CS 106A, Lecture 1

Welcome to CS 106A!

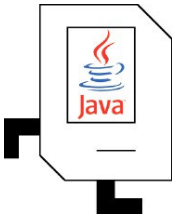
suggested reading:

Course Information handout

Karel, Ch. 1-2

Plan For Today

- Introduction
- Course Policies
- Meet Karel the Robot



Plan For Today

- Introduction
- Course Policies
- Meet Karel the Robot

What is Computer Science?

- **The art of using computing to solve complex problems.**
  - Specify *instructions* that computers execute, usually in a *programming language*
- Applicable to art, medicine, mathematics, philosophy, and more
- Touches many aspects of our daily lives

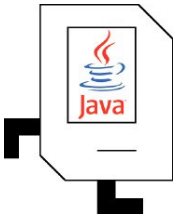



Computing is Everywhere

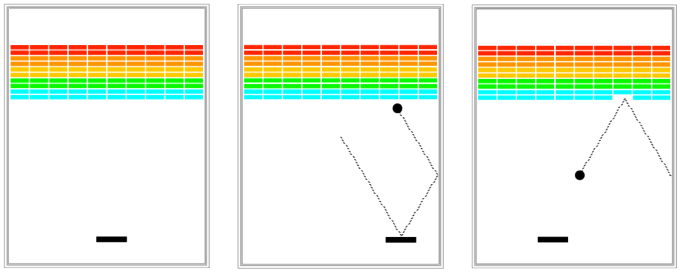
- > 3.5B users of the internet (internetlivestats.com)
- 39% owned a smartphone in 2016 (strategyanalytics.com)
- A computer recently defeated the world-champion Go player
- Machine translation has taken dramatic leaps in the past year

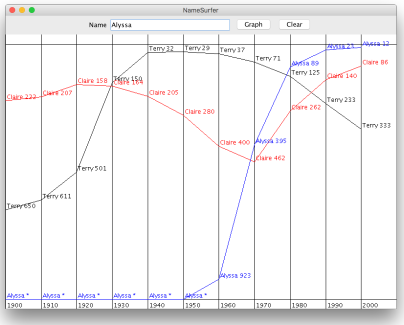
What is CS 106A?

- **Programming *Methodology***
  - Focusing on computational problem solving, not syntax
  - Uses the **Java** programming language
  - No former programming experience required!



- Topics include:
  - Karel the Robot
  - Text-based programs
  - Graphics and animation
  - Games
  - And more…





Course Website

cs106a.stanford.edu

Nice to meet you!

Instructor: *Nick Troccoli*

Head TA: *Rishi Bedi*

Section Leaders

- Helpful undergraduate assistants who will:
  - run your discussion section each week
  - grade your homework assignments and exams
  - help you when you have questions
  - ... and much more

Nice to meet you!

Aleksander Dash

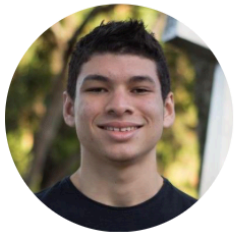
Canyon Robins

Conner Smith

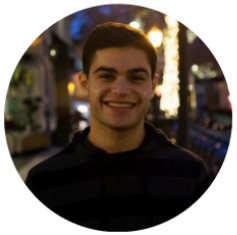
Emily Ling

Farah Uraizee

Garrick Fernandez



Gus Torres da Silva



Guy Blanc

Jared Bitz

Jestin Ma

Kate Rydberg

Katherine Erdman

Nice to meet you!

Keanu Spies

Michael Hazard

Mirae Parker

Nick Negrete

Nolan Handali

Pras Ramakrishnan

Regina Nguyen

Tobin Bell

Tyler Yep

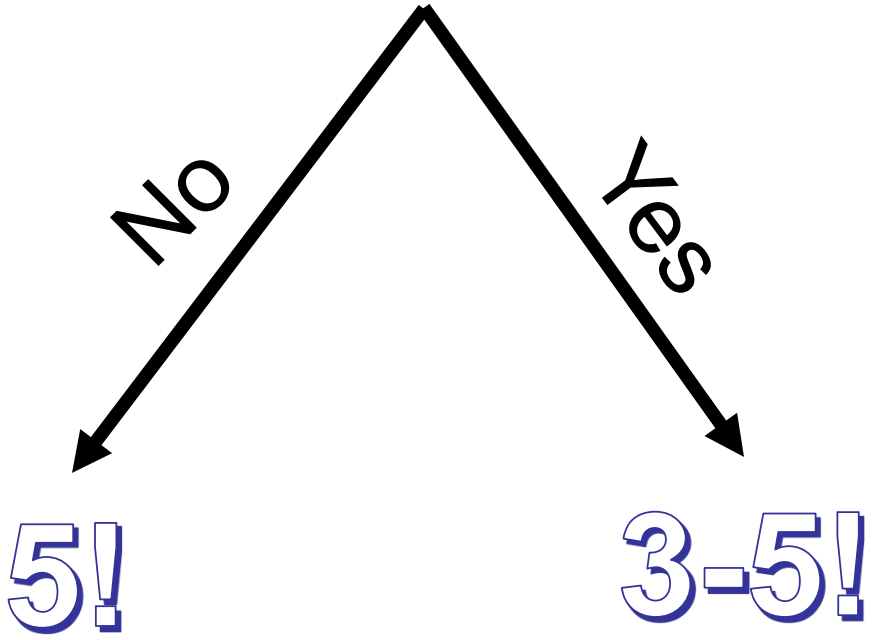
Wil Kautz

Plan For Today

- Introduction
- Course Policies
- Meet Karel the Robot

Units

Matriculated Stanford grad?

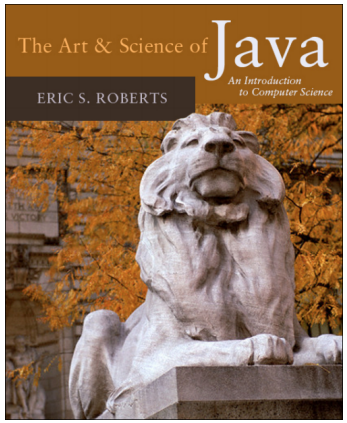


Course Website

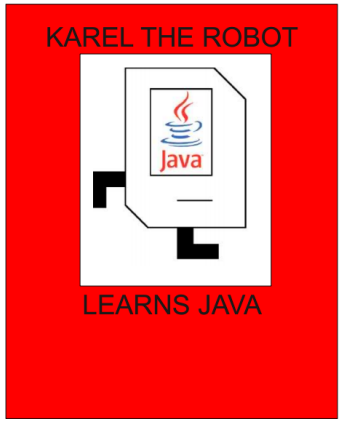
cs106a.stanford.edu

Textbooks

- *The Art & Science of Java*, by Eric Roberts
  - written here at Stanford; tailored to this course; a valuable reference
  - usable on open-book (closed-note) exams
  - available on reserve at library



- *Karel the Robot Learns Java*, coursereader (35 pages)
  - used this and next week as we introduce coding
  - usable on open-book (closed-note) exams
  - Free PDF available online



