

The background is a solid pink color. It is decorated with various hand-drawn geometric shapes in white and black. These include a dashed line in the top left, a white triangle in the top center, a black zigzag line in the top right, a white circle in the top right, two parallel black lines in the top right, a white triangle in the top right, a large black circle in the bottom right, a white circle in the bottom right, a black plus sign in the bottom left, a white circle in the bottom left, a white triangle in the bottom left, and a black plus sign in the bottom left.

# Welcome!

We'll get started shortly ...



# Welcome to Section!

## Week Two

Cameron Mohne, Maggie Lee, Surajit Bose



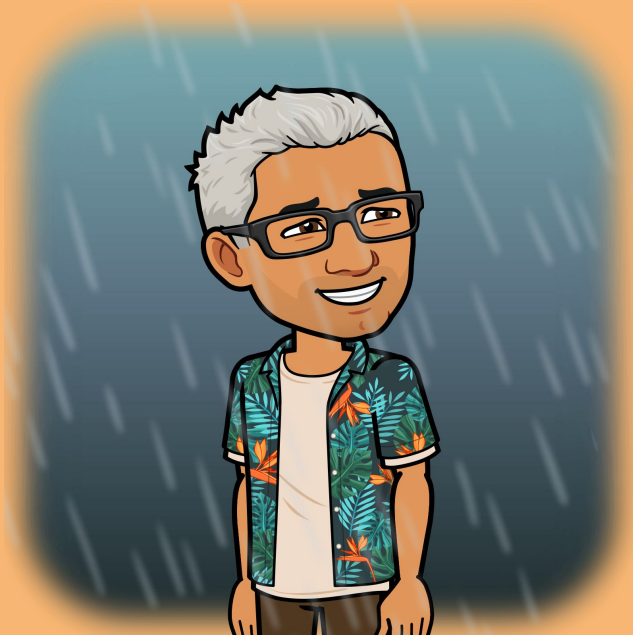


# Introductions



# A Little About Me

+



- ★ My name is Surajit and I'll be your Section Leader for CS 49
- ★ I am retired from a tech career
- ★ In my free time I like to read novels and poetry
- ★ I enjoy Indian classical music.






# What About You All?

Please go ahead and share:

1. Your name, and if you would like to share them, your pronouns
2. Where you're tuning in from: locally from the Bay Area, or farther away?
3. What you'd like to get out of this class.

If you aren't warmed up and comfortable with speaking just yet, that's fine! You can share directly to everyone in the chat or you can message me and I can read out your introduction for you!

Another question: what can I do to make you feel included? Please feel free to private message me in the chat if there's anything I can do to make you more comfortable.





# Breaking the Ice



- Share your names one more time!!!
- Icebreaker Question: What is your favorite home-cooked dish? Who makes it?
- If no one wants to share first, the person who is geographically closest to Stanford shares first!



# Lecture Review





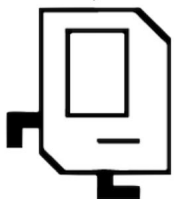
# What We've Learned

Before we get into our sample problem for today, let's review a bit. We've learned:

- The basics about Karel, the magnificent and wonderful robot
- Control Flow, loops and conditional statements which guide our programs
- Functions, a way of breaking down big problems into smaller chunks

This is a **LOT** of content, especially if you are newer to CS!

Hello, my name is Karel! Nice to meet you.



## For Loops

```
def main():  
    # repeats the body 99 times  
    for i in range(99):  
        # the "body"  
        put_beeper()
```



make\_dough()




shape\_pasta()



cook\_pasta()





The background is a solid orange color. It is decorated with various white and black geometric shapes and lines. In the top left, there is a dashed line and a small white triangle. In the top center, there is a dashed line and a white triangle. In the top right, there is a dashed line, a black zigzag line, a small white circle, and two parallel black lines. In the middle left, there is a white triangle. In the middle right, there is a white triangle and a large black arc. In the bottom left, there is a black plus sign. In the bottom center, there is a black circle and a white triangle. In the bottom right, there is a black arc and a large white circle.

**Let's review and  
refresh these  
concepts a bit!**

# Control Flow Overview



**for** loop:

- Performs some block of code, a specific amount of times.

**while** loop:

- Continuously perform a block of code until a given condition is evaluated to false.

**if** statement:

- Performs a block of code only when a condition is true, and only once.

**if-else** statement:


- Performs a block of code when a condition is true, and a different block when the condition evaluates to false. Each block is performed only once.

