Control Flow

Chris Piech

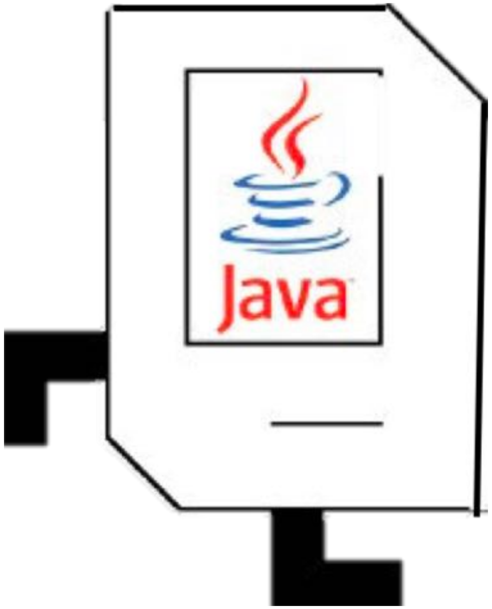
CS106A, Stanford University

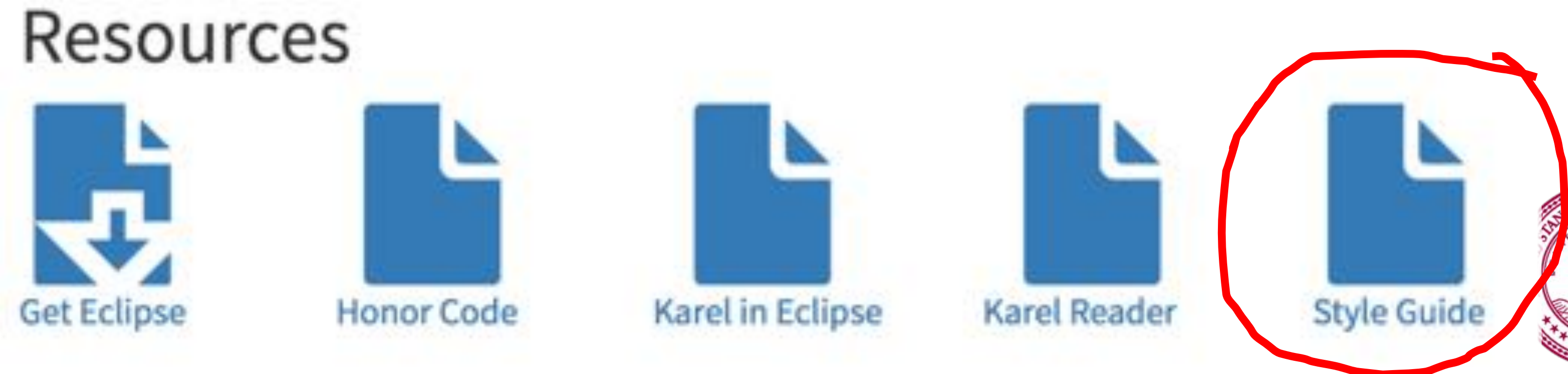
Install Eclipse

Special Eclipse office hours
LAIR, Tonight 7-9pm

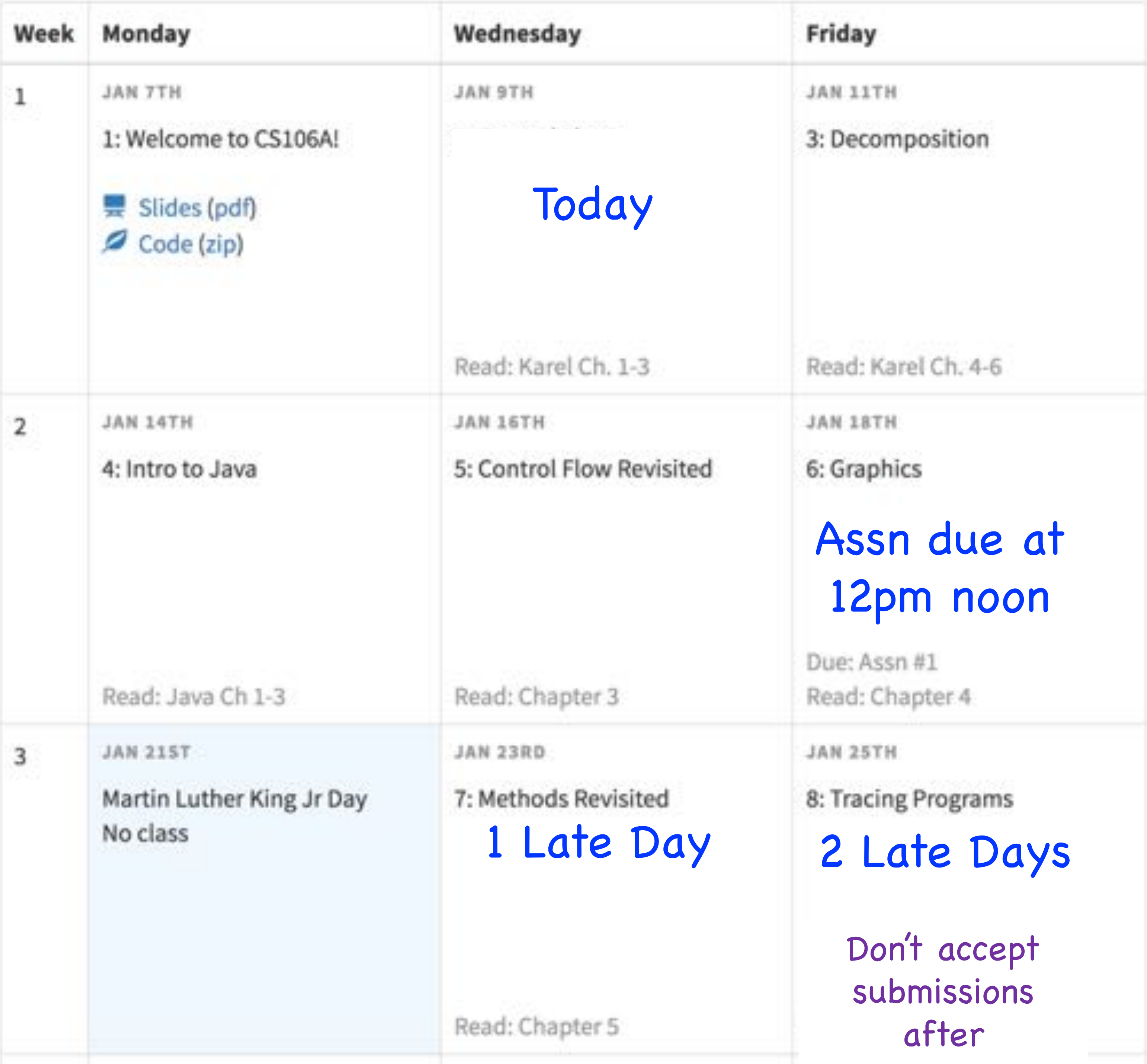
* Bottom floor of Tresidder

Assignment 1



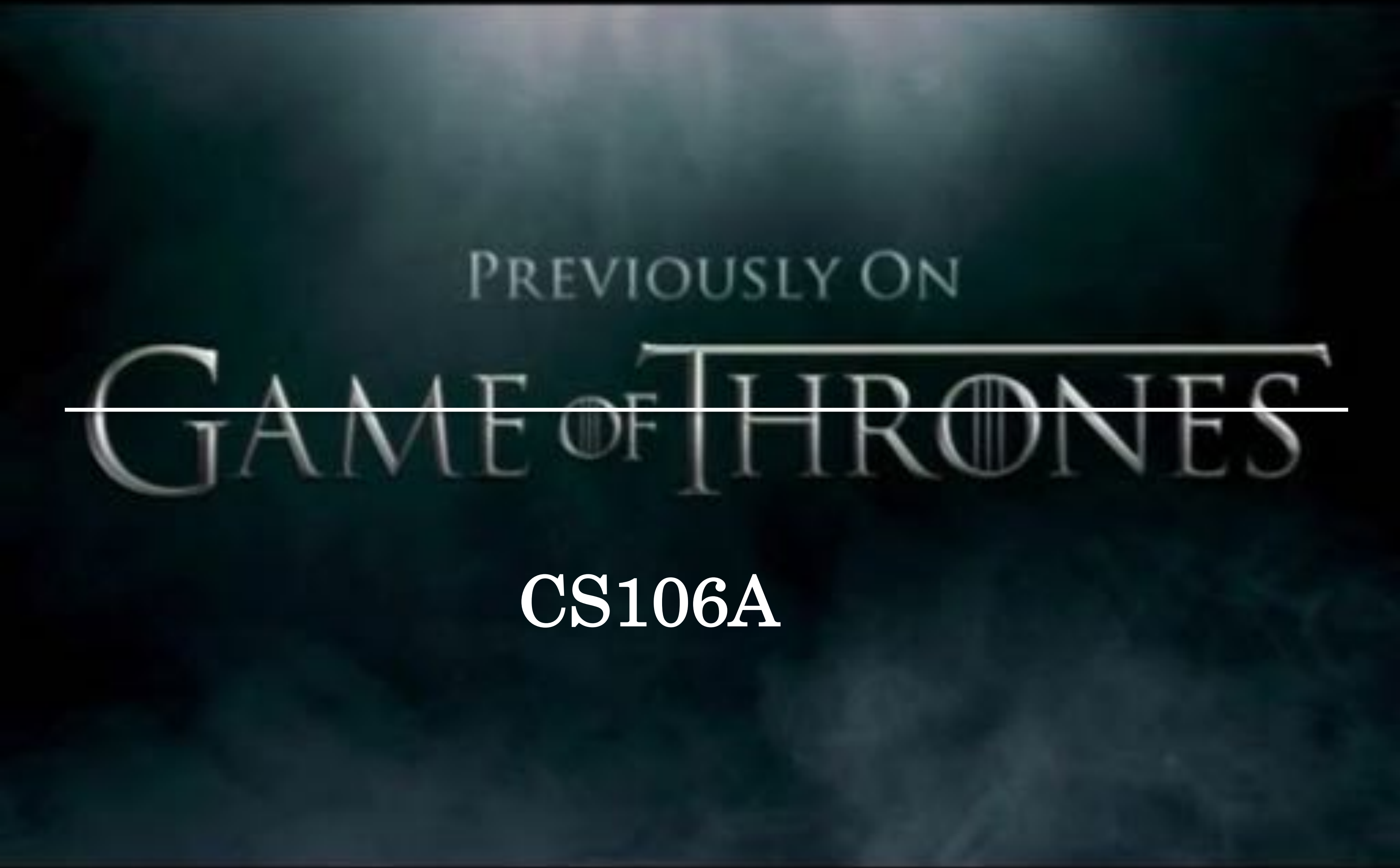


Late Days



Week	Monday	Wednesday	Friday
1	JAN 7TH 1: Welcome to CS106A! Slides (pdf) Code (zip)	JAN 9TH Today Read: Karel Ch. 1-3	JAN 11TH 3: Decomposition Read: Karel Ch. 4-6
2	JAN 14TH 4: Intro to Java Read: Java Ch 1-3	JAN 16TH 5: Control Flow Revisited Read: Chapter 3	JAN 18TH 6: Graphics Assn due at 12pm noon Due: Assn #1 Read: Chapter 4
3	JAN 21ST Martin Luther King Jr Day No class	JAN 23RD 7: Methods Revisited 1 Late Day Read: Chapter 5	JAN 25TH 8: Tracing Programs 2 Late Days Don't accept submissions after

Honor Code

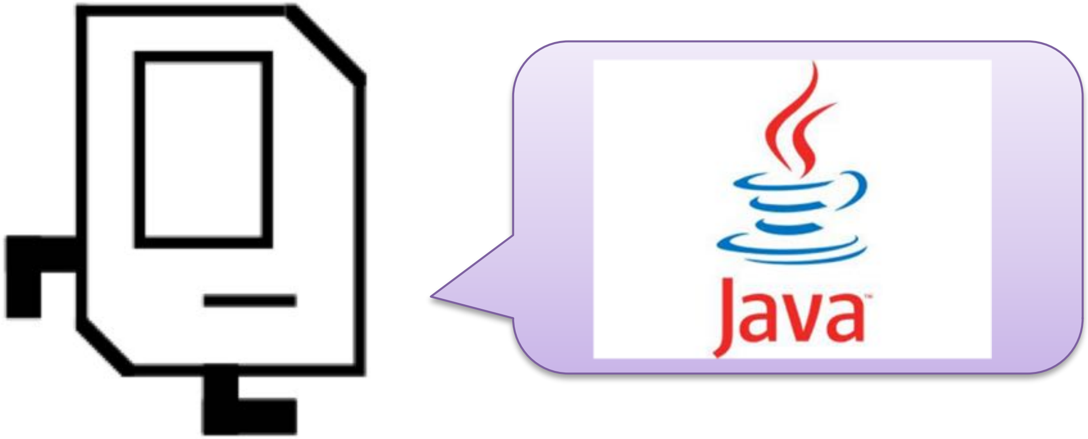


PREVIOUSLY ON

~~GAME OF THRONES~~

CS106A

Karel the Robot



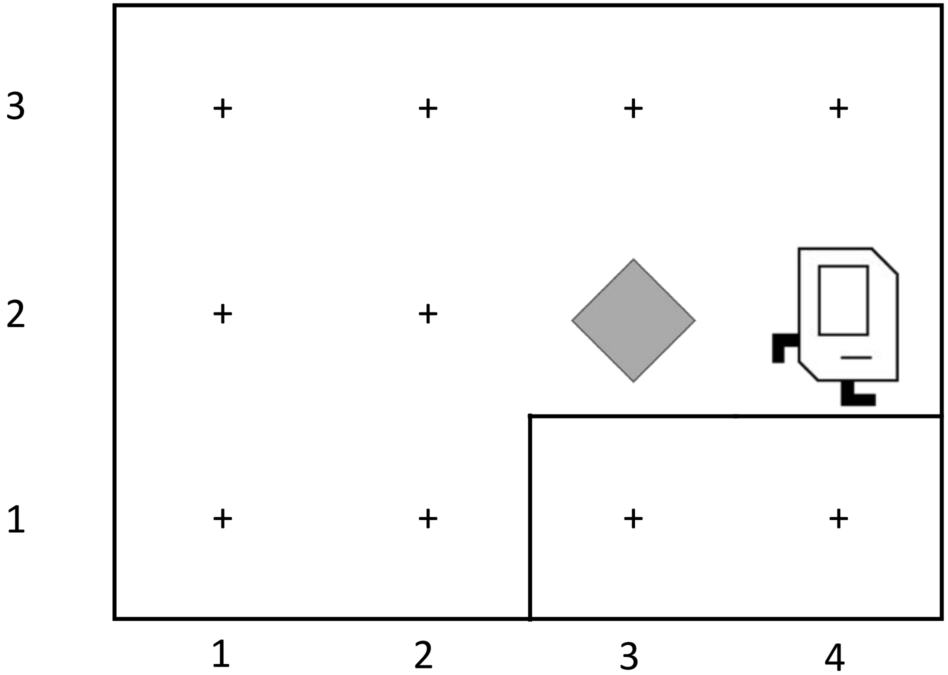
* While Karel is in Java, when you program your Karel assignment we ask that you stick to the concepts in the course reader

Method Definition

This adds a new command to Karels vocabulary

```
private void name() {
    statements in the method body
}
```

First Challenge



Anatomy of a Program

```
import stanford.karel.*;

public class OurKarelProgram extends Karel {
    public void run() {
        move();
        pickBeeper();
        move();
        turnLeft();
        move();
        turnRight();
        move();
        putBeeper();
        move();
    }

    private void turnRight() {
        turnLeft();
        turnLeft();
        turnLeft();
    }
}
```

This piece of the program's ***source code*** is called a ***method***.

Anatomy of a Program

```
import stanford.karel.*;

public class OurKarelProgram extends Karel {
    public void run() {
        move();
        pickBeeper();
        move();
        turnLeft();
        move();
        turnRight();
        move();
        putBeeper();
        move();
    }

    private void turnRight() {
        turnLeft();
        turnLeft();
        turnLeft();
    }
}
```

This line of code gives the *name* of the method (here, turnRight)

Anatomy of a Program

```
import stanford.karel.*;

public class OurKarelProgram extends Karel {
    public void run() {
        move();
        pickBeeper();
        move();
        turnLeft();
        move();
        turnRight();
        move();
        putBeeper();
        move();
    }

    private void turnRight() {
        turnLeft();
        turnLeft();
        turnLeft();
    }
}
```

This is called a ***code block***

Anatomy of a Program

```
import stanford.karel.*;

public class OurKarelProgram extends Karel {
    public void run() {
        move();
        pickBeeper();
        move();
        turnLeft();
        move();
        turnRight();
        move();
        putBeeper();
        move();
    }

    private void turnRight() {
        turnLeft();
        turnLeft();
        turnLeft();
    }
}
```

Anatomy of a Program

```java
import stanford.karel.*;

public class OurKarelProgram extends Karel {
    public void run() {
        move();
        pickBeeper();
        move();
        turnLeft();
        move();
        turnRight();
        move();
        putBeeper();
        move();
    }

    private void turnRight() {
        turnLeft();
        turnLeft();
        turnLeft();
    }
}
```

Anatomy of a Program

```
import stanford.karel.*;

public class OurKarelProgram extends Karel {
    public void run() {
        move();
        pickBeeper();
        move();
        turnLeft();
        move();
        turnRight();
        move();
        putBeeper();
        move();
    }

    private void turnRight() {
        turnLeft();
        turnLeft();
        turnLeft();
    }
}
```

Anatomy of a Program

```
import stanford.karel.*;

public class OurKarelProgram extends Karel {
    public void run() {
        move();
        pickBeeper();
        move();
        turnLeft();
        move();
        turnRight();
        move();
        putBeeper();
        move();
    }

    private void turnRight() {
        turnLeft();
        turnLeft();
        turnLeft();
    }
}
```