Grading

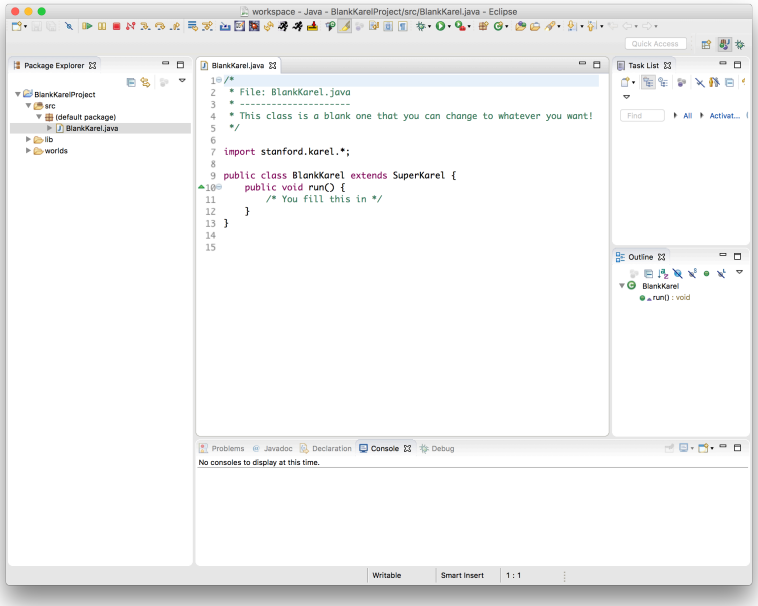
******	45%	Programming assignments
*	10%	Section Participation
**	15%	Midterm Exam
****	30%	Final Exam

Grading

******	45%	Programming assignments
*	10%	Section Participation
**	15%	Midterm Exam
****	30%	Final Exam

Programming Assignments

- 6 programming assignments (some individual, some in **pairs**), completed using **Eclipse**
  - Free software, available on course website
  - **Homework:** set up Eclipse!
  - Come to LaIR this **Wed. 7-11PM** for troubleshooting



- graded on **functionality** (behavior) and **style** (elegance)
  - Interactive grading sessions every week
  - grading scale is divided into "buckets"

The Bucket System

√	satisfactory; meets requirements, maybe a few issues

The Bucket System

√+	Well done; satisfies all assignment requirements
√	satisfactory; meets requirements, maybe a few issues

The Bucket System

√+	Well done; satisfies all assignment requirements
√	satisfactory; meets requirements, maybe a few issues
√-	Problems serious enough to fall short of assignment requirements

The Bucket System

+	Exceeds expectations; often reflects additional work
√+	Well done; satisfies all assignment requirements
√	satisfactory; meets requirements, maybe a few issues
√-	Problems serious enough to fall short of assignment requirements

The Bucket System

+	Exceeds expectations; often reflects additional work
√+	Well done; satisfies all assignment requirements
√	satisfactory; meets requirements, maybe a few issues
√-	Problems serious enough to fall short of assignment requirements
-	Extremely serious problems, a little effort and understanding

The Bucket System

++	Absolutely fantastic submission (*very rare)*
+	Exceeds expectations; often reflects additional work
√+	Well done; satisfies all assignment requirements
√	satisfactory; meets requirements, maybe a few issues
√-	Problems serious enough to fall short of assignment requirements
-	Extremely serious problems, a little effort and understanding

The Bucket System

++	Absolutely fantastic submission (*very rare)*
+	Exceeds expectations; often reflects additional work
√+	Well done; satisfies all assignment requirements
√	satisfactory; meets requirements, maybe a few issues
√-	Problems serious enough to fall short of assignment requirements
-	Extremely serious problems, a little effort and understanding
--	Little effort

The Bucket System

++	Absolutely fantastic submission (*very rare)*
+	Exceeds expectations; often reflects additional work
√+	Well done; satisfies all assignment requirements
√	satisfactory; meets requirements, maybe a few issues
√-	Problems serious enough to fall short of assignment requirements
-	Extremely serious problems, a little effort and understanding
--	Little effort
0	No submission

Getting Help

- Visit the SLs in the **LaIR** (1st floor of Tresidder Union)
  - open Sun-Wed, 7PM – 11PM, starting this Wednesday
  - staffed with multiple section leaders to answer questions

- other help resources:
  - instructor office hours
  - head TA office hours
  - email SL, TA, instructor

- Eclipse troubleshooting session **Wednesday 6/28 7-11PM @ LaIR**

2 Minds are Better Than 1

- Some assignments may optionally be done in **pairs**
- Both partners receive the same grade
- A chance to brainstorm ideas and work with another programmer
- **MUST be in the same section!**
- More info in handout #1 and on the course website

Interactive Grading

- For each assignment (except for the last), you will get feedback via an **Interactive Grading** (IG) session, scheduled with your section leader.
- Go over assignment feedback, strengths, things to improve

Late Days

- **Start out with 3 “free late days”**: each late day allows you to submit an assignment 24 hours late without penalty.
- Hard deadline 48 hours after original due date
- 1-bucket deduction per day late after late days are exhausted
- Pair late days are assessed individually
- ”Pre-granted extensions” – additional extensions granted only in *very special* circumstances. **Head TA** must approve extensions.

Grading

******	45%	Programming assignments
*	10%	Section Participation
**	15%	Midterm Exam
****	30%	Final Exam

Discussion Sections

- Weekly 50-minute sections led by your section leader
- Go over lecture material, do practice problems, answer questions
- Graded on section attendance + participation (+IG attendance)
- **Homework**: sign up for section on the course website!

Grading

******	45%	Programming assignments
*	10%	Section Participation
**	15%	Midterm Exam
****	30%	Final Exam

Exams

- **Midterm exam** – Monday, July 24th, 7-9PM
  - Contact me by *July 17* if you have an academic or University conflict
- **Final exam** – Friday, August 18th, 12:15-3:15PM
  - No alternate final! You ***MUST*** be able to take the final exam at the scheduled time.

- Both exams are *open-book, closed-notes, closed-electronic-device*. You will be provided with a syntax reference sheet.

Grading

******	45%	Programming assignments
*	10%	Section Participation
**	15%	Midterm Exam
****	30%	Final Exam

Stanford Honor Code

- The **Honor Code** is an undertaking of the students, individually and collectively:
  - that they will not give or receive aid in examinations; that they will not give or receive unpermitted aid in class work, in the preparation of reports, or in any other work that is to be used by the instructor as the basis of grading;
  - that they will do their share and take an active part in seeing to it that others as well as themselves uphold the spirit and letter of the Honor Code.

- The faculty on its part manifests its confidence in the honor of its students by refraining from proctoring examinations and from taking unusual and unreasonable precautions to prevent the forms of dishonesty mentioned above. The faculty will also avoid, as far as practicable, academic procedures that create temptations to violate the Honor Code.

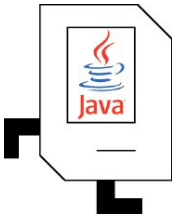
- While the faculty alone has the right and obligation to set academic requirements, the students and faculty will work together to establish optimal conditions for honorable academic work.

