Student-Computer Interaction Design

for Introductory Computer Science

John DeNero

How A Computer Science Course Begins

Photos by Brian Ly, Daily Cal Staff

How A Computer Science Course Begins

Photos by Brian Ly, Daily Cal Staff

How Students Actually Learn to Program

Photo by Alice Oh, Daily Cal Staff

Student–Computer Interaction Design

Projects & Assignments	Lab & Discussion	Lecture & Reading	Question & Answer

Student–Computer Interaction Design

Projects & Assignments	Lab & Discussion	Lecture & Reading	Question & Answer

Design Principle: Create interactions that are consistently productive and challenging.

Student–Computer Interaction Design

Projects & Assignments	Lab & Discussion	Lecture & Reading	Question & Answer

Design Principle: Create interactions that are consistently productive and challenging.

- No prolonged periods of frustration or confusion

Student–Computer Interaction Design

Projects & Assignments	Lab & Discussion	Lecture & Reading	Question & Answer

Design Principle: Create interactions that are consistently productive and challenging.

- No prolonged periods of frustration or confusion
- Every distinct activity involves a new idea

Student–Computer Interaction Design

Projects & Assignments	Lab & Discussion	Lecture & Reading	Question & Answer

Design Principle: Create interactions that are consistently productive and challenging.

- No prolonged periods of frustration or confusion
- Every distinct activity involves a new idea
- Students solve problems that they didn't think they could solve (especially ones worth solving)

Student–Computer Interaction Design

Projects & Assignments	Lab & Discussion	Lecture & Reading	Question & Answer

Design Principle: Create interactions that are consistently productive and challenging.

- No prolonged periods of frustration or confusion
- Every distinct activity involves a new idea
- Students solve problems that they didn't think they could solve (especially ones worth solving)

Design from the Student's Perspective

Student–Computer Interaction Design

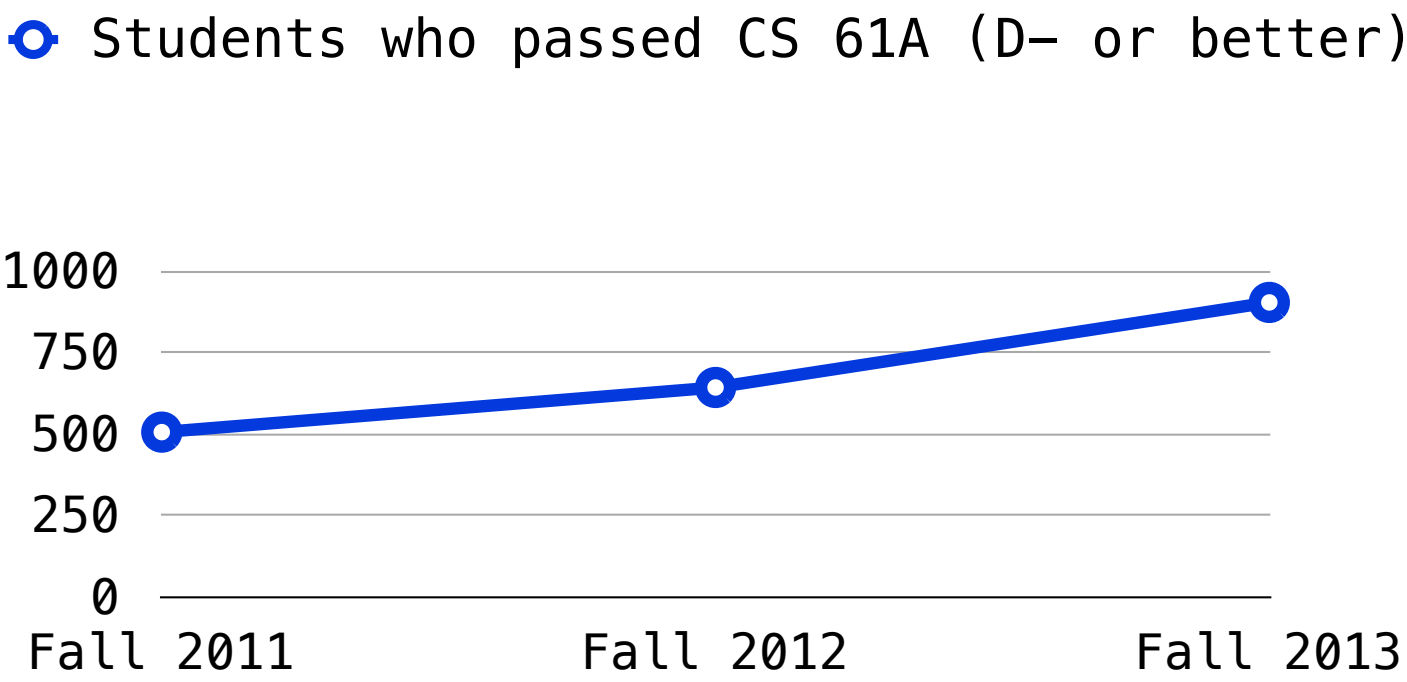
Projects & Assignments	Lab & Discussion	Lecture & Reading	Question & Answer

Design Principle: Create interactions that are consistently productive and challenging.

Student–Computer Interaction Design

Projects & Assignments	Lab & Discussion	Lecture & Reading	Question & Answer

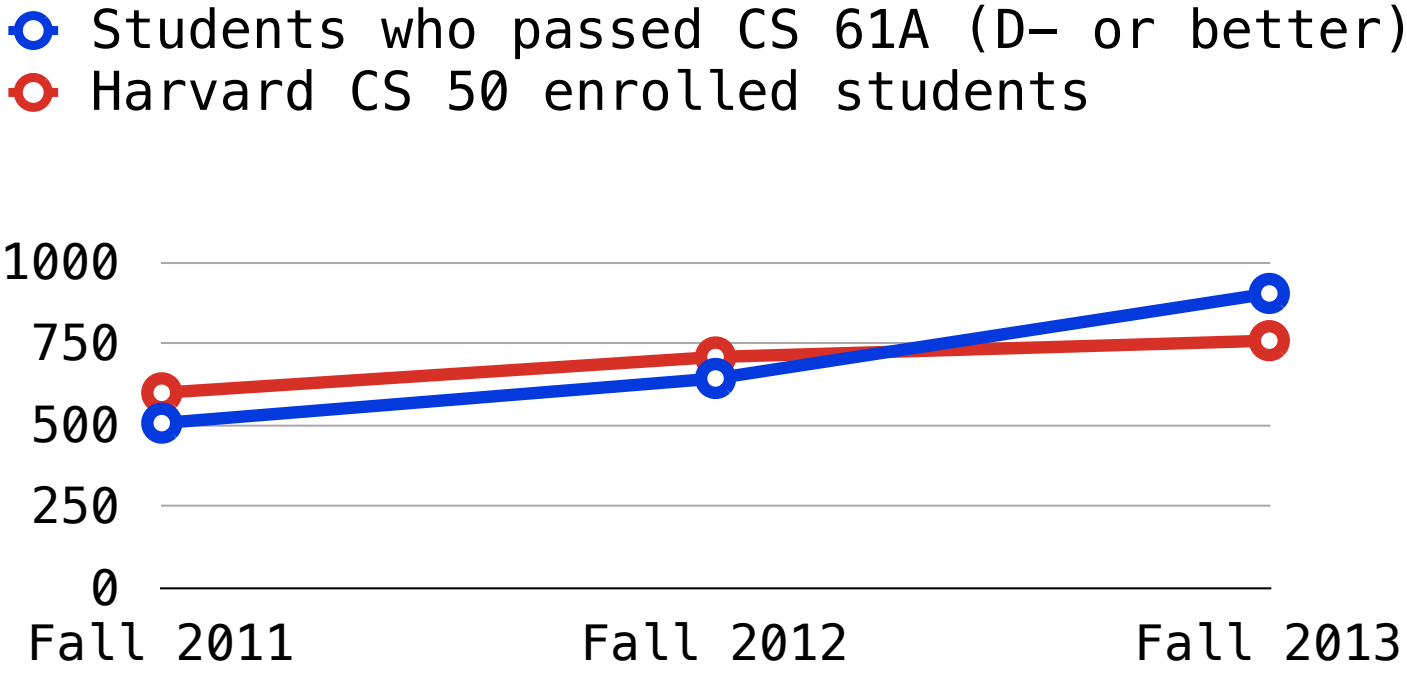
Design Principle: Create interactions that are consistently productive and challenging.



Student–Computer Interaction Design

Projects & Assignments	Lab & Discussion	Lecture & Reading	Question & Answer

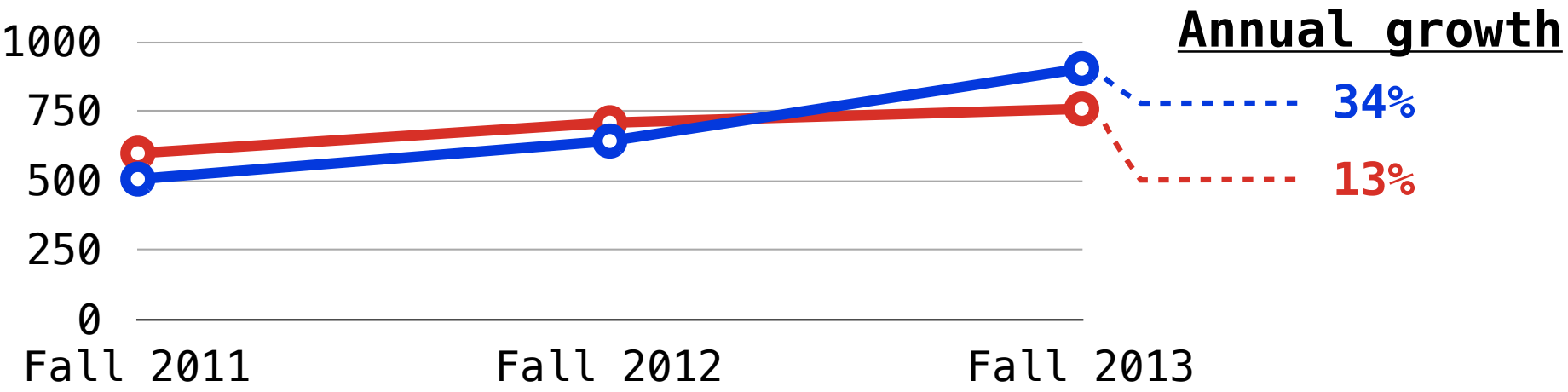
Design Principle: Create interactions that are consistently productive and challenging.



Student–Computer Interaction Design

Projects & Assignments	Lab & Discussion	Lecture & Reading	Question & Answer

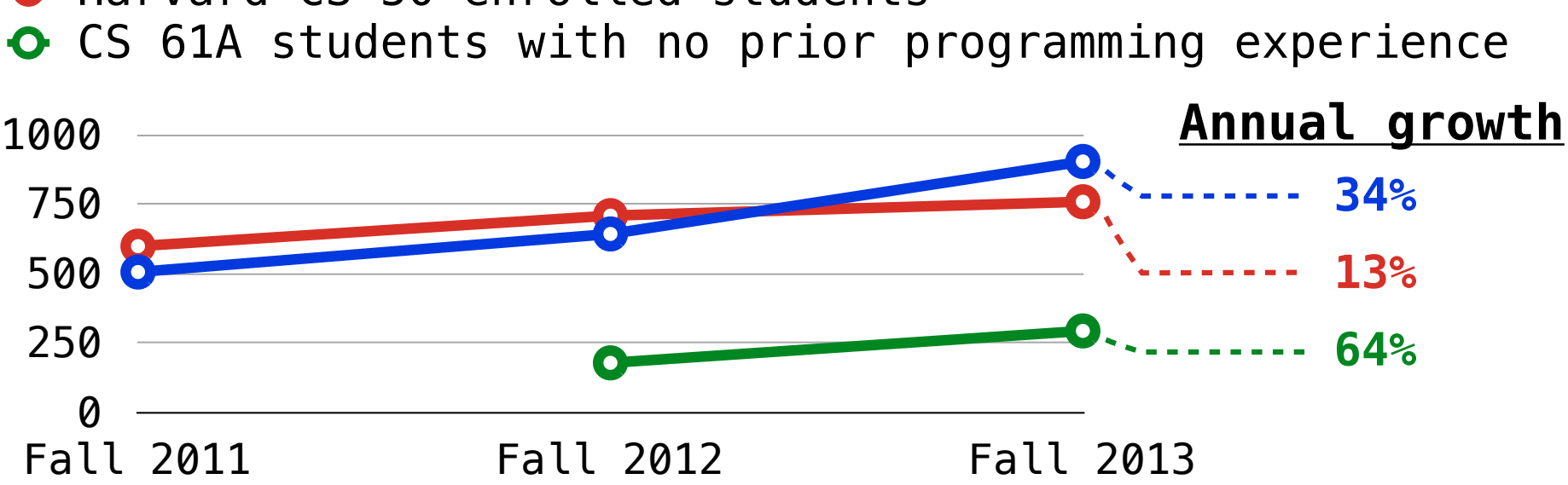
Design Principle: Create interactions that are consistently productive and challenging.



Student–Computer Interaction Design

Projects & Assignments	Lab & Discussion	Lecture & Reading	Question & Answer

Design Principle: Create interactions that are consistently productive and challenging.



Programming Projects

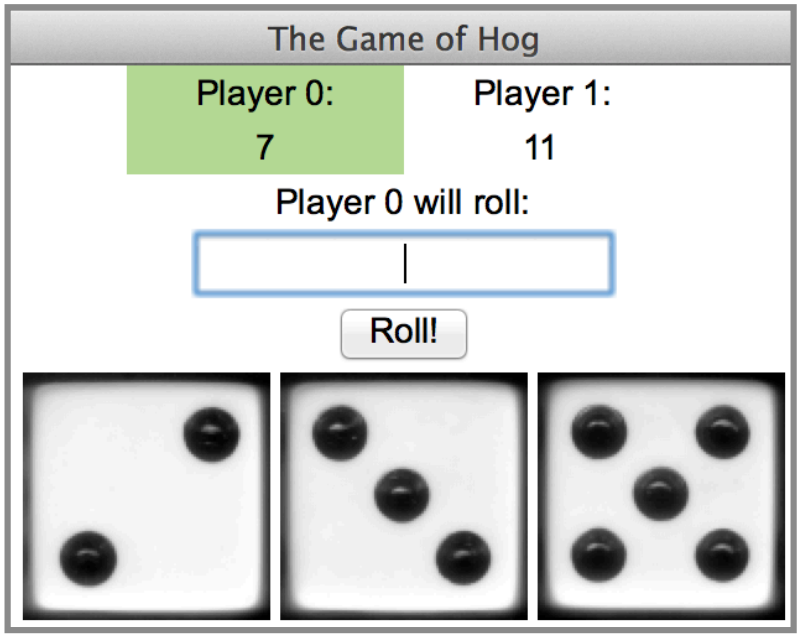
Scaffolded Programming Projects

Scaffolded Programming Projects

Fill-in-the-blank starter code and a full test suite

Scaffolded Programming Projects

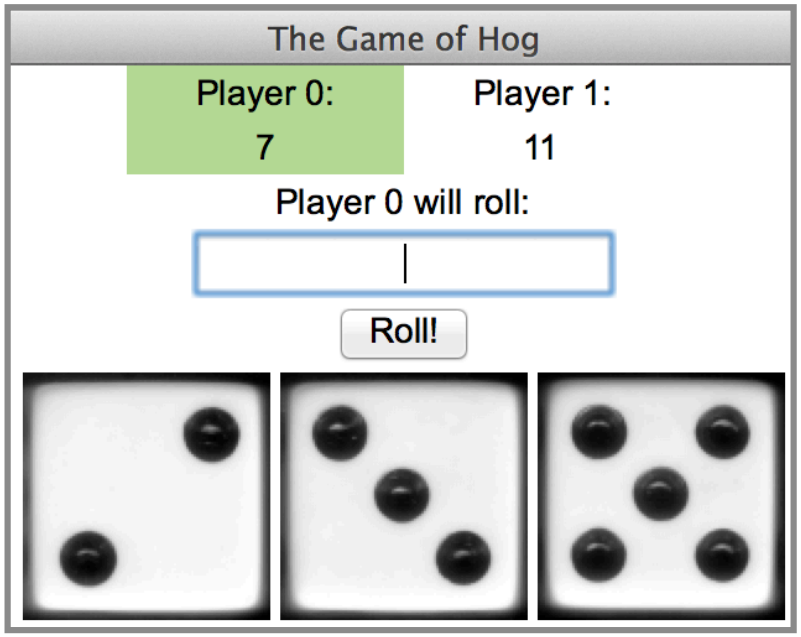
Fill-in-the-blank starter code and a full test suite



