1][currentY - 2] =0;R=  true;}
    else if (c=='N' && 0 <= currentY    && currentY  <= y   && map[currentX - 1][currentY    ] == 1){map[currentX - 1][currentY    ] =0;R=  true;}
    else if (c== 'H' && map[currentX - 1][currentY - 1] == 1 ){map[currentX - 1][currentY - 1] =0;R = true;}
    return R;
}
```

7- `boolean beepersOn(String c)`

This method is same as `beepersOn(char c)` but doesn’t delete the beeper from the solution map

```
public boolean beeperOn(String c) {
    boolean R = false;
    if      (c.equals("W") && 0 <= currentX-2  && currentX-2 <= x  && map[currentX - 2][currentY - 1] == 1){R=  true;}
    else if (c.equals("E") && 0 <= currentX    && currentX   < x   && map[currentX    ][currentY - 1] == 1){R=  true;}
    else if (c.equals("S") && 0 <= currentY -2 && currentY-2  <= y && map[currentX - 1][currentY - 2] == 1){R=  true;}
    else if (c.equals("N") && 0 <= currentY    && currentY  <= y   && map[currentX - 1][currentY    ] == 1){R=  true;}
    else if (c.equals("H") && map[currentX - 1][currentY - 1] == 1 ){R = true;}
    return R;}
```

More explanation and solving.

1- To get Karel’s world dimensions I made a Method called `getDimensions()`

Sence This method will make Karel move to the end East side of the map calculating the X-axis , then move to the end North side of the map calculating the Y-axis, and return a 2-d integer array

the size of map dimensions.

```
public int[][] getDimensions(){
    while (frontIsClear()){
        moveAndCount(); x++;}
    turnLeft();
    while (frontIsClear()){
        moveAndCount(); y++;}
    turnLeft();
    if(x > 6 && y > 6 ){
    return new int[x][y];
    }
    else {System.out.println("-------------- \nmap is too small
        return new int[0][0];}
}
```

2- to Split the map into 4 equal spaces I made a Method called `splitMapEvenly()`

It goes over the center lines for any given map to draw the lines that split it to 4 equal square spaces.

```
public void splitMapEvenly(){ // this function will split the map into 4 equal spaces
    centerPointX = (x / 2); centerPointY = (y / 2);
    if (x % 2 == 1) { // odd x axis
        for (int i = 0; i < y; i++) {
            map[centerPointX][i] = 1;}
    } else { // even x axis
        for (int i = 0; i < y; i++) {
            map[centerPointX][i] = 1;
            map[centerPointX - 1][i] = 1;
        }}
    if (y % 2 == 1) { // odd y axis
        for (int i = 0; i < x; i++) {
            map[i][centerPointY] = 1;
        }
    } else { // even y axis
        for (int i = 0; i < x; i++) {
            map[i][centerPointY] = 1;
            map[i][centerPointY - 1] = 1;}
    }
}
```

3- After that , this method will draw the inner chambers based on the X and Y axis.

`drawInnerChambers()`

this method takes on consideration if one of the axis is bigger than the other, hence the chamber size depends on the smaller axis to minimize the amount of beepers used where :

Chamber size = (the smaller axis - 2)/2 .

Bellow shows if the axis are even*even or odd*odd

```
public void drawInnerChambers() {
    if (x%2==0 && y%2==0  ){  //even x even
        int minVar = Math.min(x, y);
        int cSize = (minVar - 2) / 2;
        for (int i = centerPointY - cSize; i < centerPointY + cSize; i++) {
            map[centerPointX - cSize][i] = 1;
            map[centerPointX + cSize - 1][i] = 1;
        }
        for (int i = centerPointX - cSize; i < centerPointX + cSize; i++) {
            map[i][centerPointY - cSize] = 1;
            map[i][centerPointY + cSize - 1] = 1;
        }
    }
    else if (x%2==1 && y%2==1 ){// odd x odd
        int minVar = Math.min(x, y);
        int cSize = (minVar - 2) / 2;
        for (int i = centerPointY - cSize; i < centerPointY + cSize; i++) {
            map[centerPointX - cSize][i] = 1;
            map[centerPointX + cSize ][i] = 1;
        }
        for (int i = centerPointX - cSize; i <= centerPointX + cSize; i++) {
            map[i][centerPointY - cSize] = 1;
            map[i][centerPointY + cSize ] = 1;
        }
    }
```

Here it draws the inner chambers when the map is even*odd