see also: http://honorcode.stanford.edu/

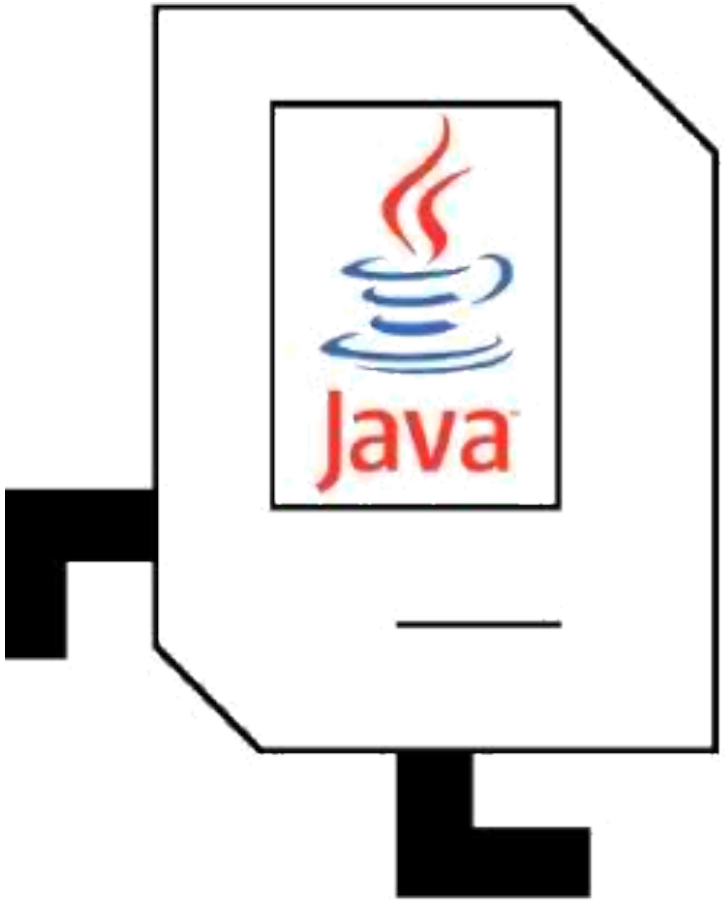
Honor Code and CS 106A

- Please help us ensure academic integrity:
  - Indicate any assistance received on HW (books, web sites, friends).
  - Do not look at other people's solution code *(outside of your pair)*.
  - Do not give your solution code to others, or post it on the web.
  - Report any inappropriate activity you see performed by others.
- Assignments are checked regularly for similarity with help of software tools.
- If you realize that you have made a mistake, you may retract your submission to any assignment at any time, no questions asked.
- If you need help, please contact us and we will help you.
  - We do not want you to feel any pressure to violate the Honor Code in order to succeed in this course.

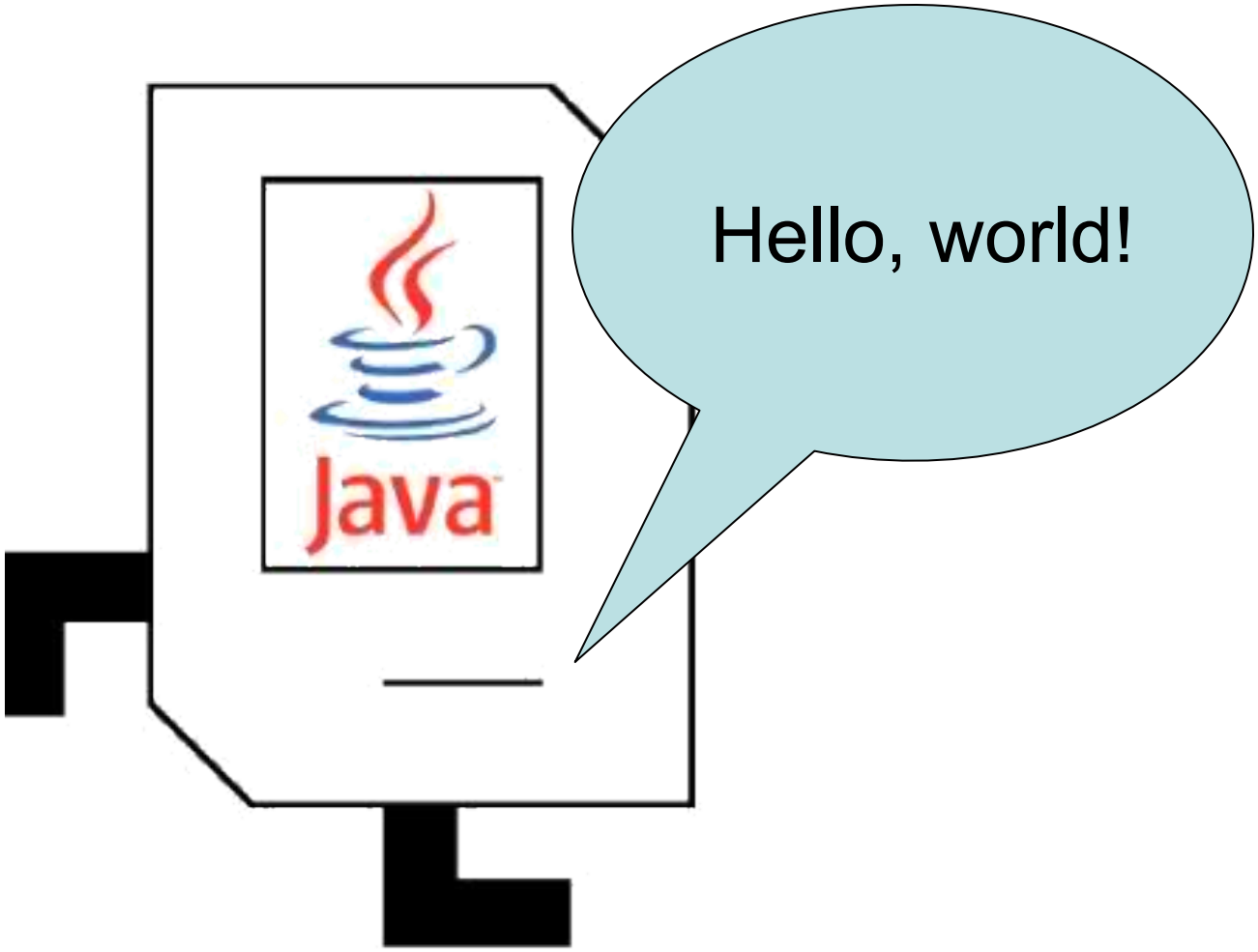
Plan For Today

- Introduction
- Course Policies
- Meet Karel the Robot 

Meet Karel the Robot!

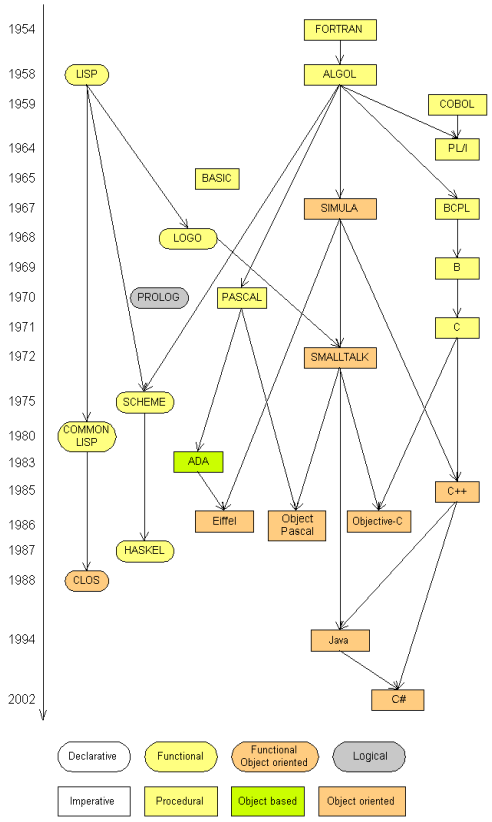


Meet Karel the Robot!

