



CS Bridge



Teaching at Stanford

CS106A

Programming
Methodologies

CURRENT

CS106B

Programming
Abstractions

LAST: FALL 2016

CS109

Probability for Computer
Scientists

LAST: FALL 2018

CS221

Intro to Artificial
Intelligence

LAST: SUM 2013

Stanford?



Stanford



Near San Francisco



Kenya

Kenya

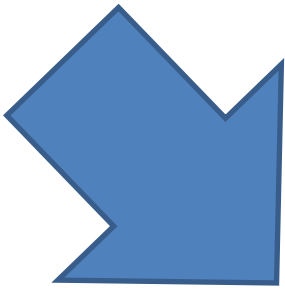


Kenya



Logistics

Course Website

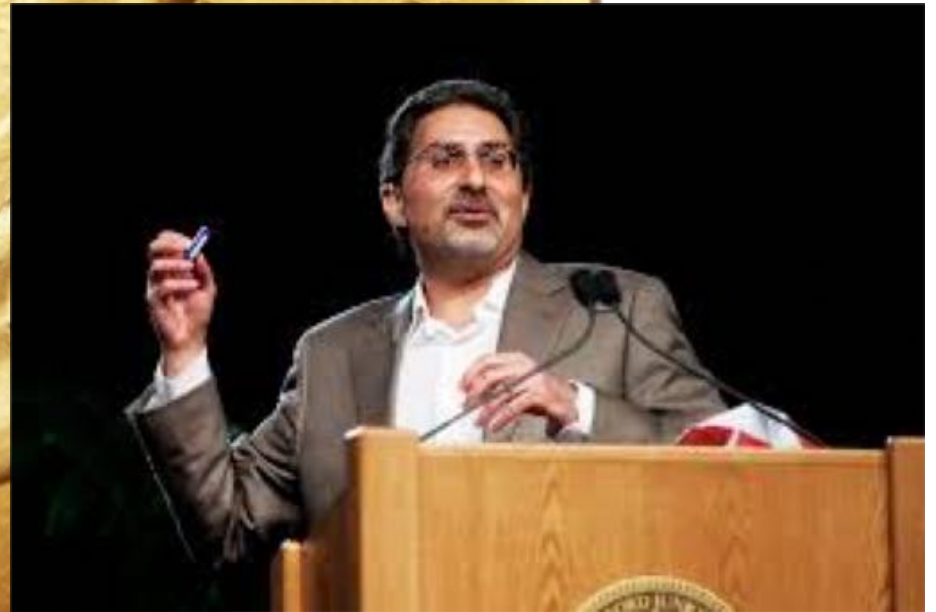
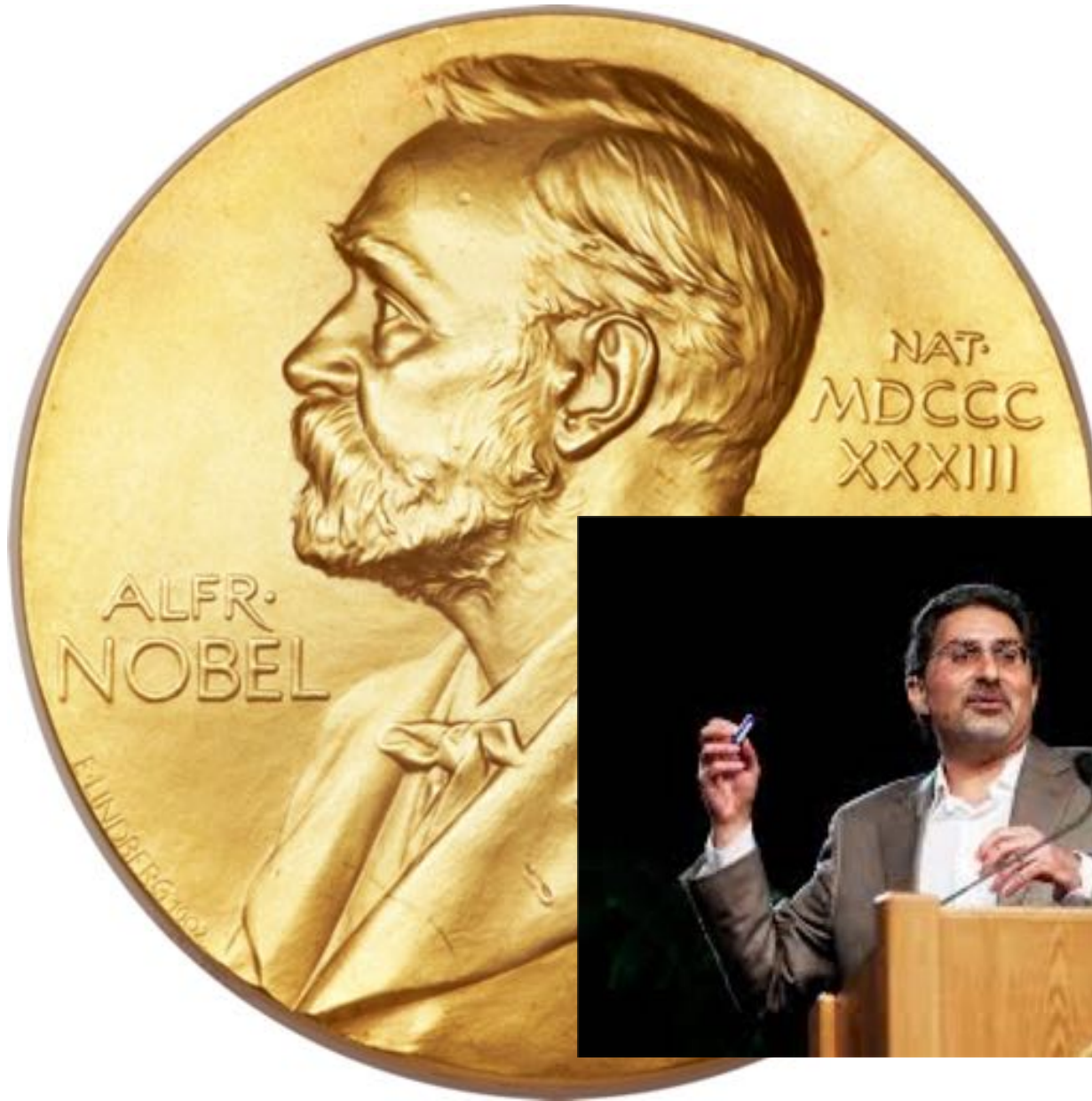