Anatomy of a Program

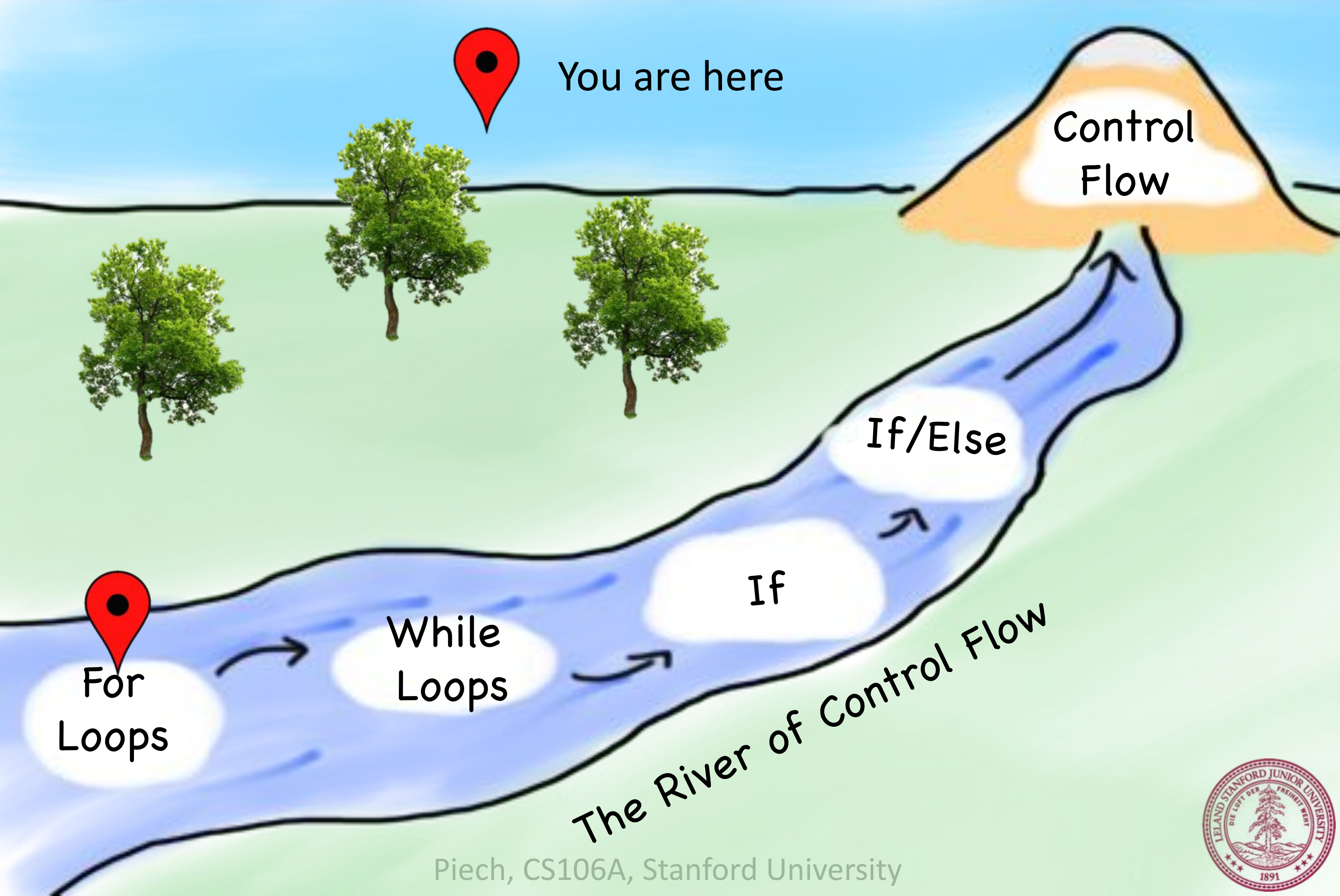
```
import stanford.karel.*;

public class OurKarelProgram extends Karel {
    public void run() {
        move();
        pickBeeper();
        move();
        turnLeft();
        move();
        turnRight();
        move();
        putBeeper();
        move();
    }

    private void turnRight() {
        turnLeft();
        turnLeft();
        turnLeft();
    }
}
```

Methods are defined one after another

Today's Goal

1. Code using loops and conditions
2. Trace programs that use loops and conditions

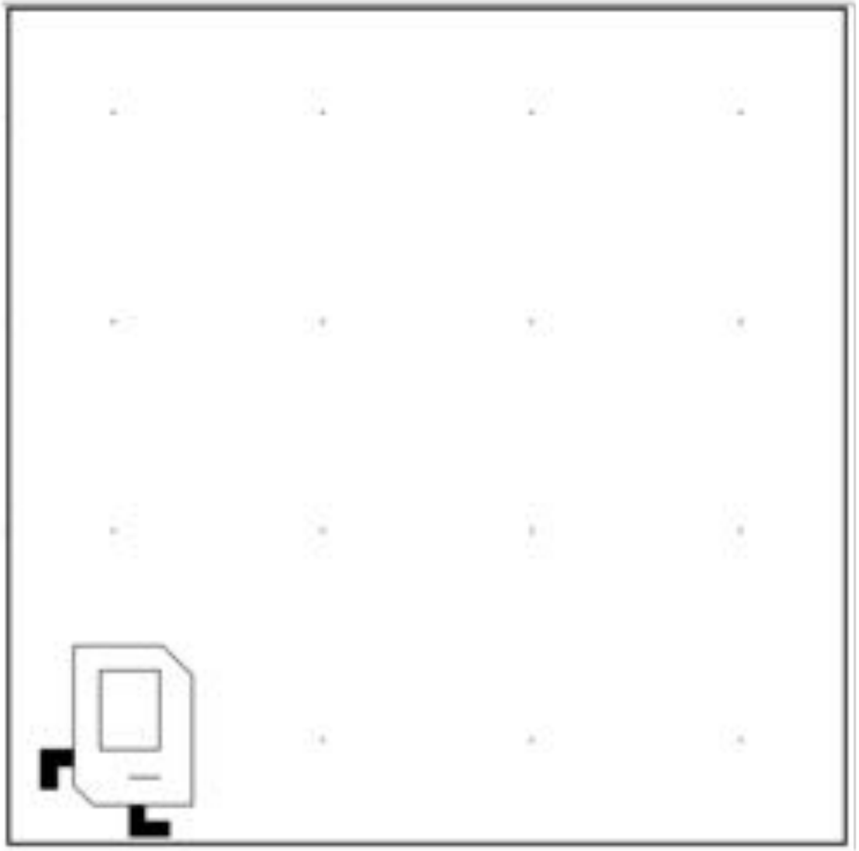
Today's Route



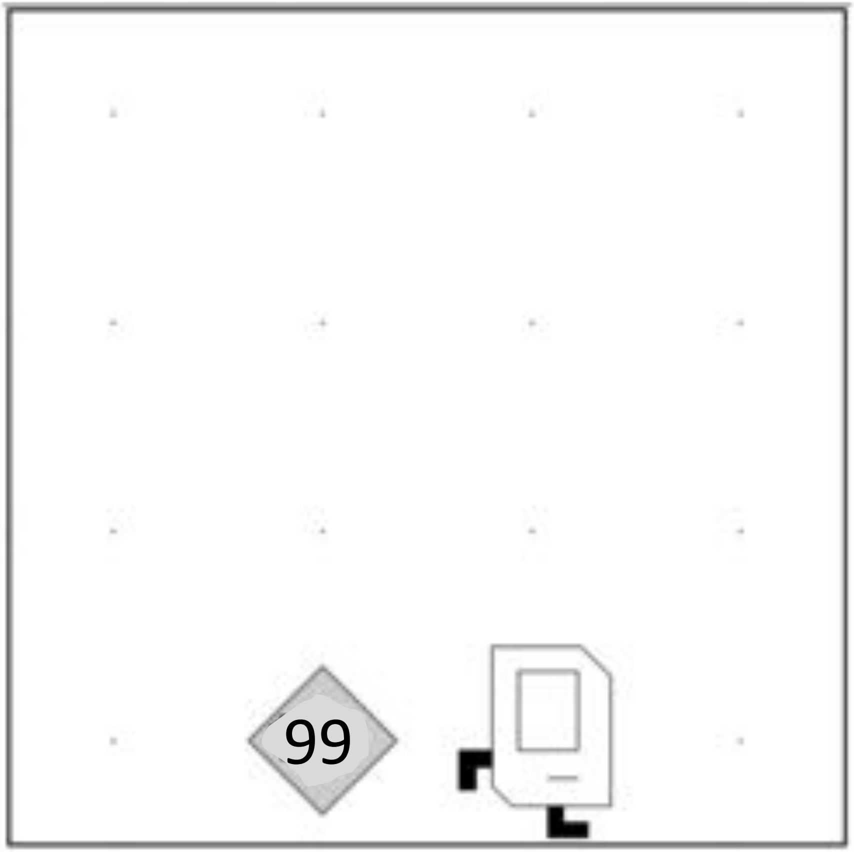
For loops,
While loops,
If/Else statements

Place 99 beepers?

Before



After



Place 99 beepers

```
public class Place99Beepers extends SuperKarel {
    public void run() {
        move();
        repeat(99) {
            putBeeper();
        }
        move();
    }
}
```

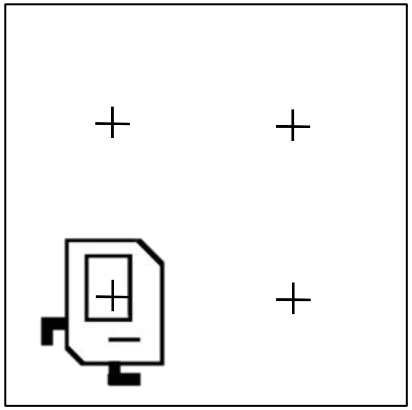
Place 99 beepers

```
public class Place99Beepers extends SuperKarel {
    public void run() {
        move();
        for(int i = 0; i < 99; i++) {
            putBeeper();
        }
        move();
    }
}
```

This "for loop" repeats the code in its "body" 99 times

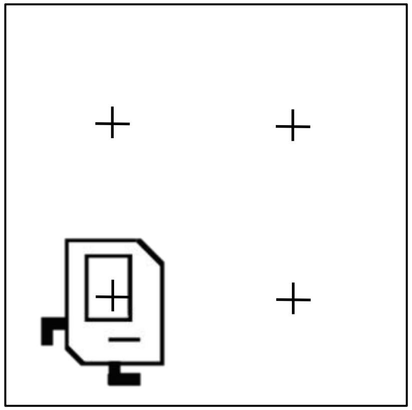
Place Beeper Square

```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



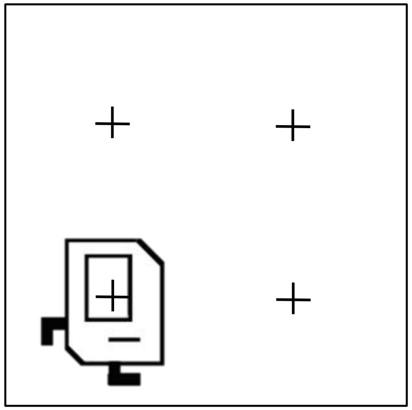
Place Beeper Square

```
public class BeeperSquare extends SuperKarel {
   public void run() {
      for(int i = 0; i < 4; i++) {
         putBeeper();
         move();
         turnLeft();
      }
   }
}
```



Place Beeper Square

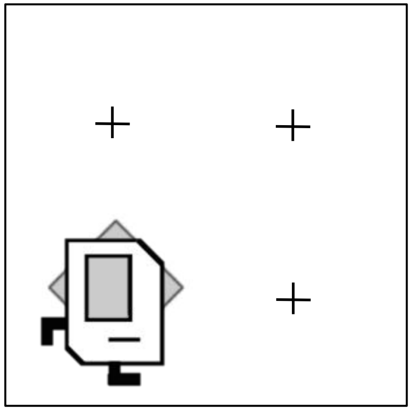
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



First time through the loop

Place Beeper Square

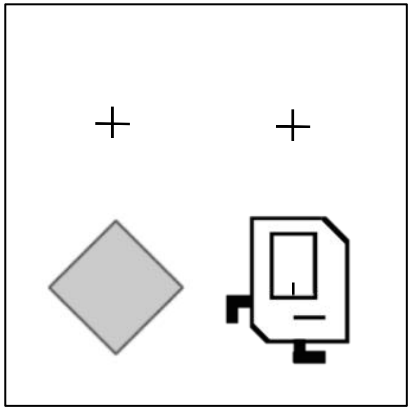
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



First time through the loop

Place Beeper Square

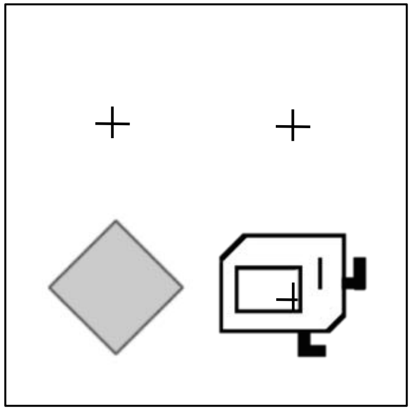
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



First time through the loop

Place Beeper Square

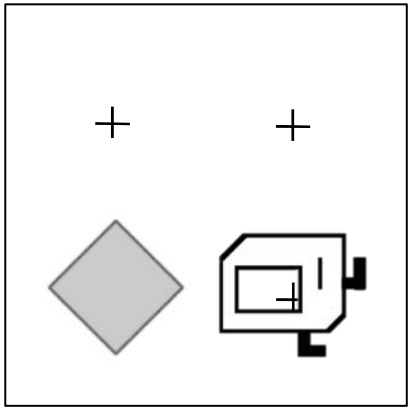
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



First time through the loop

Place Beeper Square

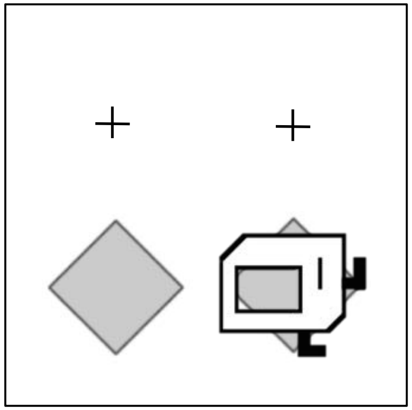
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



Second time through the loop

Place Beeper Square

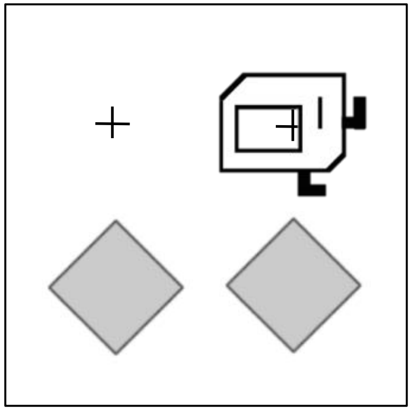
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



Second time through the loop

Place Beeper Square

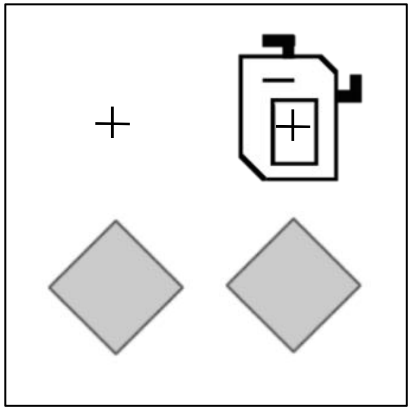
```
public class BeeperSquare extends SuperKarel {
   public void run() {
      for(int i = 0; i < 4; i++) {
         putBeeper();
         move();
         turnLeft();
      }
   }
}
```



Second time through the loop

Place Beeper Square

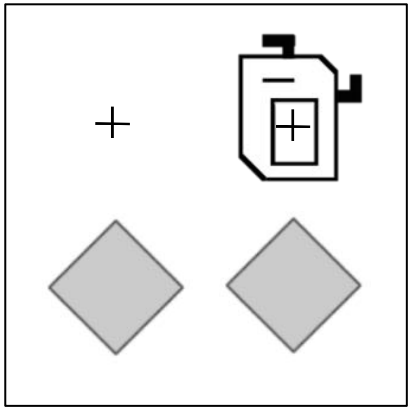
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



Second time through the loop

Place Beeper Square

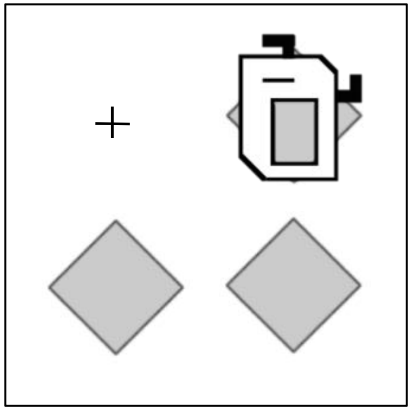
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



Third time through the loop

Place Beeper Square

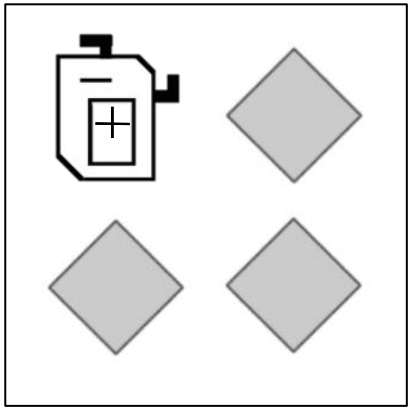
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



Third time through the loop

Place Beeper Square

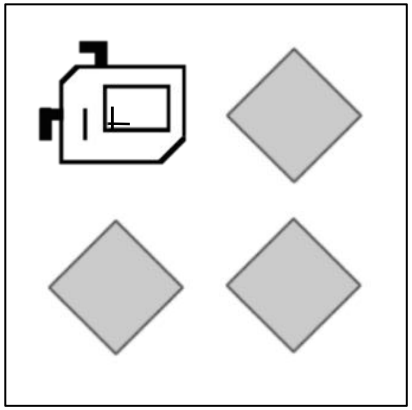
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



Third time through the loop

Place Beeper Square

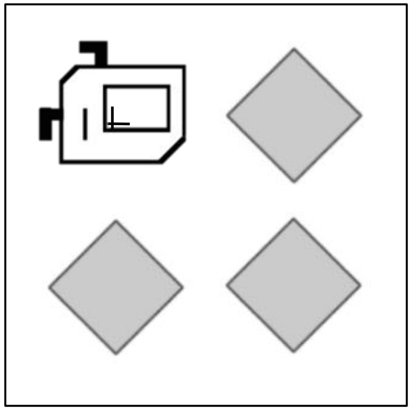
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



Third time through the loop

Place Beeper Square

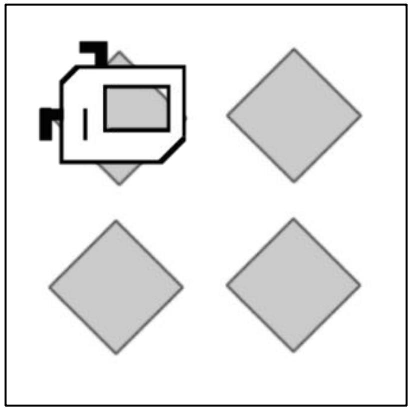
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



Fourth time through the loop

Place Beeper Square

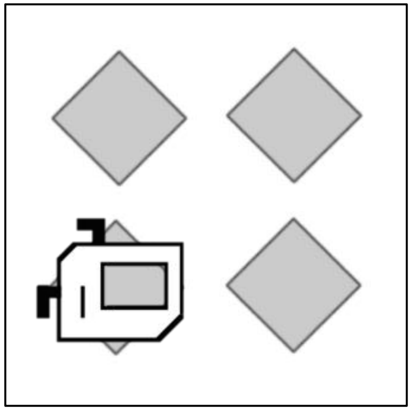
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



Fourth time through the loop

Place Beeper Square

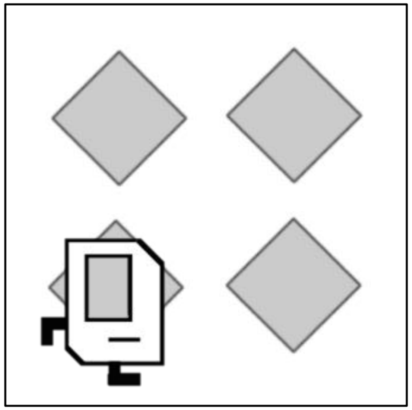
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



Fourth time through the loop

Place Beeper Square

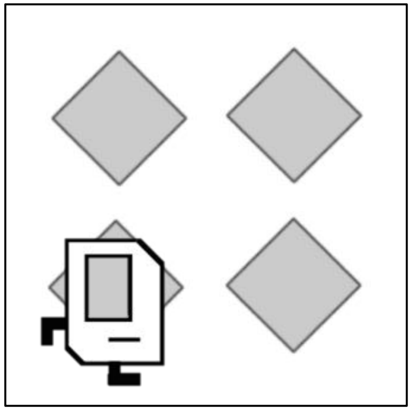
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



Fourth time through the loop

Place Beeper Square

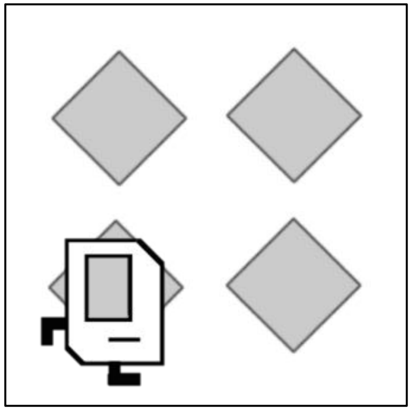
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



Fourth time through the loop

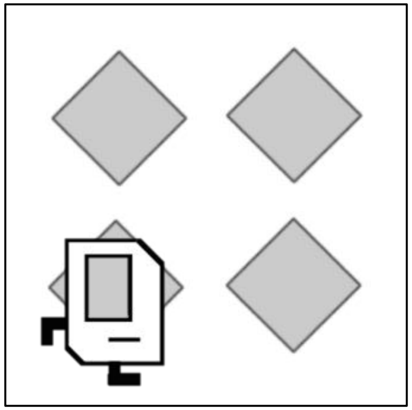
Place Beeper Square

```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



What if you remove the turnLeft?