Scaffolded Programming Projects

Fill-in-the-blank starter code and a full test suite

Advantages of scaffolding:

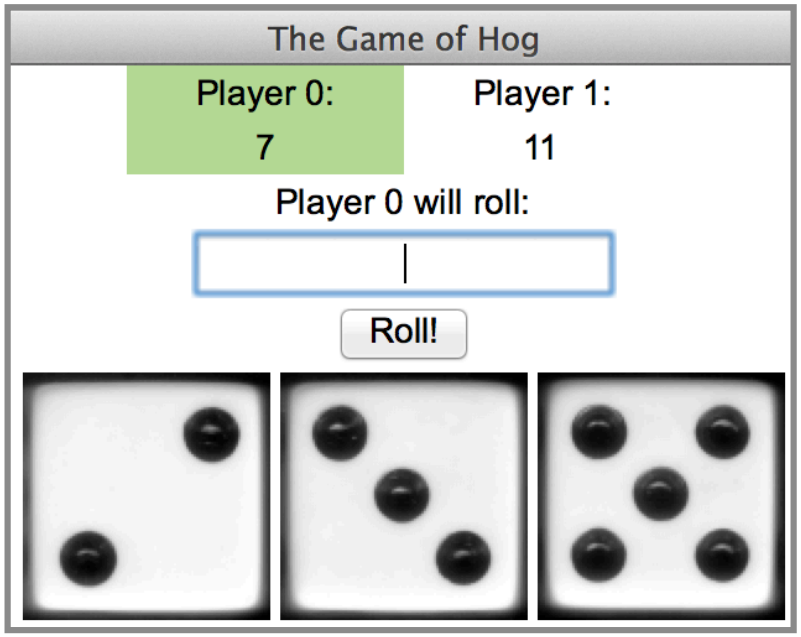


Scaffolded Programming Projects

Fill-in-the-blank starter code and a full test suite

Advantages of scaffolding:

- Modular design taught by example

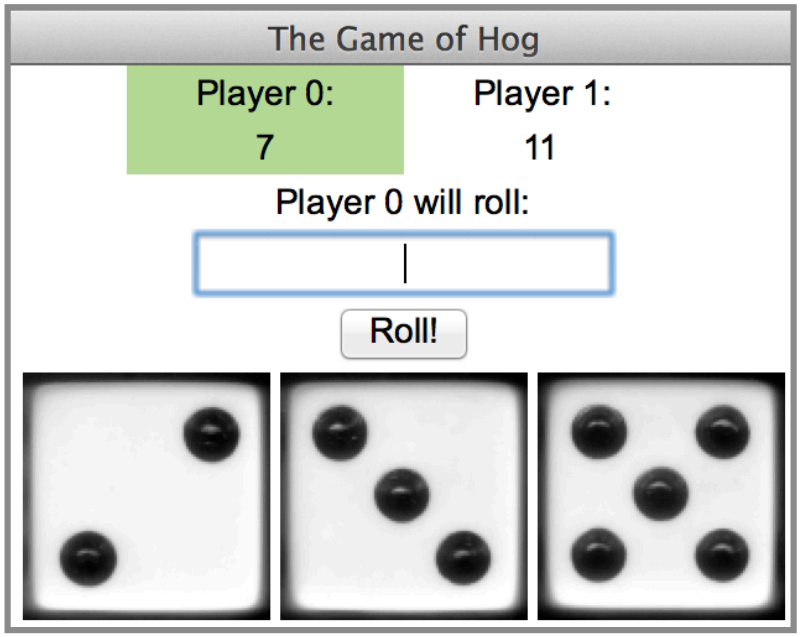


Scaffolded Programming Projects

Fill-in-the-blank starter code and a full test suite

Advantages of scaffolding:

- Modular design taught by example



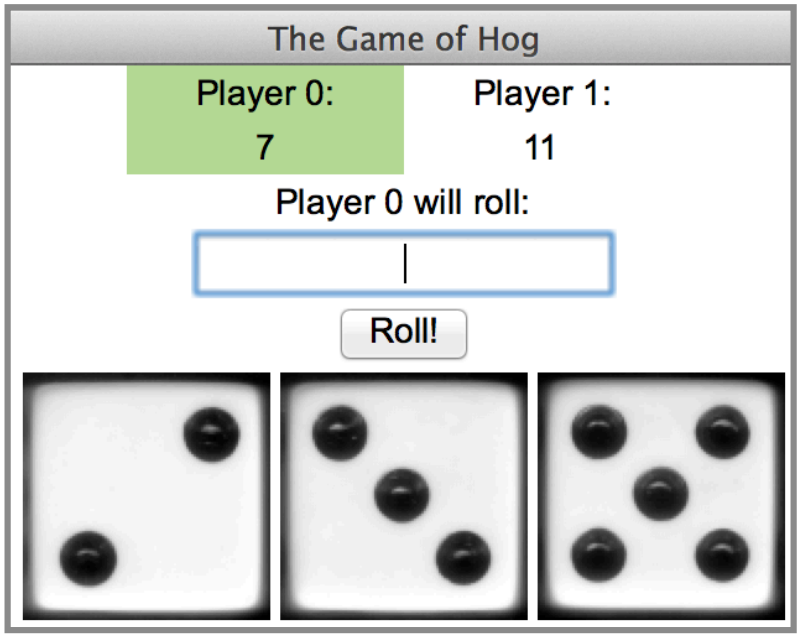

```
def make_averaged(fn, num_samples=1000):
    """Return a function that returns the average
    return value of FN called NUM_SAMPLES times.

    >>> dice = make_test_dice(3, 1, 5, 6)
    >>> averaged_dice = make_averaged(dice, 1000)
    >>> averaged_dice()
    3.75
    """
    "*** YOUR CODE HERE ***"
```

Scaffolded Programming Projects

Fill-in-the-blank starter code and a full test suite

Advantages of scaffolding:

- Modular design taught by example
- Test-driven development from the first assignment



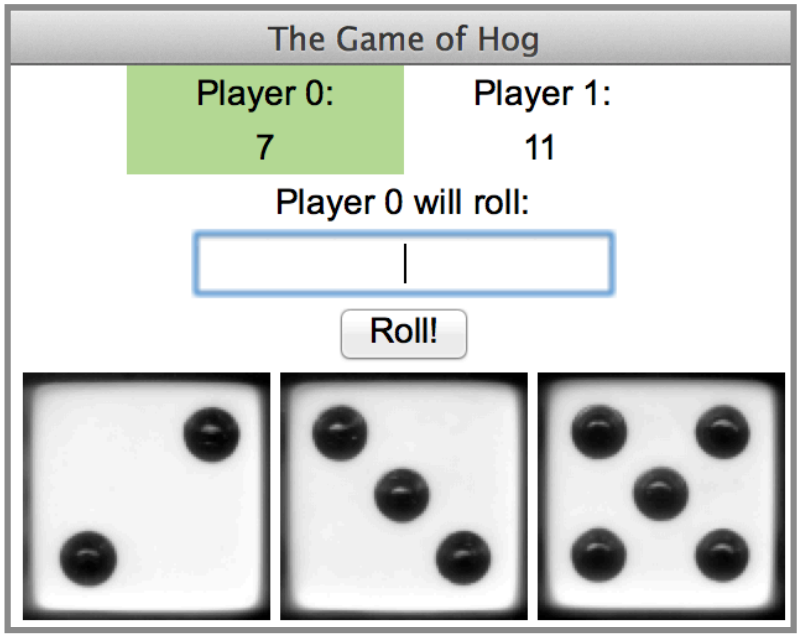

```
def make_averaged(fn, num_samples=1000):
    """Return a function that returns the average
    return value of FN called NUM_SAMPLES times.

    >>> dice = make_test_dice(3, 1, 5, 6)
    >>> averaged_dice = make_averaged(dice, 1000)
    >>> averaged_dice()
    3.75
    """
    "*** YOUR CODE HERE ***"
```

Scaffolded Programming Projects

Fill-in-the-blank starter code and a full test suite

Advantages of scaffolding:

- Modular design taught by example
- Test-driven development from the first assignment



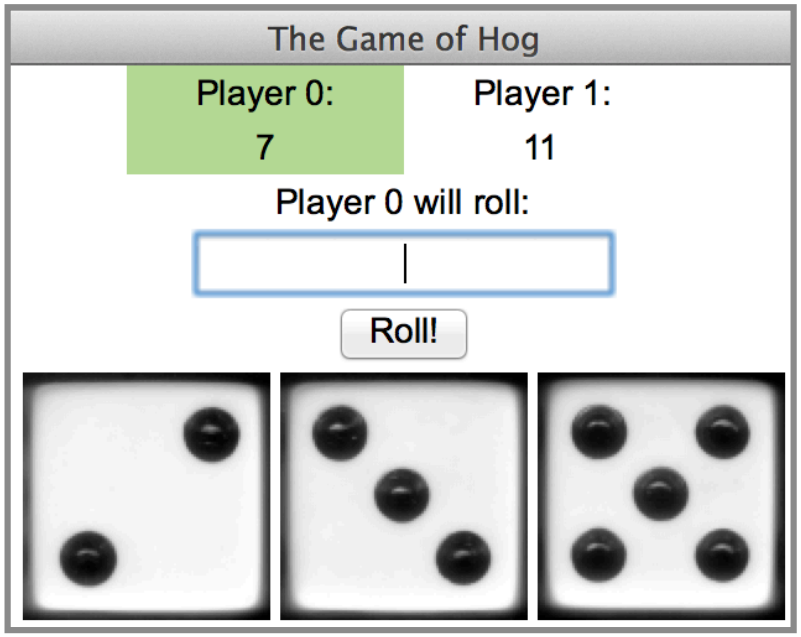

```
def make_averaged(fn, num_samples=1000):
    """Return a function that returns the average
    return value of FN called NUM_SAMPLES times.

    >>> dice = make_test_dice(3, 1, 5, 6)
    >>> averaged_dice = make_averaged(dice, 1000)
    >>> averaged_dice()
    3.75
    """
    "*** YOUR CODE HERE ***"
```

Scaffolded Programming Projects

Fill-in-the-blank starter code and a full test suite

Advantages of scaffolding:

- Modular design taught by example
- Test-driven development from the first assignment
- Automated feedback localizes problems



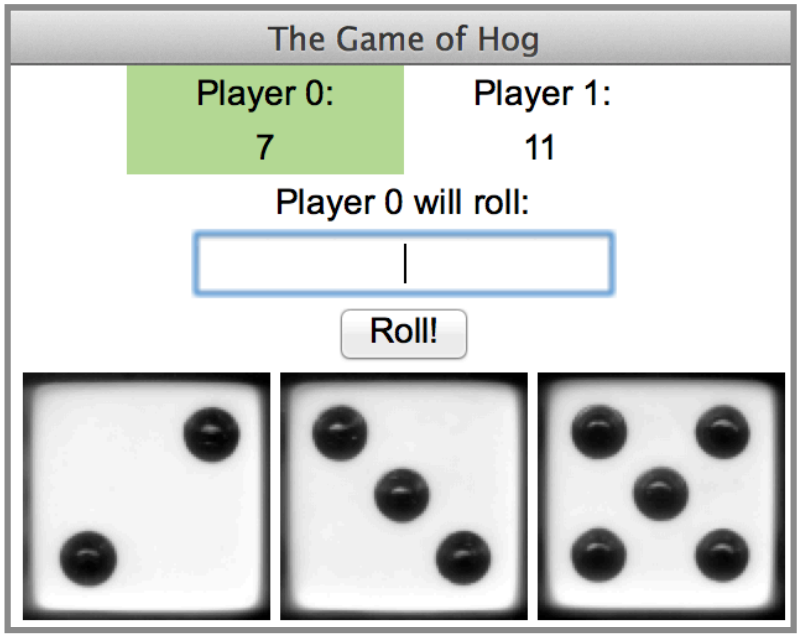

```
def make_averaged(fn, num_samples=1000):
    """Return a function that returns the average
    return value of FN called NUM_SAMPLES times.

    >>> dice = make_test_dice(3, 1, 5, 6)
    >>> averaged_dice = make_averaged(dice, 1000)
    >>> averaged_dice()
    3.75
    """
    "*** YOUR CODE HERE ***"
```

Scaffolded Programming Projects

Fill-in-the-blank starter code and a full test suite

Advantages of scaffolding:

- Modular design taught by example
- Test-driven development from the first assignment
- Automated feedback localizes problems




```
def make_averaged(fn, num_samples=1000):
    """Return a function that returns the average
    return value of FN called NUM_SAMPLES times.

    >>> dice = make_test_dice(3, 1, 5, 6)
    >>> averaged_dice = make_averaged(dice, 1000)
    >>> averaged_dice()
    3.75
    """
    "*** YOUR CODE HERE ***"
```

Teaching Introductory Artificial Intelligence with Pac-Man, EAAI 2010

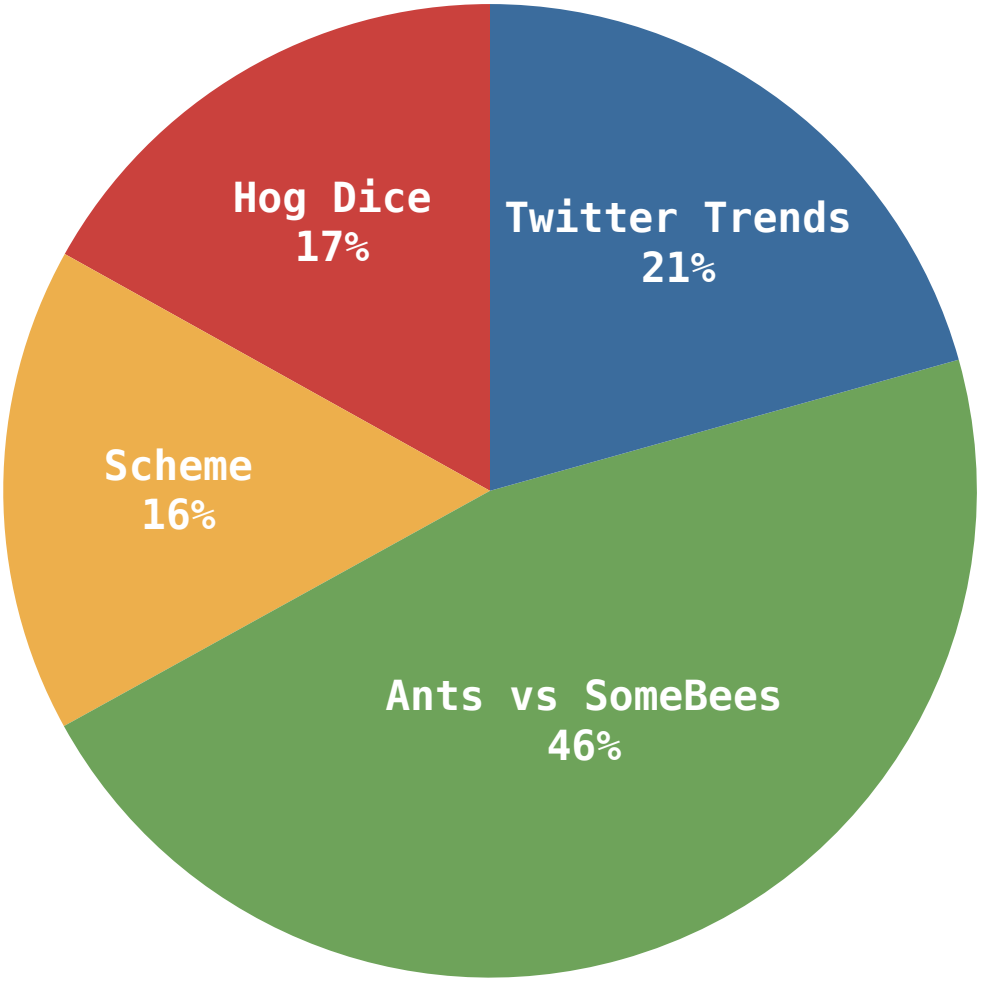
Rewarding Project Outcomes

Rewarding Project Outcomes

Which project did you enjoy the most (Fall 2013)?