<http://ctu.csbridge.org>



*note that its **org** not **com**

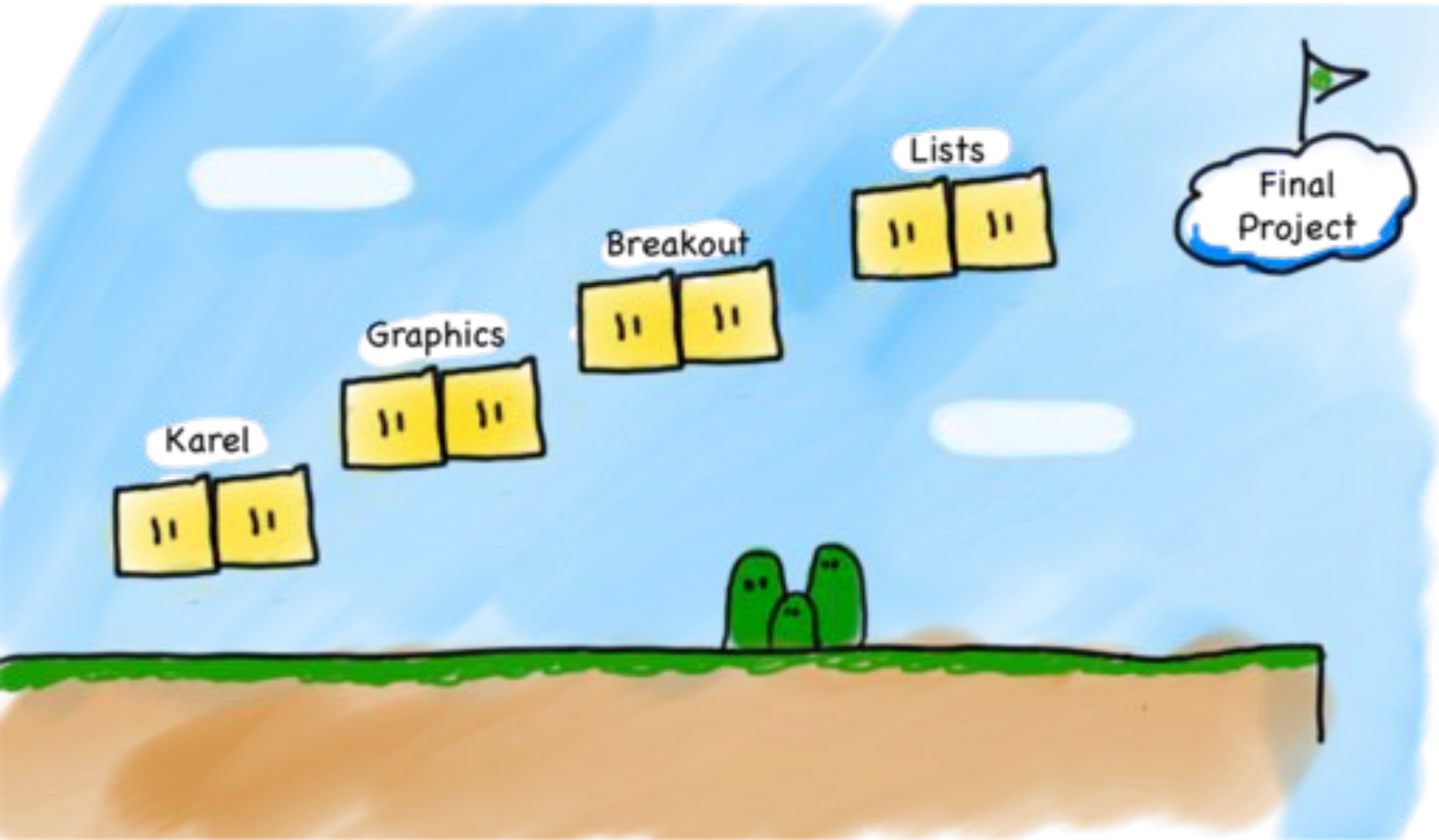
Prize Policy



Are there any questions?

About the class

Very High Level



Joint Effort



Great Team



Nick T.



Eliška



Ondra



Chris



Julia



Matyáš



Emily



Jaroslav



Radek



Glenn



Zach



Honza



Marek



Nick M.



Asena

Prerequisites



Prerequisite Test



Art of Computer *Science*

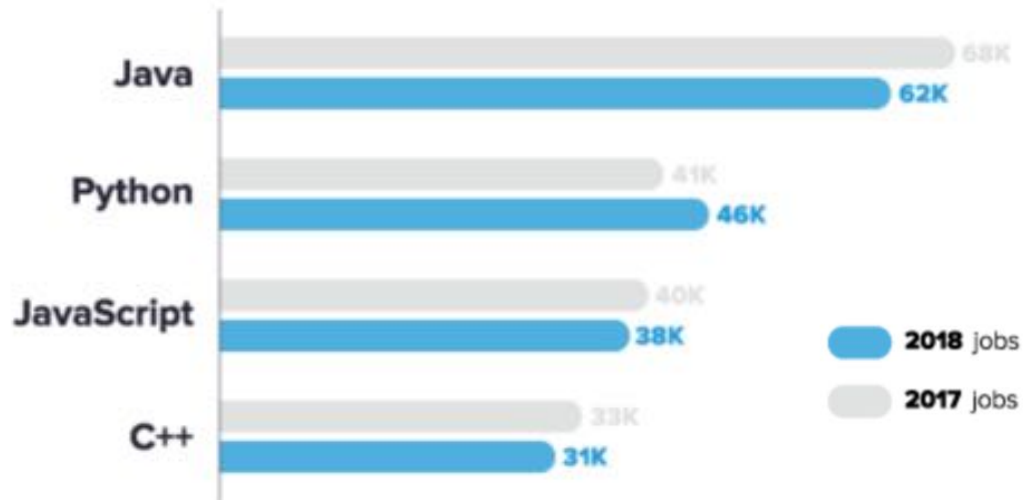


Why Java?