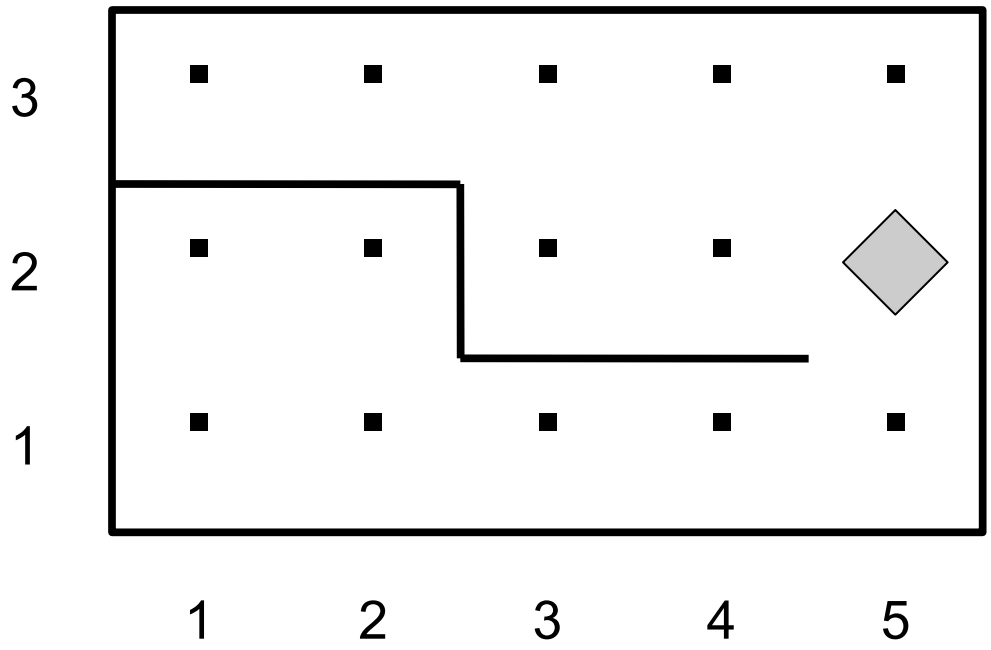


Programming languages

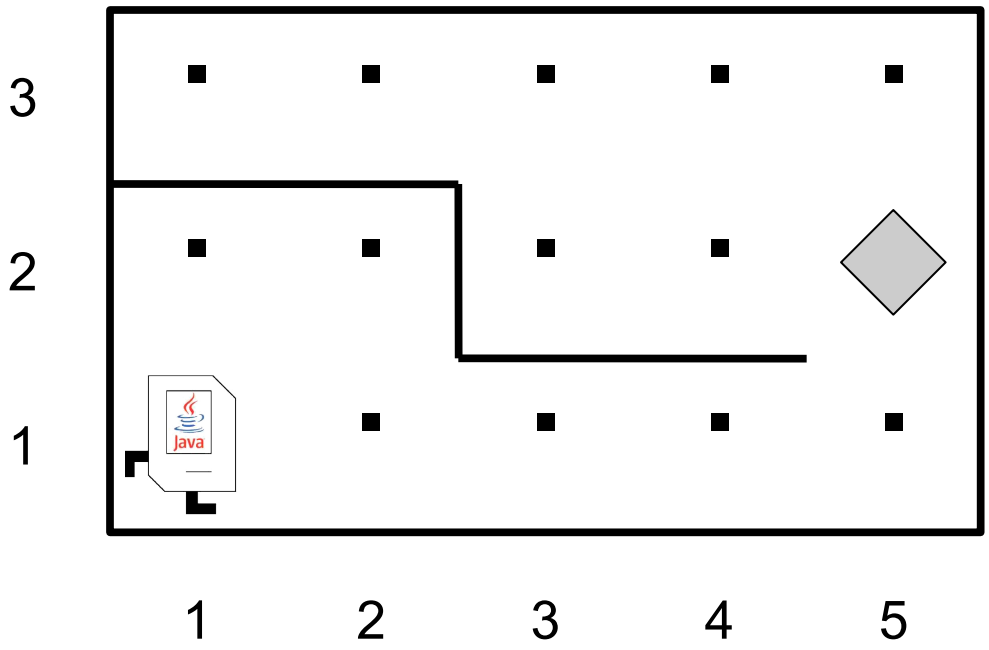
- *procedural languages*: programs are a series of commands
  - **Pascal** (1970): designed for education
  - **C** (1972): low-level operating systems and devices
- *functional programming*: functions map inputs to outputs
  - **Lisp** (1958) / **Scheme** (1975), **ML** (1973), **Haskell** (1990)
- *object-oriented languages*: programs use interacting "objects"
  - **Smalltalk** (1980): first major object-oriented language
  - **C++** (1985): "object-oriented" improvements to C
    - successful in industry; used to build OSes such as Windows
  - **Java** (1995): designed for embedded systems, web apps
    - Runs on many platforms (Windows, Mac, Linux, cell phones...)
    - The language taught in this course and our textbook