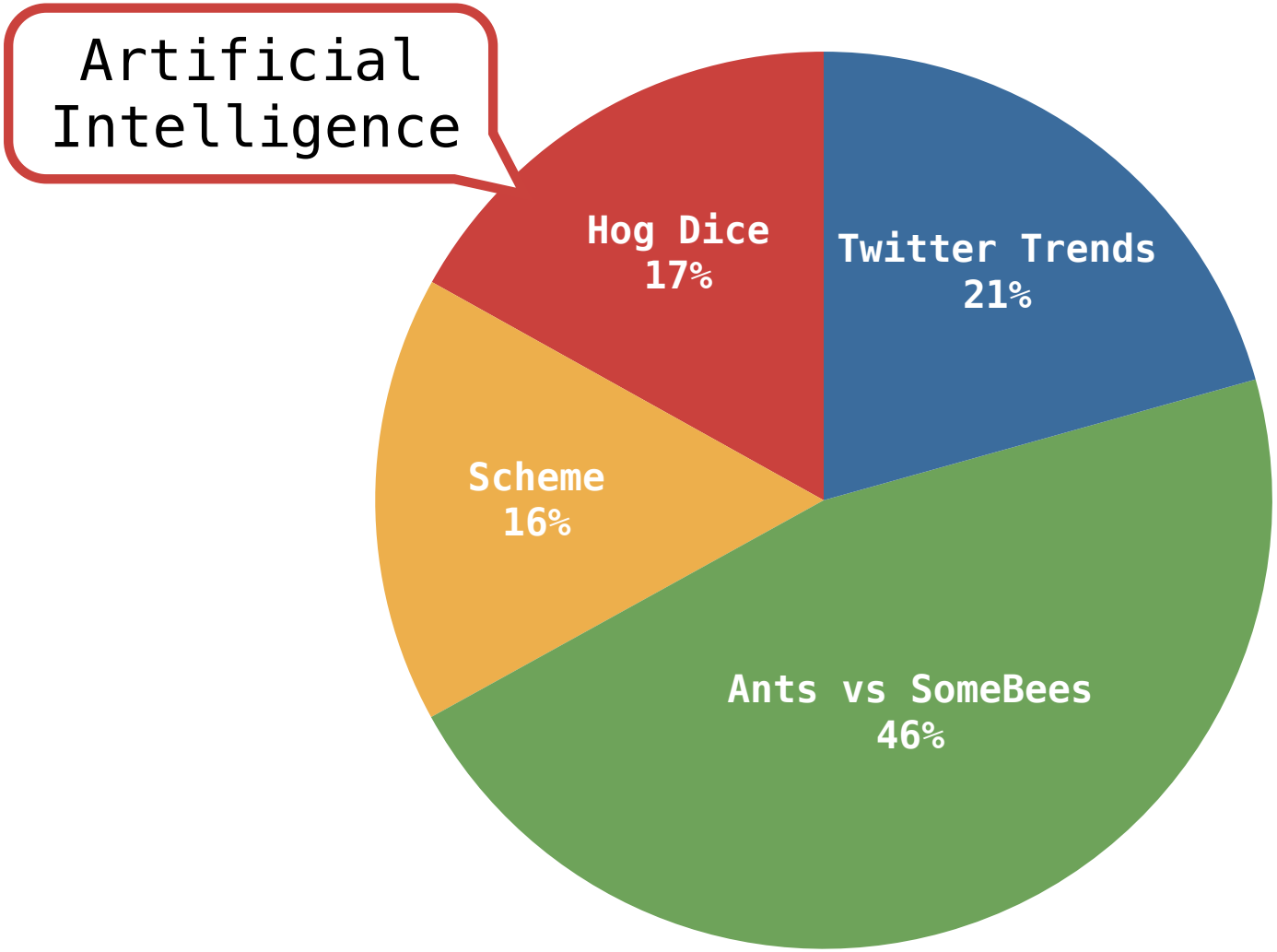
Rewarding Project Outcomes

Which project did you enjoy the most (Fall 2013)?



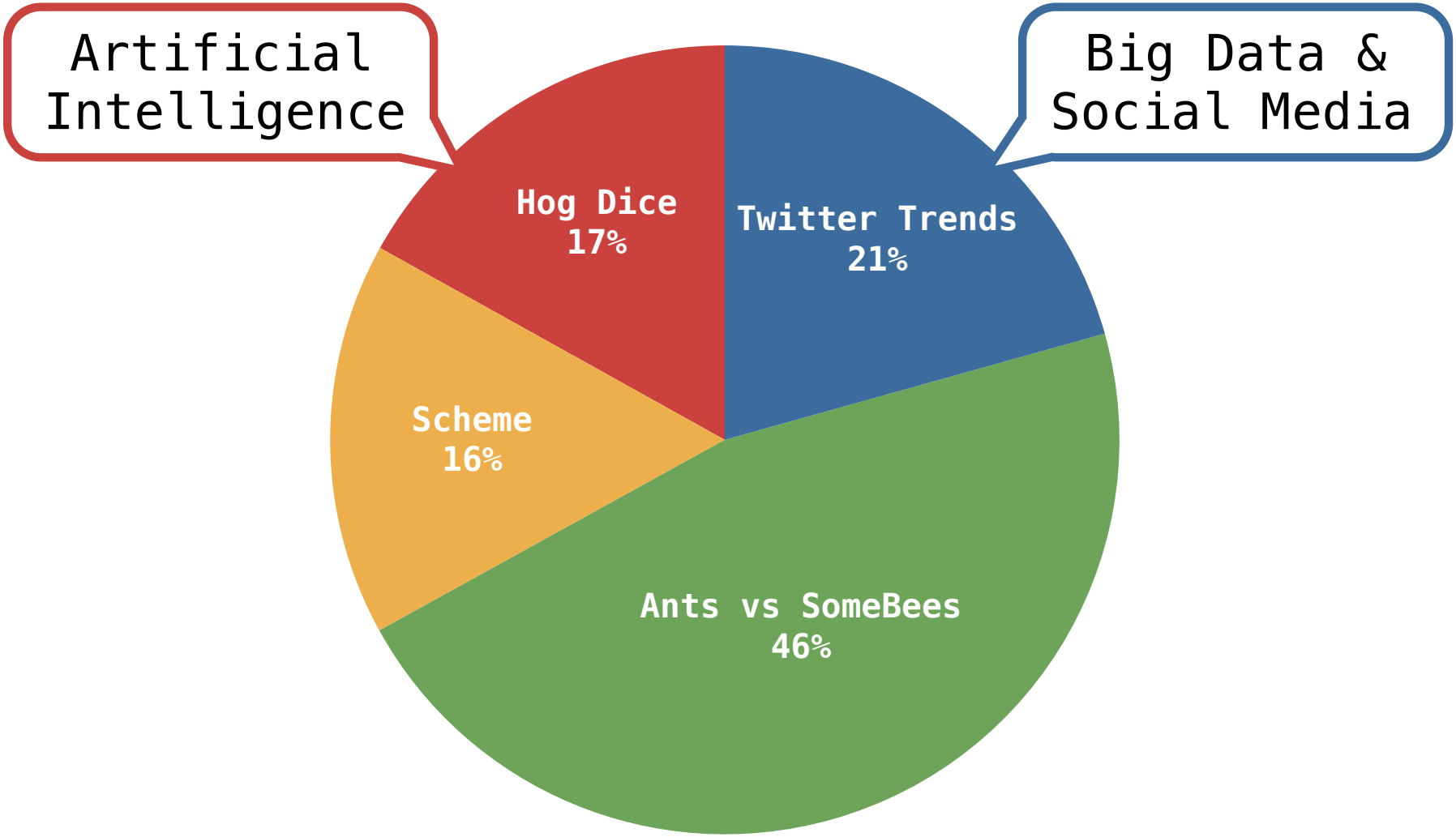
Rewarding Project Outcomes

Which project did you enjoy the most (Fall 2013)?



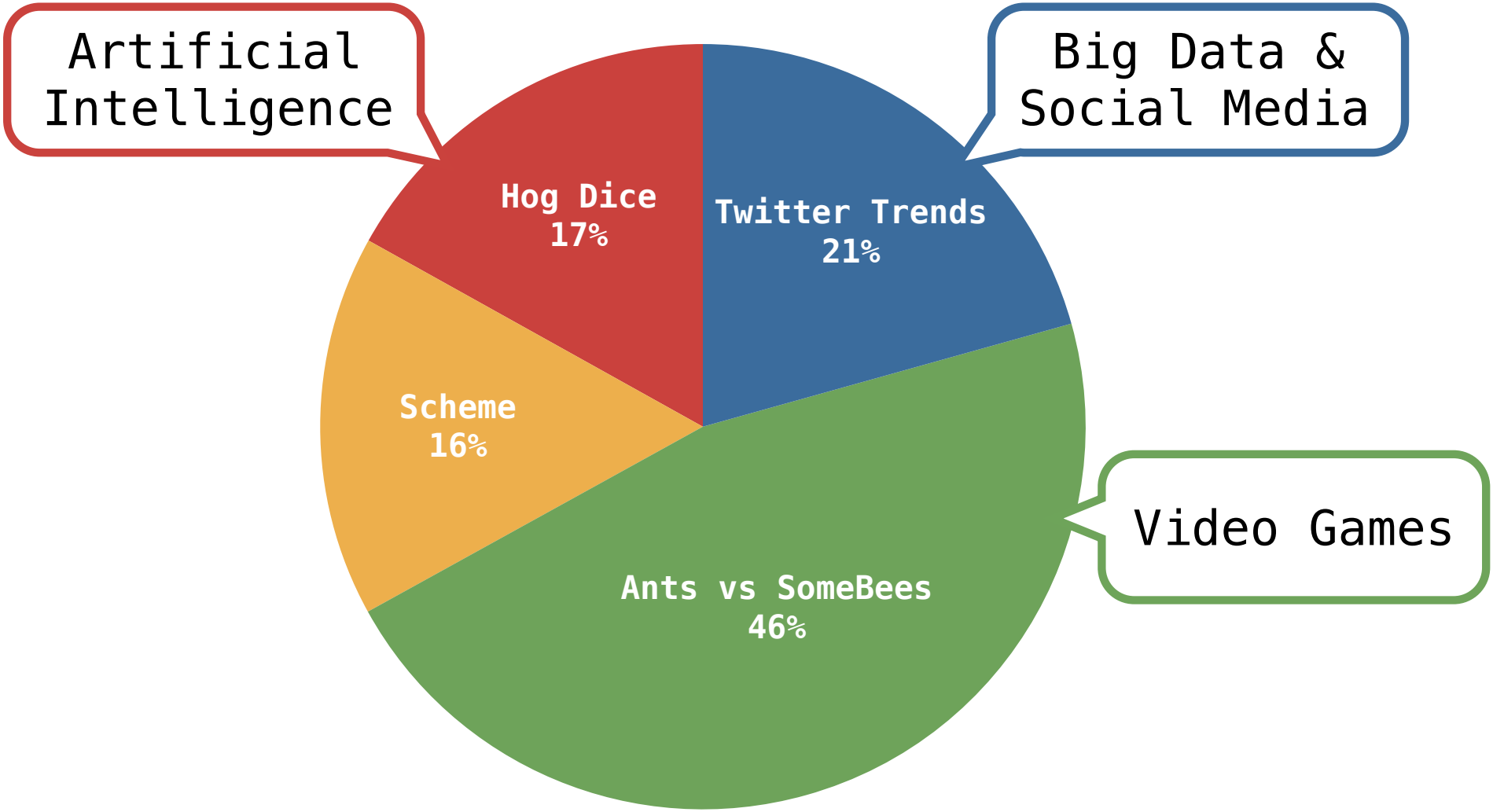
Rewarding Project Outcomes

Which project did you enjoy the most (Fall 2013)?



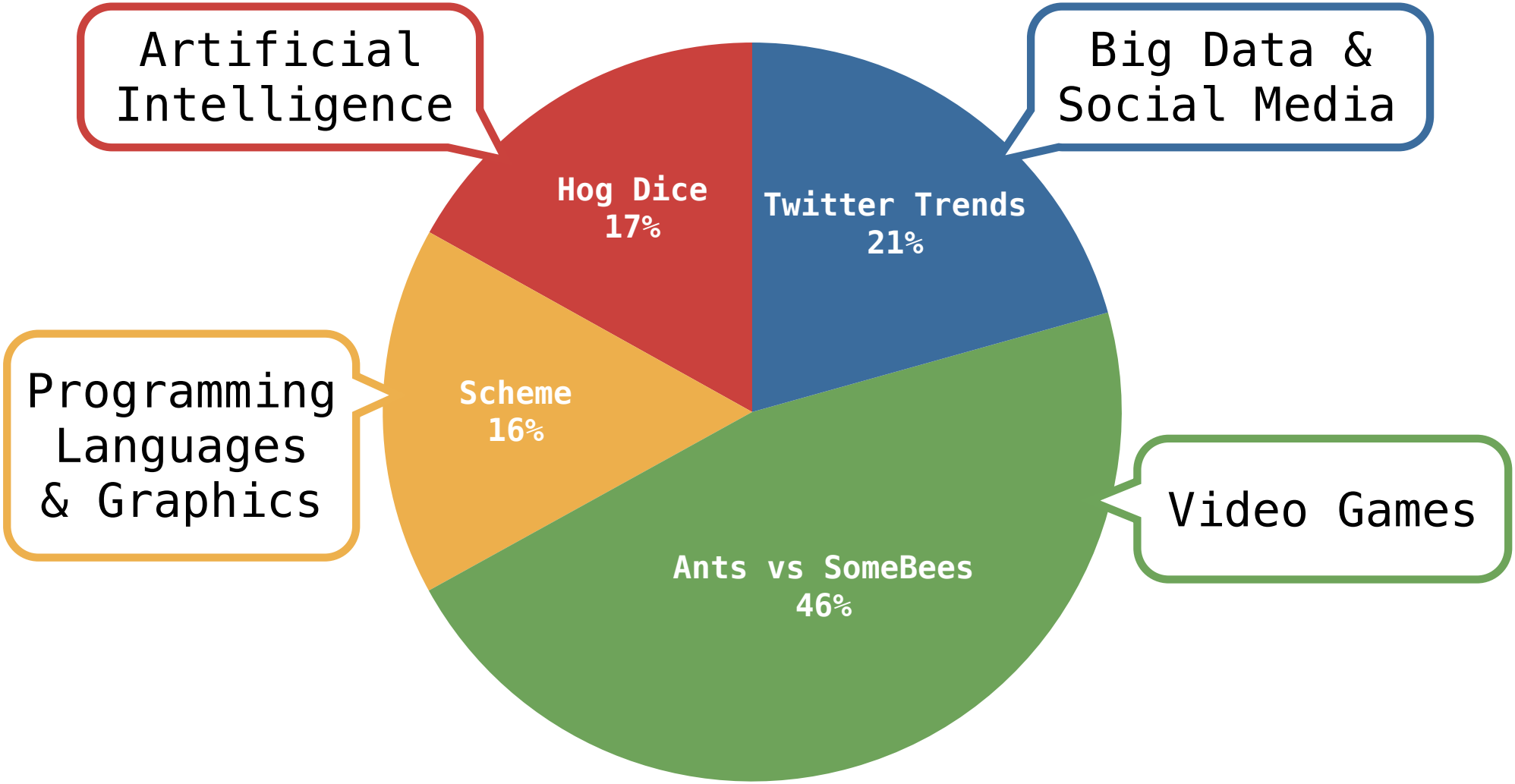
Rewarding Project Outcomes

Which project did you enjoy the most (Fall 2013)?



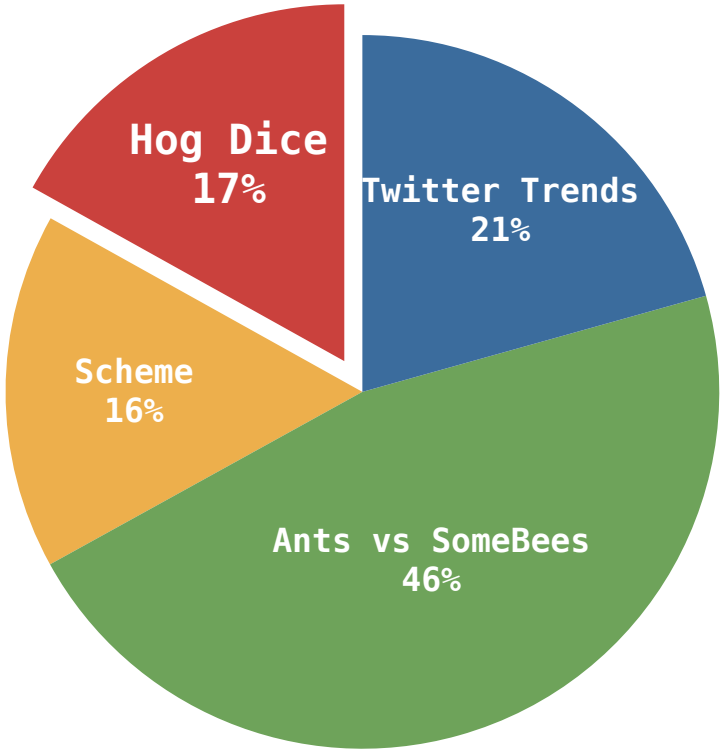
Rewarding Project Outcomes

Which project did you enjoy the most (Fall 2013)?