1

Job postings containing top languages

Indeed.com - November, 17th 2017



2



Breakout

What if I fall behind?

To Try is to Succeed

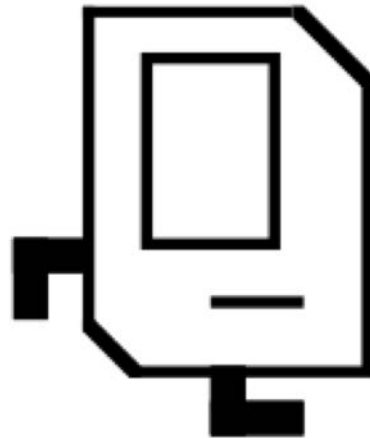


Lets Get Started



Meet Karel!

Ahoy!

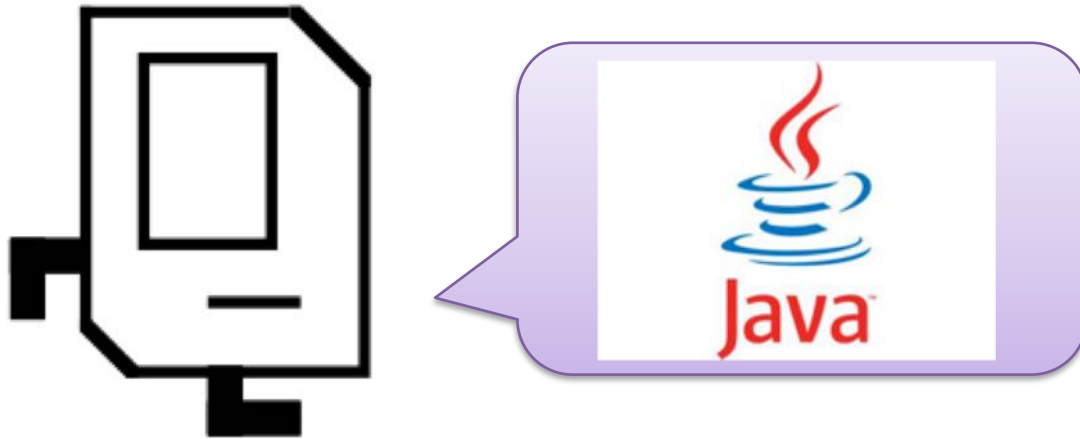


jak se máš?

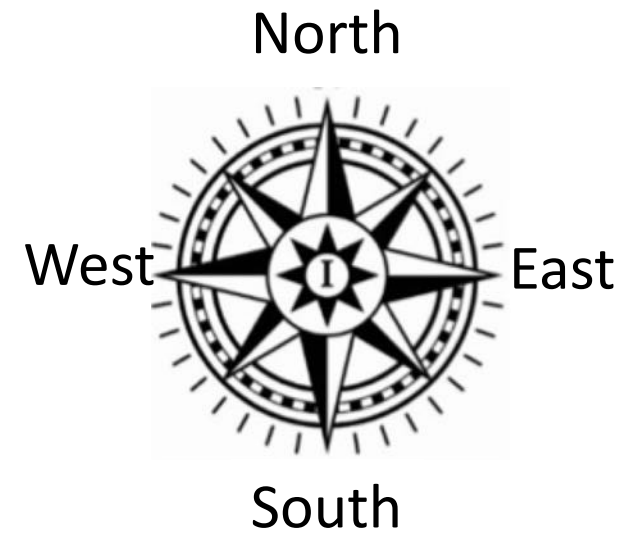
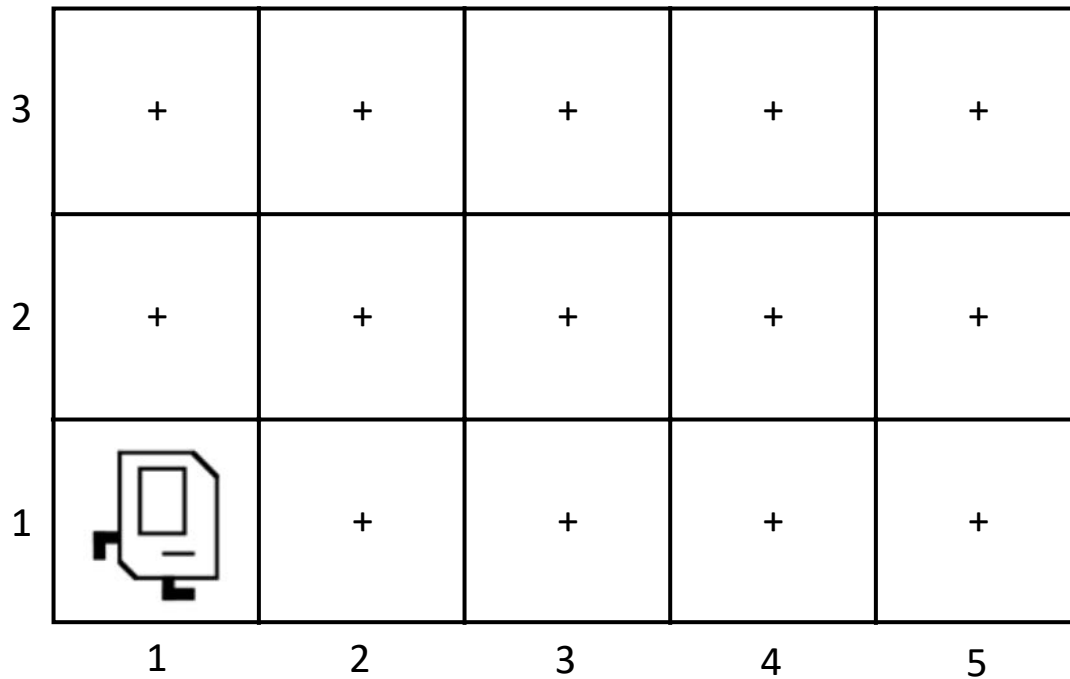