# for Loop




An example for-loop that you may see and use with Karel:

```
def turn_right():  
    for i in range(3):  
        turn_left()
```



This loop is also called a *definite loop* because we know where it ends, when *i* reaches 3.



# while Loop



An example while-loop that you may see and use with Karel:

```
def move_to_wall():  
    while front_is_clear():  
        move()
```



This loop is also called an *indefinite loop* because it will run until the associated condition becomes false.




# if Statement




An example if-statement that you may see and use with Karel:

```
def safe_move():  
    if front_is_clear():  
        move()
```



An if-statement runs code inside of it when the associated statement is evaluated to true.




# if-else Statement



An if-else statement runs the code inside the `if` block when the associated statement is evaluated to true, and the code inside the `else` block if the statement is evaluated to false. An example:

```
def safe_move_or_turn():  
    if front_is_clear():  
        move()  
    else:  
        turn_left()
```



We will get into more flexible statements later on in the course!




The background is a solid orange color. It is decorated with various hand-drawn geometric shapes in white and black. These include: a dashed line in the top left; a white triangle in the top center; a black zigzag line in the top right; a white circle in the top right; two parallel black lines in the top right; a white triangle in the top right; a large black circle in the bottom right; a white circle in the bottom right; a black plus sign in the bottom left; a white circle in the bottom center; a white triangle in the bottom center; and a black plus sign in the bottom center.

**Any Questions?**

# Functions



Given a problem to solve:

- Start with the big picture
  - Break the problem down into smaller, self-contained building blocks
    - These smaller building blocks are functions
    - The process of breaking down the problem into functions is **decomposition**
    - Any set of steps that will need to be repeated is a good candidate for a function
  - Assume the building blocks are done (use **pass** keyword)
  - Assemble building in **main** to solve the big problem
  - Implement each building block!
- 



# Functions



**Example:** Karel needs to walk from the first corner  $[1, 1]$  to the end of the row. Every time Karel lands on a beeper, it needs to spin  $360^\circ$ .

Big picture:

- Karel starts on  $[1, 1]$
- It moves until it is on a corner with a beeper.
- It spins, then moves forward again.
- This process continues until Karel reaches a wall.

What is the small building block that will be useful?

- What action does Karel not yet know how to do, but will need to do repeatedly?

# Functions

spin()



# Any Questions?



# Worked Example

Using functions and control flow  
to make Karel spin when it is on  
a beeper



# Solve the big problem



```
def main():  
    while front_is_clear():  
        move()  
        if beepers_present():  
            spin()
```

```
def spin():  
    pass      # Placeholder
```



# Implement the building blocks



```
def main():  
    while front_is_clear():  
        move()  
        if beepers_present():  
            spin()
```

```
def spin():  
    for i in range(4):  
        turn_left()
```



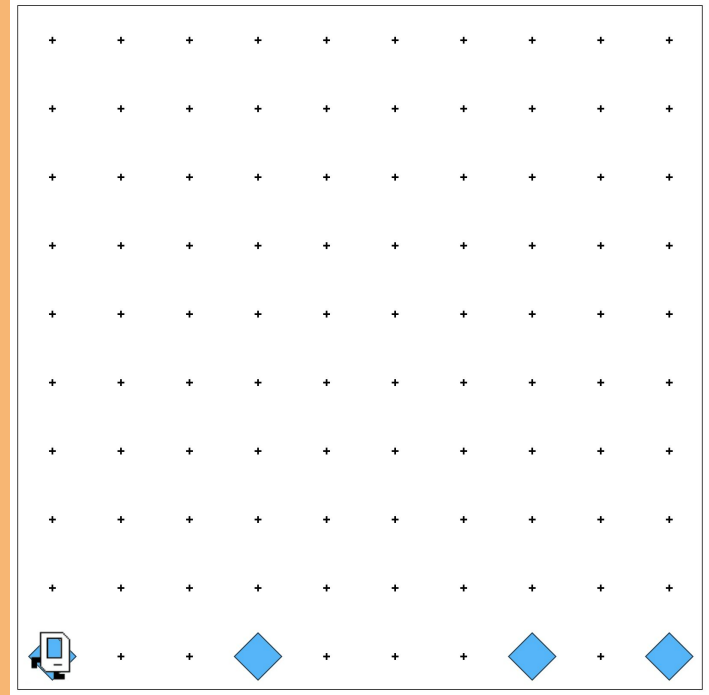
# Test and refine the entire solution

```
def main():  
    if beepers_present():          # Fencepost problem: there  
        spin()                    # could be a beeper at [1,1]  
    while front_is_clear():  
        move()  
        if beepers_present():  
            spin()  
  
def spin():  
    for i in range(4):  
        turn_left()
```

# spin() in action



You can see Karel spinning as desired [here](#). The implementation uses the code on the previous slide.