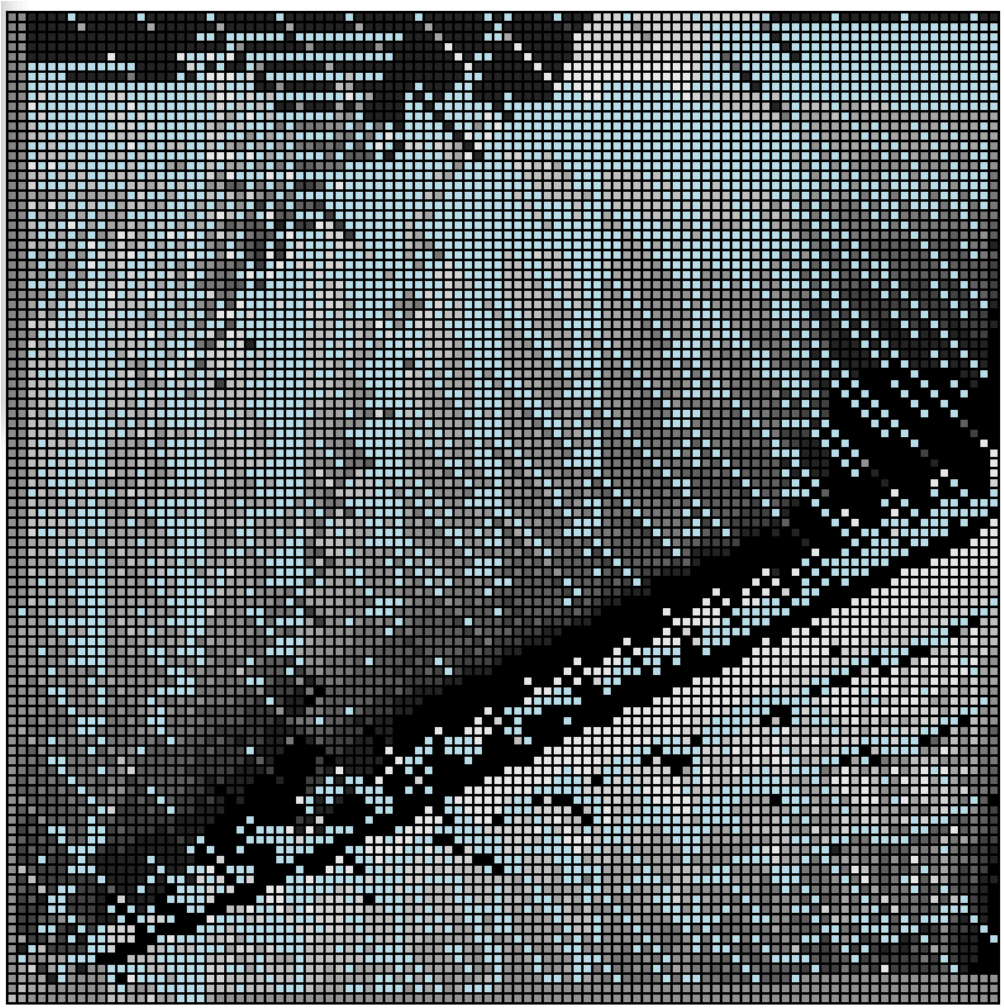
Rewarding Project Outcomes: Hog Dice

The Hog strategy contest encourages exploration

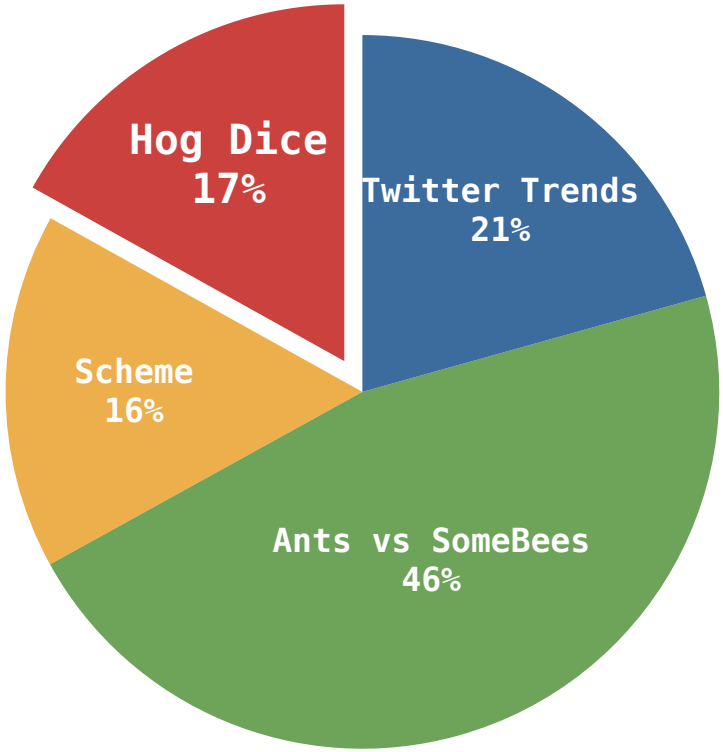


Rewarding Project Outcomes: Hog Dice

The Hog strategy contest encourages exploration

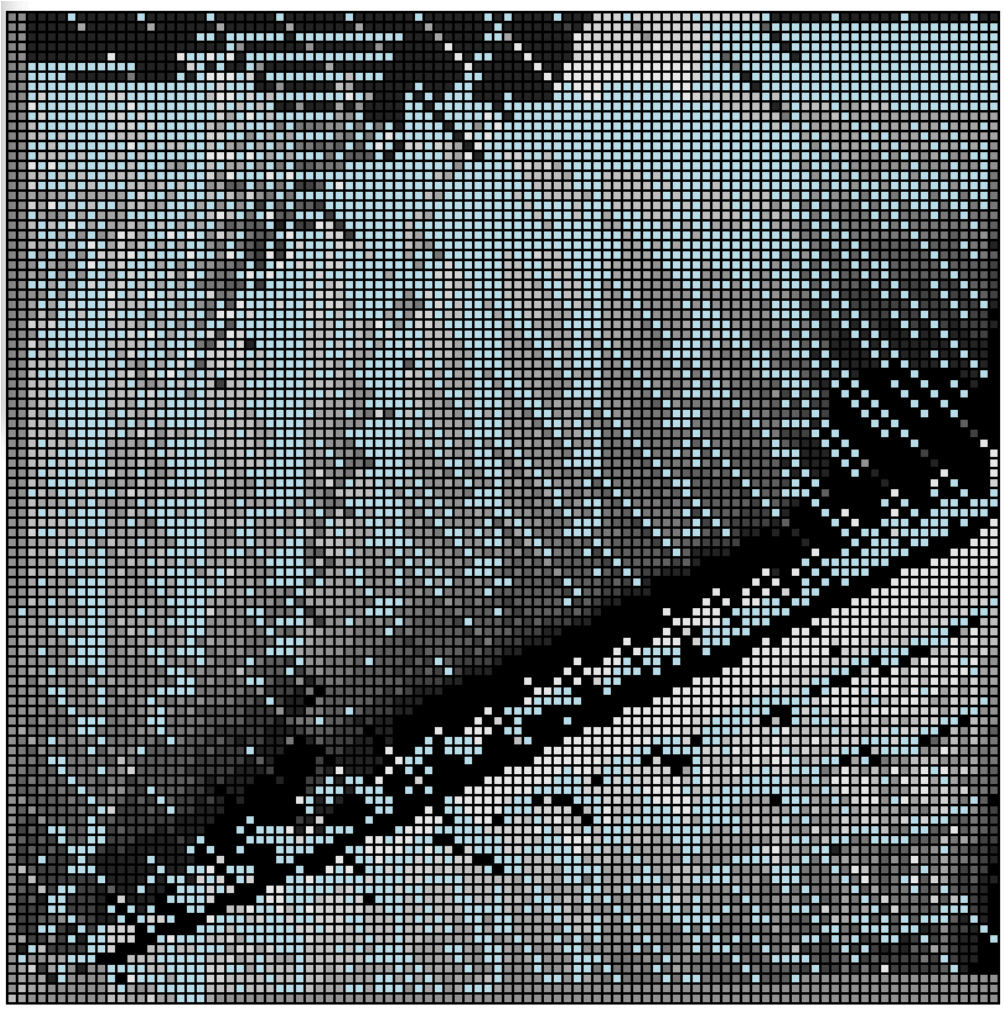


Strategy computed by Chenyang Yuan (class of Fall '12)
Visualization by Kevin Chen (class of Fall '13)



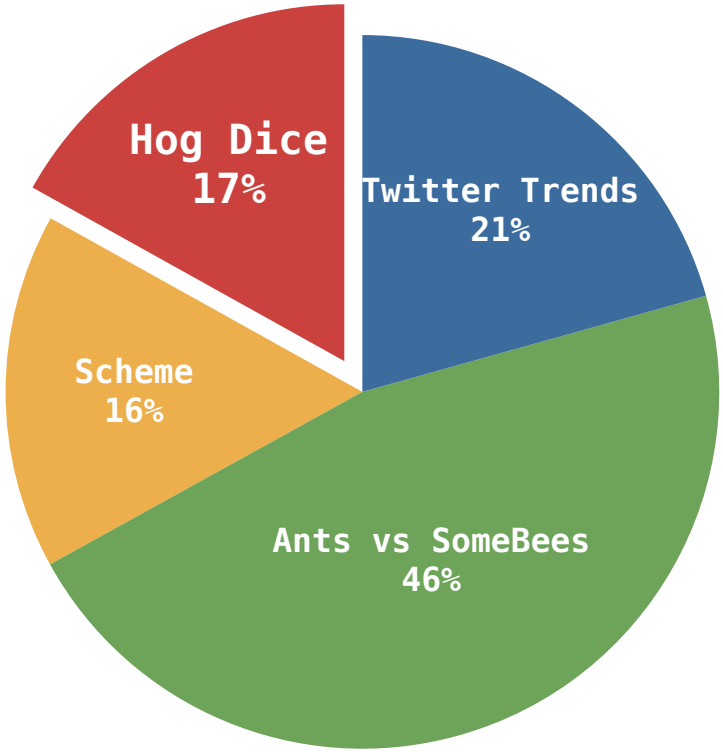
Rewarding Project Outcomes: Hog Dice

The Hog strategy contest encourages exploration



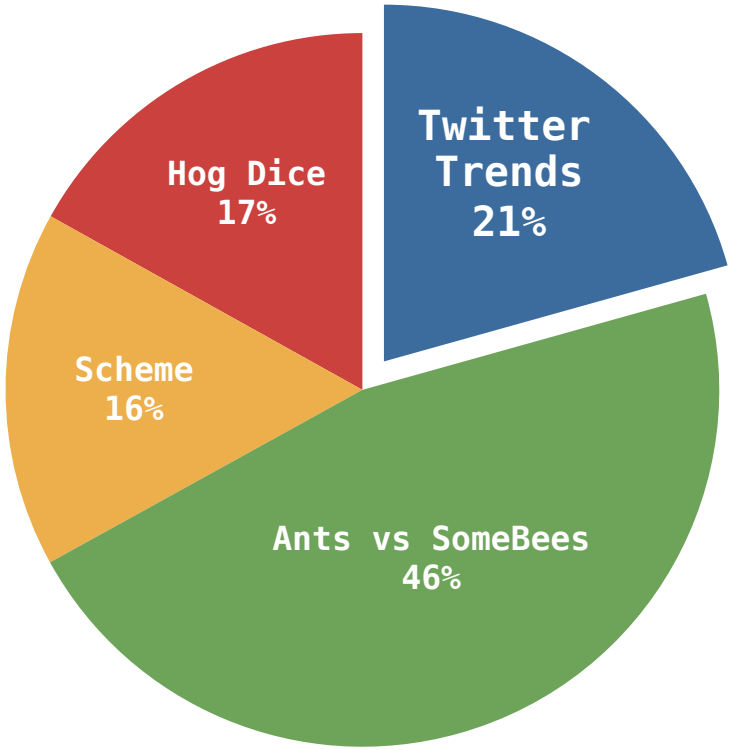
Strategy computed by Chenyang Yuan (class of Fall '12)
Visualization by Kevin Chen (class of Fall '13)

Visualization created for a student blog post



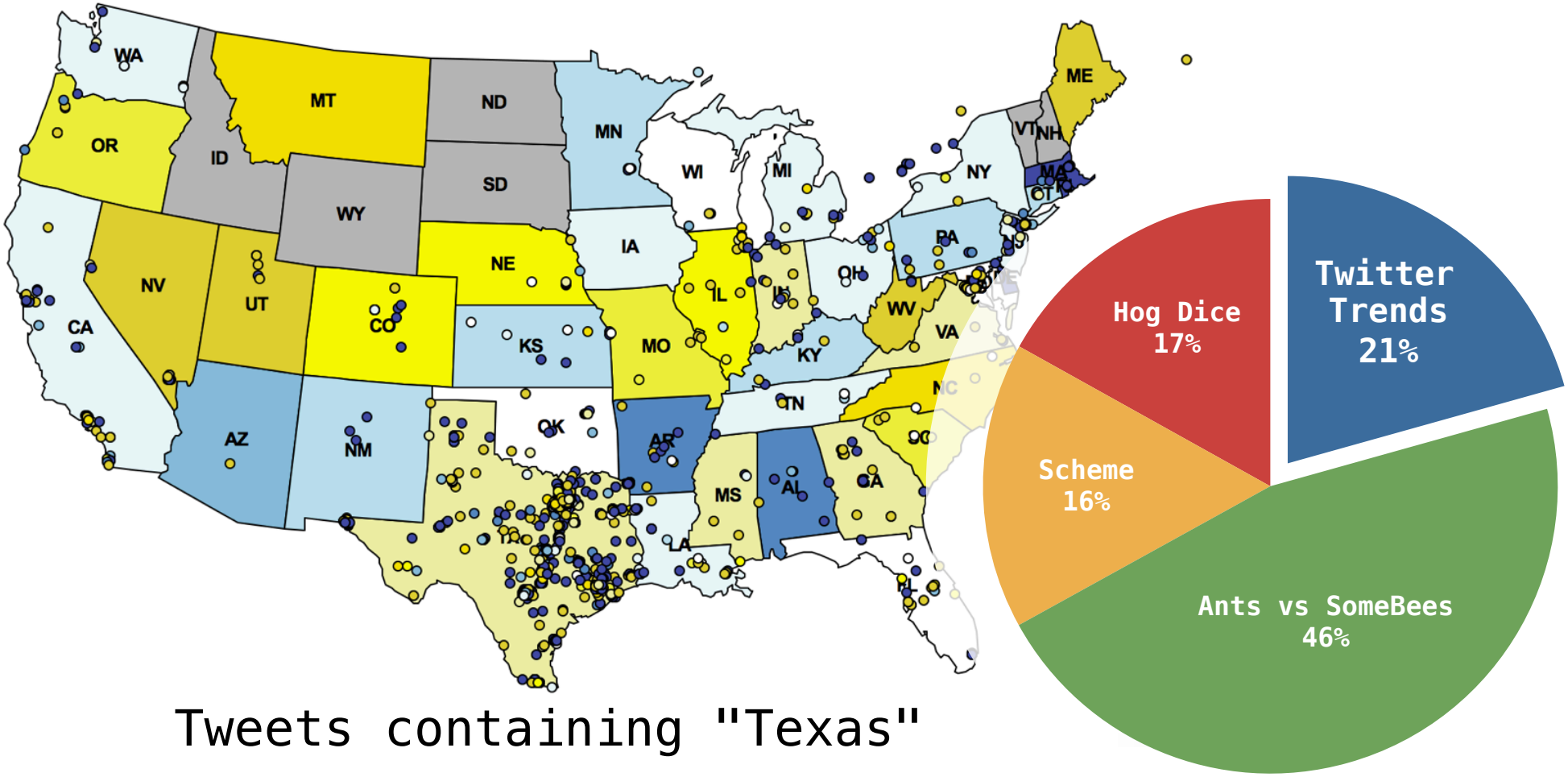
Rewarding Project Outcomes: Twitter Trends

The Twitter Trends project plots the average sentiment of Tweets, aggregated by US state.