Karel Čapek

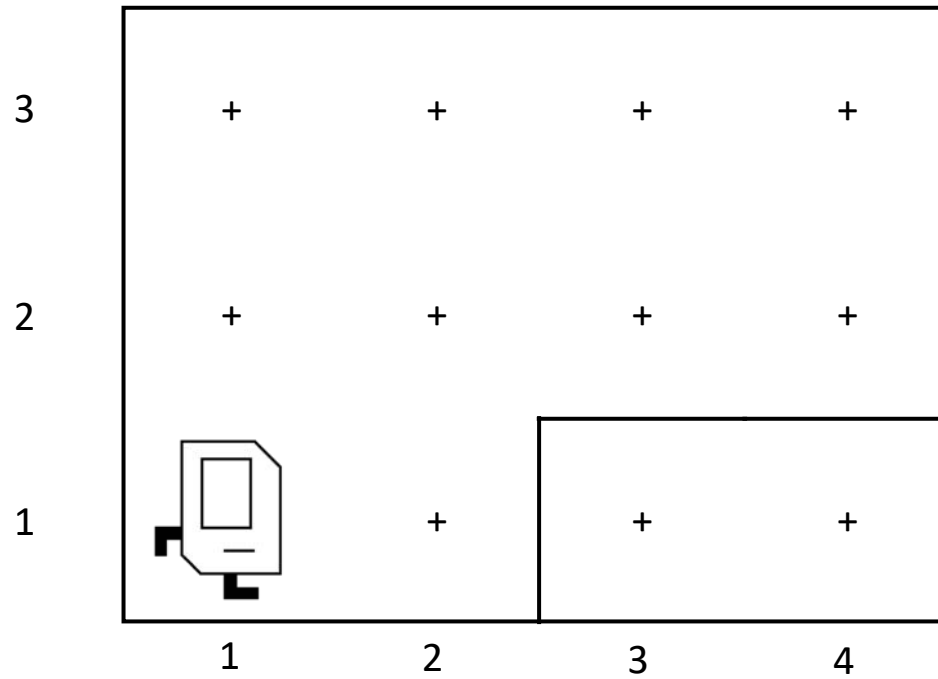
Karel Speaks Java



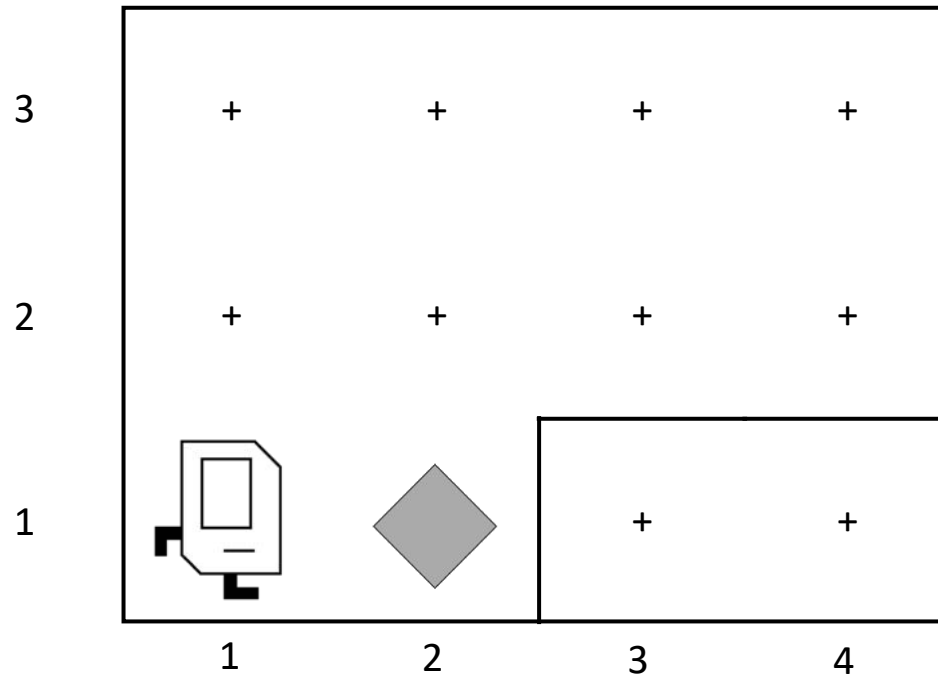
Karel's World



Walls



Beepers



Knows Four Commands



```
move ( ) ;
```

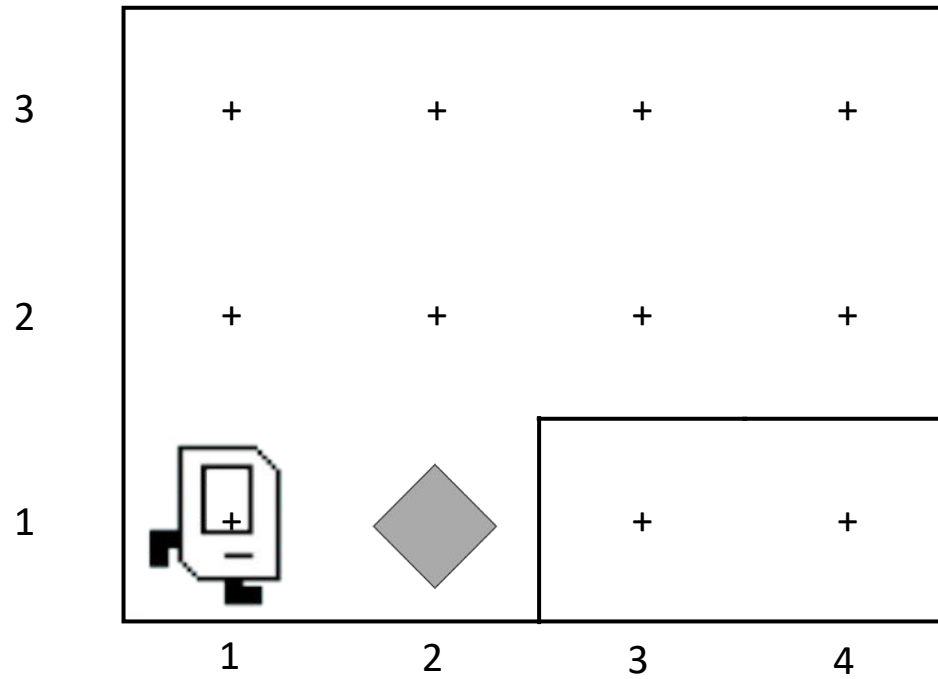
```
turnLeft ( ) ;
```

```
putBeeper ( ) ;
```