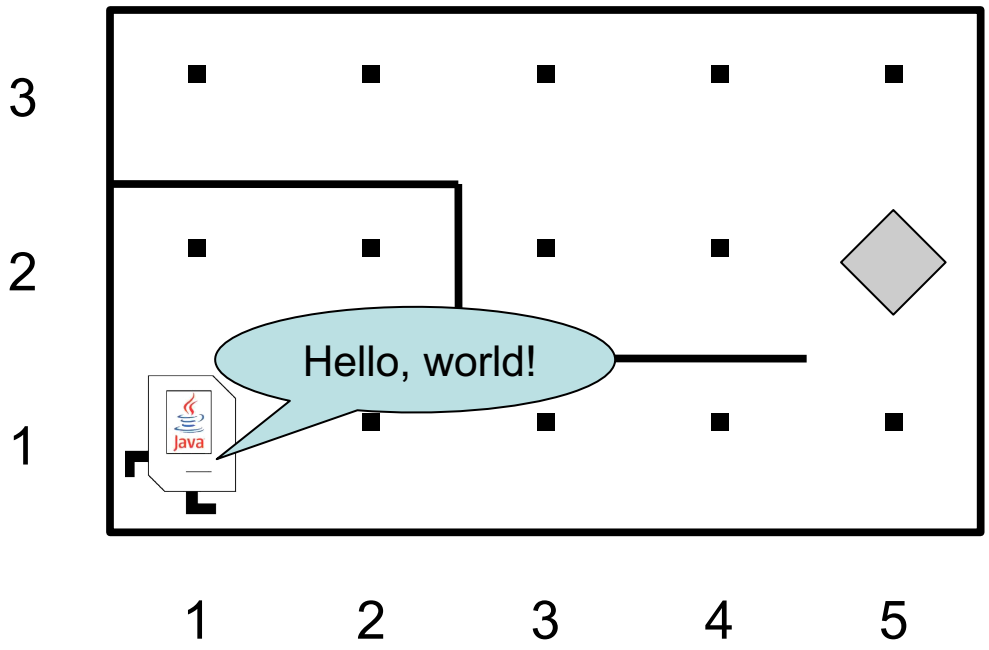
Karel's World



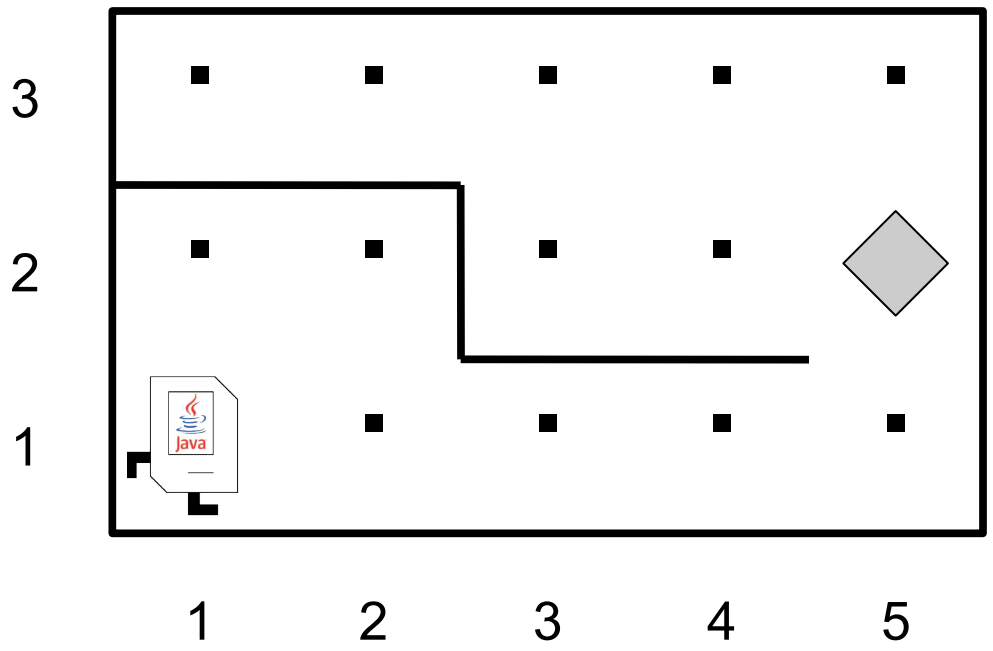
Karel's World



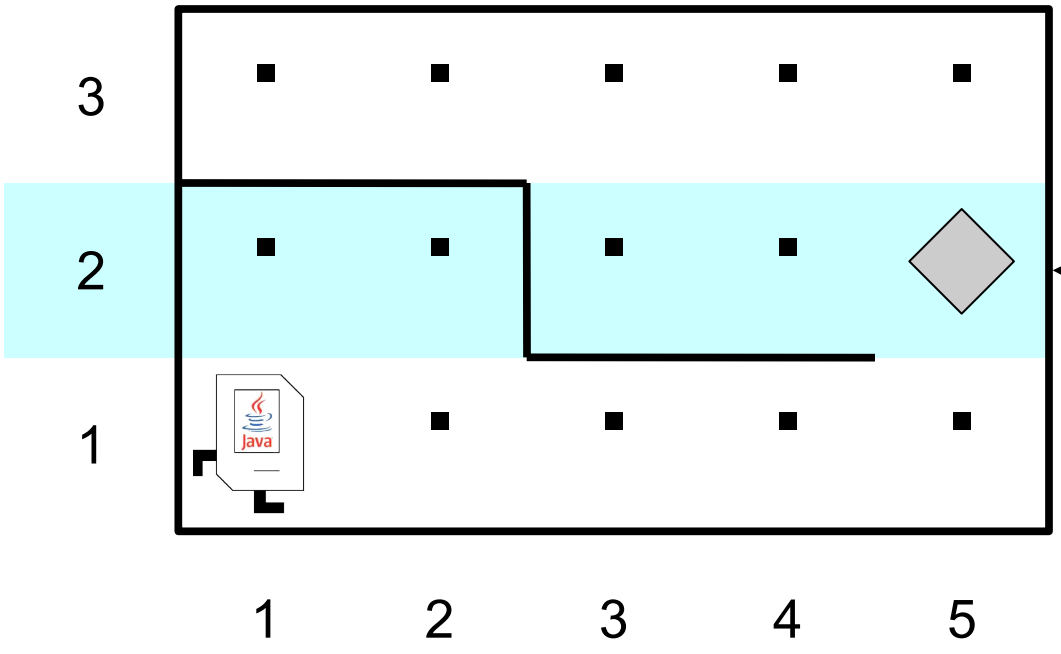
Karel's World



Karel's World

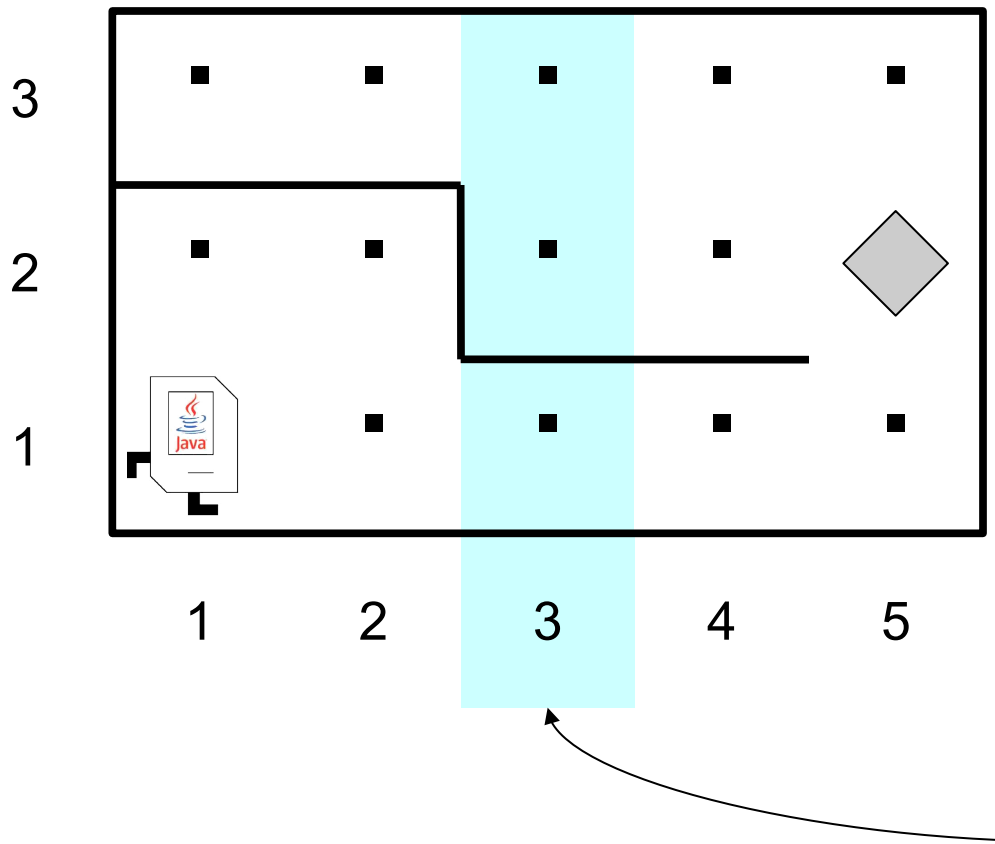


Streets (rows)