Rewarding Project Outcomes: Twitter Trends

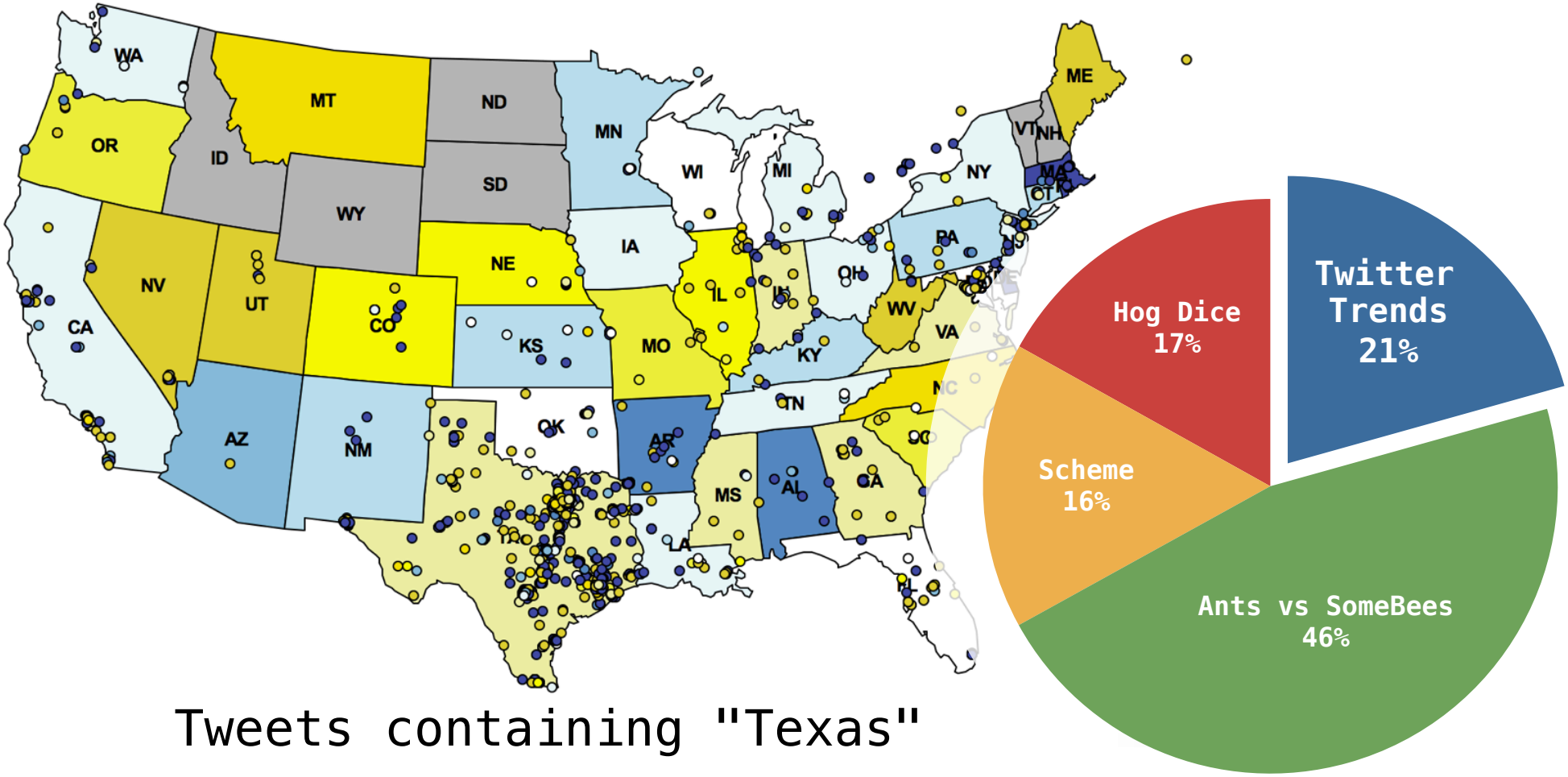
The Twitter Trends project plots the average sentiment of Tweets, aggregated by US state.



Tweets containing "Texas"

Rewarding Project Outcomes: Twitter Trends

The Twitter Trends project plots the average sentiment of Tweets, aggregated by US state.

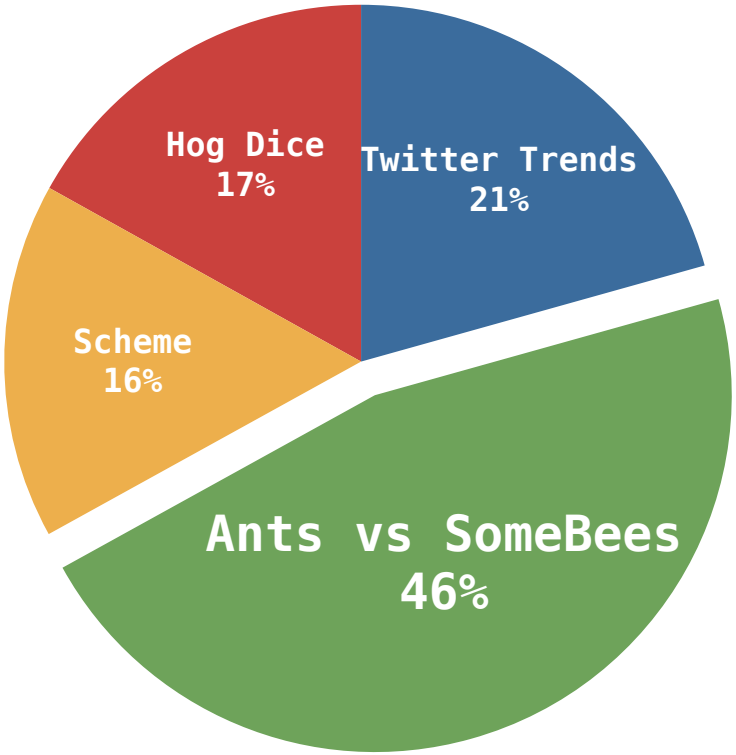


Tweets containing "Texas"

Nifty Assignments Track, SIGCSE 2013

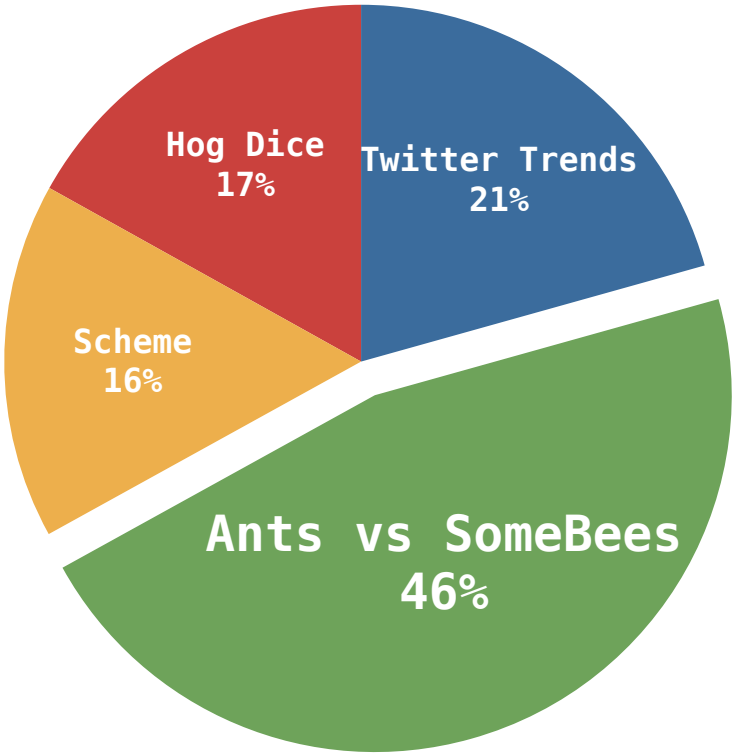
Rewarding Project Outcomes: Ants vs

Ants vs **SomeBees** is a clone of a popular game, **Plants** vs **Zombies**



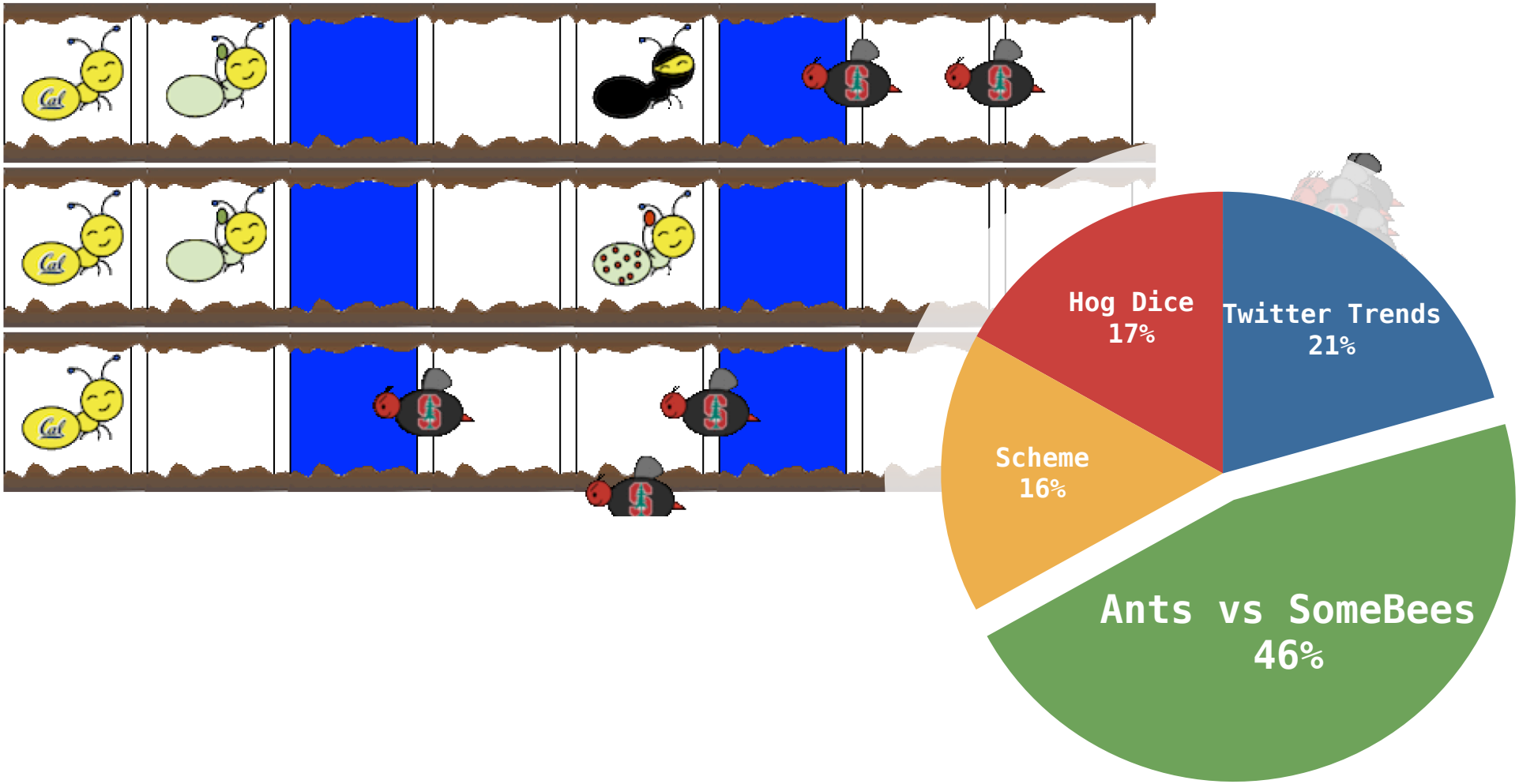
Rewarding Project Outcomes: Ants vs

Ants vs **SomeBees** is a clone of a popular game, **Plants** vs **Zom bies**



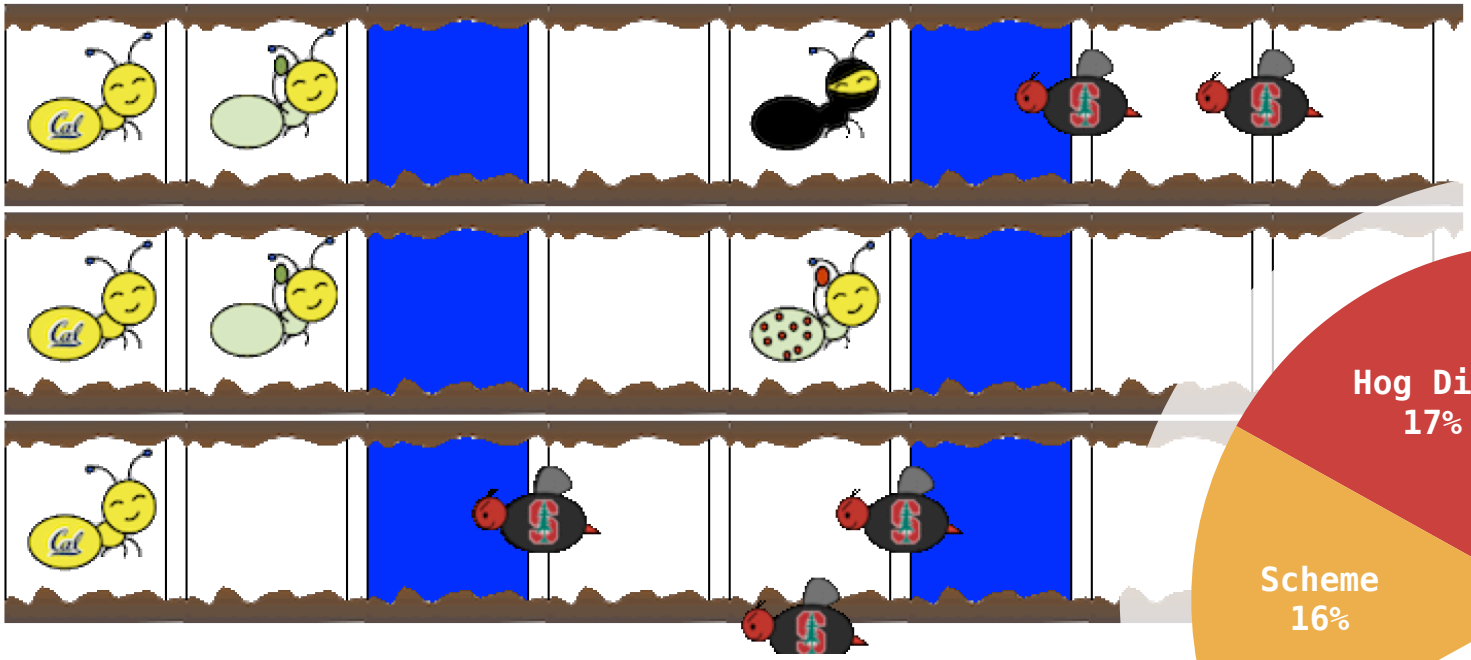
Rewarding Project Outcomes: Ants vs

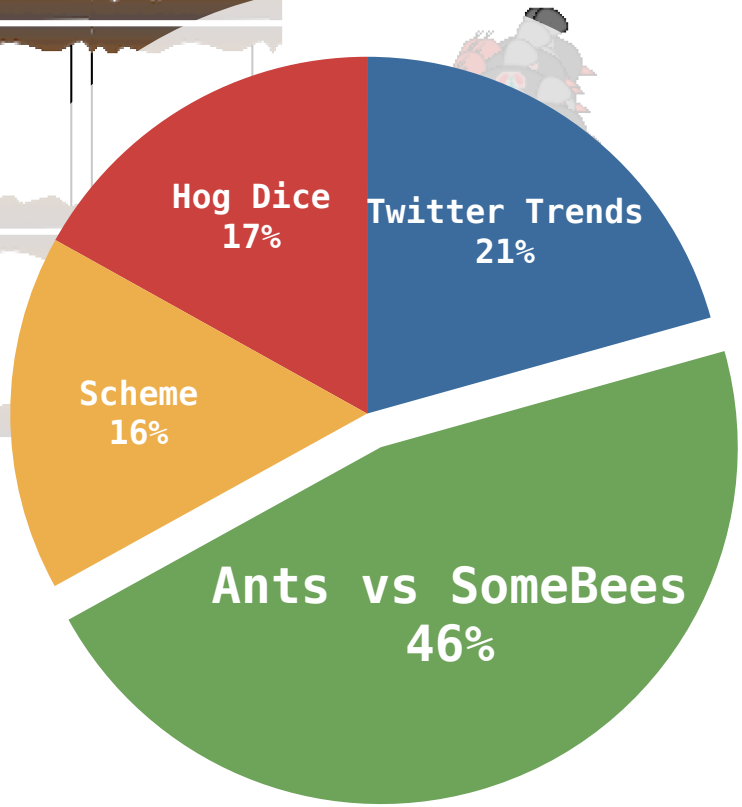
Ants vs **SomeBees** is a clone of a popular game, **Plants** vs **Zom bies**