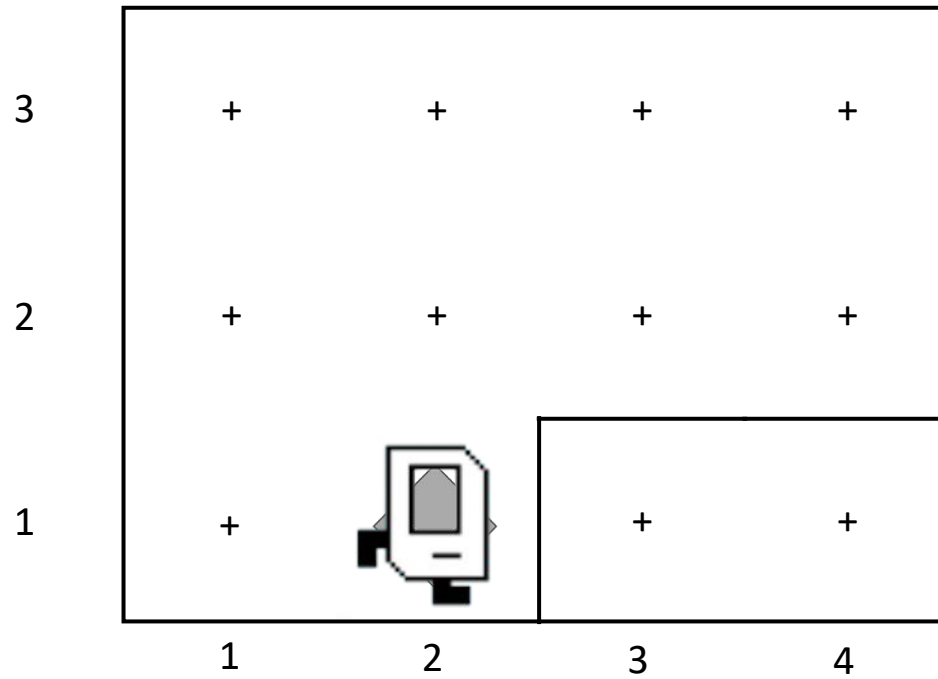
```
pickBeeper ( ) ;
```

```
move ( ) ;
```

move () ;

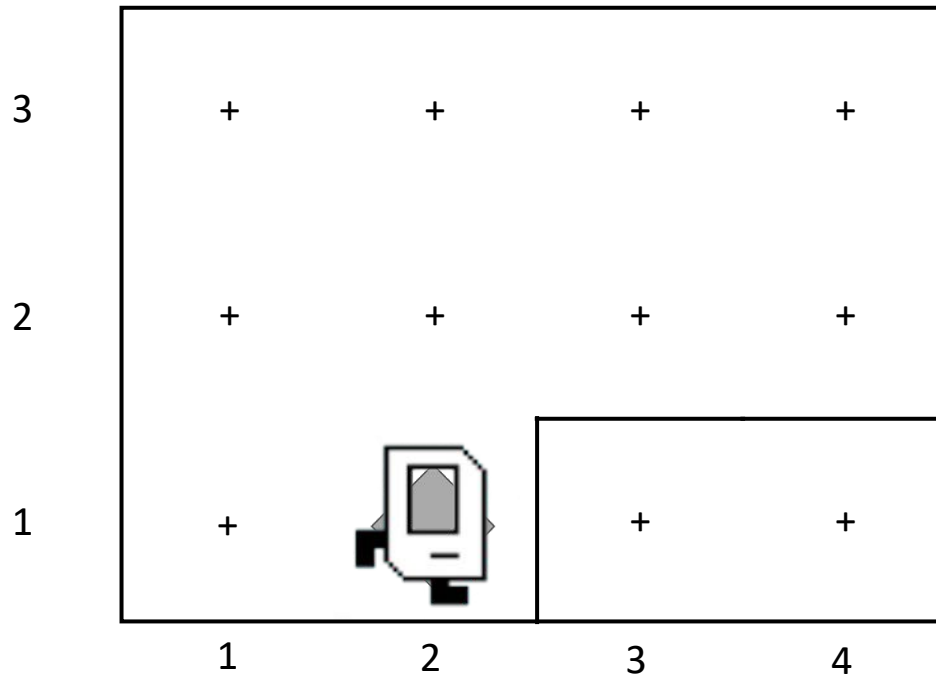


move () ;

