Karel Reference Card

Karel command methods:

```
moveForward();
turnLeft();
turnAround();
turnRight();
pickBeeper();
dropBeeper();
```

Karel method structure:

```
void methodName {
      // your code goes here
}
```

Karel conditional methods:

```
onBeeper()           beeperAhead()
leftIsClear()        frontIsClear()
rightIsClear()
```

Karel conditional operators:

!	means NOT
&&	means AND
||	means inclusive OR

Permitted conditional statements:

```
if (condition) {
      statements executed if condition is true
}

if (condition) {
      statements executed if condition is true
} else {
      statements executed if condition is false
}
```

Permitted iterative statements:

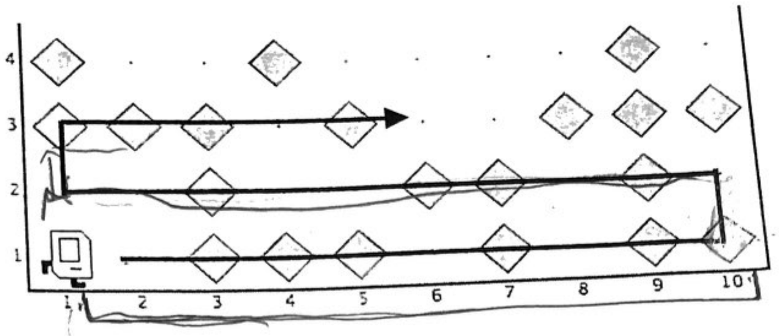
```
repeat (n) {
      statements to be repeated n times
}

while (condition) {
      statements to be repeated
}
```

In general:

- ➡Everything must be done with the set of methods and statements given in this reference.
- ➡NO variable declarations are allowed.
- ➡NO parameter passing and NO return values.

Now that Karel can clean one street, we need to get him to do all streets. Assume the world is just one big rectangular room with no obstructions, and beepers are everywhere. You will have Karel use a zig-zag pattern to pick up all the beepers, like below.



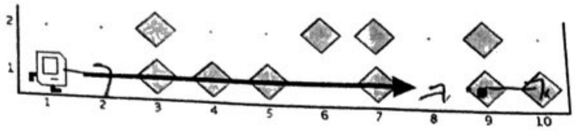
Assume that Karel starts in the lower left corner facing east. Your solution must work with even and odd numbers of streets and avenues. You may use the method in part a) and assume it works correctly regardless of what you wrote.

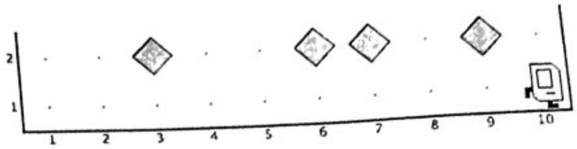
```
/**
 *   Karel picks up all beepers in a room using a zig-zag pattern.
 *   Precondition: Karel starts in the lower left corner facing east.
 *   Postcondition: All of the beepers are picked up.
 */
public void cleanClutter()
```

[illegible]

7) Karel problem: Karel has made a mess! Beepers are everywhere. He needs to pick them all up and make the room uncluttered. You may only use the accepted Karel syntax from the Karel Reference Card on the last page of this test in writing your code.

a) First, Karel needs to pick up a street (see below).





You will write a method for Karel to pick up the beepers on one street. Assume Karel starts at the eastern or western wall and is facing away from the wall. When Karel finishes, he will be on the same street facing the wall opposite from which he came. Make sure Karel picks up all beepers including those at the beginning and end of the street.

```
/**
 *   Karel picks up beepers on one street
 *   Precondition: Karel is next to the eastern or western wall and
 *   facing away from the wall.
 *   Postcondition: Karel is next to the eastern or western wall and
 *   facing towards the wall.
 */
public void cleanStreet() {
while((frontIsClear){
 if(onBeeper()){
 pickBeeper();
moveforward();
if(onBeeper){
pickBeeper();
 }
}
}
```