Rewarding Project Outcomes: Ants vs

Ants vs **SomeBees** is a clone of a popular game, **Plants** vs **Zom bies**

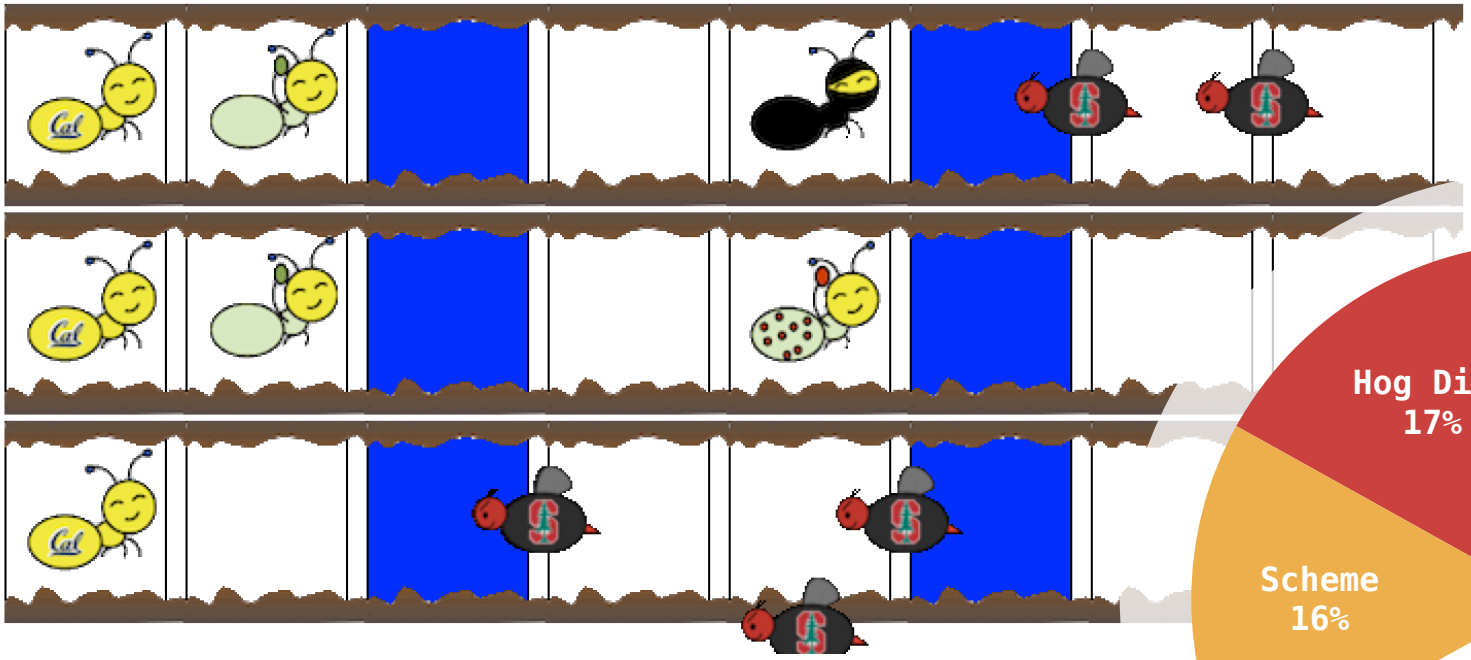


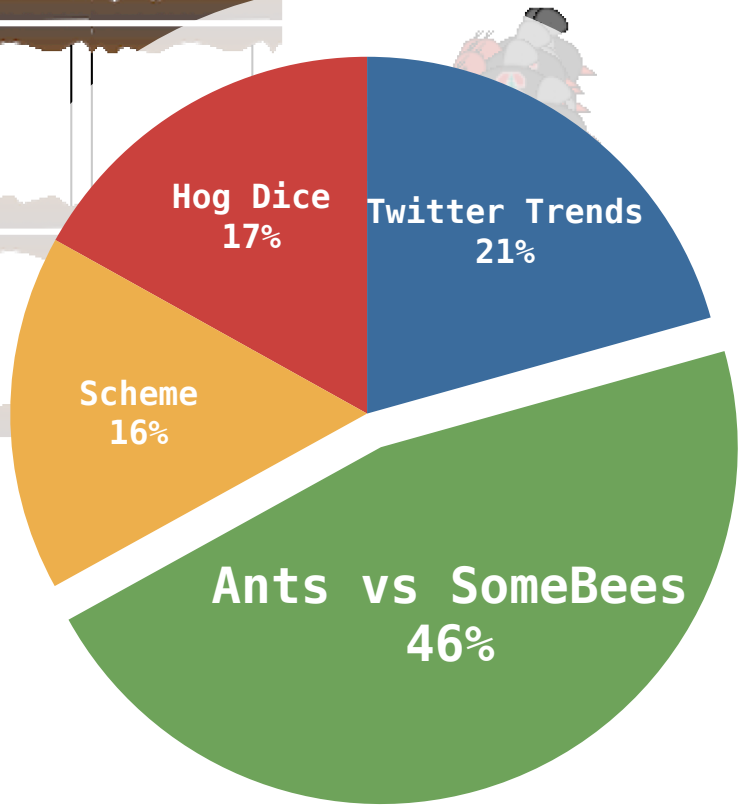


- Students define behavior

Rewarding Project Outcomes: Ants vs

Ants vs **SomeBees** is a clone of a popular game, **Plants** vs **Zom bies**

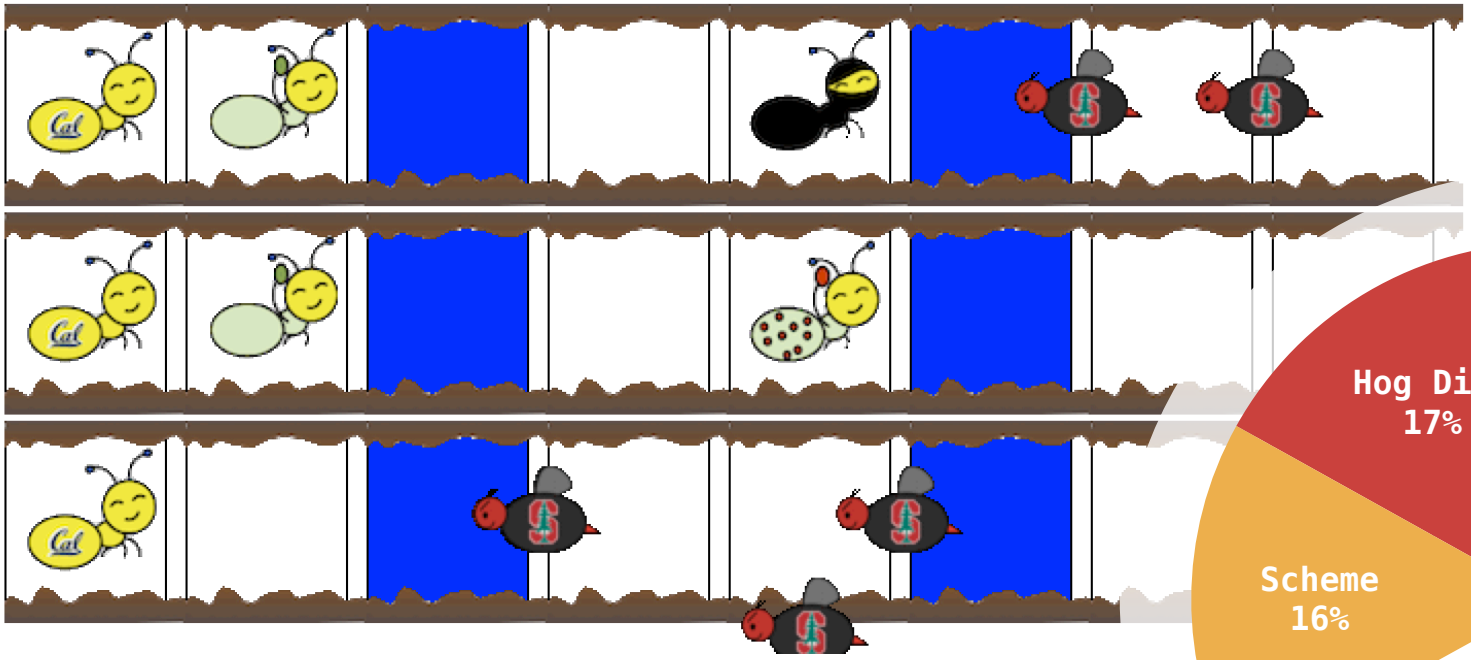


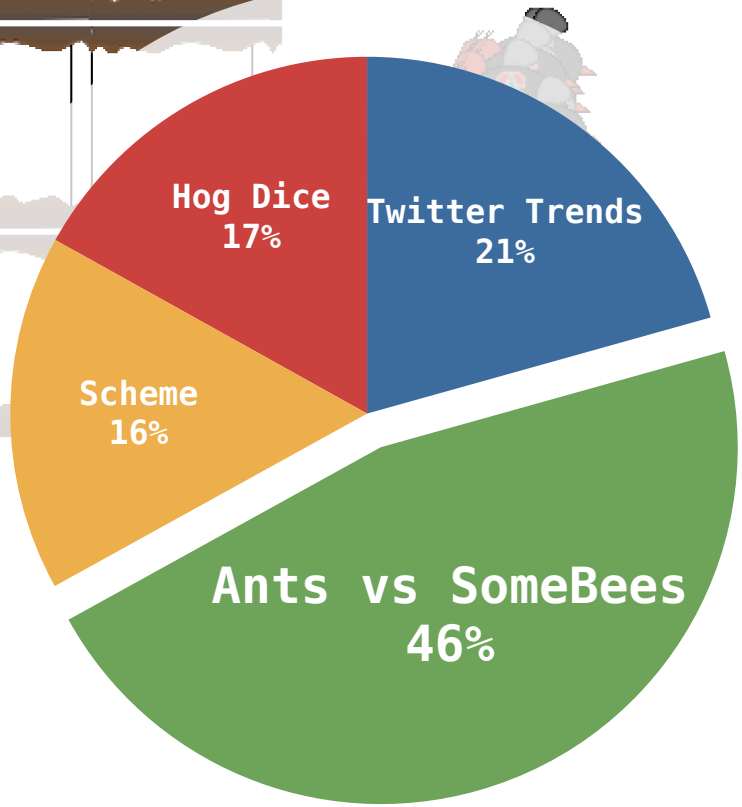


- Students define behavior
- GUI displays interactions

Rewarding Project Outcomes: Ants vs

Ants vs **SomeBees** is a clone of a popular game, **Plants** vs **Zom bies**



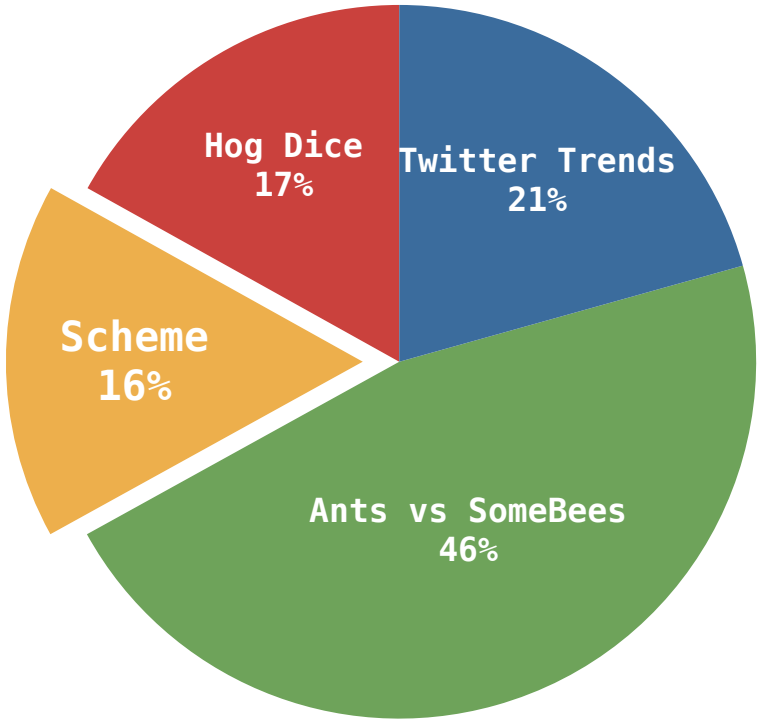


- Students define behavior
- GUI displays interactions

Nifty Assignments Track, SIGCSE 2014

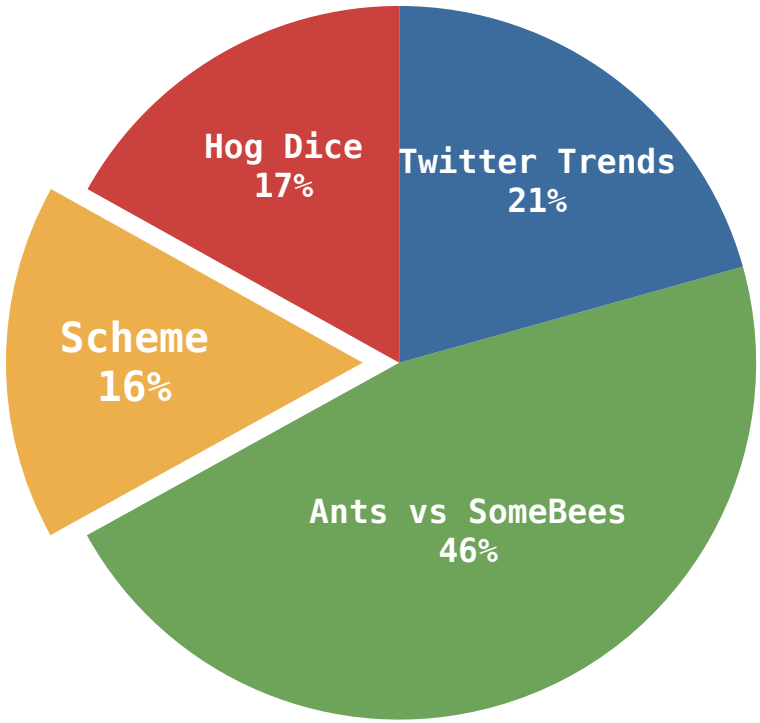
Rewarding Project Outcomes: Scheme

In the Scheme Recursive Art Contest, students draw using Turtle commands interpreted by their own code