The background is a solid orange color. It is decorated with various hand-drawn geometric shapes in white and black. These include: a dashed line in the top left; a white triangle in the top center; a black zigzag line in the top right; a white circle in the top right; two parallel black lines in the top right; a white triangle in the top right; a large black circle on the right edge; a white triangle in the bottom left; a black plus sign in the bottom left; a white circle in the bottom center; a white triangle in the bottom center; a dashed line in the bottom center; a black plus sign in the bottom center; a black circle in the bottom center; a white circle in the bottom right; and a black circle in the bottom right.

**Any Questions?**



# Section Problem: Hospital Karel

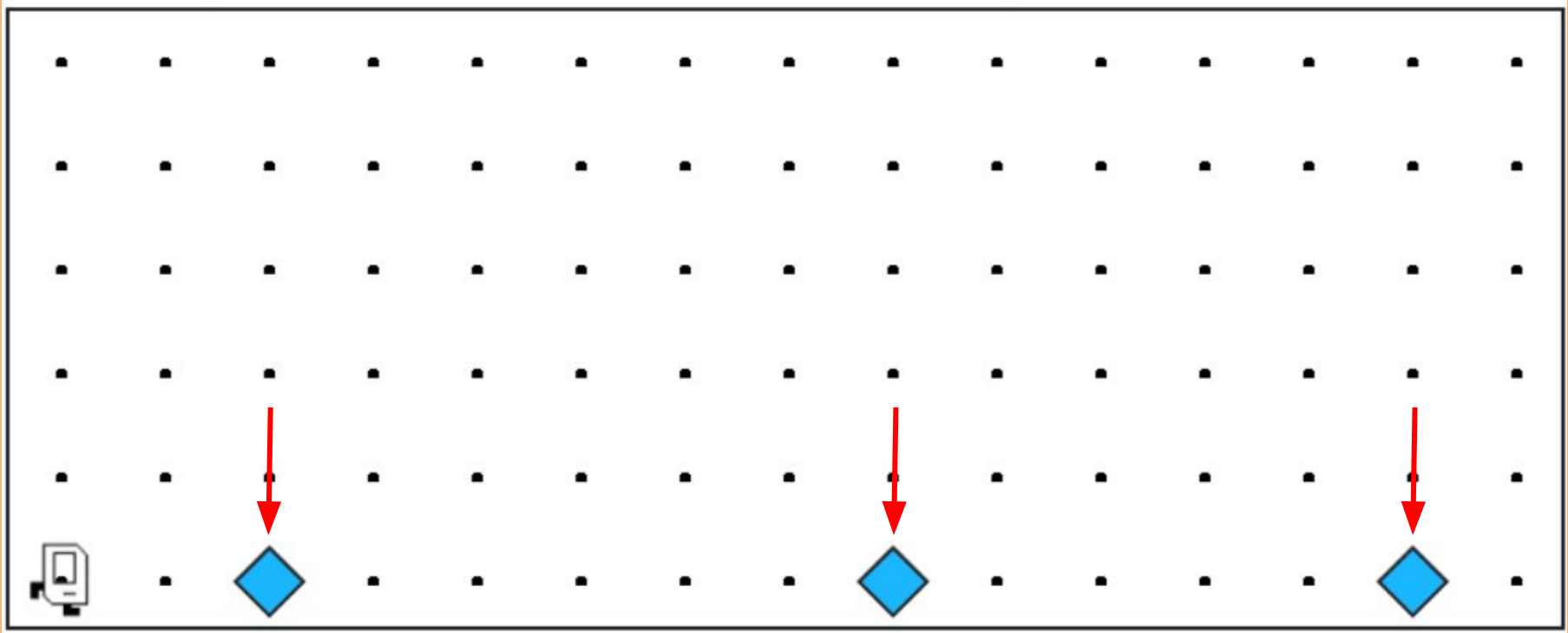


The background is a solid orange color. It is decorated with various hand-drawn geometric shapes in white and black. These include a dashed line in the top left, a white triangle in the top center, a black zigzag line in the top right, a white circle in the top right, two parallel black lines in the top right, a white triangle in the top left, a black plus sign in the bottom left, a white circle in the bottom center, a white triangle in the bottom center, a black plus sign in the bottom center, a black circle in the bottom center, and a white circle in the bottom right.