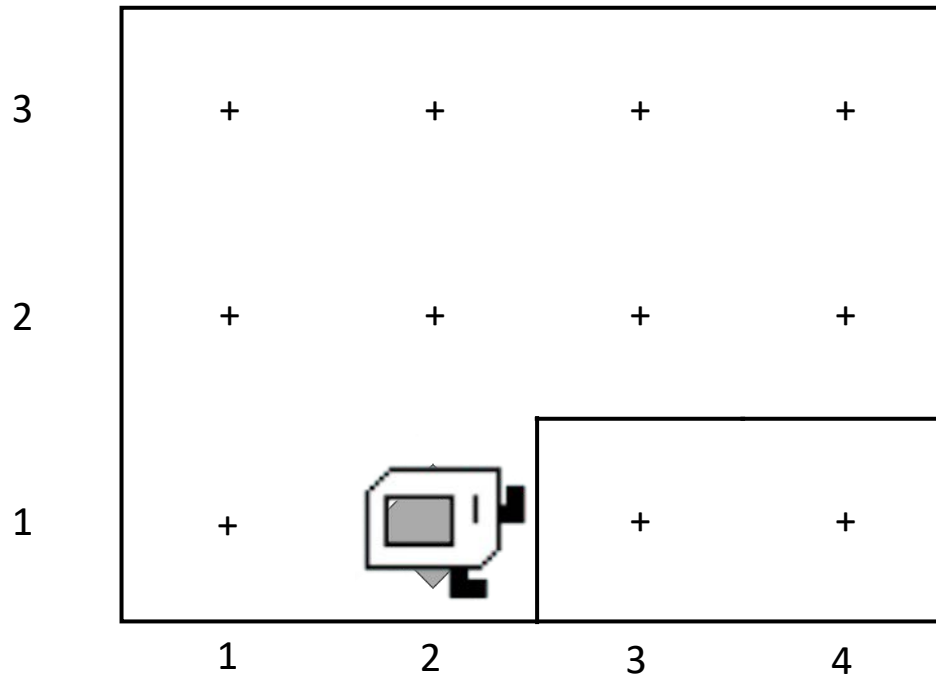


```
turnLeft();
```

turnLeft();

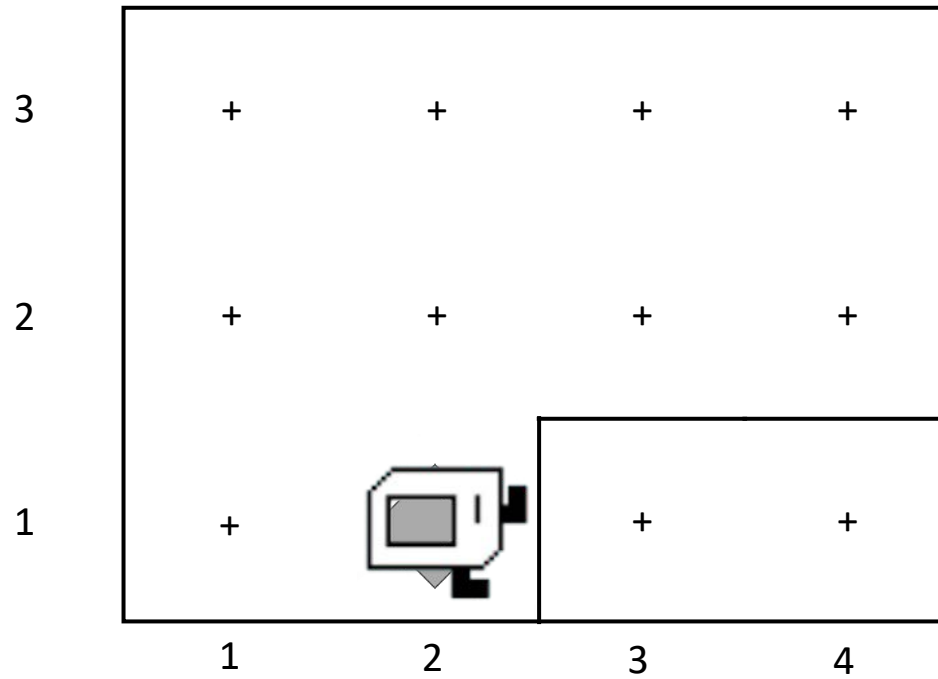


turnLeft();

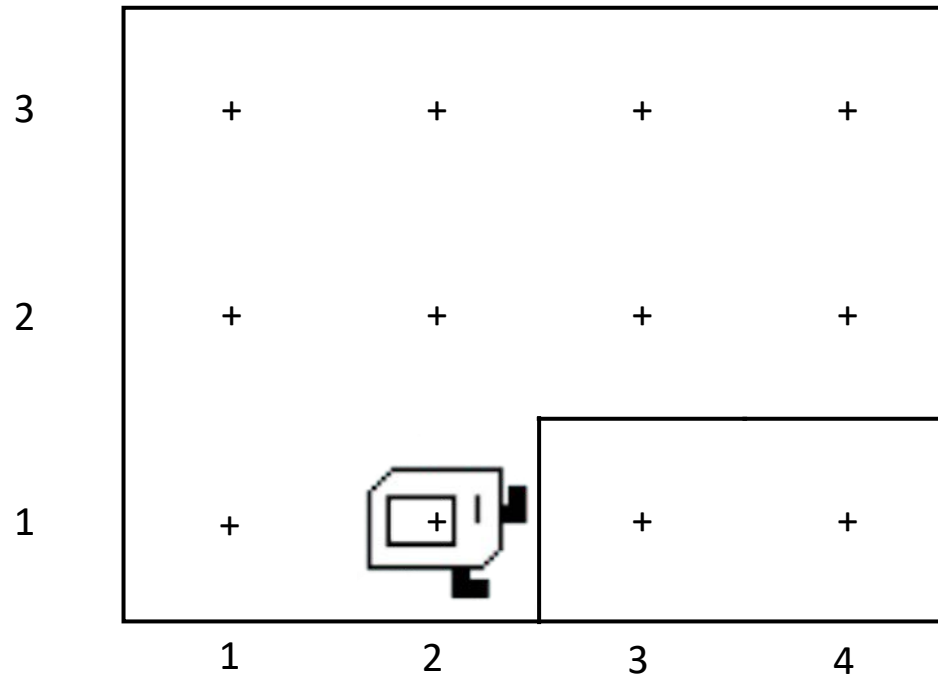


```
pickBeeper ( ) ;
```

pickBeeper();