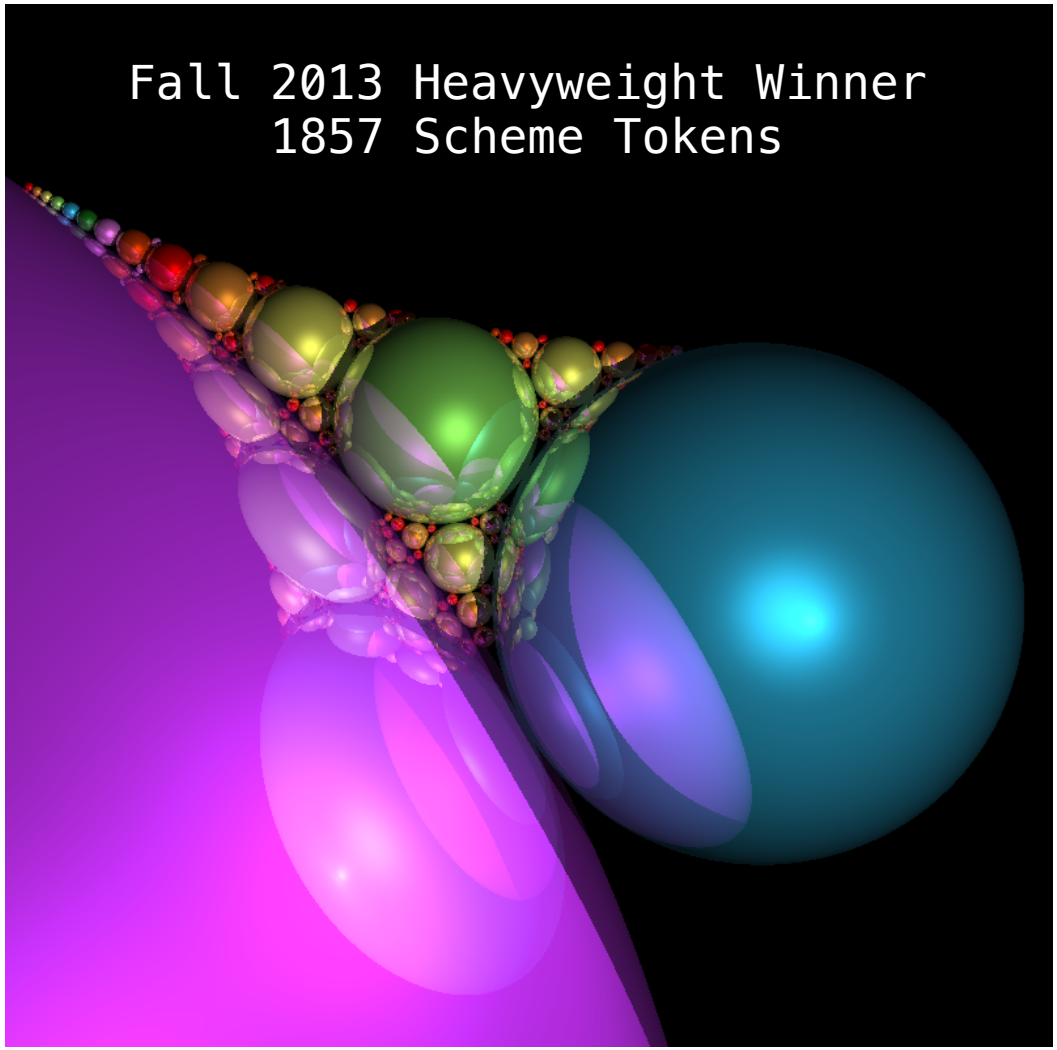
Rewarding Project Outcomes: Scheme

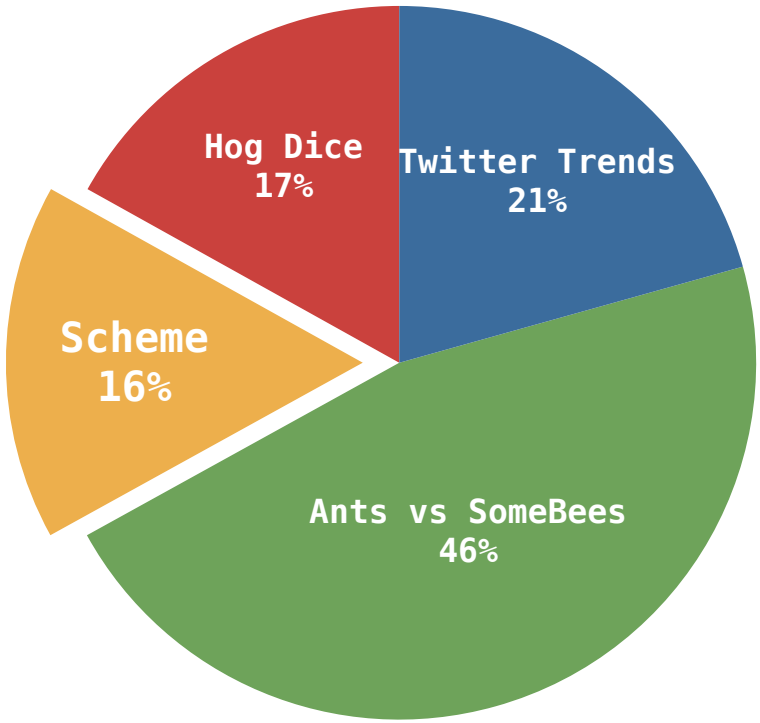
In the Scheme Recursive Art Contest, students draw using Turtle commands interpreted by their own code



Rewarding Project Outcomes: Scheme

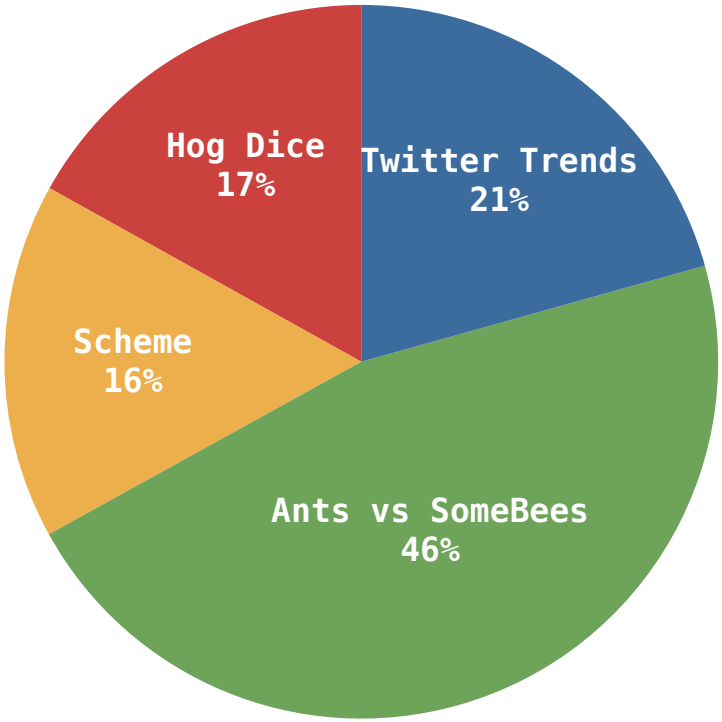
In the Scheme Recursive Art Contest, students draw using Turtle commands interpreted by their own code





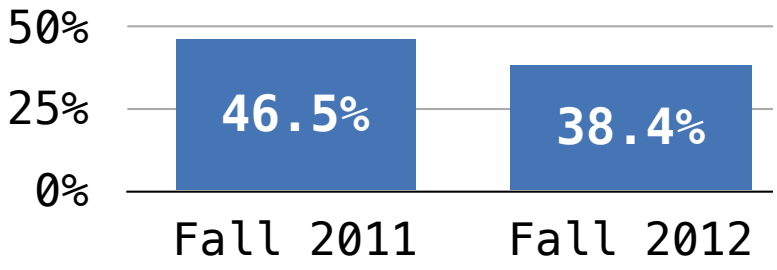
Student Responses to Projects

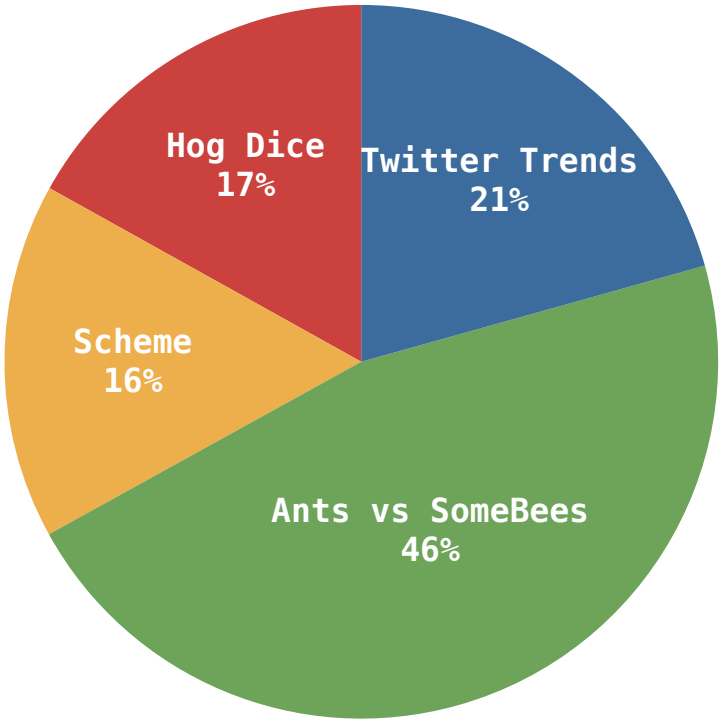
When asked, "what worked best for you in CS 61A," did students mention the projects?



Student Responses to Projects

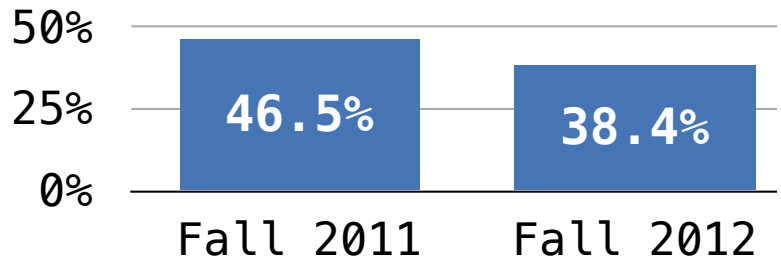
When asked, "what worked best for you in CS 61A," did students mention the projects?

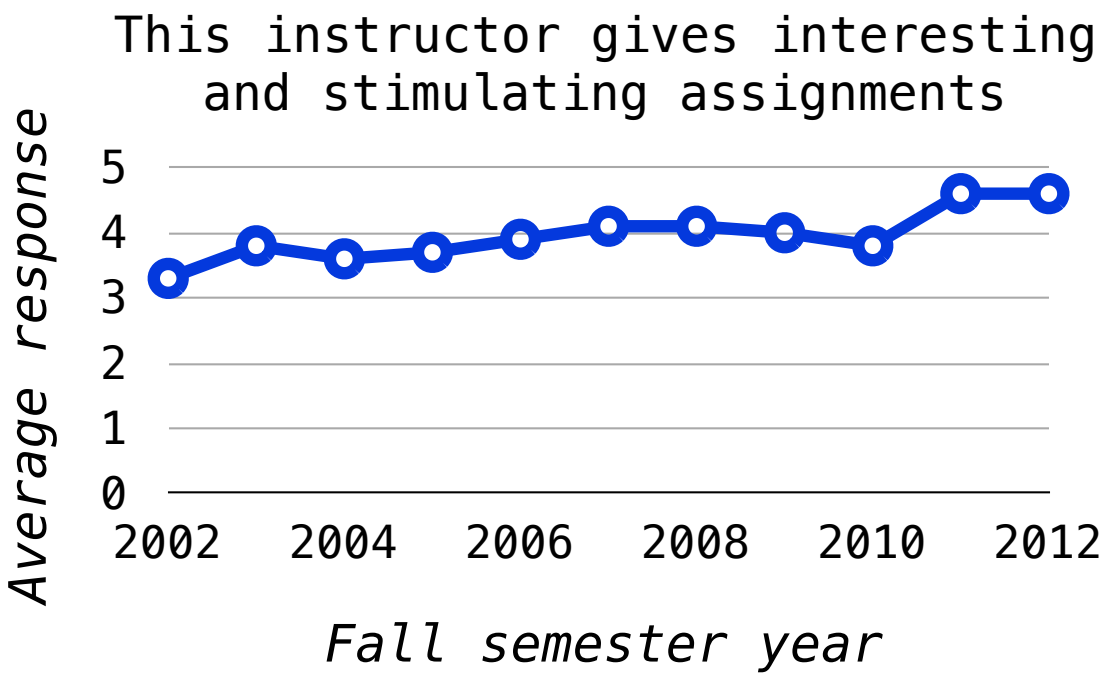


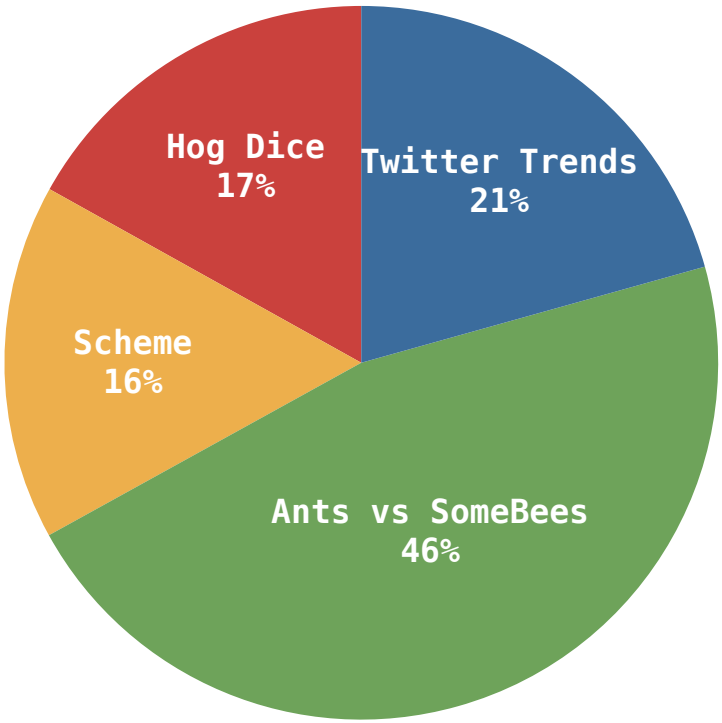


Student Responses to Projects

When asked, "what worked best for you in CS 61A," did students mention the projects?

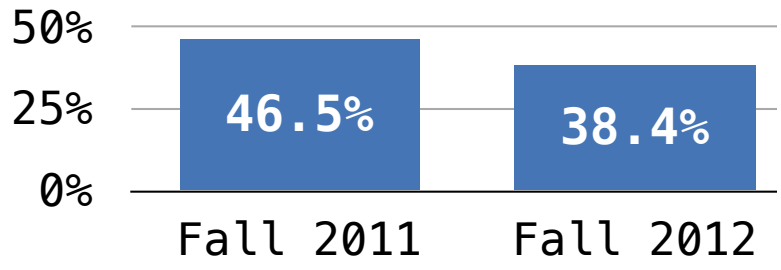


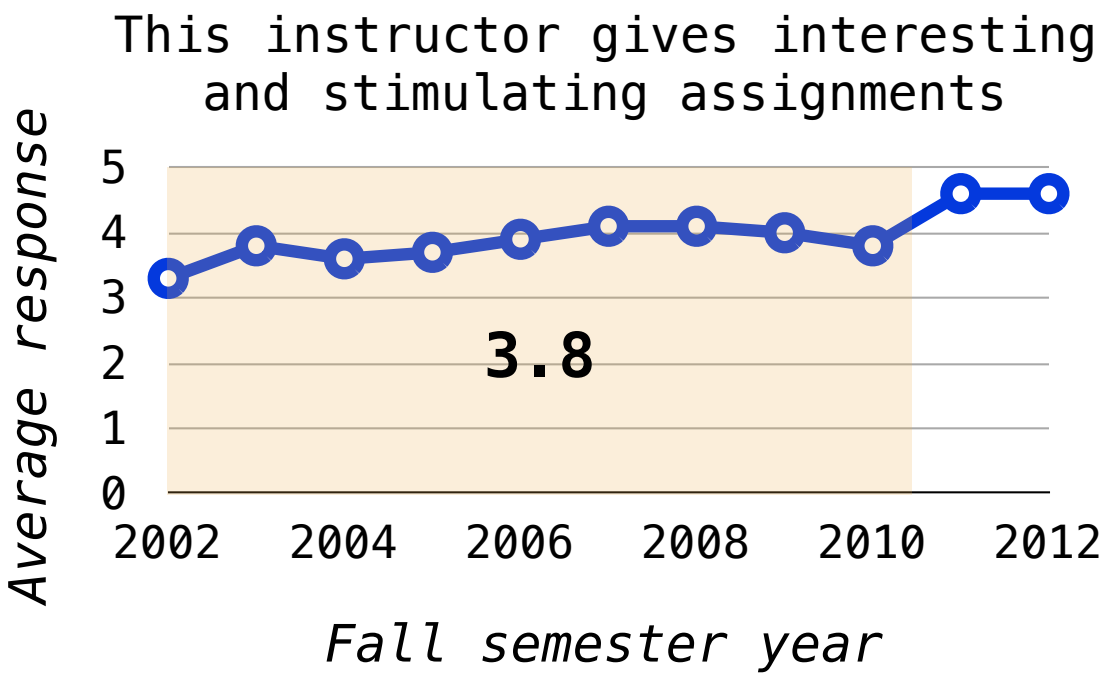


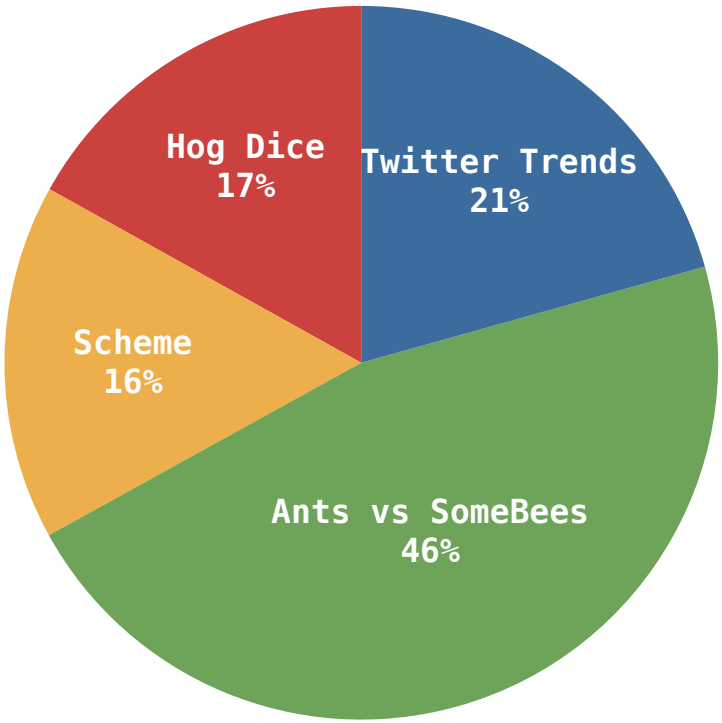


Student Responses to Projects

When asked, "what worked best for you in CS 61A," did students mention the projects?