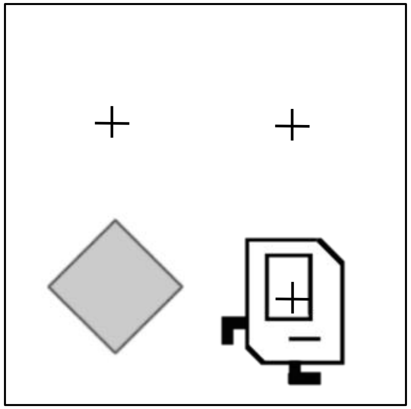
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```



What if you remove the turnLeft?

```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            // no turn left
        }
    }
}
```

After first iteration:

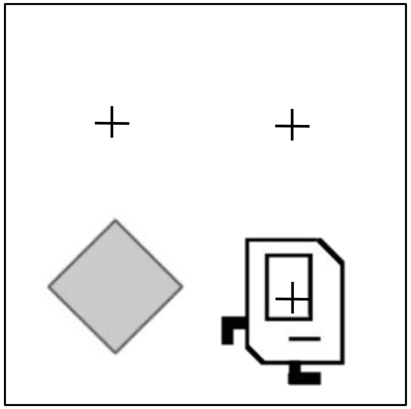


You need the **postcondition** of a loop to match the **precondition**

What if you remove the turnLeft?

```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            // precondition
            putBeeper();
            move();
            // postcondition
        }
    }
}
```

After first iteration:

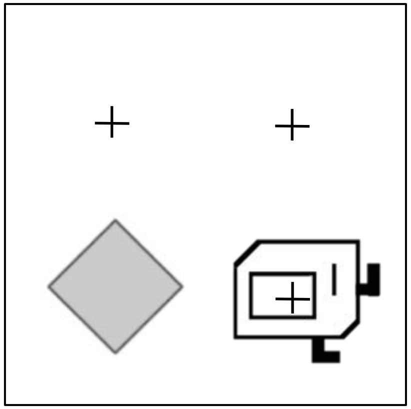


You need the **postcondition** of a loop to match the **precondition**

Place Beeper Square

```
public class BeeperSquare extends SuperKarel {
   public void run() {
      for(int i = 0; i < 4; i++) {
         putBeeper();
         move();
         turnLeft();
      }
   }
}
```

After first iteration:



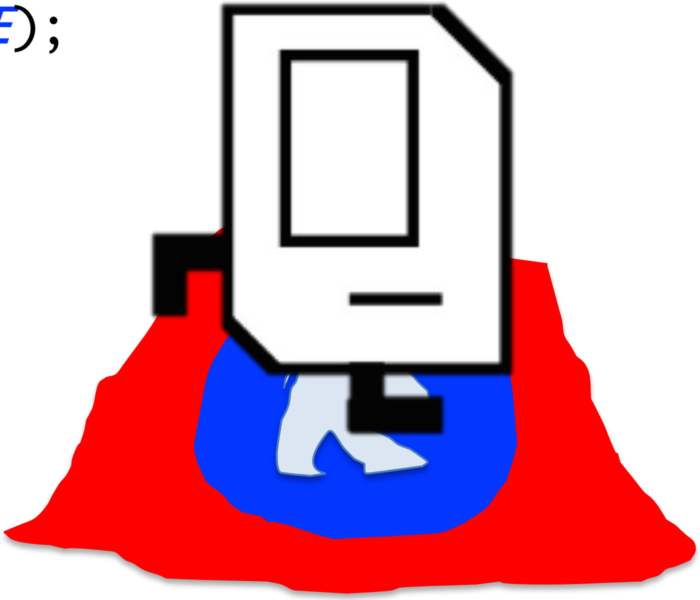
Exciting!

Aside: Super Karel

```
public class BeeperSquare extends SuperKarel {
    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
            turnLeft();
        }
    }
}
```

Aside: Super Karel

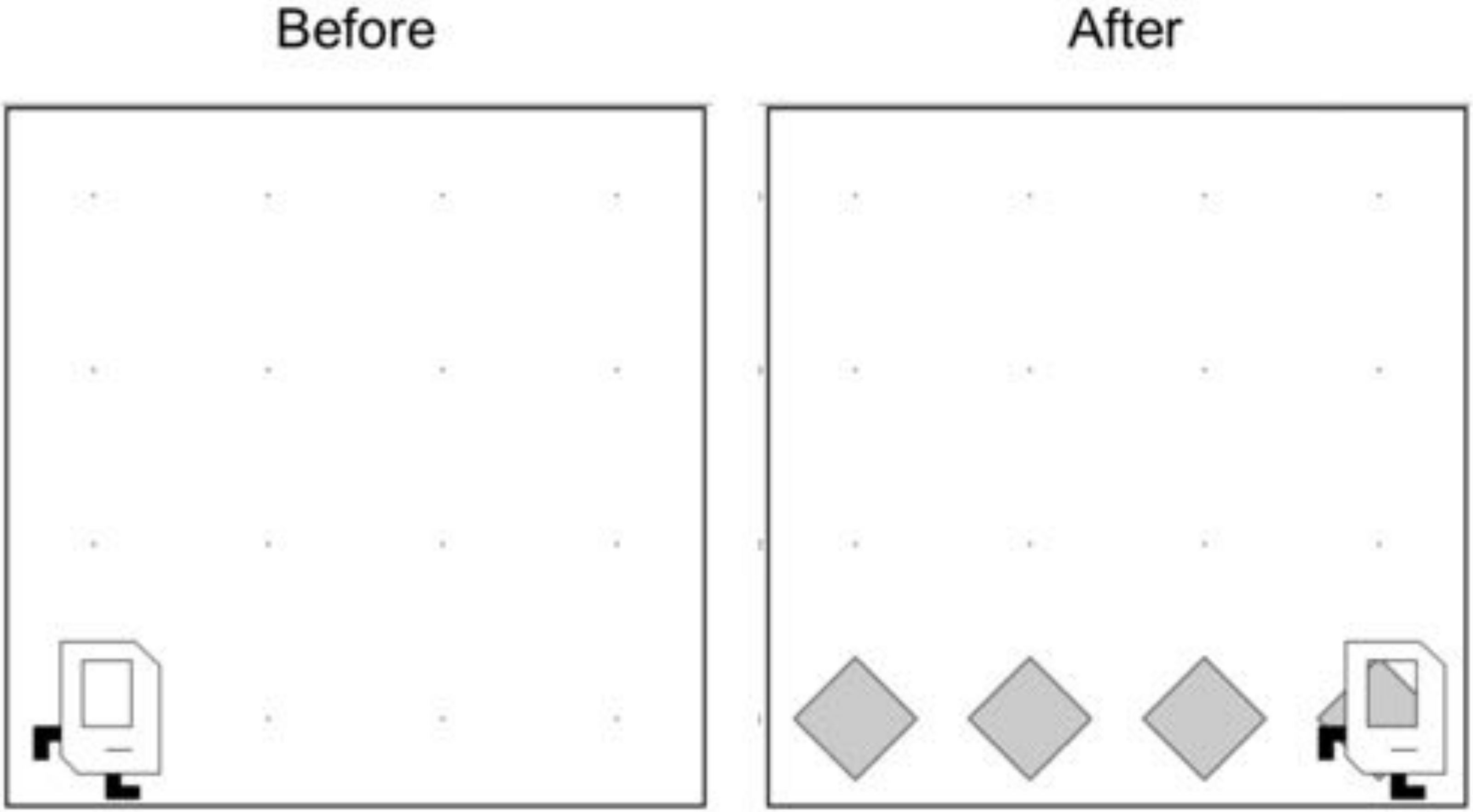
```
public class BeeperSquare extends SuperKarel {
    public void run() {
        // super karel has a few more commands
        turnRight();
        turnAround();
        paintCorner(BLUE);

        putBeeper();
        move();
    }
}
```



Next task

Place Beeper Line

Try and solve it!

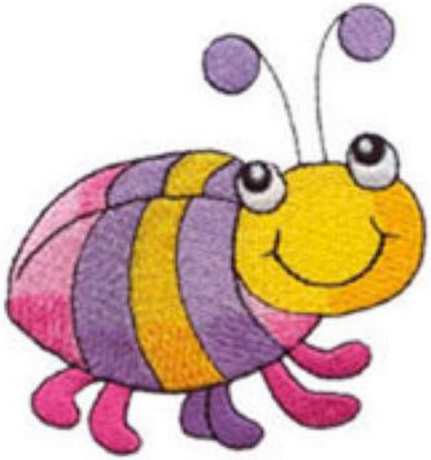


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

   public void run() {
      for(int i = 0; i < 4; i++) {
         putBeeper();
         move();
      }
   }
}
```

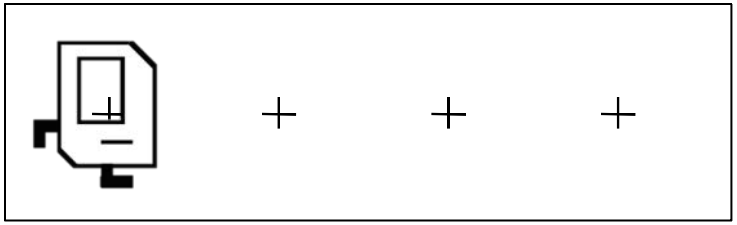
Place Beeper Line

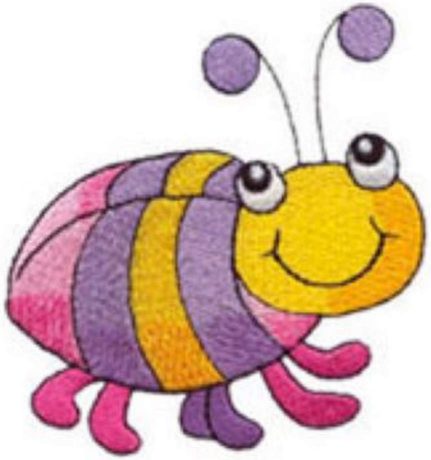
```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
        }
    }
}
```



Place Beeper Line

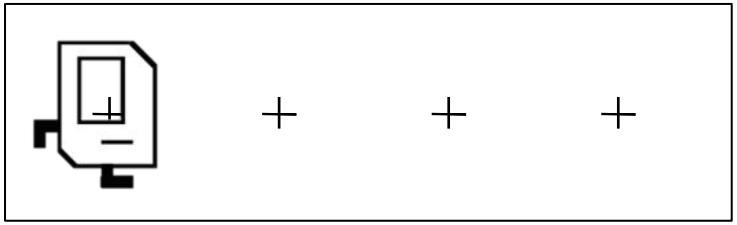
```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
       for(int i = 0; i < 4; i++) {
          putBeeper();
          move();
       }
    }
}
```

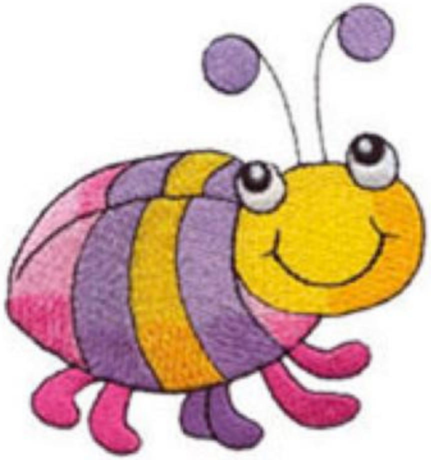




Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
        }
    }
}
```

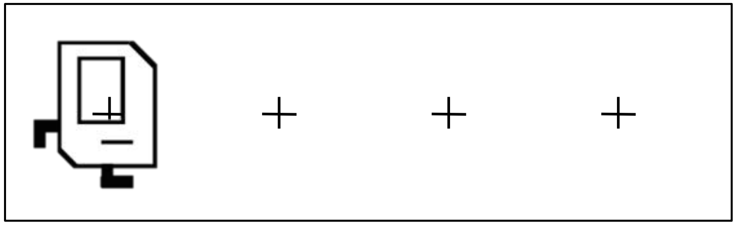


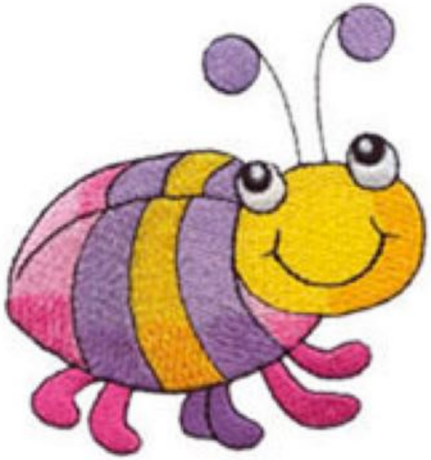


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
        }
    }
}
```

1st time through loop

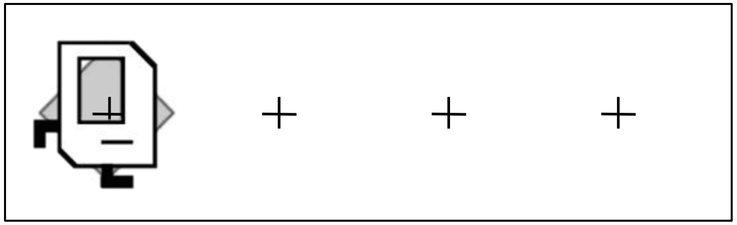


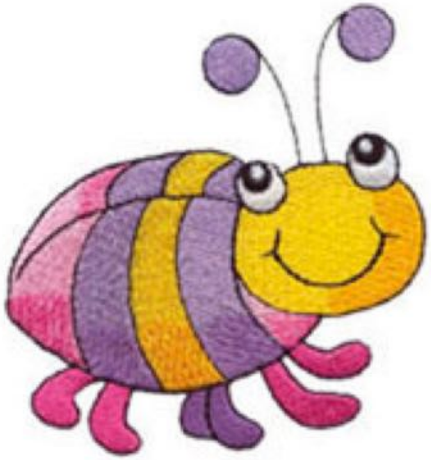


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
       for(int i = 0; i < 4; i++) {
          putBeeper();
          move();
       }
    }
}
```

1st time through loop

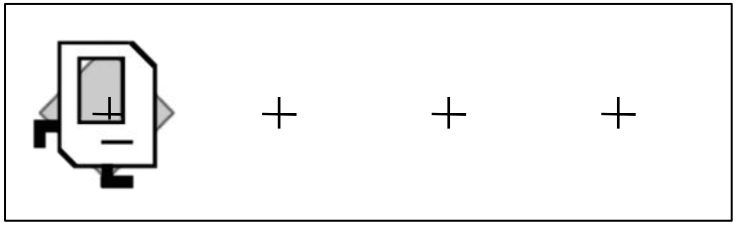


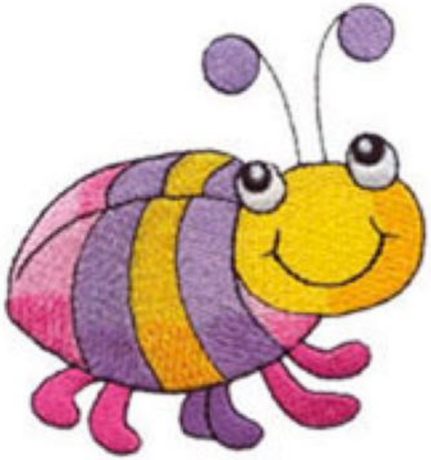


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
        }
    }
}
```

1st time through loop

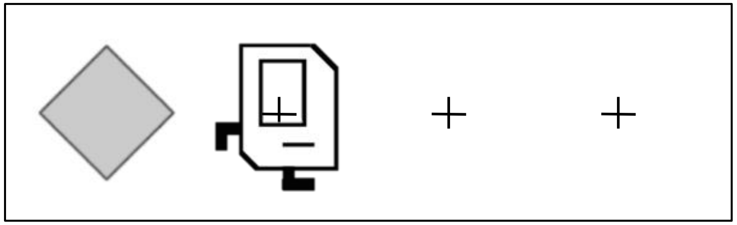


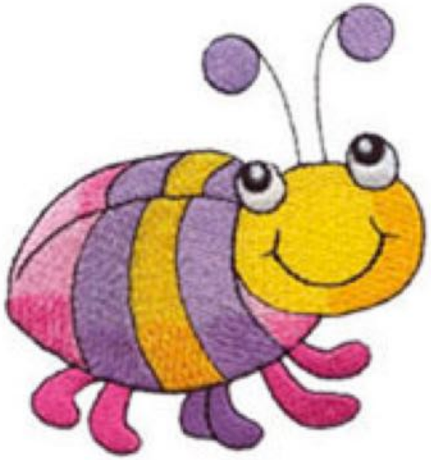


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
       for(int i = 0; i < 4; i++) {
          putBeeper();
          move();
       }
    }
}
```