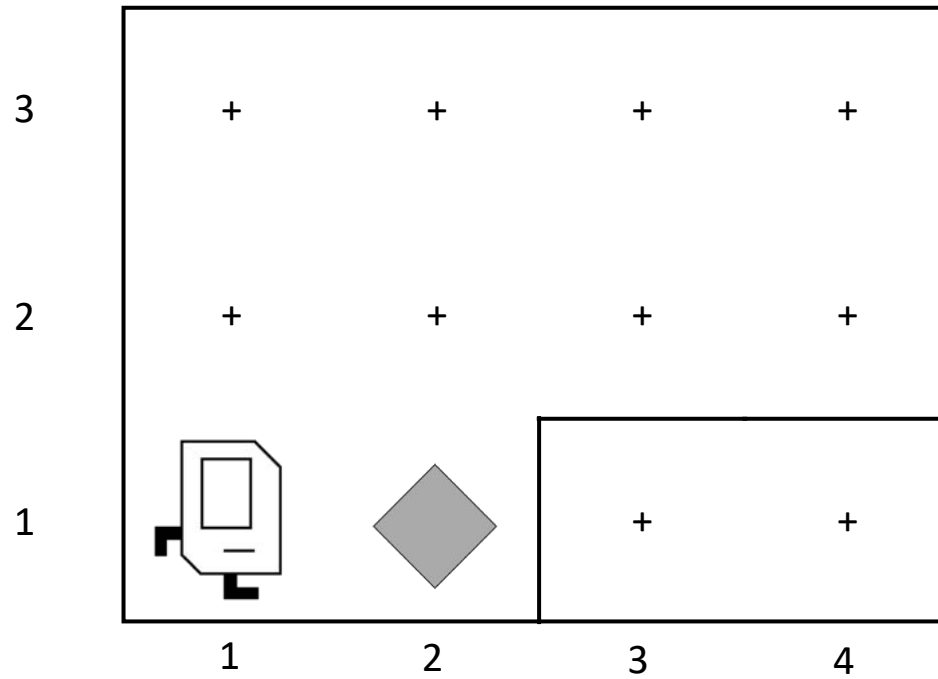


pickBeeper();

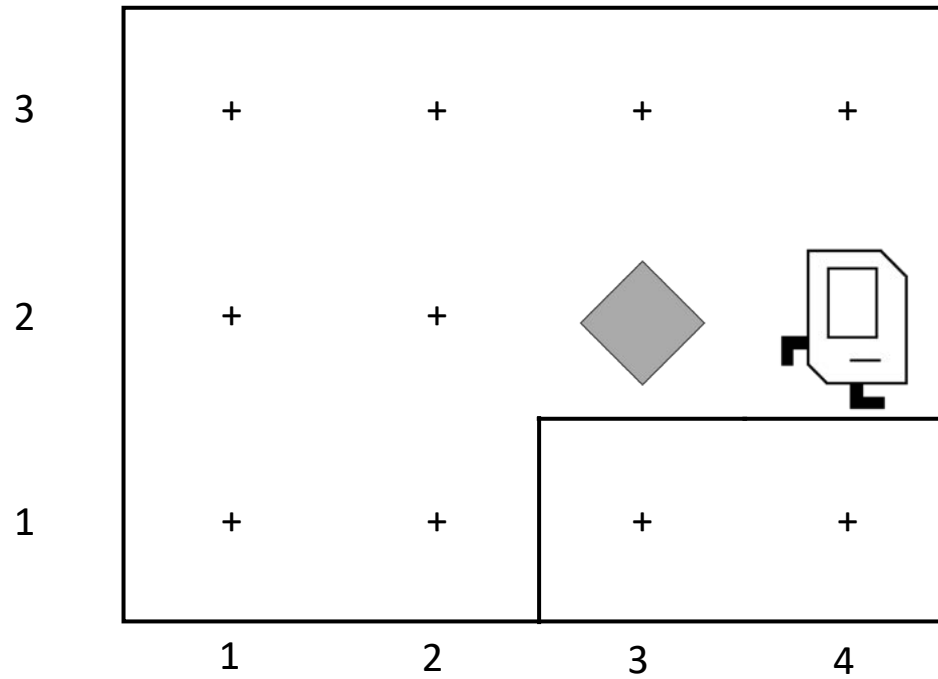


Make Sense?

First Challenge



First Challenge



Need a Volunteer



Lets Try It

A stylized graphic of an eclipse. A dark blue circle is partially obscured by a larger, lighter blue circle. The word "eclipse" is written in white, lowercase letters across the center of the circles. The background is a light blue gradient with some faint lines and a soft glow.

eclipse

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();  
    }  
}
```

This is the program's
source code

Anatomy of a Program

```
import stanford.karel.*;
```

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

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

```
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();
```

```
    }
```

```
}
```



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

Anatomy of a Program

```
import stanford.karel.*;
```

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

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

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

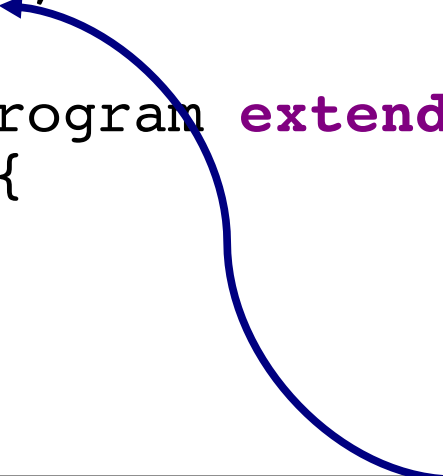
```
}
```

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();
    }
}
```



This is called an *import statement*. It tells Java what Karel is.

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();
```

```
    }
```

```
}
```

Method Definition

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

This adds a new
command to Karel's
vocabulary

Method Definition

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

This adds a new
command to Karel's
vocabulary

Method Definition

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

This adds a new
command to Karel's
vocabulary

Method Definition

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