```
else if ((x%2==0 && y%2==1 )){//even x odd
    if(x>y) {
        int minVar = Math.min(x, y);
        int cSize = (minVar - 2) / 2;
        for (int i = centerPointY - cSize; i <= centerPointY + cSize; i++) {
            map[centerPointX - cSize - 1][i] = 1;
            map[centerPointX + cSize][i] = 1;
        }
        for (int i = centerPointX - cSize; i <= centerPointX + cSize; i++) {
            map[i][centerPointY - cSize] = 1;
            map[i][centerPointY + cSize] = 1;
        }
    }
    else if (y>x){
        int minVar = Math.min(x, y);
        int cSize = (minVar - 2) / 2;
        for (int i = centerPointY - cSize+1; i < centerPointY + cSize-1; i++) {
            map[centerPointX - cSize ][i] = 1;
            map[centerPointX + cSize-1][i] = 1;
        }
        for (int i = centerPointX - cSize; i < centerPointX + cSize; i++) {
            map[i][centerPointY - cSize+1] = 1;
            map[i][centerPointY + cSize-1] = 1;
```

And finally it draws the inner chambers if the map is odd* even

```
else if (x%2==1 && y%2==0) {  //odd even
    if(x>y){
        int minVar = Math.min(x, y);
        int cSize = (minVar - 2) / 2;
        for (int i = centerPointY - cSize; i < centerPointY + cSize; i++) {
            map[centerPointX - cSize+1][i] = 1;
            map[centerPointX + cSize -1][i] = 1;
        }
        for (int i = centerPointX - cSize+1; i < centerPointX + cSize; i++) {
            map[i][centerPointY - cSize] = 1;
            map[i][centerPointY + cSize -1 ] = 1;
        }
    } else if (y > x) {
        int minVar = Math.min(x, y);
        int cSize = (minVar - 2) / 2;
        for (int i = centerPointY - cSize; i < centerPointY + cSize; i++) {
            map[centerPointX - cSize][i] = 1;
            map[centerPointX + cSize ][i] = 1;
        }
        for (int i = centerPointX - cSize; i < centerPointX + cSize+1; i++) {
            map[i][centerPointY - cSize - 1] = 1;
            map[i][centerPointY + cSize  ] = 1;
```

4- After finishing drawing the solution map, comes the time to put the beepers in place while using the minimum amount of moves.

`putBeepers()`

Here it shows the followed procedure to go on all the points when the map is even*even .