2nd time through loop

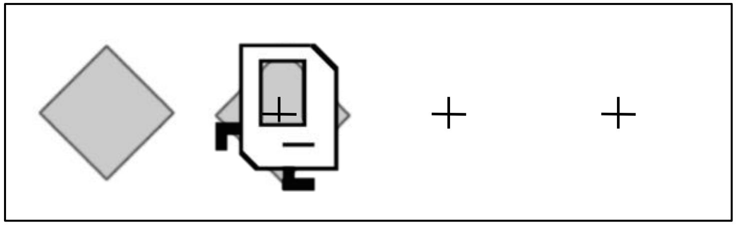


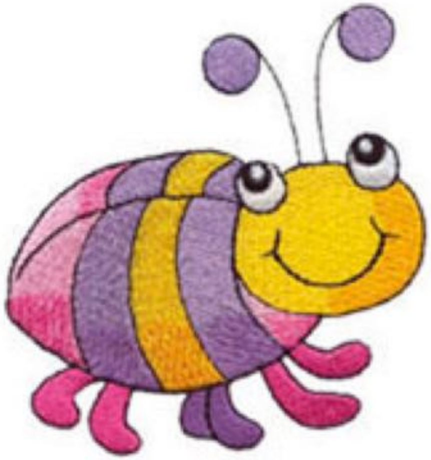


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
        }
    }
}
```

2nd time through loop

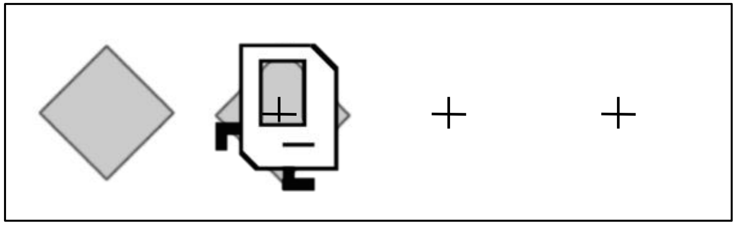


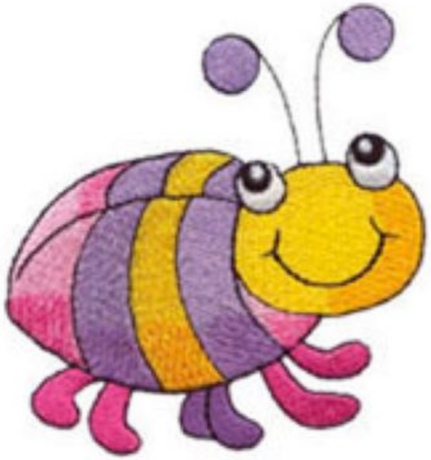


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
        }
    }
}
```

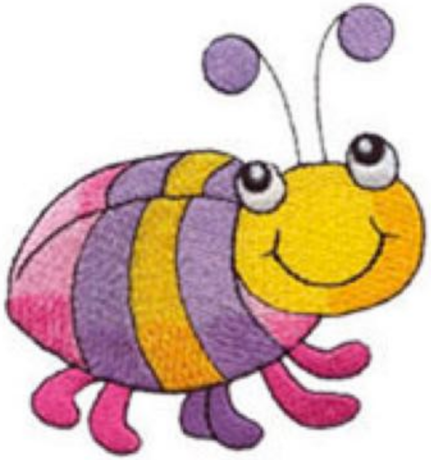
2nd time through loop



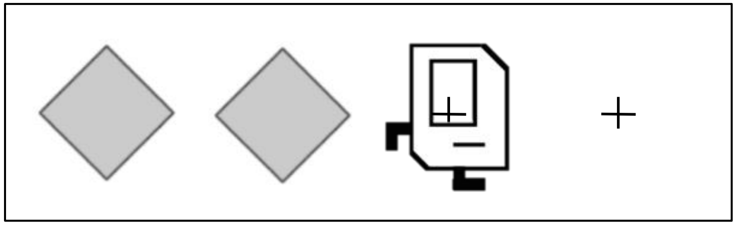


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
        }
    }
}
```



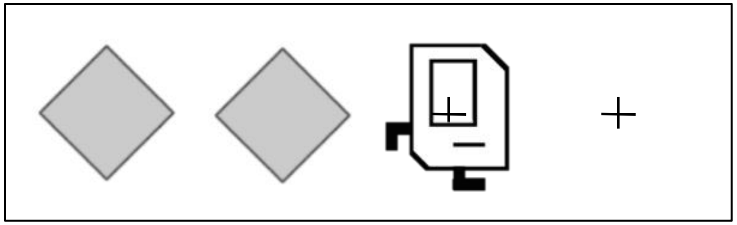
2nd time through loop

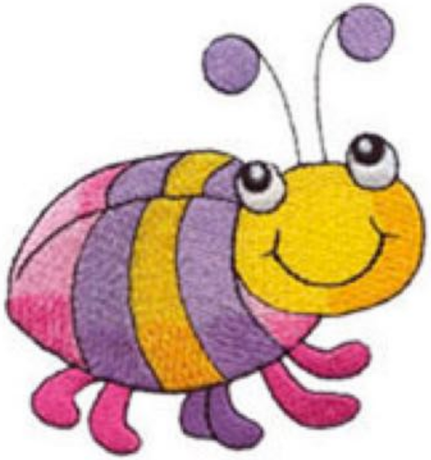


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

      public void run() {
         for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
         }
      }
}
```

3rd time through loop

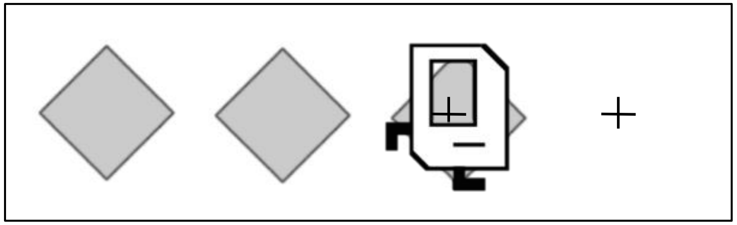


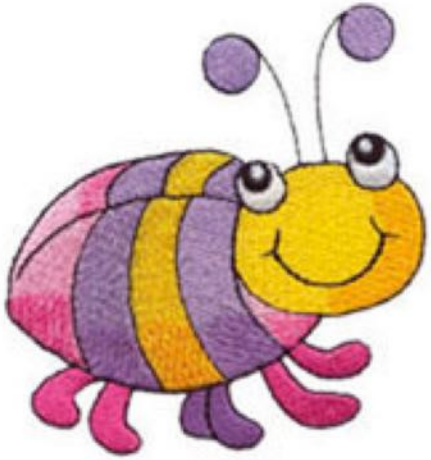


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
        }
    }
}
```

3rd time through loop

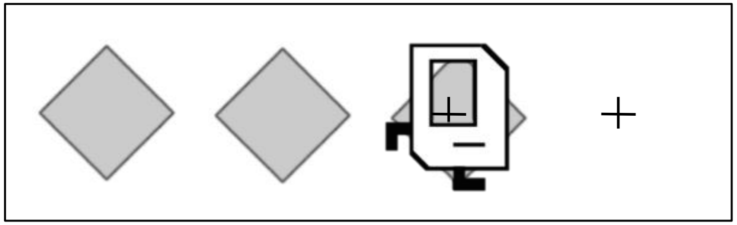


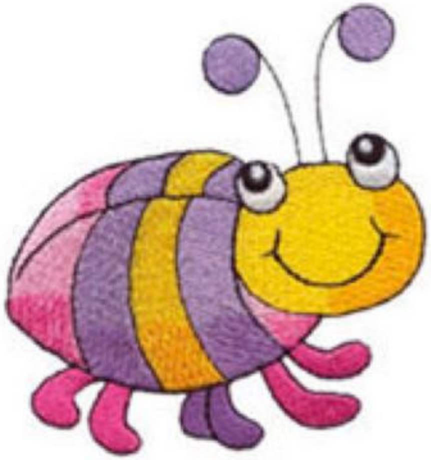


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
        }
    }
}
```

3rd time through loop

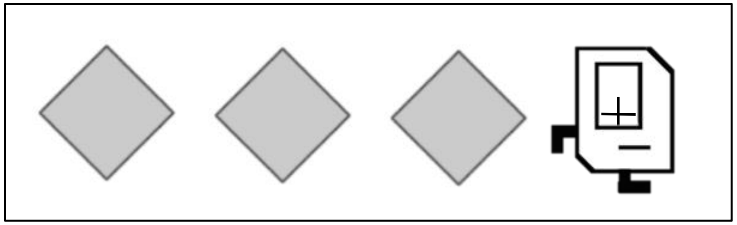


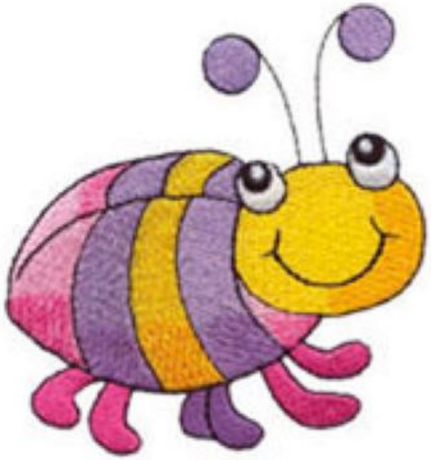


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
        }
    }
}
```

4th time through loop

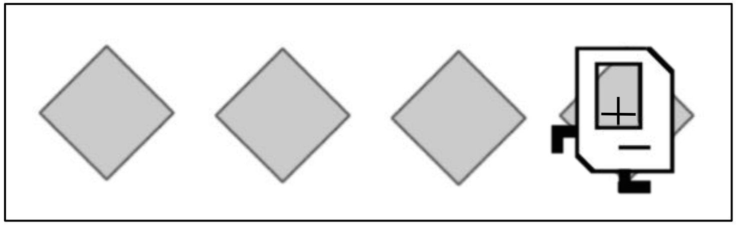


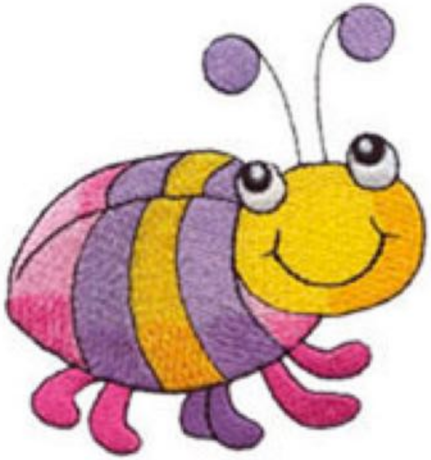


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        for(int i = 0; i < 4; i++) {
            putBeeper();
            move();
        }
    }
}
```

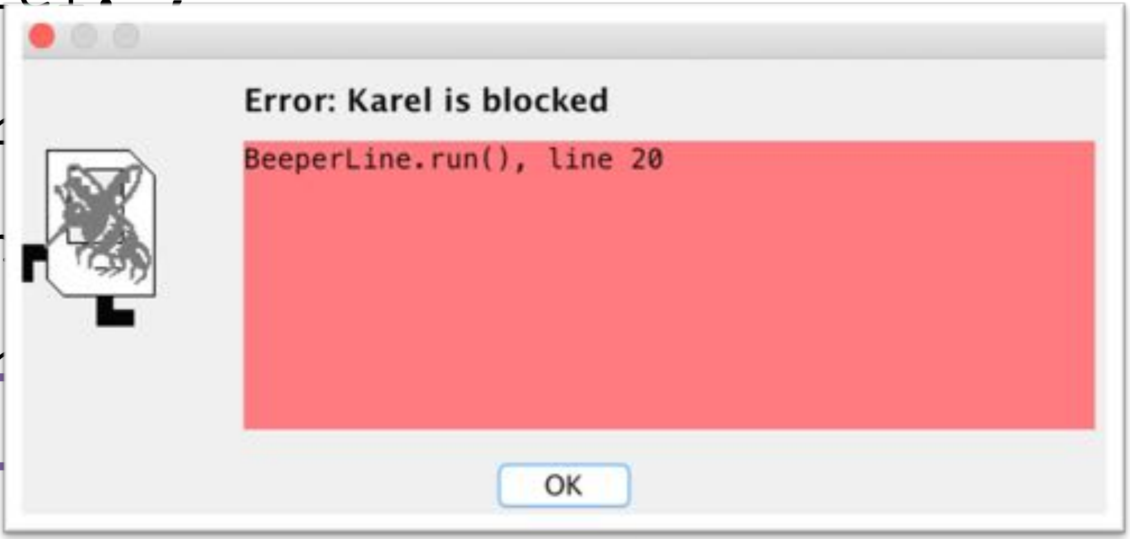
4th time through loop





Place Beeper Line

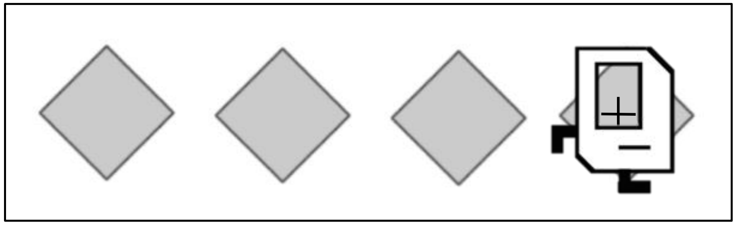
```
import stanford.karel.*;

public class Beeper

    public void run
        for(int i =
            putBeeper
            move();
        }
    }
}
```

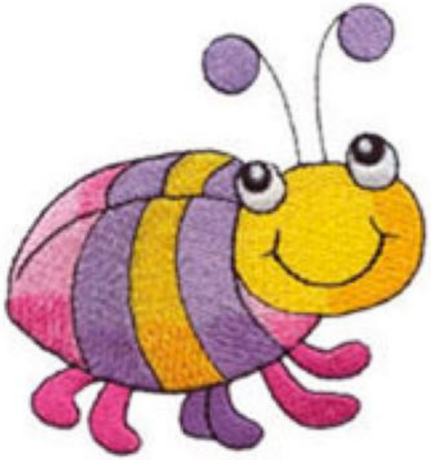


Error: Karel is blocked

```
BeeperLine.run(), line 20
```

4th time through loop

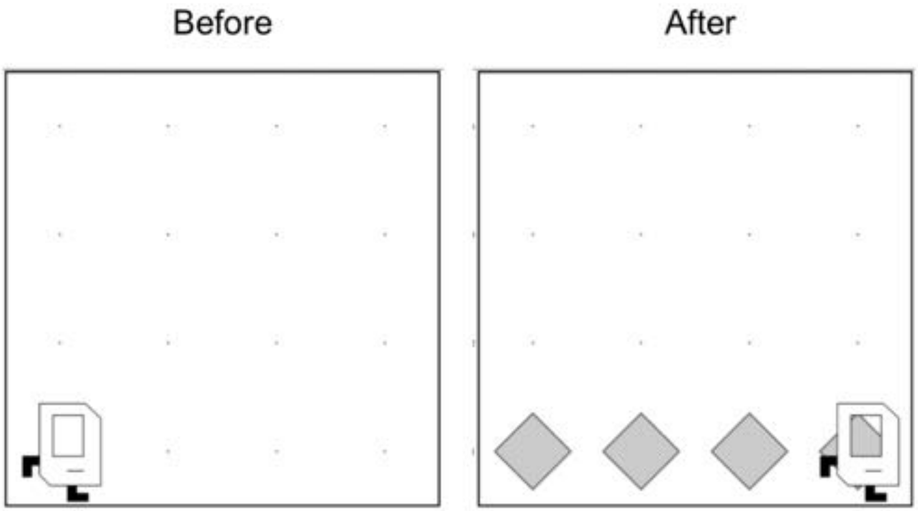


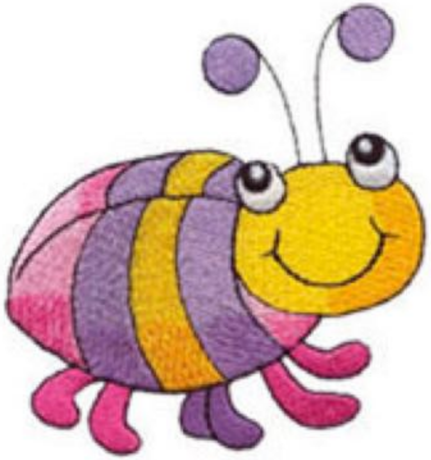


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

     public void run() {
         for(int i = 0; i < 4; i++) {
             putBeeper();
             move();
         }
     }
}
```

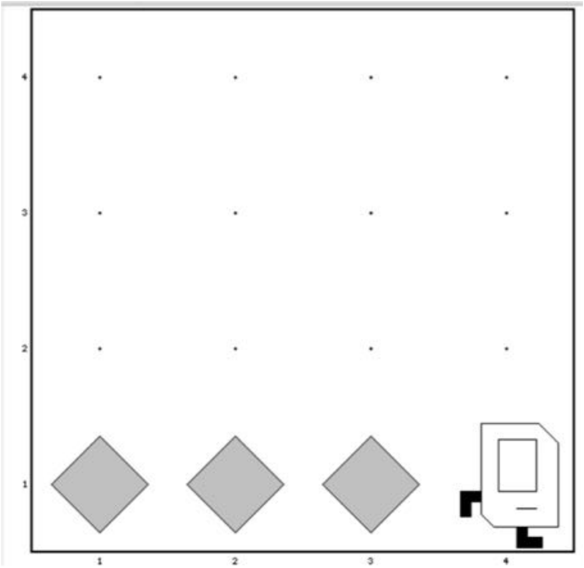
What we want

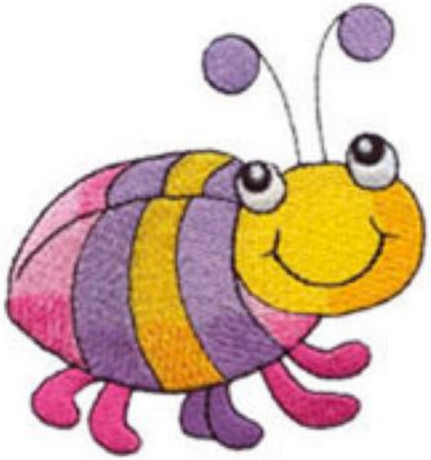




Place Beeper Line

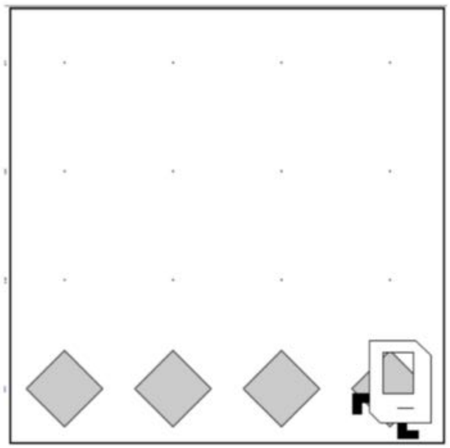
```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        for(int i = 0; i < 3; i++) {
            putBeeper();
            move();
        }
    }