```
}
```

This adds a new
command to Karel's
vocabulary

Method Definition

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

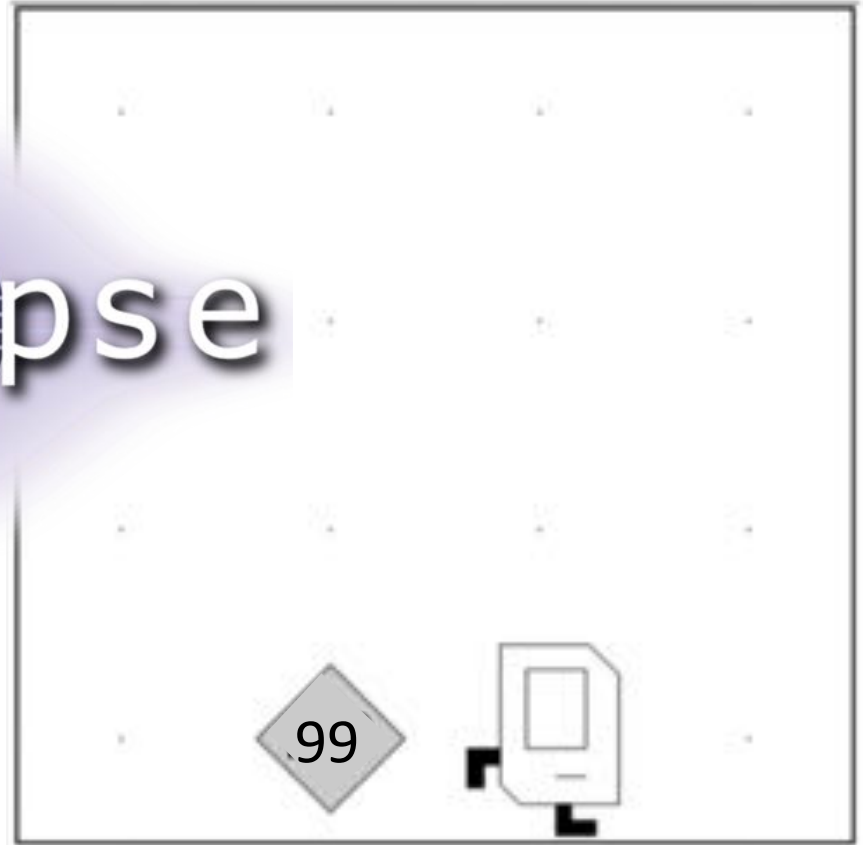
This adds a new
command to Karel's
vocabulary

Place 99 beeper?

Before



After



Place 99 beepers

```
public class Place99Beepers extends Karel {  
    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 Karel {  
    public void run() {  
        move();  
        for(int i = 0; i < 4; i++) {  
            putBeeper();  
            move();  
            turnLeft();  
        }  
    }  
}
```

CS Bridge Handouts Projects Examples Slides Bonus Forms

Karel Reader
Karel Reference

Intro to Computer Science

Joy of Building



Joy of Building



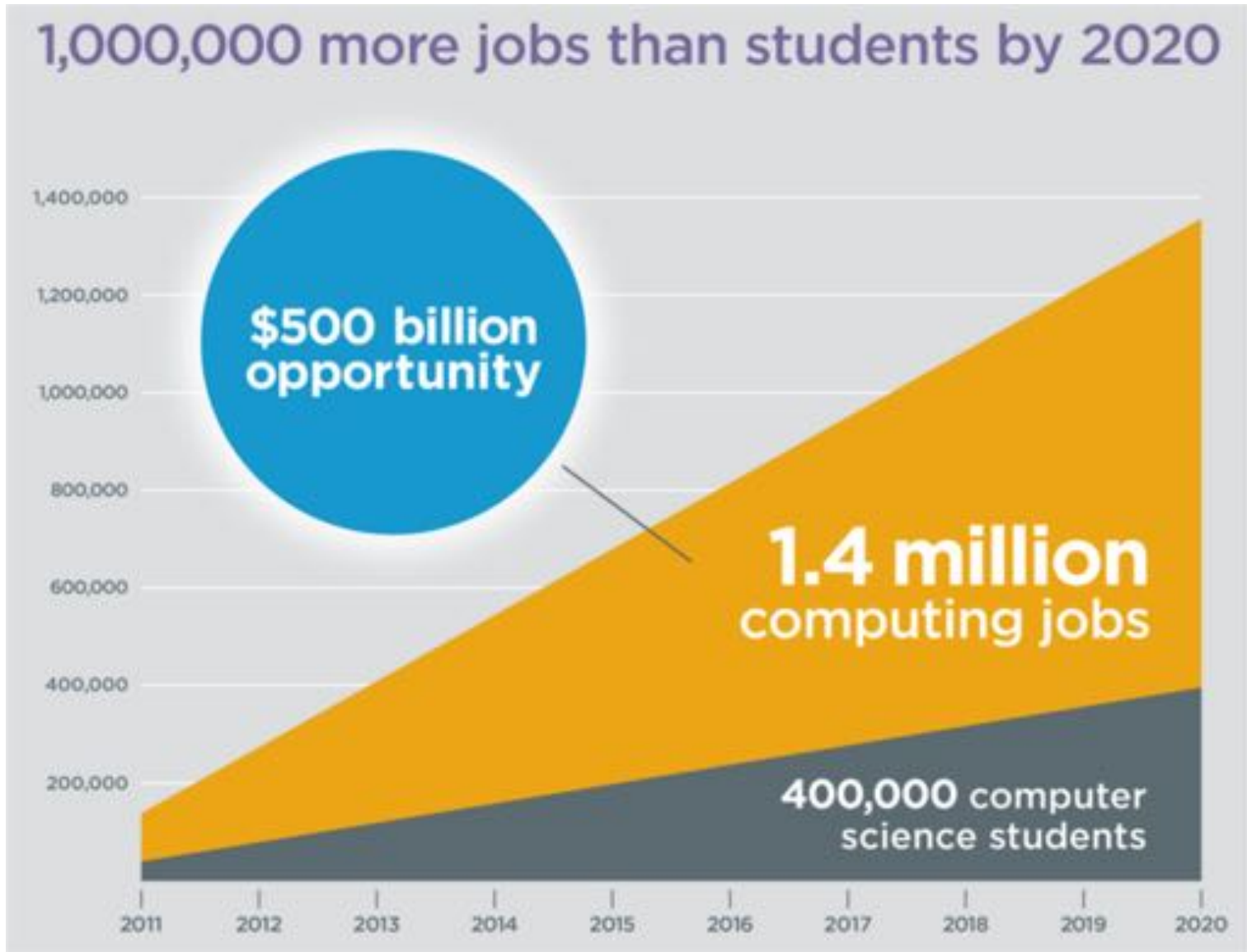
Closest Thing To Magic



Now is the Time



Oh and Its Useful



Everyone is Welcome



Learn By Doing