```
public void putBeepers() {
    int minVar = Math.min(x, y);
    int cSize = (minVar - 2) / 2;
    goTo( nextX: centerPointX +1  ,y);
    System.out.println("Start drawing-------");
    //even x even
    if (x%2==0 && y%2==0){
        while((currentY!=centerPointY+cSize)){
            if (beeperOn( c: 'H') && !beepersPresent()) {putBeeperAndCount();}
            else if ((beeperOn( c: 'W'))) { go( c: 'W');putBeeperAndCount(); }
            else if ((beeperOn( c: 'E'))) { go( c: 'E');putBeeperAndCount(); }
            else if ((beeperOn( c: 'S'))) { go( c: 'S');putBeeperAndCount(); }
        }
        while (currentY!= 1 ) {
            if (beeperOn( c: 'H') && !beepersPresent()) {putBeeperAndCount();}
            else if (beeperOn( c: 'W') ) { go( c: 'W');putBeeperAndCount(); } //if b on west
            else if (beeperOn( c: 'S')) {go( c: 'S');putBeeperAndCount();}
            else break;
        }
        while (currentY != y) {
            if (beeperOn( c: 'H') && !beepersPresent()) {putBeeperAndCount();}
            else if (beeperOn( c: 'S')) {go( c: 'S');putBeeperAndCount();}
            else if (beeperOn( c: 'S') &&beeperOn( c: 'E')) {go( c: 'S');putBeeperAndCount();}
            else if (beeperOn( c: 'E') ) { go( c: 'E');putBeeperAndCount(); } //if b on east
            else break;
        }
        while (currentY != y) {
            if (beeperOn( c: 'H') && !beepersPresent()) {putBeeperAndCount();}
            else if (beeperOn( c: 'E') ) { go( c: 'E');putBeeperAndCount(); } //if b on east
            else if (beeperOn( c: 'N')) {go( c: 'N');putBeeperAndCount();}
            else if ( beeperOn( c: 'W')) {go( c: 'W');putBeeperAndCount();}
            else break;
        }
        if (!beeperOn( c: 'W')){go( c: 'W');}
        while(true){
            if      (beeperOn( c: 'E')) { go( c: 'E');putBeeperAndCount();}
            else if (beeperOn( c: 'S')) { go( c: 'S');putBeeperAndCount();}
            else if (beeperOn( c: 'W')) { go( c: 'W');putBeeperAndCount();}
            else if (beeperOn( c: 'N')) { go( c: 'N');putBeeperAndCount();}
            else break;
        }
```

Here it shows the followed procedure to go on all the points when the map is odd*odd .

```
else if (x%2==1 && y%2==1) {

    while(true){
        if (beeperOn( c: "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (beeperOn( c: "S")) {go( c: 'S');if(!beepersPresent())putBeeperAndCount();}
        if(currentY == 1 ){goTo( nextX: 1, nextY: centerPointY+1);break;}
    }// will end at cx , 1

    while(true){
        if (beeperOn( c: "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (beeperOn( c: "E")) {go( c: 'E');if(!beepersPresent())putBeeperAndCount();}
        if(currentX == x ){break;}
    }

    goTo( nextX: centerPointX +cSize +1 ,  nextY: centerPointY+2);

    while(true){
        if (beeperOn( c: "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (beeperOn( c: "N") && !beeperOn( c: "W")) {go( c: 'N');if(!beepersPresent())putBeeperAndCount();}
        else if (beeperOn( c: "W")) {go( c: 'W');if(!beepersPresent())putBeeperAndCount();}
        else break;
    }
    while(currentX != centerPointX+1 ){
        if (beeperOn( c: "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (beeperOn( c: "S")) {go( c: 'S');if(!beepersPresent())putBeeperAndCount();}
        else if (beeperOn( c: "E")) {go( c: 'E');if(!beepersPresent())putBeeperAndCount();}
        else break;
    }
    while(currentX != centerPointX+cSize +2 && currentY !=centerPointY+1  ){
        if (beeperOn( c: "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (beeperOn( c: "E")) {go( c: 'E');if(!beepersPresent())putBeeperAndCount();}
        else if (beeperOn( c: "N") ) {go( c: 'N');if(!beepersPresent())putBeeperAndCount();}
        else break;
    }
```

Here it shows the followed procedure to go on all the points when the map is even*odd .