}
```



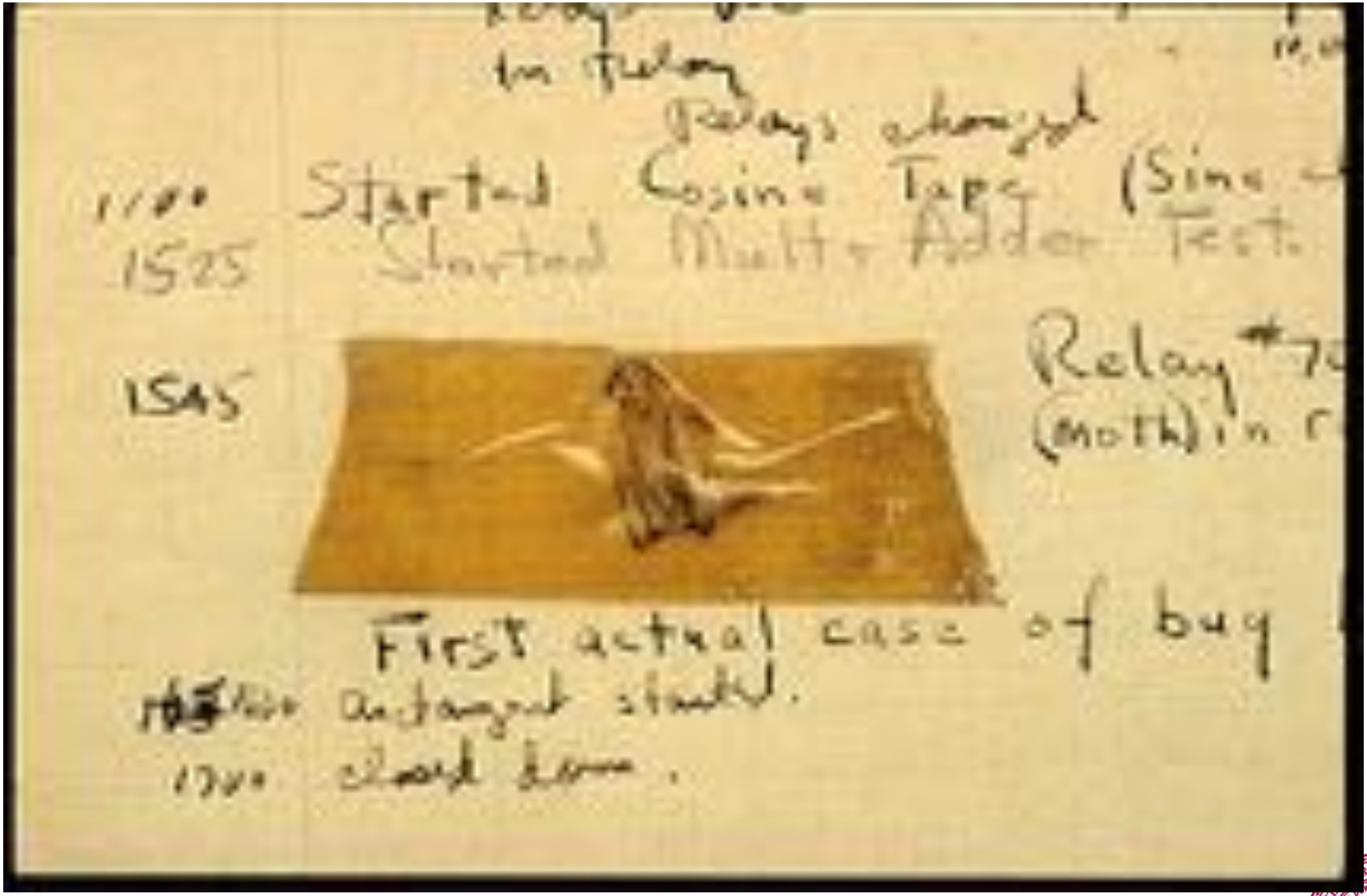


Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        for(int i = 0; i < 3; i++) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```



Actual Bug from Marc II



in relay

Relays changed

1100 Started Cosine Tape (Sine

1525 Started Multi Adder Test.

1545 Relay #70

(moth) in r

First actual case of bug

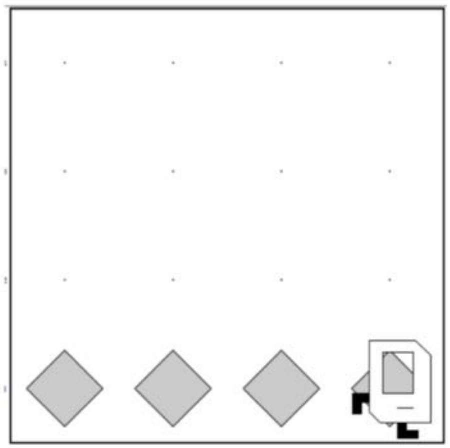
1630 Antagent started.

1700 closed down.

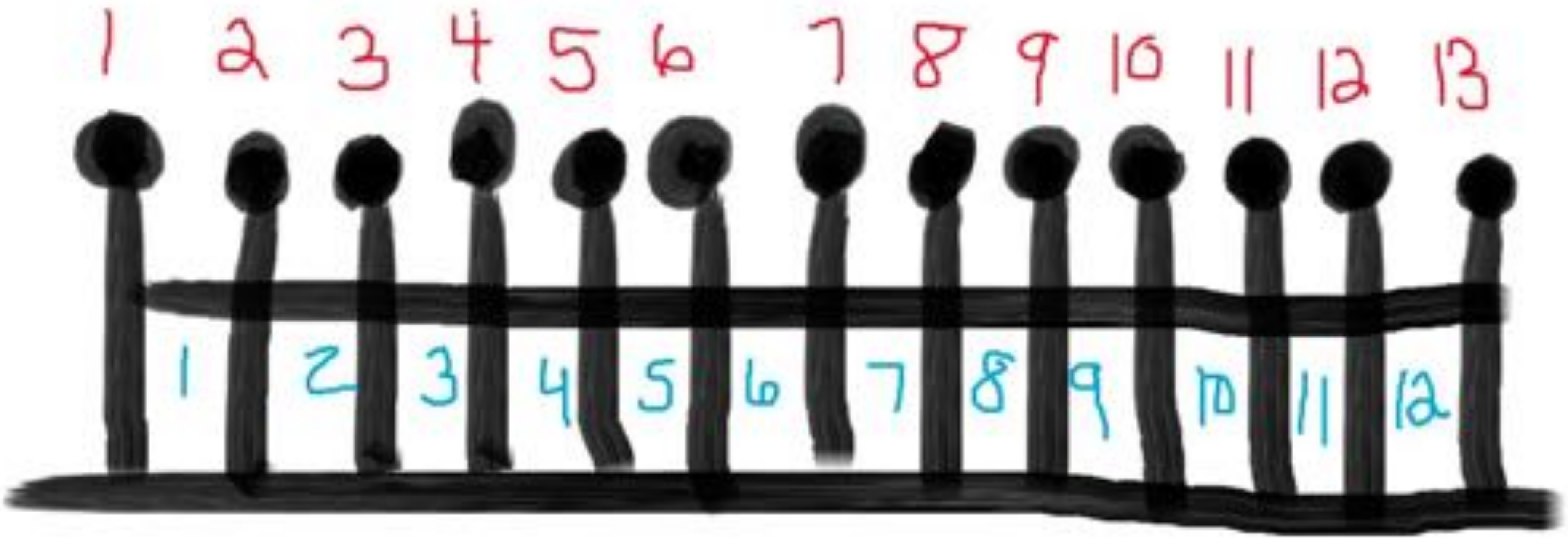
Grace Hopper

Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
       for(int i = 0; i < 3; i++) {
          putBeeper();
          move();
       }
       // extra put beeper
       putBeeper();
    }

}
```



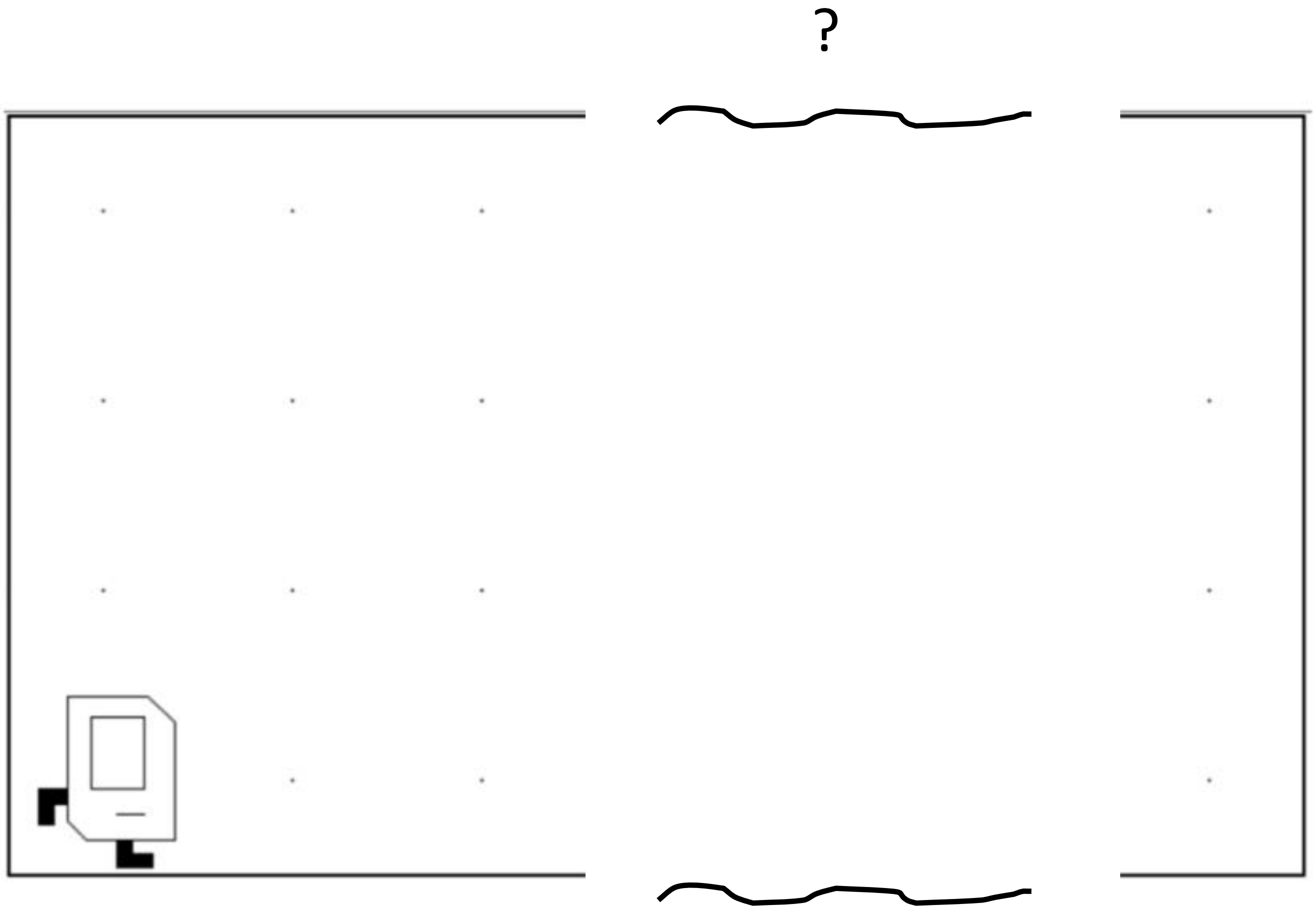
Fence Post Problem



* Also sometimes called an Off By One Error

Unstoppable

Don’t Know World Size



While Loop

While Loop

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

       // example while loop
       while(condition) {
          code to repeat
       }

    }

}
```

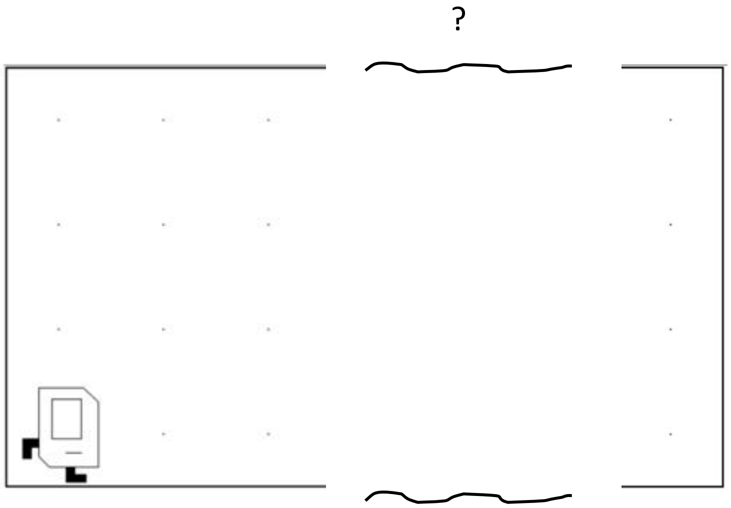
Possible Conditions

Test	***Opposite***	***What it checks***
`frontIsClear()`	`frontIsBlocked()`	Is there a wall in front of Karel?
`leftIsClear()`	`leftIsBlocked()`	Is there a wall to Karel's left?
`rightIsClear()`	`rightIsBlocked()`	Is there a wall to Karel's right?
`beepersPresent()`	`noBeepersPresent()`	Are there beepers on this corner?
`beepersInBag()`	`noBeepersInBag()`	Any there beepers in Karel's bag?
`facingNorth()`	`notFacingNorth()`	Is Karel facing north?
`facingEast()`	`notFacingEast()`	Is Karel facing east?
`facingSouth()`	`notFacingSouth()`	Is Karel facing south?
`facingWest()`	`notFacingWest()`	Is Karel facing west?

*This is **Table 1** on page 18 of the Karel coursereader.*

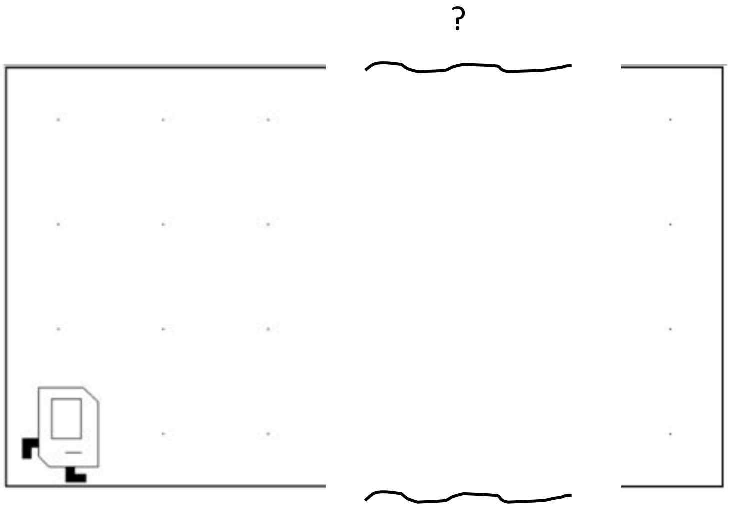
Move to Wall

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

     public void run() {

          // example while loop
          while(condition) {
               code to repeat
          }

     }

}
```



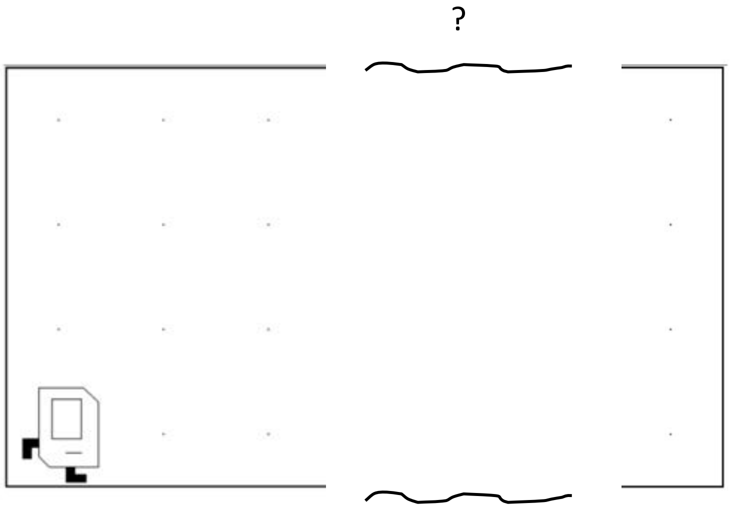
Move to Wall

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

     public void run() {

          // example while loop
          while(frontIsClear()) {
               code to repeat
          }

     }

}
```



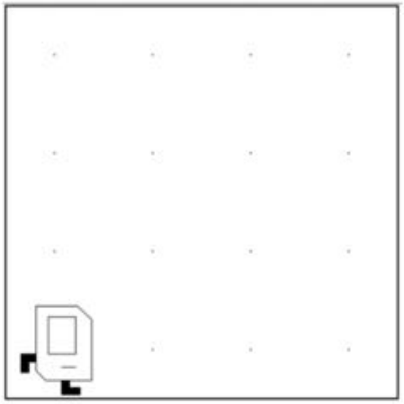
Move to Wall

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

     public void run() {

          // example while loop
          while(frontIsClear()) {
               move();
          }

     }

}
```

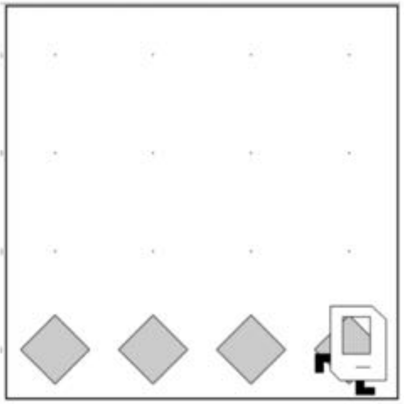


Code that Works in Any World

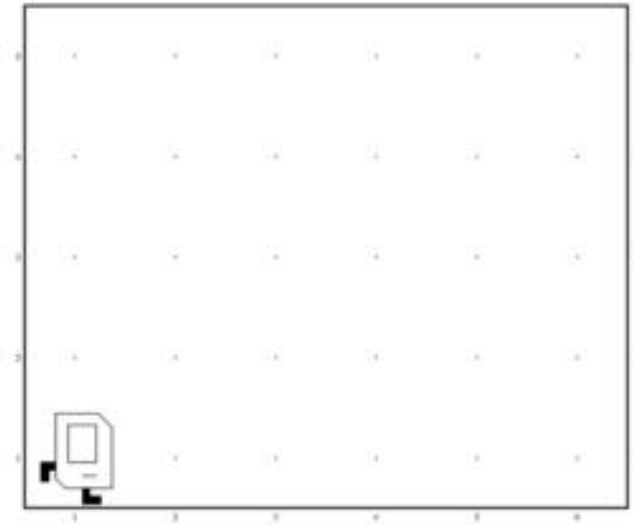
Before



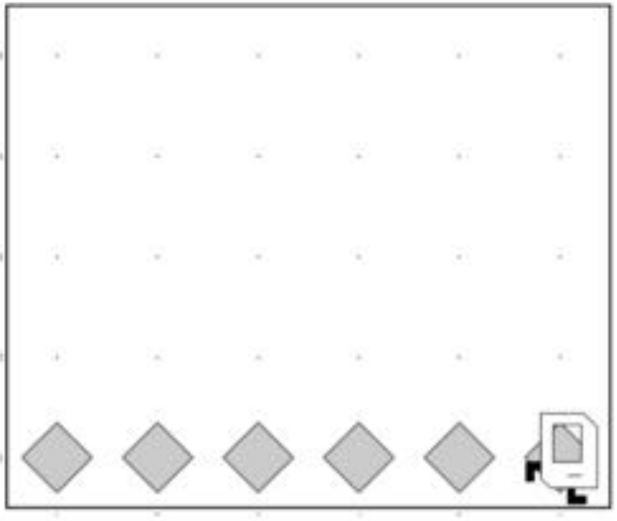
After



Before



After



Place Beeper Line

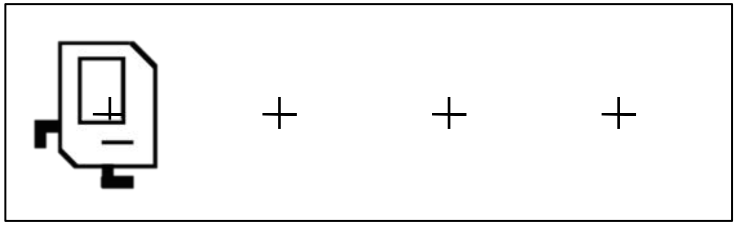
```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

       // example while loop
       while(frontIsClear()) {
          move();
       }
    }

}
```

Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

       // example while loop
       while(frontIsClear()) {
          putBeeper();
          move();
       }
    }

}
```

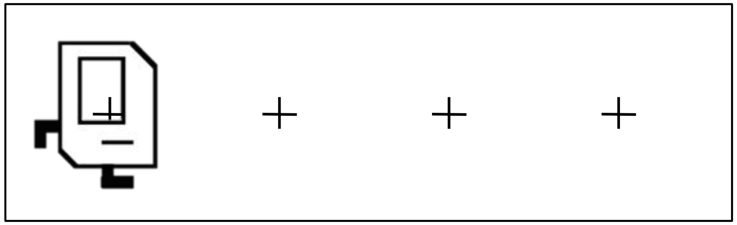
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
    }

}
```



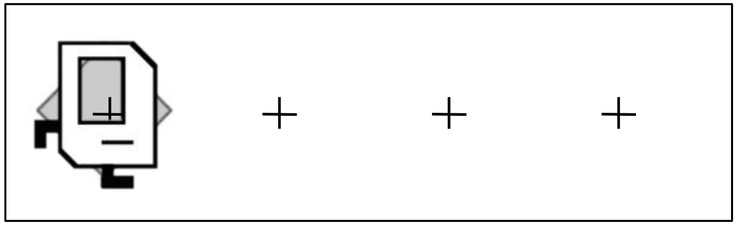
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
    }

}
```



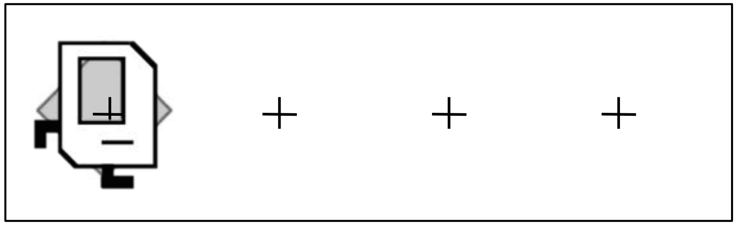
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

     public void run() {

         // example while loop
         while(frontIsClear()) {
            putBeeper();
            move();
         }
     }

}
```



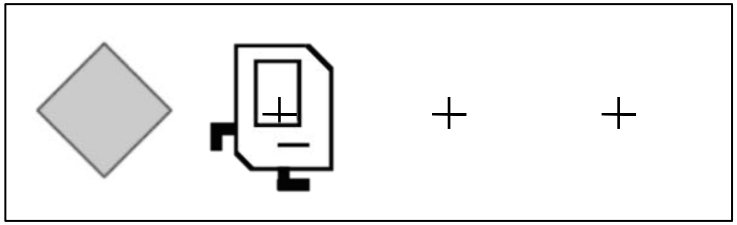
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

     public void run() {

          // example while loop
          while(frontIsClear()) {
             putBeeper();
             move();
          }
     }

}
```



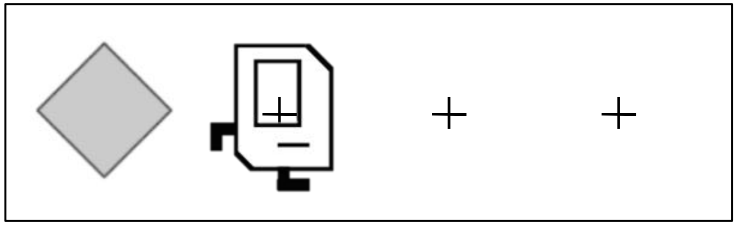
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
    }

}
```



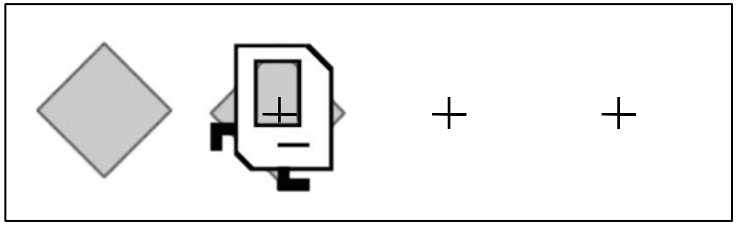
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
    }

}
```



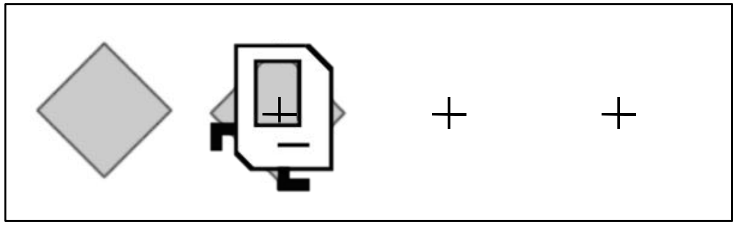
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

   public void run() {

      // example while loop
      while(frontIsClear()) {
         putBeeper();
         move();
      }
   }

}
```



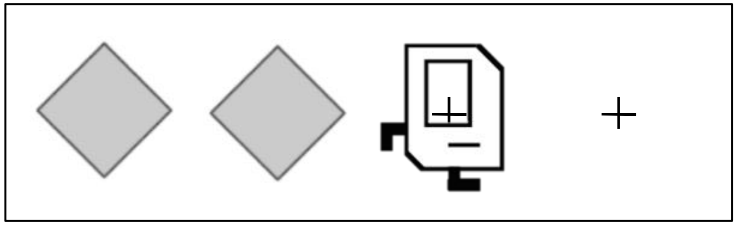
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

     public void run() {

          // example while loop
          while(frontIsClear()) {
               putBeeper();
               move();
          }
     }

}
```



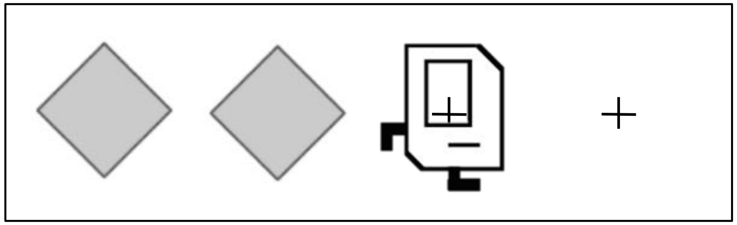
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

     public void run() {

          // example while loop
          while(frontIsClear()) {
               putBeeper();
               move();
          }
     }

}
```



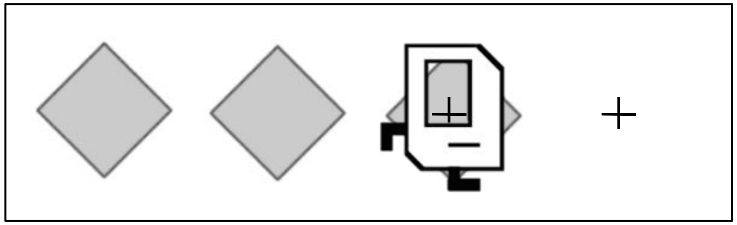
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
    }

}
```



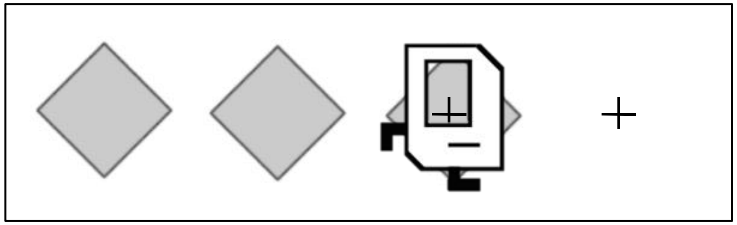
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
    }

}
```



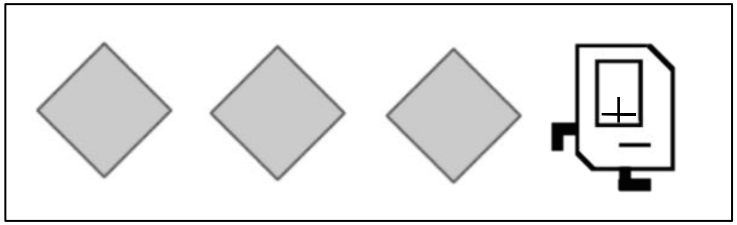
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
    }

}
```



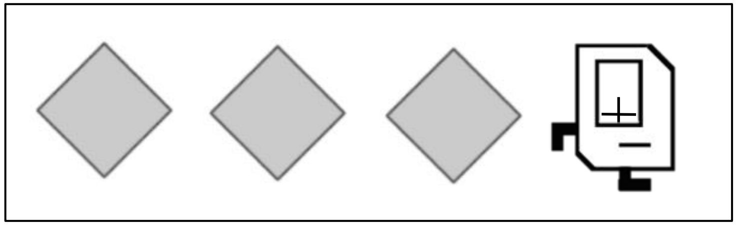
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

     public void run() {

          // example while loop
          while(frontIsClear()) {
               putBeeper();
               move();
          }
     }

}
```



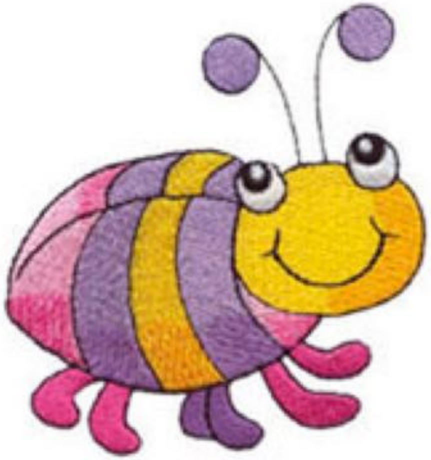
Place Beeper Line

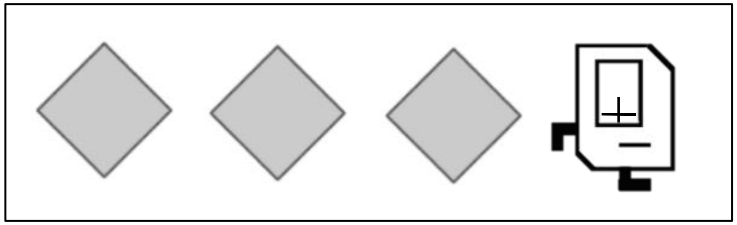
```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

     public void run() {

          // example while loop
          while(frontIsClear()) {
               putBeeper();
               move();
          }
     }

}
```



Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
    }

}
```



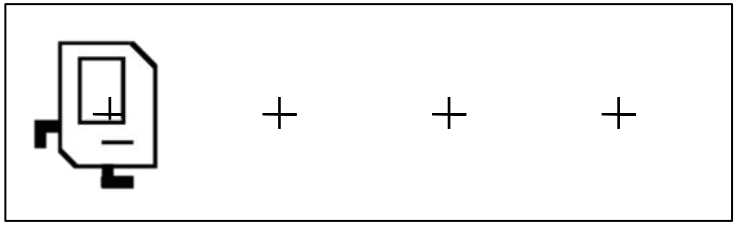


Place Beeper Line

```java
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```

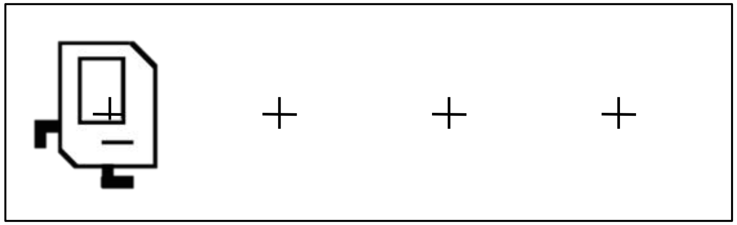
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```



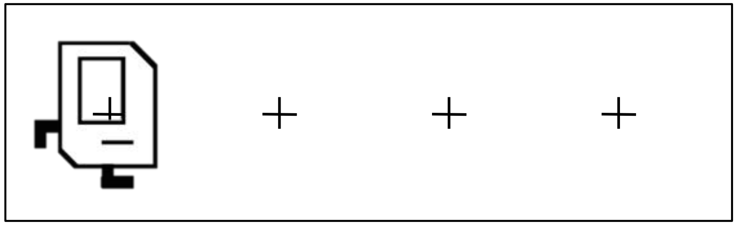
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```



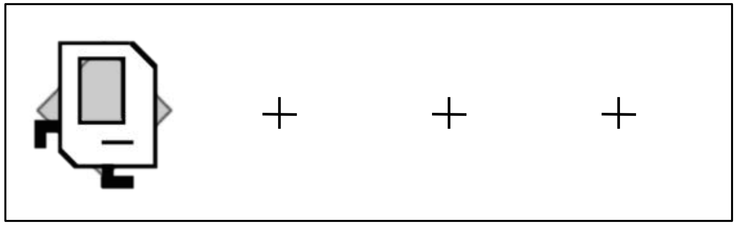
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```



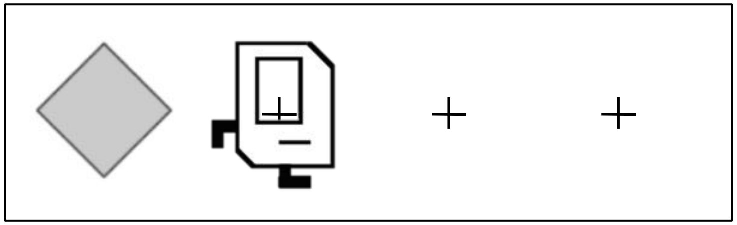
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

       // example while loop
       while(frontIsClear()) {
          putBeeper();
          move();
       }
       // extra put beeper
       putBeeper();
    }

}
```



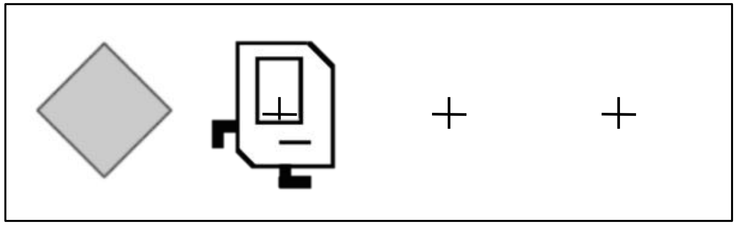
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```



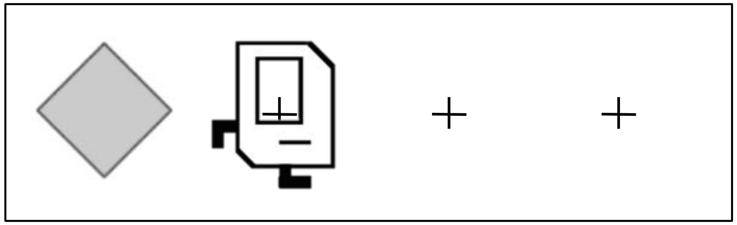
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```



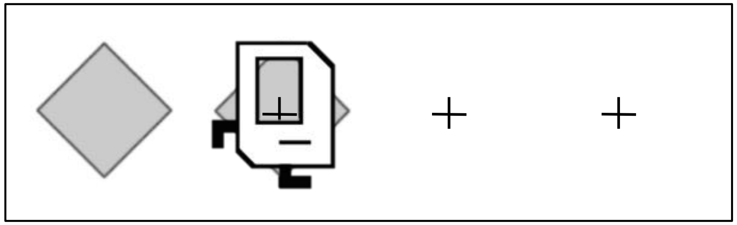
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```



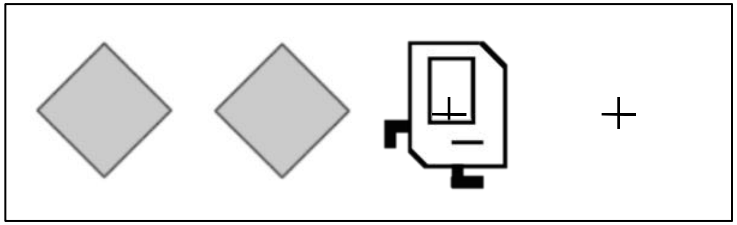
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```



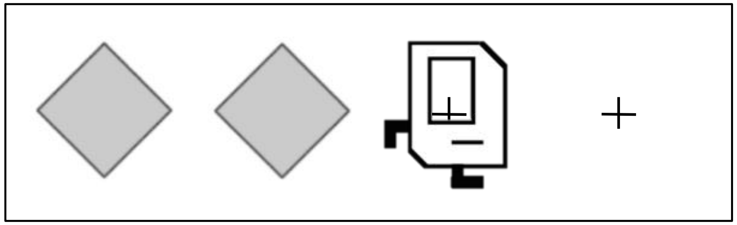
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```



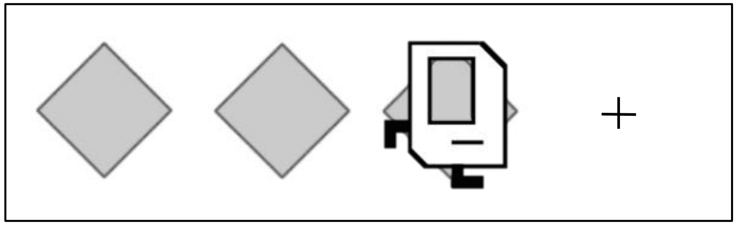
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```



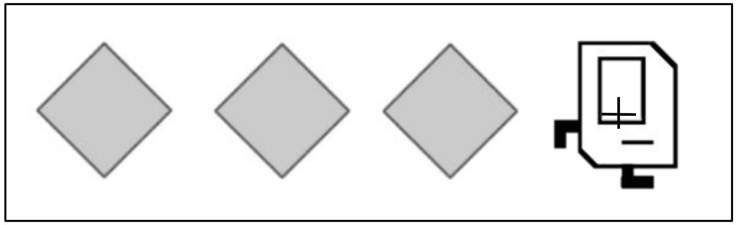
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

     public void run() {

          // example while loop
          while(frontIsClear()) {
             putBeeper();
             move();
          }
          // extra put beeper
          putBeeper();
      }

}
```