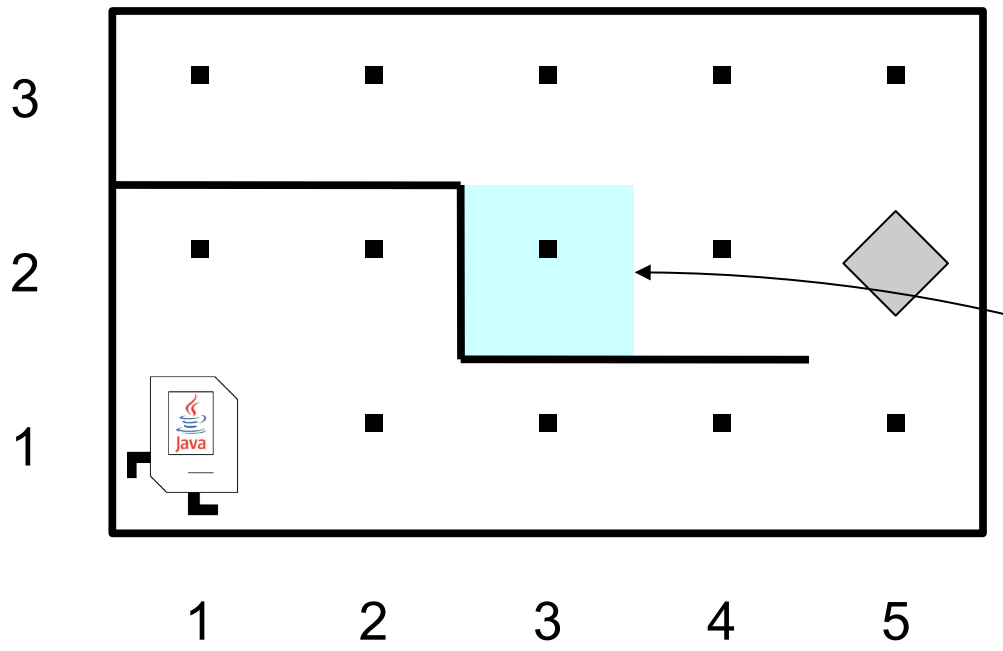
Each row is called a street.

Avenues (columns)



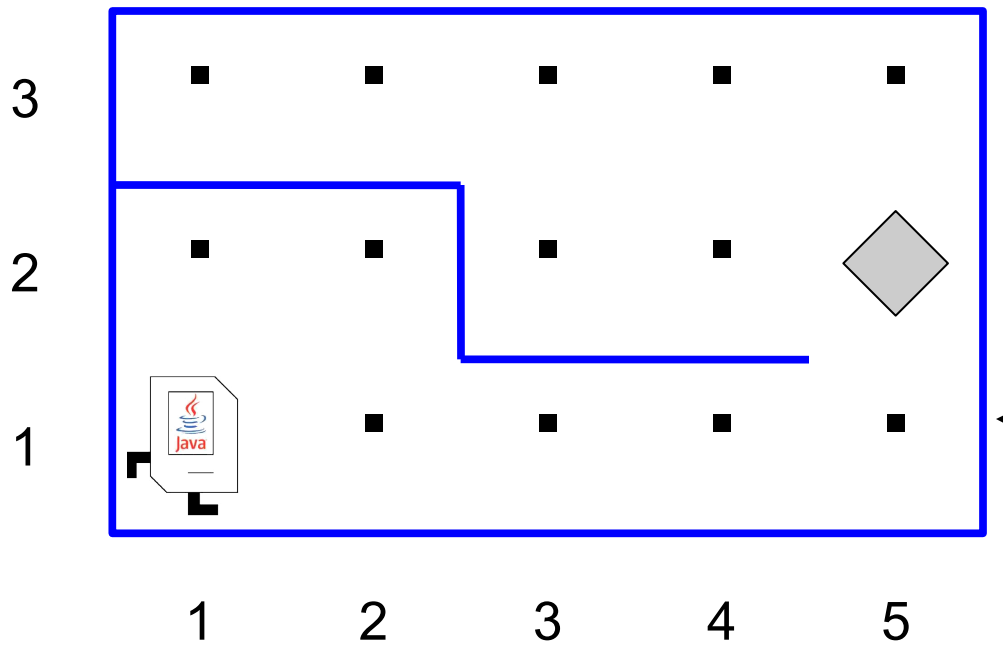
Each column is called an avenue.

Corners (locations)



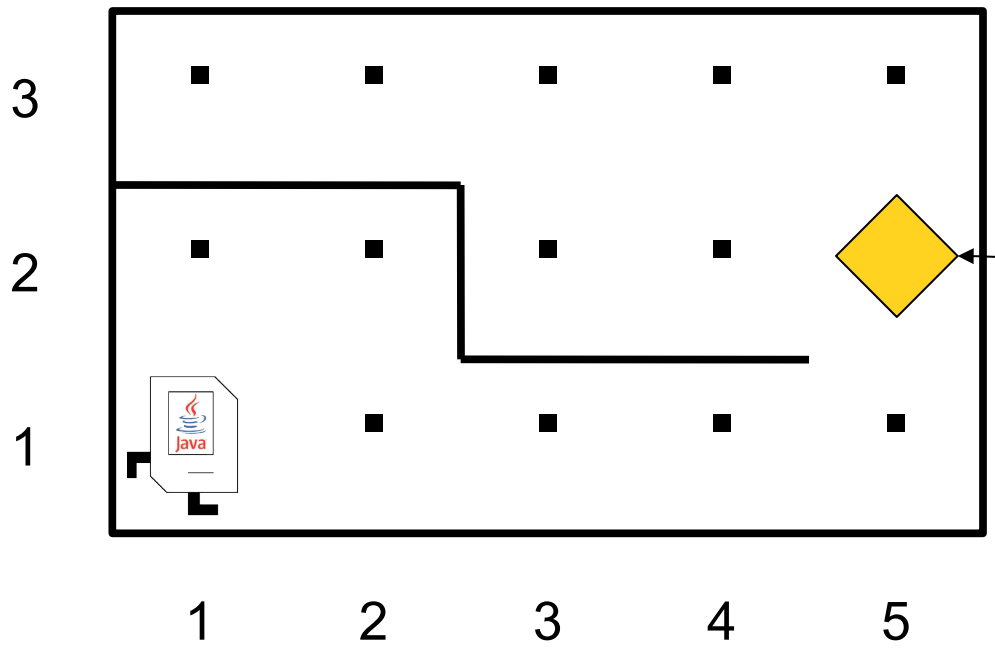
The intersection of a street and an avenue is a corner.

Walls



Karel cannot move through walls.

Beepers



Beepers mark locations in Karel's world. Karel can pick them up and put them down.

Karel Knows 4 Commands

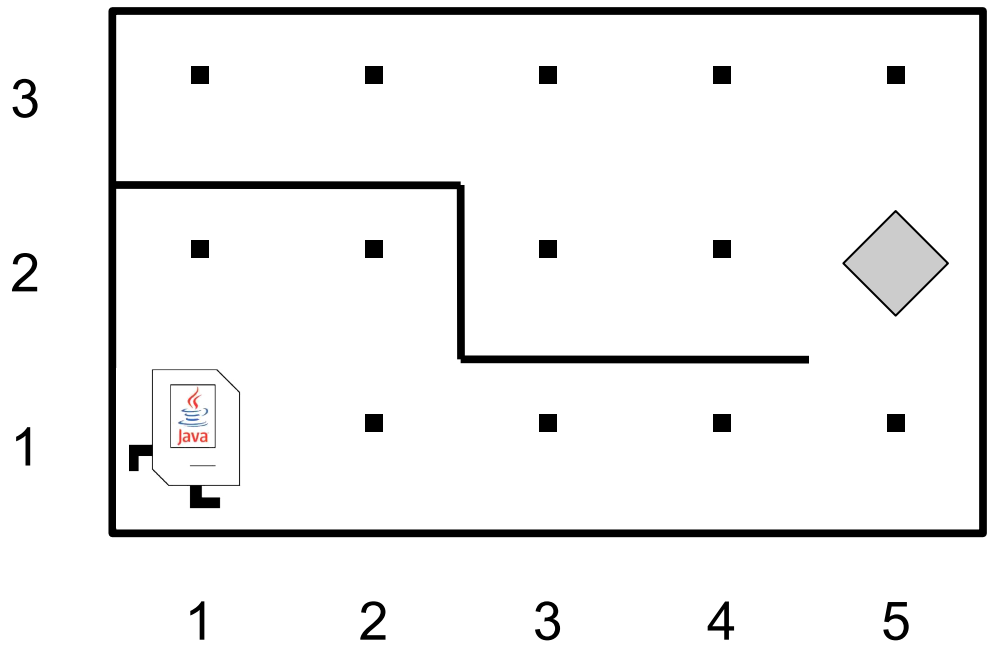
```
move

turnLeft

putBeeper

pickBeeper
```