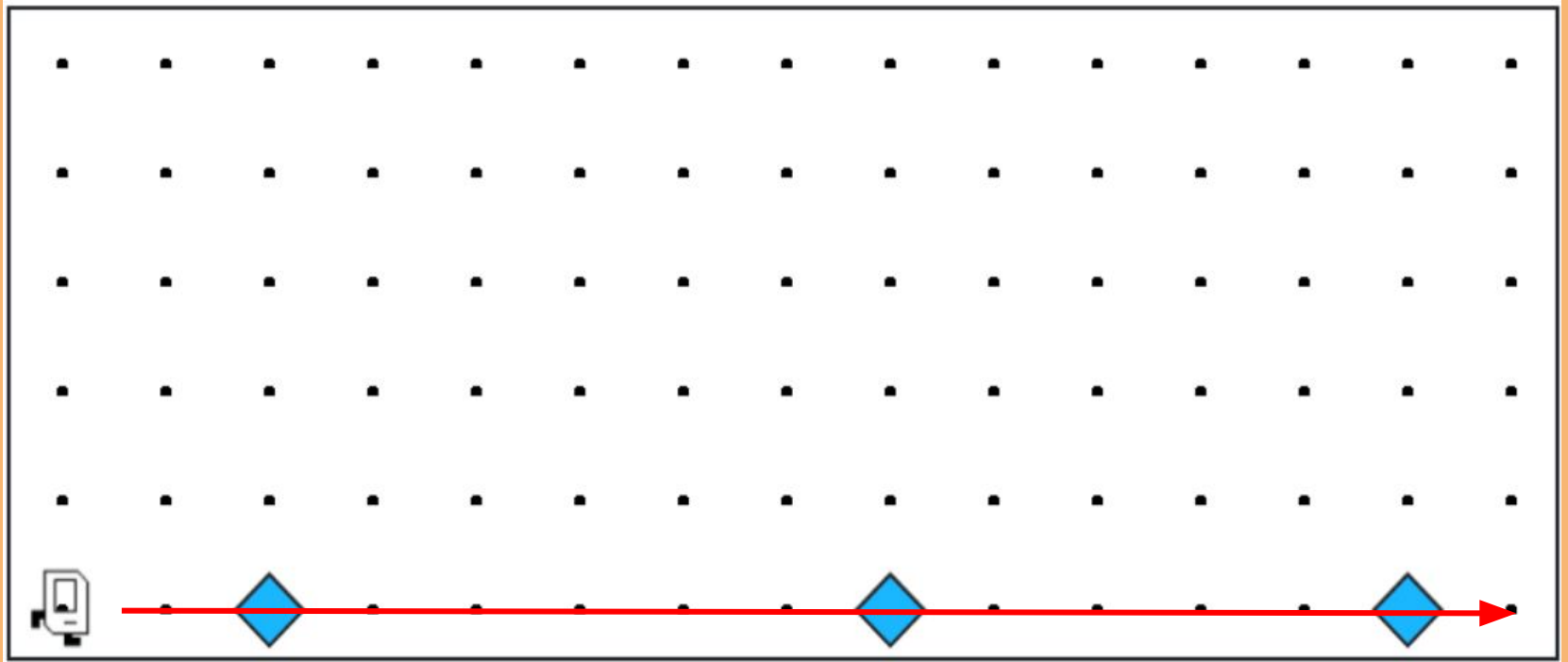
# Setting Context

Countries around the world are dispatching hospital-building robots to make sure anyone who gets sick can be treated. They have decided to enlist Karel robots. Your job is to program those robots.