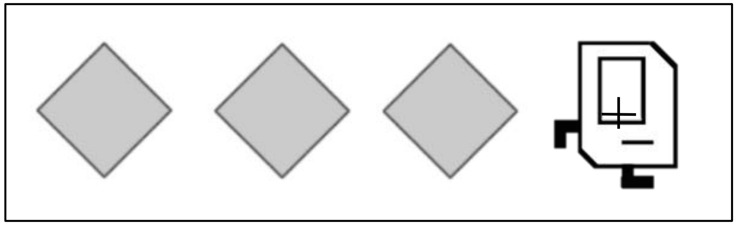
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```



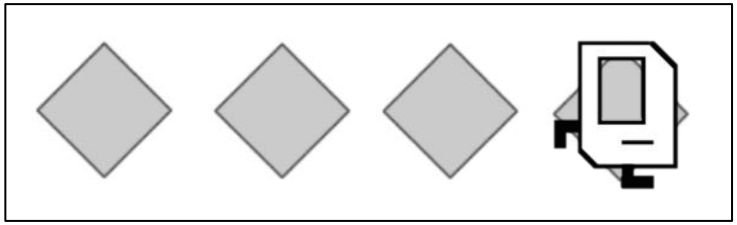
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```



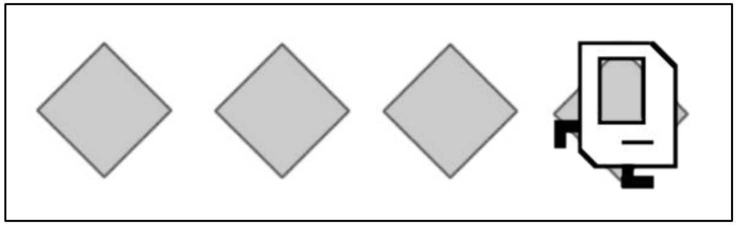
Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```



Place Beeper Line

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {

        // example while loop
        while(frontIsClear()) {
            putBeeper();
            move();
        }
        // extra put beeper
        putBeeper();
    }

}
```

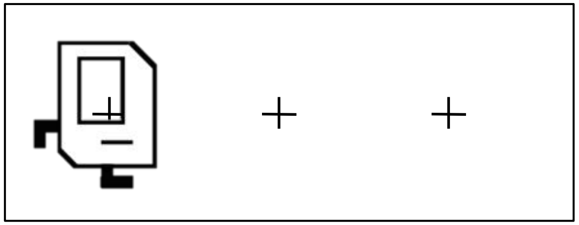


🎉🙌🙌

Common misconception:

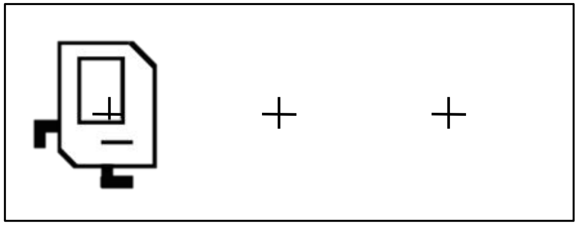
Place Beeper Line: Redux

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        // place a first beeper
        putBeeper();

        // example while loop
        while(frontIsClear()) {
            move();
            putBeeper();
        }
    }

}
```



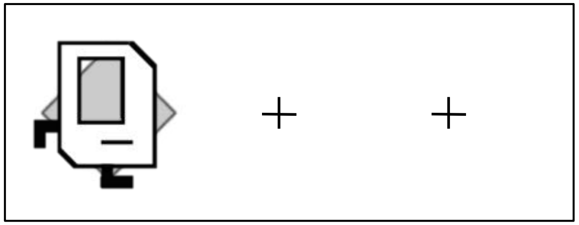
Place Beeper Line: Redux

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        // place a first beeper
        putBeeper();

        // example while loop
        while(frontIsClear()) {
            move();
            putBeeper();
        }
    }

}
```



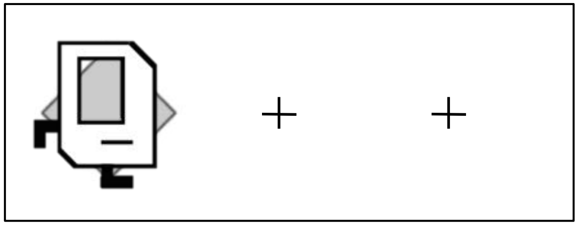
Place Beeper Line: Redux

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

     public void run() {
        // place a first beeper
        putBeeper();

        // example while loop
        while(frontIsClear()) {
           move();
           putBeeper();
        }
     }

}
```



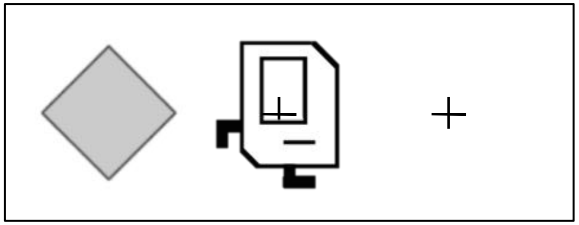
Place Beeper Line: Redux

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        // place a first beeper
        putBeeper();

        // example while loop
        while(frontIsClear()) {
            move();
            putBeeper();
        }
    }

}
```



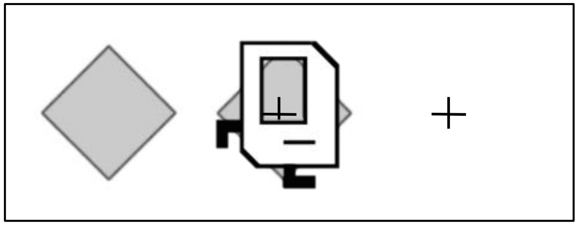
Place Beeper Line: Redux

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        // place a first beeper
        putBeeper();

        // example while loop
        while(frontIsClear()) {
            move();
            putBeeper();
        }
    }

}
```



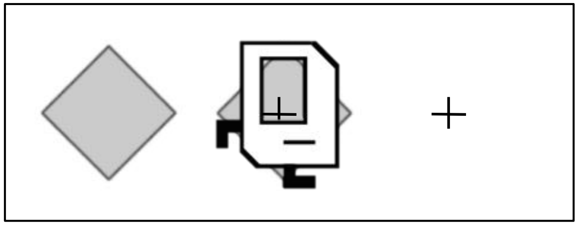
Place Beeper Line: Redux

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        // place a first beeper
        putBeeper();

        // example while loop
        while(frontIsClear()) {
            move();
            putBeeper();
        }
    }

}
```



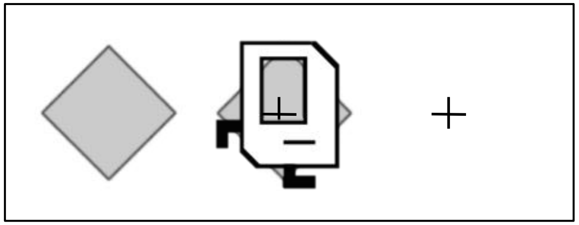
Place Beeper Line: Redux

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        // place a first beeper
        putBeeper();

        // example while loop
        while(frontIsClear()) {
            move();
            putBeeper();
        }
    }

}
```



Place Beeper Line: Redux

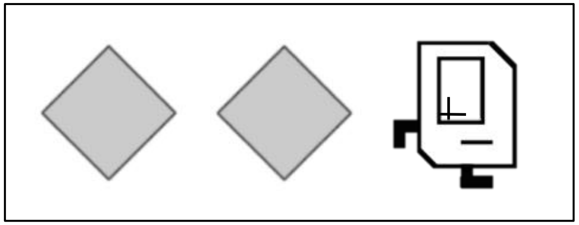
```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        // place a first beeper
        putBeeper();

        // example while loop
        while(frontIsClear()) {
            move();
            putBeeper();
        }
    }

}
```



Place Beeper Line: Redux

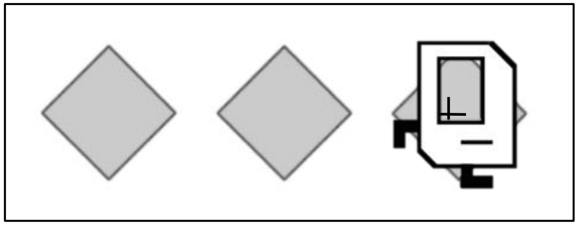
```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        // place a first beeper
        putBeeper();

        // example while loop
        while(frontIsClear()) {
            move();
            putBeeper();
        }
    }

}
```

This is incredibly important!



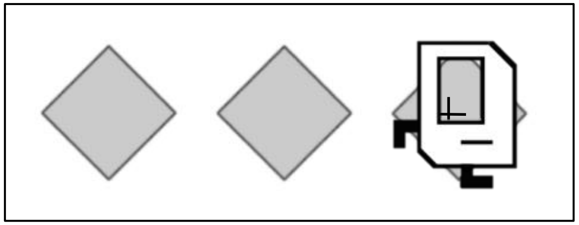
Place Beeper Line: Redux

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        // place a first beeper
        putBeeper();

        // example while loop
        while(frontIsClear()) {
            move();
            putBeeper();
        }
    }

}
```



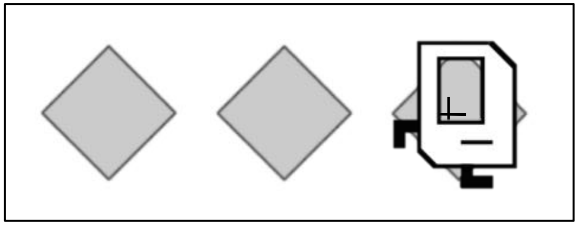
Place Beeper Line: Redux

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        // place a first beeper
        putBeeper();

        // example while loop
        while(frontIsClear()) {
            move();
            putBeeper();
        }
    }

}
```



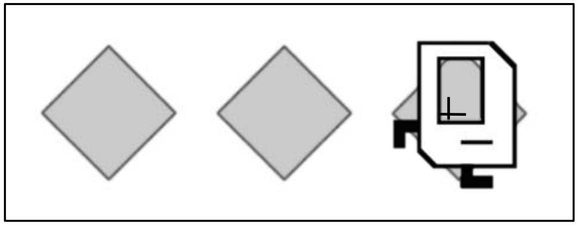
Place Beeper Line: Redux

```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        // place a first beeper
        putBeeper();

        // example while loop
        while(frontIsClear()) {
            move();
            putBeeper();
        }
    }

}
```



Place Beeper Line: Redux

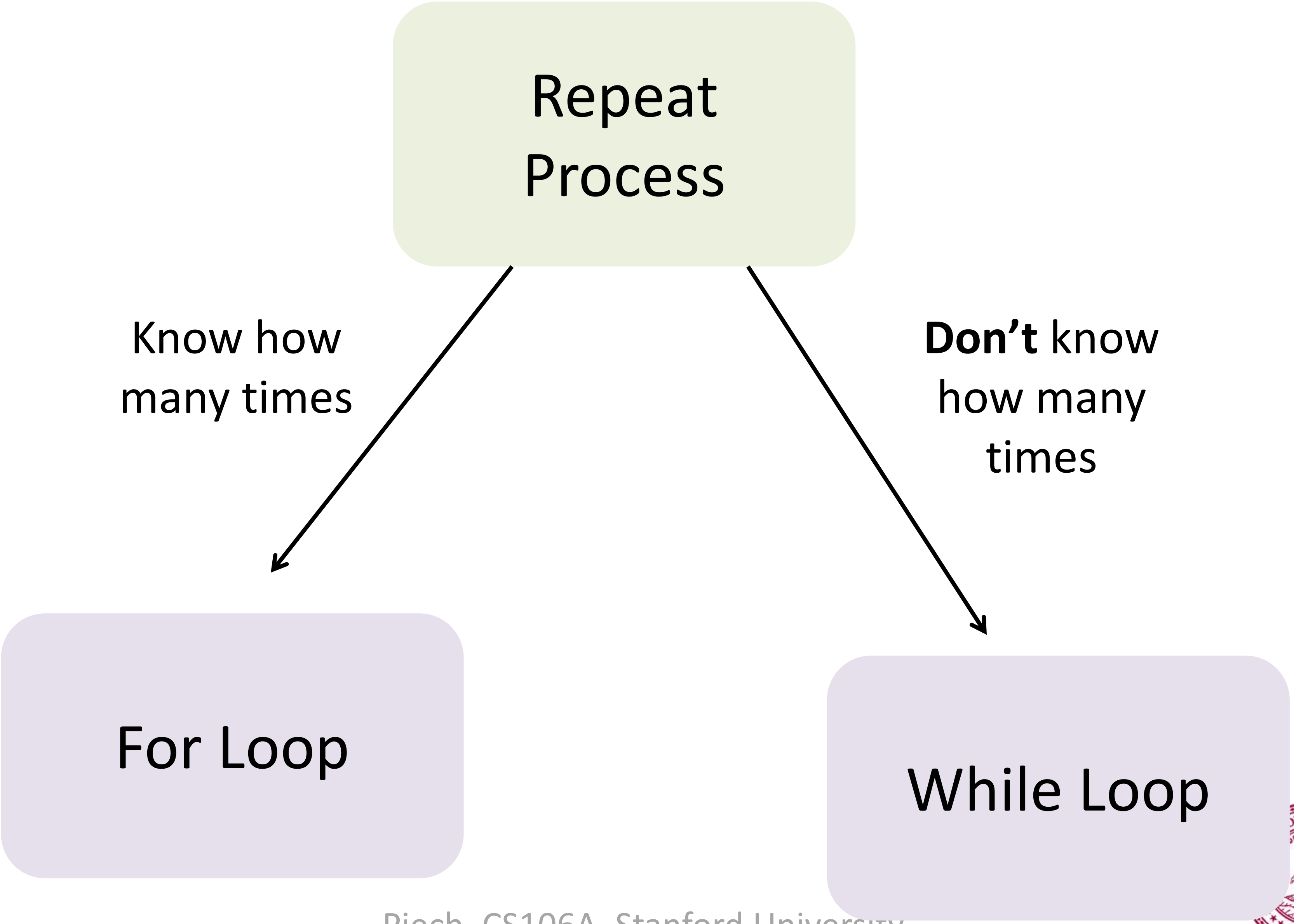
```
import stanford.karel.*;

public class BeeperLine extends SuperKarel {

    public void run() {
        // place a first beeper
        putBeeper();

        // example while loop
        while(frontIsClear()) {
            move();
            putBeeper();
        }
    }

}
```



A program executes one line at a time.

The while loop checks its condition only at the start of the code block and before repeating.

Which Loop



Repeat Process

Know how many times

Don't know how many times

For Loop

While Loop

What if you only want to repeat one time?

If statement

If Statement

```
import stanford.karel.*;

public class IfExample extends SuperKarel {

    public void run() {

       // example of an if statement
       if(condition) {
          code to run if condition is true
       }

    }

}
```

If Statement

```
import stanford.karel.*;

public class IfExample extends Pretend{

    public void run() {

       // example of an if statement
       if(youLikeBeyonce()) {
          makeSomeNoise();
       }

    }

}
```

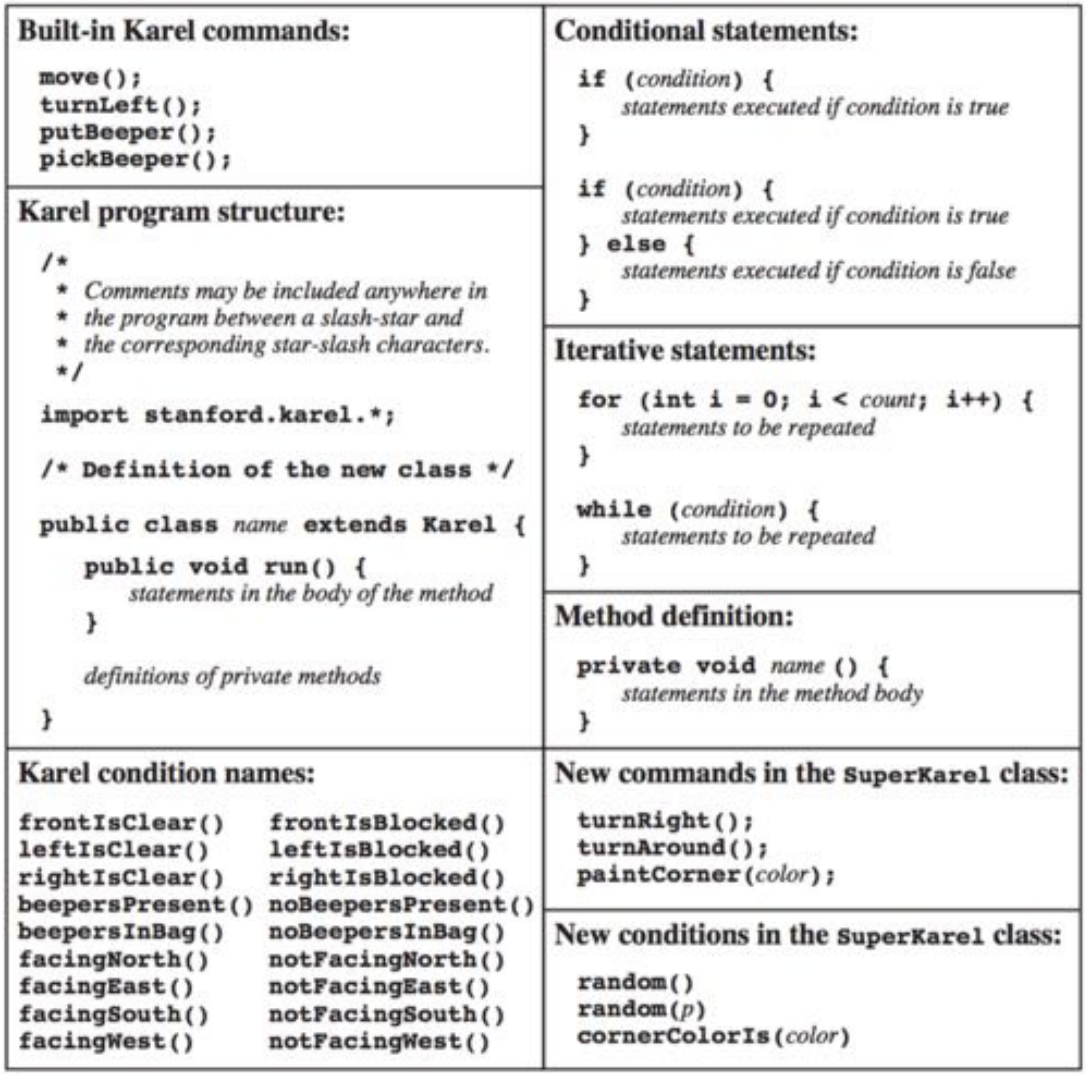
If Statement

```
import stanford.karel.*;

public class IfExample extends SuperKarel{

    public void run() {
       safeMove();
    }

    private void safeMove() {
       if(frontIsClear()) {
           move();
       }
    }

}
```