Karel commands: move



Karel Commands

move

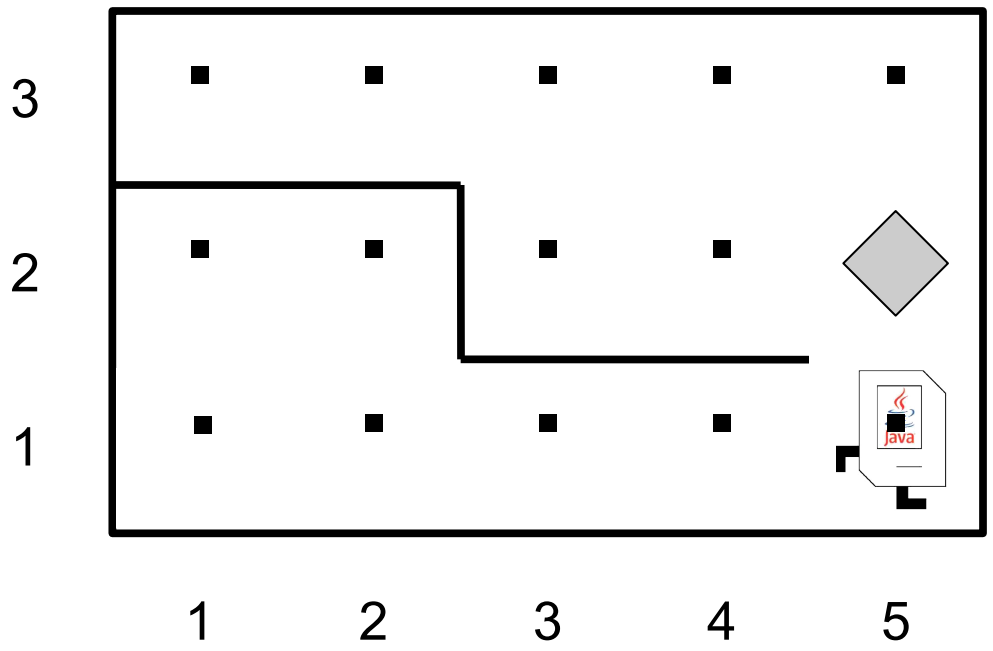
turnLeft

pickBeeper

putBeeper

– move makes Karel move forward one square in the direction it is facing.

Commands: turnLeft

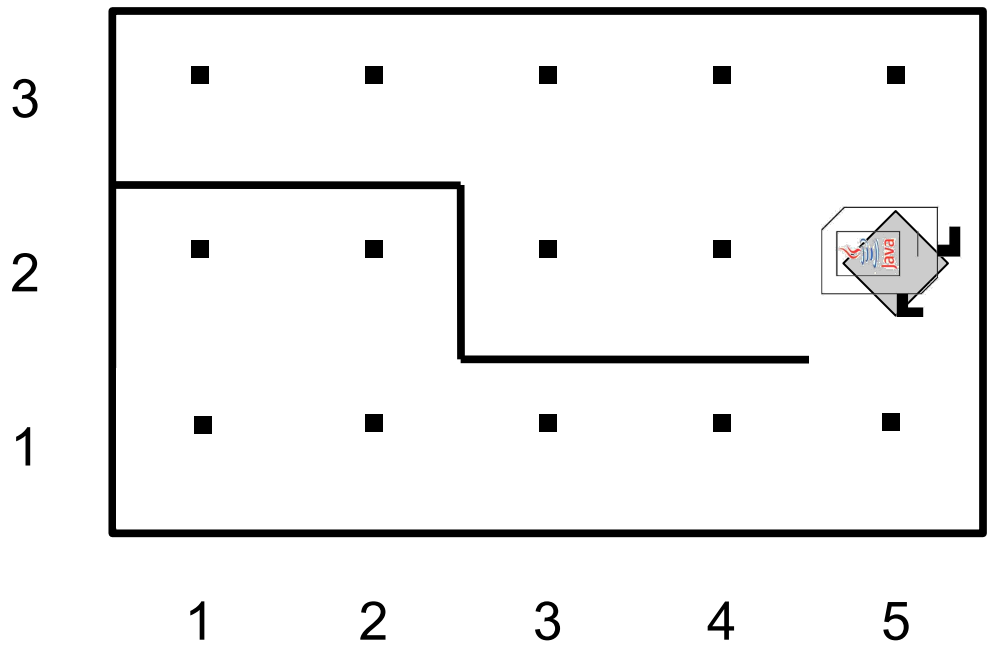


Karel Commands

`move`
`turnLeft`
`pickBeeper`
`putBeeper`

- `turnLeft` makes Karel rotate 90° counter-clockwise.
- There is no `turnRight` command. (Why not?)

Commands: pickBeeper

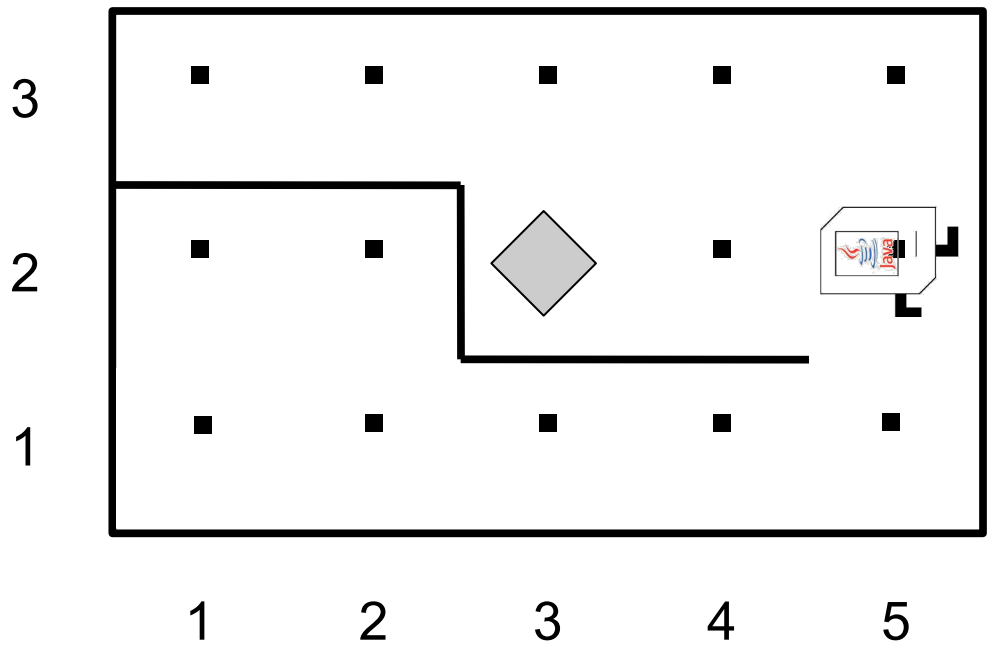


Karel Commands

```
move
turnLeft
pickBeeper
putBeeper
```

- `pickBeeper` makes Karel pick up the beeper at the current corner. Karel can hold multiple beepers at a time in its "beeper bag".