```
//even x odd
else if (x%2==0 && y%2==1) {
    if(y>x){cSize--;}
    while(true){
        if (beeperOn( c "H") && !beepersPresent()) {putBeeperAndCount();}
        else if ((beeperOn( c "W") && beeperOn( c "S"))) { go( c 'W');putBeeperAndCount();go( c 'S');putBeeperAndCount();go( c 'E');putBeeperAndCount();go( c 'S'); }
        else if ((beeperOn( c "S"))) { go( c 'S'); }

        if (currentY == 1 ){break;}

    }
    while(true){
        if (beeperOn( c "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (beeperOn( c "W")) {go( c 'W');if(!beepersPresent())putBeeperAndCount();}
        if(currentY == 1 && currentX == centerPointX){break;}
    }
    goTo( nextX: 1 ,  nextY: centerPointY+1);
    while(true){
        if (beeperOn( c "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (beeperOn( c "E")) {go( c 'E');if(!beepersPresent())putBeeperAndCount();}
        if(currentX == x){break;}
    }
    goTo( nextX: centerPointX +cSize +1 ,  nextY: centerPointY+2);

    while(true){
        if (beeperOn( c "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (beeperOn( c "N") && !beeperOn( c "W")) {go( c 'N');if(!beepersPresent())putBeeperAndCount();}
        else if (beeperOn( c "W")) {go( c 'W');if(!beepersPresent())putBeeperAndCount();}
        else break;
    }
    while(true){
        if (beeperOn( c "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (beeperOn( c "S")) {go( c 'S');if(!beepersPresent())putBeeperAndCount();}
        else if (beeperOn( c "W")) {go( c 'W');if(!beepersPresent())putBeeperAndCount();}
        else break;
    }

    while(currentX != centerPointX+cSize +2 && currentY !=centerPointY+1 ){
        if (beeperOn( c "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (beeperOn( c "E") ) {go( c 'E');if(!beepersPresent())putBeeperAndCount();}
        else if (beeperOn( c "N")) {go( c 'N');if(!beepersPresent())putBeeperAndCount();}
        else break;
    }
```

Here it shows the followed procedure to go on all the points when the map is odd*even .

```
// odd x even
else if (x%2==1 && y%2==0) {
    if (y > x){cSize++;} // because of integer division the result comes decremented by one
    while(true){
        if (beeperOn( c "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (beeperOn( c "S")) {go( c 'S');if(!beepersPresent())putBeeperAndCount();}
        else break;
    }
    goTo( nextX: 1 , centerPointY);
    if (beeperOn( c "H") && !beepersPresent()) {putBeeperAndCount();}
    while(true){
        if (beeperOn( c "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (beeperOn( c "N") && beeperOn( c 'E')) { go( c 'N');if(!beepersPresent()){putBeeperAndCount();} go( c 'E');if(!beepersPresent()){putBeeperAndCount();}g
        else if ((beeperOn( c "E"))) { go( c 'E'); }
        else {go( c 'N');if (beeperOn( c "H") && !beepersPresent()) {putBeeperAndCount();}break;}
    }
    goTo( nextX: centerPointX +cSize  , nextY: centerPointY+2);
    while(true){
        if (beeperOn( c "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (beeperOn( c "N")) {go( c 'N');if(!beepersPresent())putBeeperAndCount();}
        else if (currentY == centerPointY + cSize){break;}
    }
    while(true){
        if (beeperOn( c "W")) {go( c 'W');if(!beepersPresent())putBeeperAndCount();}
        else if(currentX == centerPointX ){go( c 'W');break;}
        else break;

    }
    while(currentY != centerPointY - cSize){
        if (beeperOn( c "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (beeperOn( c "S")) {go( c 'S');if(!beepersPresent())putBeeperAndCount();}
        else if(currentY == centerPointY +1 ){go( c 'S');go( c 'S');}
        else break;


    }
    go( c 'E');
    while(true){
        if (beeperOn( c "H") && !beepersPresent()) {putBeeperAndCount();}
        else if (currentY == centerPointY){break;}
        else if (beeperOn( c "E") ) {go( c 'E');if(!beepersPresent())putBeeperAndCount();}
        else if (beeperOn( c "N")) {go( c 'N');if(!beepersPresent())putBeeperAndCount();}

        else break;
    }}
```