If / Else Statement

```
import stanford.karel.*;

public class IfExample extends SuperKarel{

    public void run() {
       invertBeeper();
    }

    private void invertBeeper() {
       if(beepersPresent()) {
           pickBeeper();
       } else {
           putBeeper();
       }
    }

}
```

The Full Karel



Built-in Karel commands:

```
move();
turnLeft();
putBeeper();
pickBeeper();
```

Karel program structure:

```
/*
 * Comments may be included anywhere in
 * the program between a slash-star and
 * the corresponding star-slash characters.
 */

import stanford.karel.*;

/* Definition of the new class */

public class name extends Karel {

    public void run() {
        statements in the body of the method
    }

    definitions of private methods
}
```

Karel condition names:

```
frontIsClear()    frontIsBlocked()
leftIsClear()     leftIsBlocked()
rightIsClear()    rightIsBlocked()
beepersPresent()  noBeepersPresent()
beepersInBag()    noBeepersInBag()
facingNorth()     notFacingNorth()
facingEast()      notFacingEast()
facingSouth()     notFacingSouth()
facingWest()      notFacingWest()
```

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
}
```

Iterative statements:

```
for (int i = 0; i < count; i++) {
    statements to be repeated
}

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

Method definition:

```
private void name () {
    statements in the method body
}
```

New commands in the `SuperKarel` class:

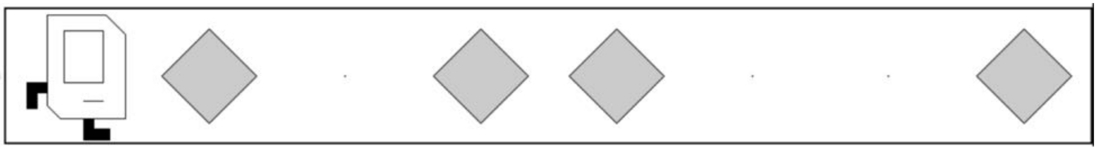
```
turnRight();
turnAround();
paintCorner(color);
```

New conditions in the `SuperKarel` class:

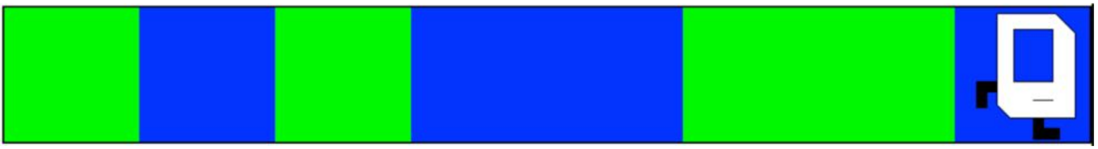
```
random()
random(p)
cornerColorIs(color)
```

Random Painter

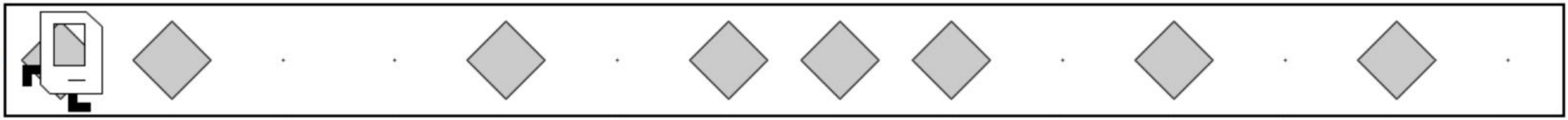
Before:



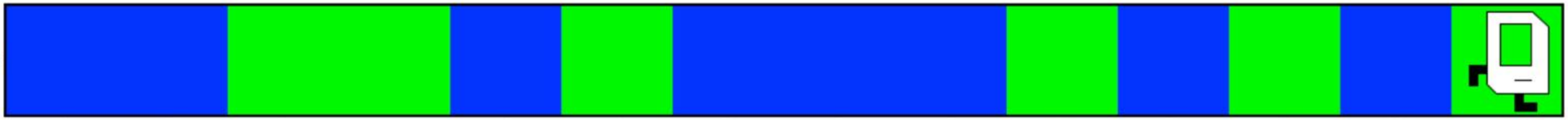
After:



Before:



After:

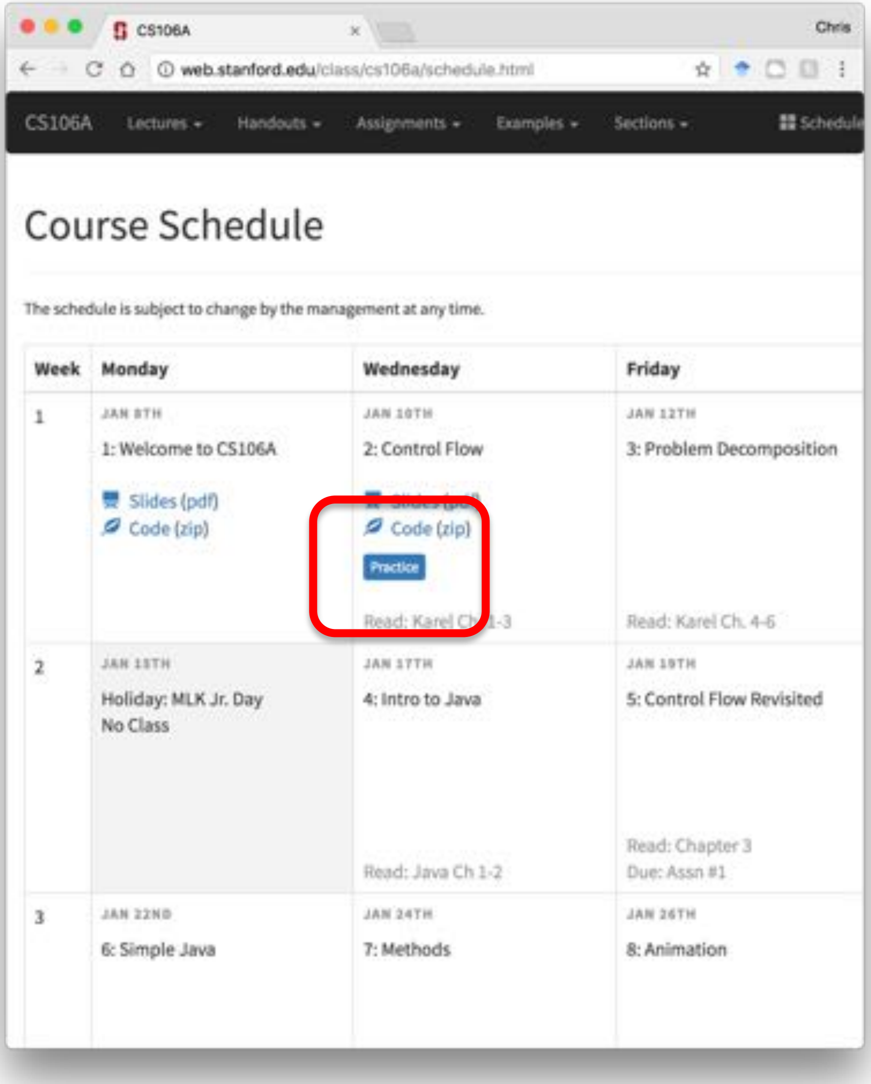


You just learned most of programming “control flow”

Today’s Goal

1. Code using loops and conditions
2. Trace programs that use loops and conditions

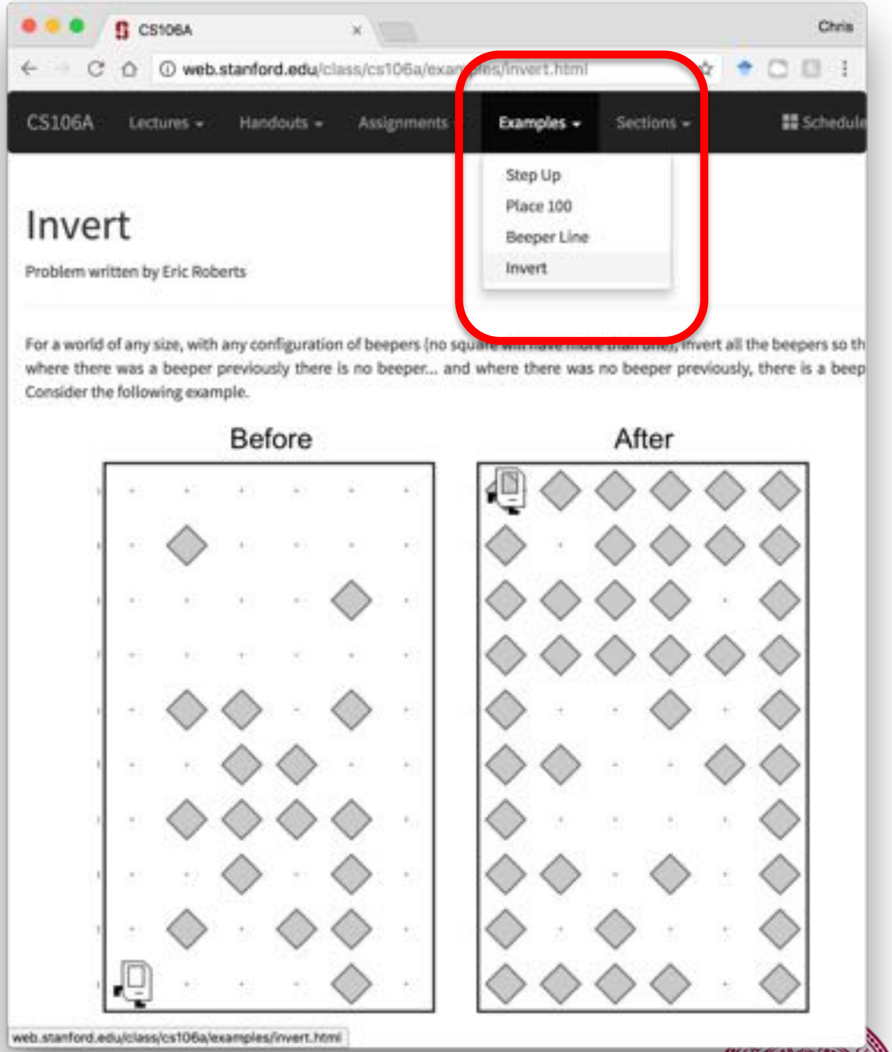
Examples and Practice



Course Schedule

The schedule is subject to change by the management at any time.

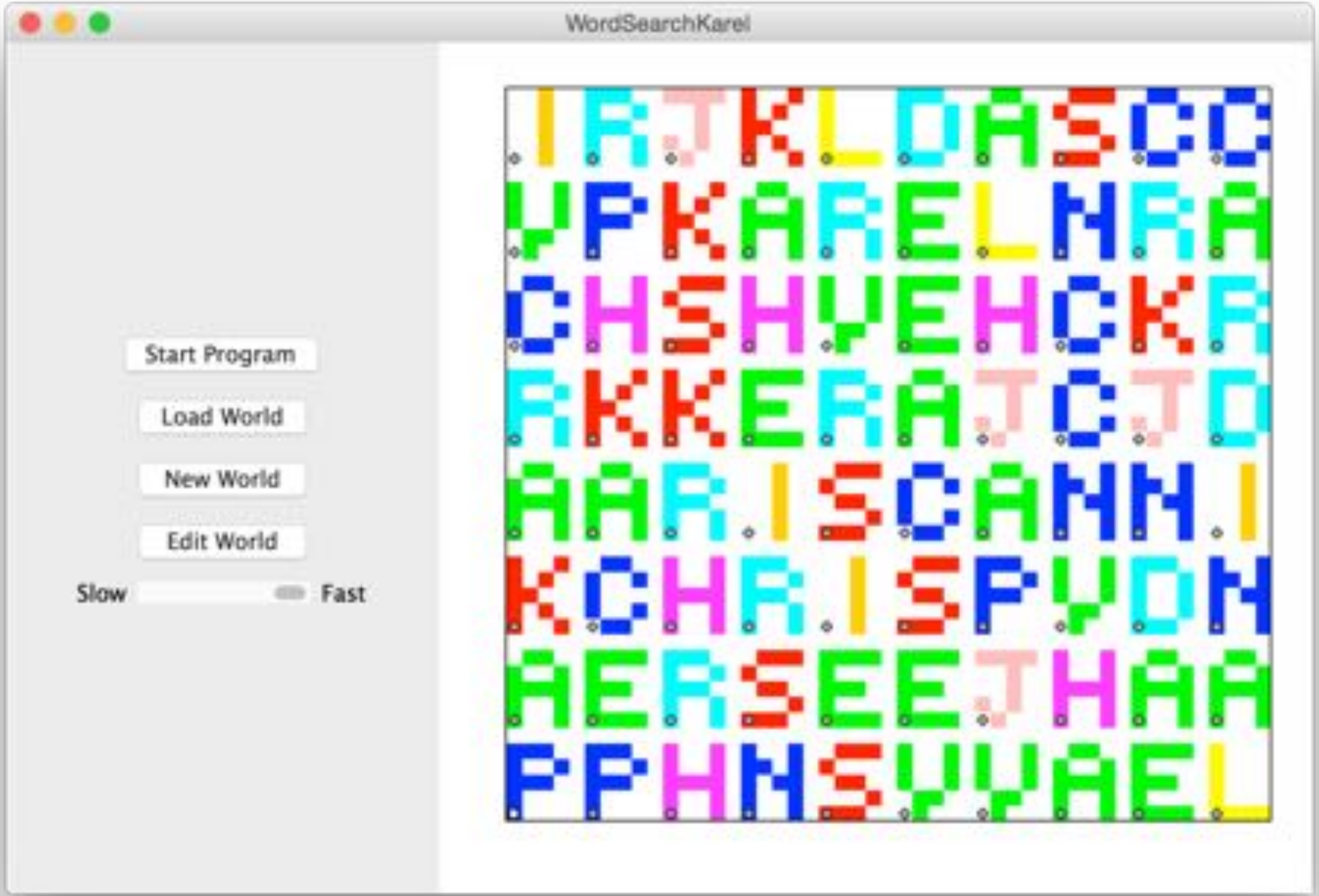
Week	Monday	Wednesday	Friday
1	JAN 8TH 1: Welcome to CS106A Slides (pdf) Code (zip)	JAN 10TH 2: Control Flow Slides (pdf) Code (zip) Practice Read: Karel Ch 1-3	JAN 12TH 3: Problem Decomposition Read: Karel Ch. 4-6
2	JAN 15TH Holiday: MLK Jr. Day No Class	JAN 17TH 4: Intro to Java Read: Java Ch 1-2	JAN 19TH 5: Control Flow Revisited Read: Chapter 3 Due: Assn #1
3	JAN 22ND 6: Simple Java	JAN 24TH 7: Methods	JAN 26TH 8: Animation



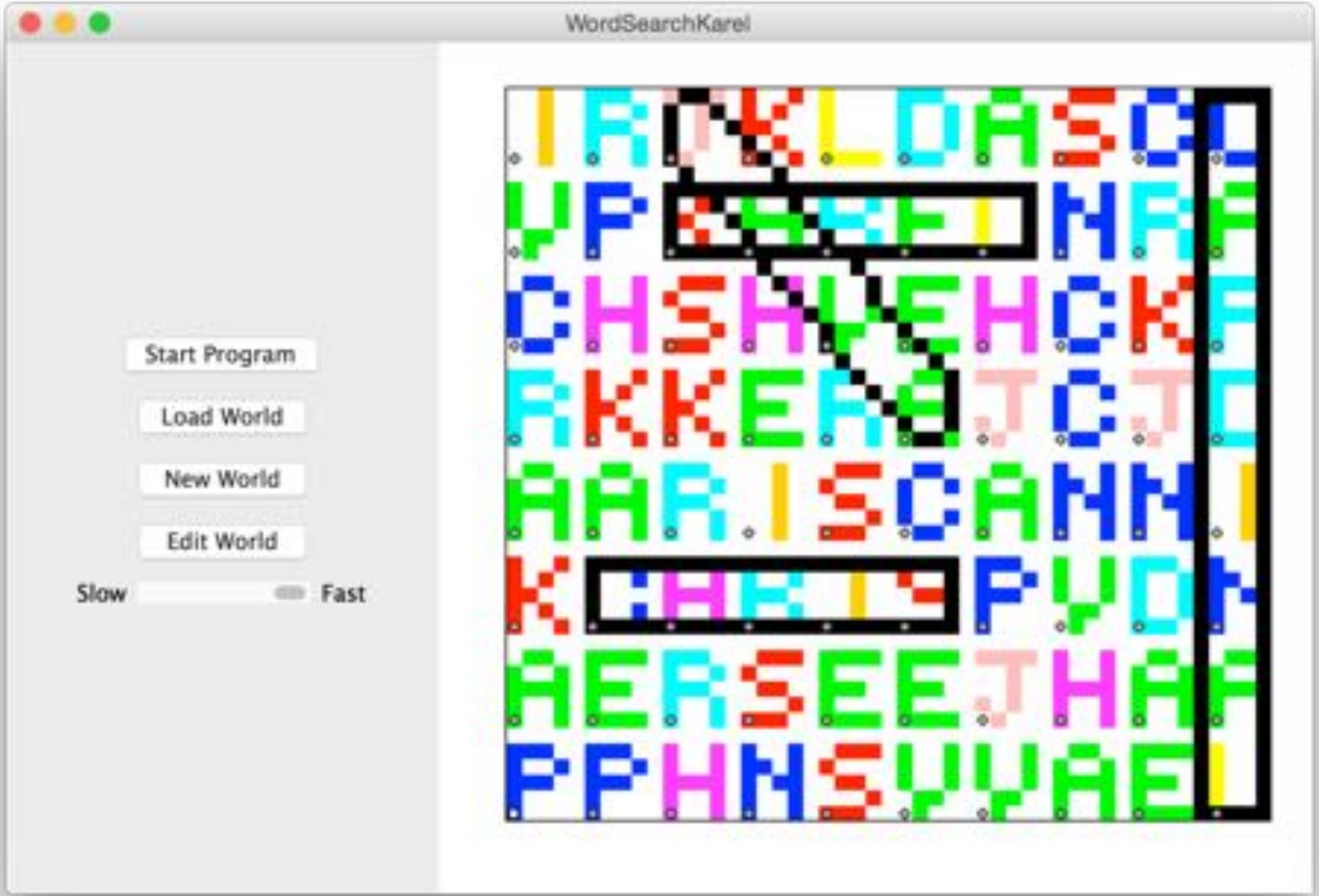
Invert

Problem written by Eric Roberts

For a world of any size, with any configuration of beepers (no square will have more than one), invert all the beepers so th
where there was a beeper previously there is no beeper... and where there was no beeper previously, there is a beep
Consider the following example.



```
I R J K L D A S C C
V P K A R E L N R A
C H S H V E H C K R
R K K E R A J C J D
A A R I S C A N N I
K C H R I S P V D N
A E R S E E J H A A
P P H N S V V A E L
```


WordSearchKarel
Start Program
Load World
New World
Edit World
Slow
Fast