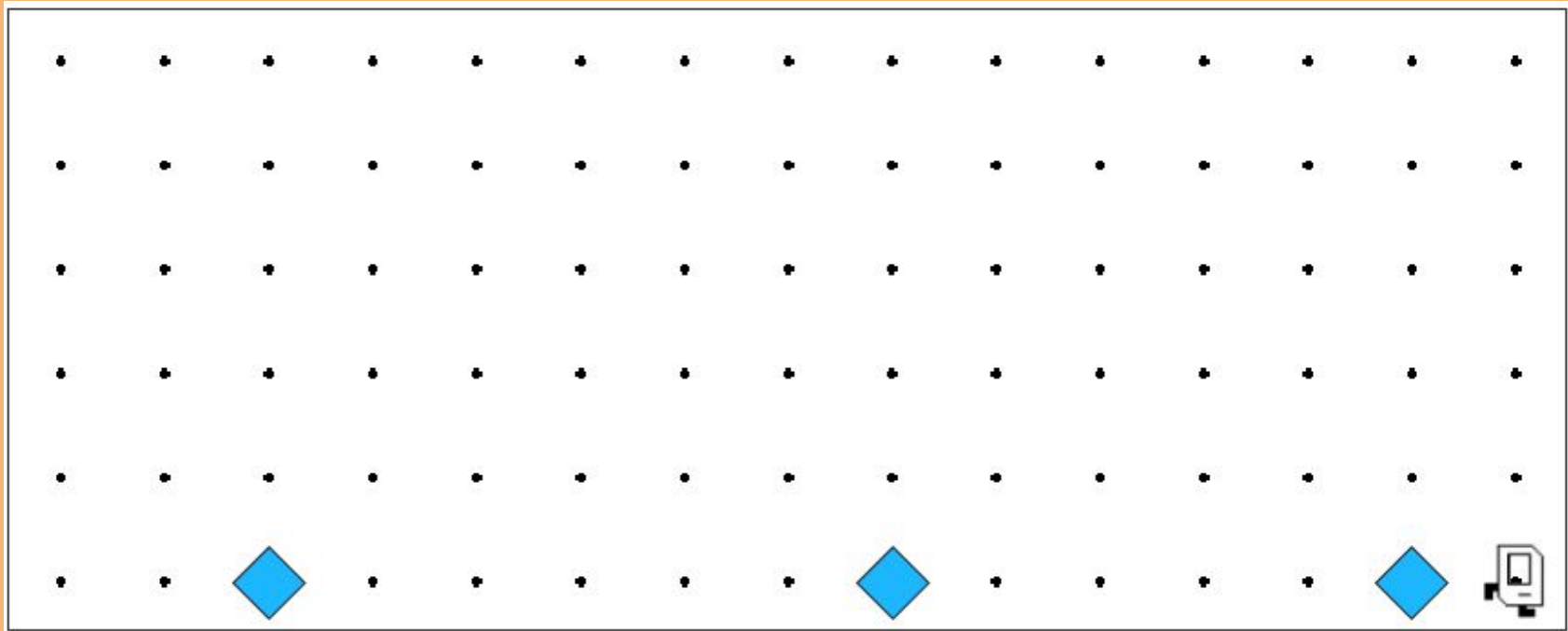
Each beeper in the figure represents a pile of supplies.



Karel's job is to walk along the row and build a new hospital in the places marked by each beeper.



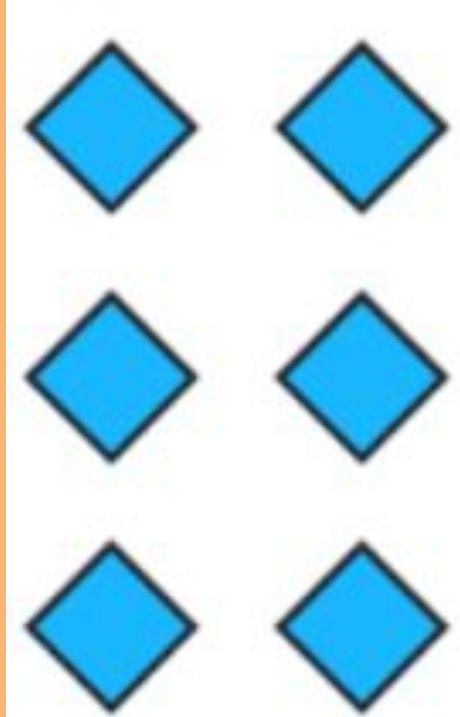
Karel's job is to walk along the row and build a new hospital in the places marked by each beeper.



Karel's job is to walk along the row and build a new hospital in the places marked by each beeper.