Commands: putBeeper

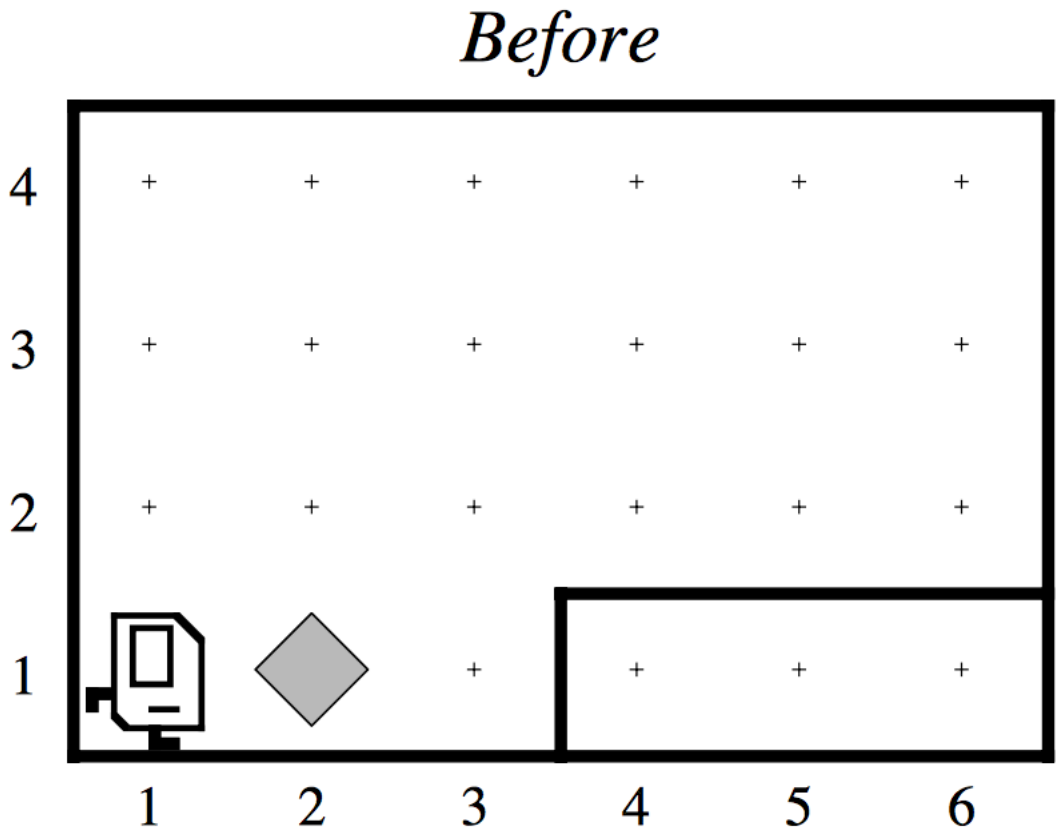


Karel Commands

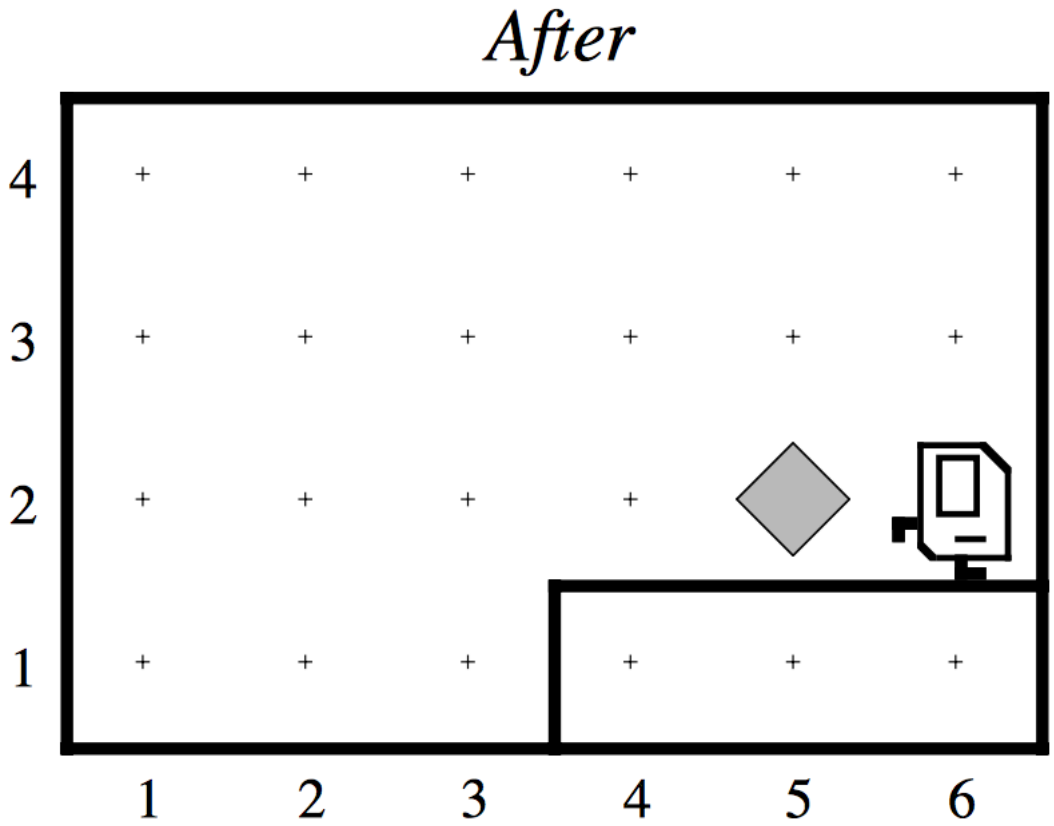
```
move
turnLeft
pickBeeper
putBeeper
```

- `putBeeper` makes Karel put a beeper down at its current location.
  - `pickBeeper` and `putBeeper` are used to move beepers around.

Our First Karel Program

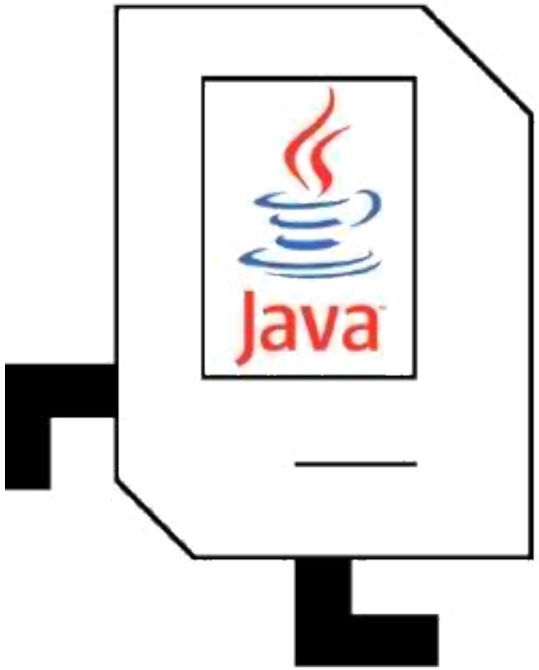


Our First Karel Program

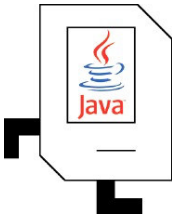


Karel Summary

- Karel lives in a grid world with walls and beepers.
- 4 commands: **move**, **turnLeft**, **pickBeeper** and **putBeeper**.
- Seems simple; but you can do amazing things with Karel!



Wrap-up

- Introduction √
- Course Policies √
- Meet Karel the Robot  √

Next time: more programming with Karel!