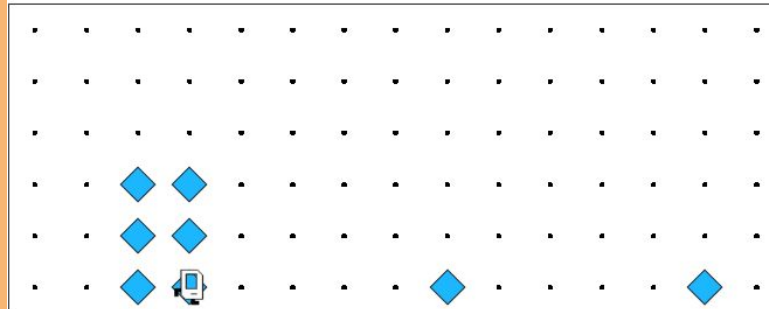
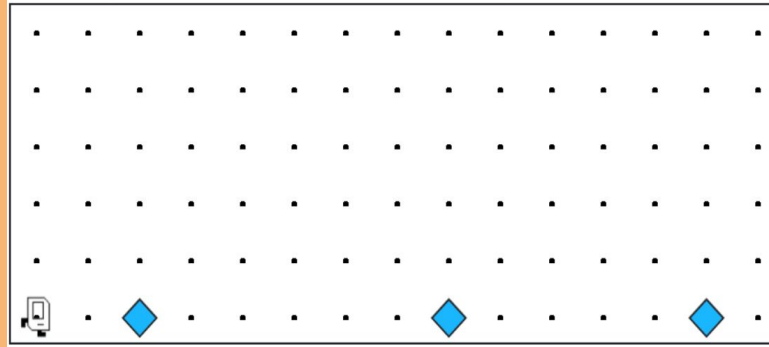


Hospitals look like this: a 3x2 rectangle of beepers!

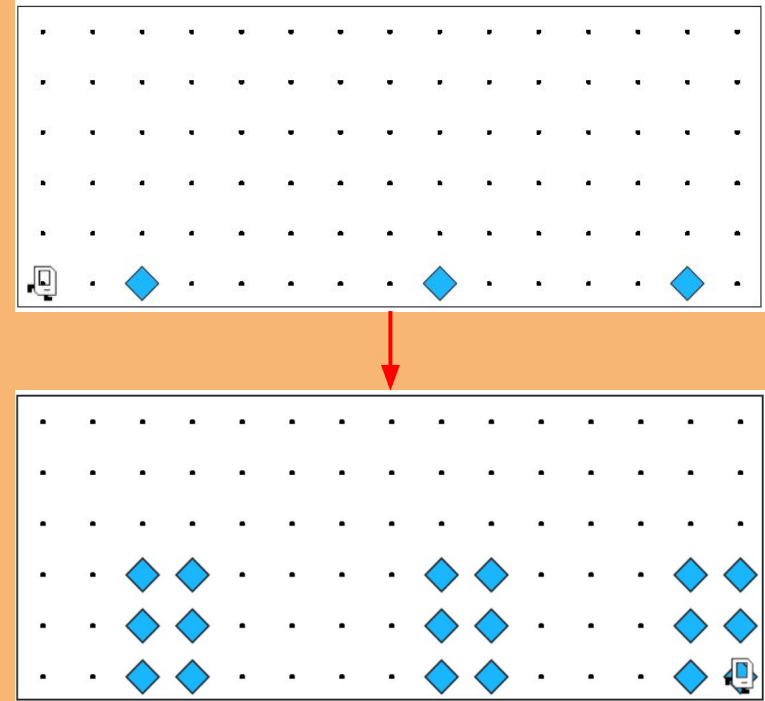




The new hospital should have their corner at the point at which the pile of supplies was left.



At the end of the run, Karel should be at the end of the row having created a set of hospitals. For the initial conditions shown, the result would look like this:





# Notes to Keep in Mind



- Karel starts facing east at  $[1, 1]$  with an infinite number of beepers in its beeper bag
- The beepers indicating the positions at which hospitals should be built will be spaced so that there is room to build the hospitals without overlapping or hitting walls
- There will be no supplies left on the last column
- Karel should not run into a wall if it builds a hospital that extends into that final corner.

The background is a solid orange color. It is decorated with various hand-drawn geometric shapes and lines in white and black. These include: a dashed line in the top left; a white triangle in the top center; a black zigzag line in the top right; a white circle in the top right; two parallel black lines in the top right; a white triangle in the top right; a black plus sign in the bottom left; a white circle in the bottom center; a white triangle in the bottom center; a black plus sign in the bottom center; a black circle in the bottom right; and a white circle in the bottom right.

# Questions Before We Begin?

The background is a solid pink color. It is decorated with various white and black geometric shapes and symbols. In the top left, there is a dashed white line and a solid white triangle. In the top center, there is a dashed white line and a solid white triangle. In the top right, there is a solid black zigzag line, a solid black circle, and two parallel solid black lines. In the bottom left, there is a solid black plus sign. In the bottom center, there is a solid black circle and a solid white triangle. In the bottom right, there is a solid black circle and a solid white circle. The text "Let's get to work!" is centered in the middle of the image in a white, sans-serif font.

Let's get to work!