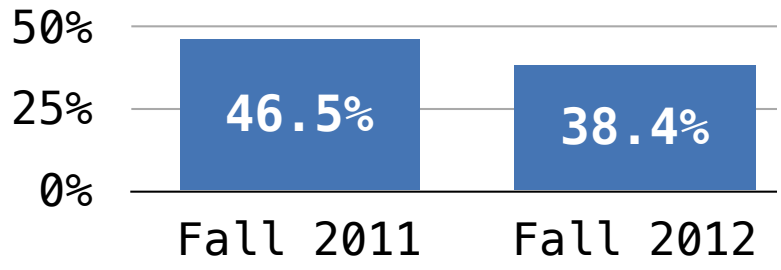


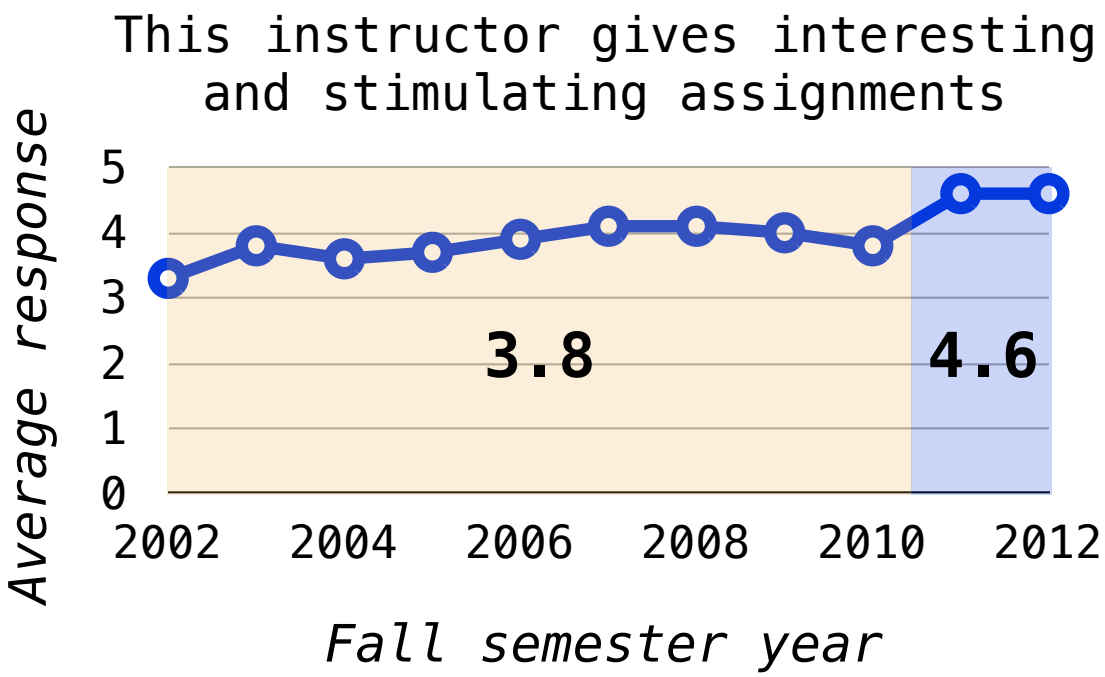


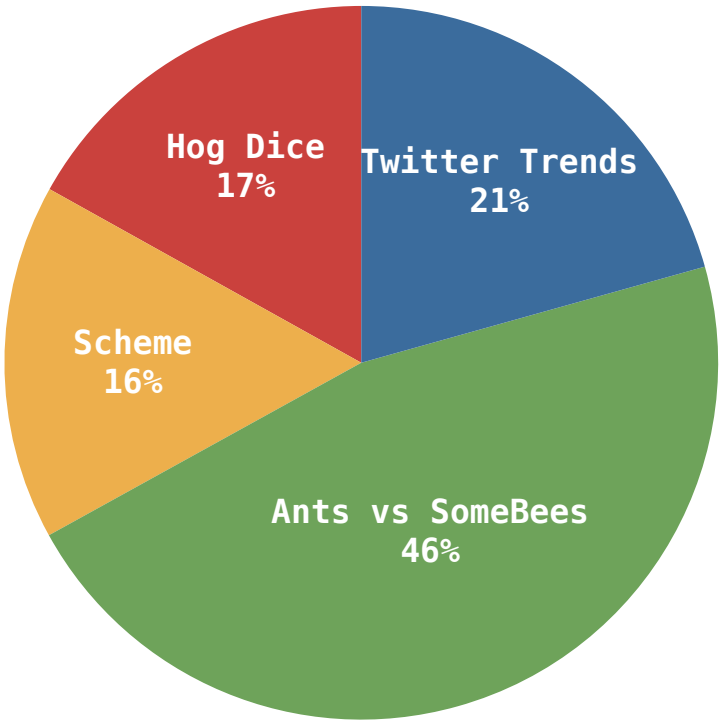


Student Responses to Projects

When asked, "what worked best for you in CS 61A," did students mention the projects?

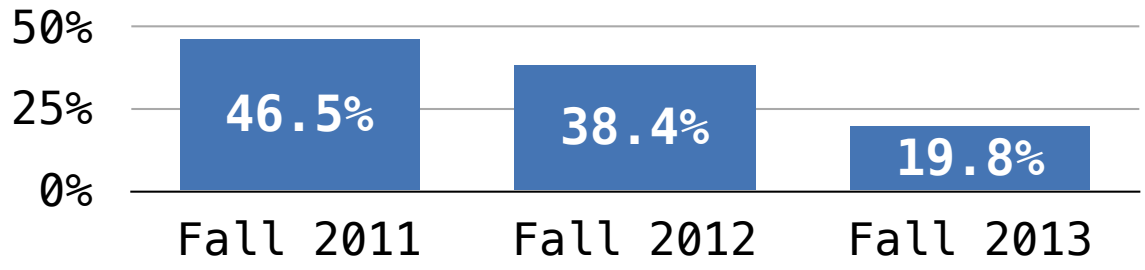


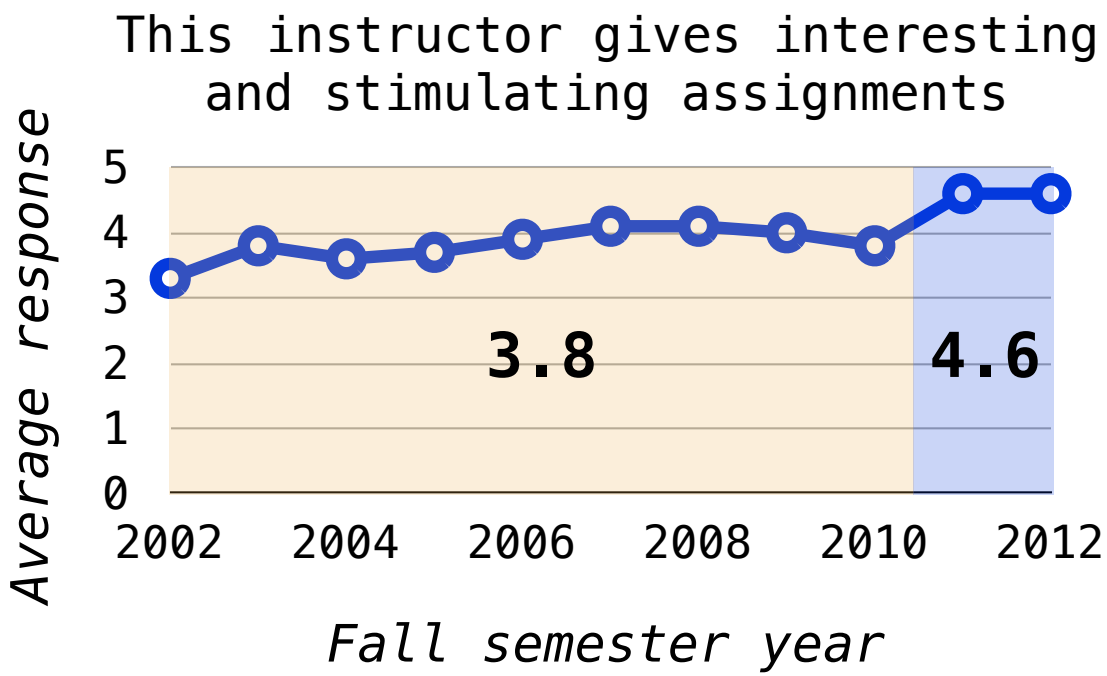


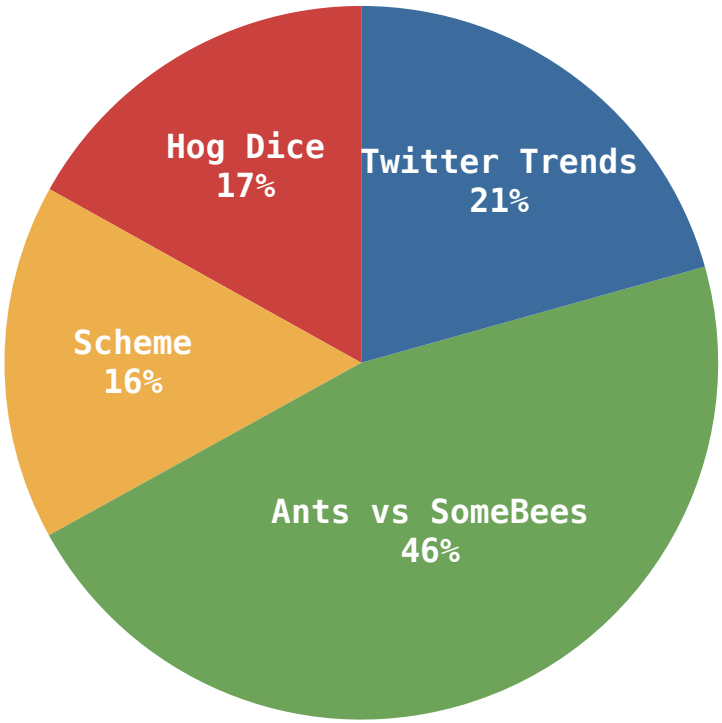


Student Responses to Projects

When asked, "what worked best for you in CS 61A," did students mention the projects?

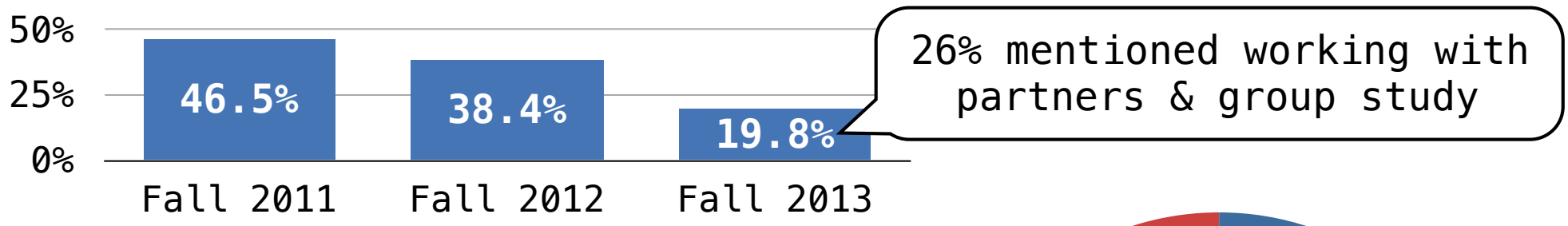


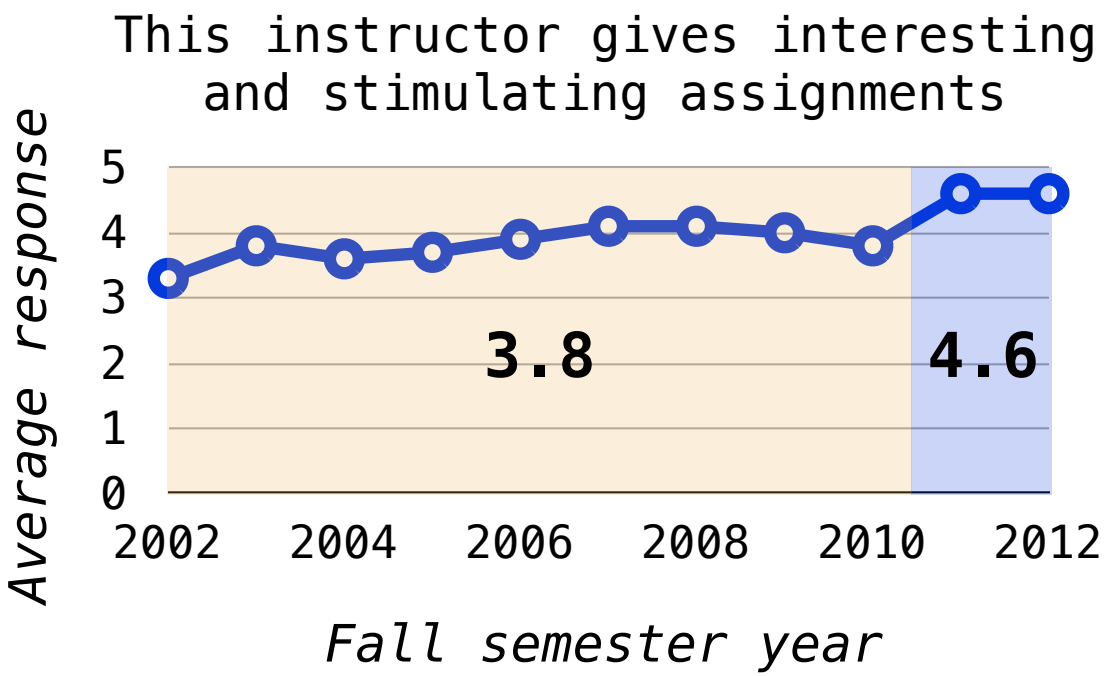


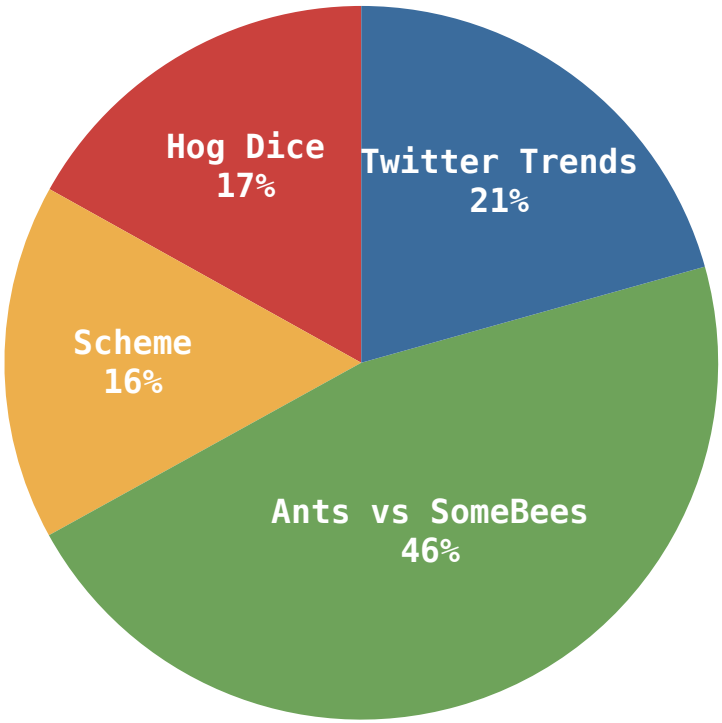


Student Responses to Projects

When asked, "what worked best for you in CS 61A," did students mention the projects?







Community

Community and Collaboration Guidelines

Community and Collaboration Guidelines

Computing is a collaborative, social discipline

Community and Collaboration Guidelines

Computing is a collaborative, social discipline

I strongly encourage students to:

Community and Collaboration Guidelines

Computing is a collaborative, social discipline

I strongly encourage students to:

- Work with a partner on projects

Community and Collaboration Guidelines

Computing is a collaborative, social discipline

I strongly encourage students to:

- Work with a partner on projects

61% had a partner for all projects

True for **67%** of students with no prior programming experience

Community and Collaboration Guidelines

Computing is a collaborative, social discipline

I strongly encourage students to:

- Work with a partner on projects
- Discuss problems with classmates

61% had a partner for all projects

True for **67%** of students with no prior programming experience

Community and Collaboration Guidelines

Computing is a collaborative, social discipline

I strongly encourage students to:

- Work with a partner on projects
- Discuss problems with classmates
- Attend office hours with questions

61% had a partner for all projects

True for **67%** of students with no prior programming experience

Community and Collaboration Guidelines

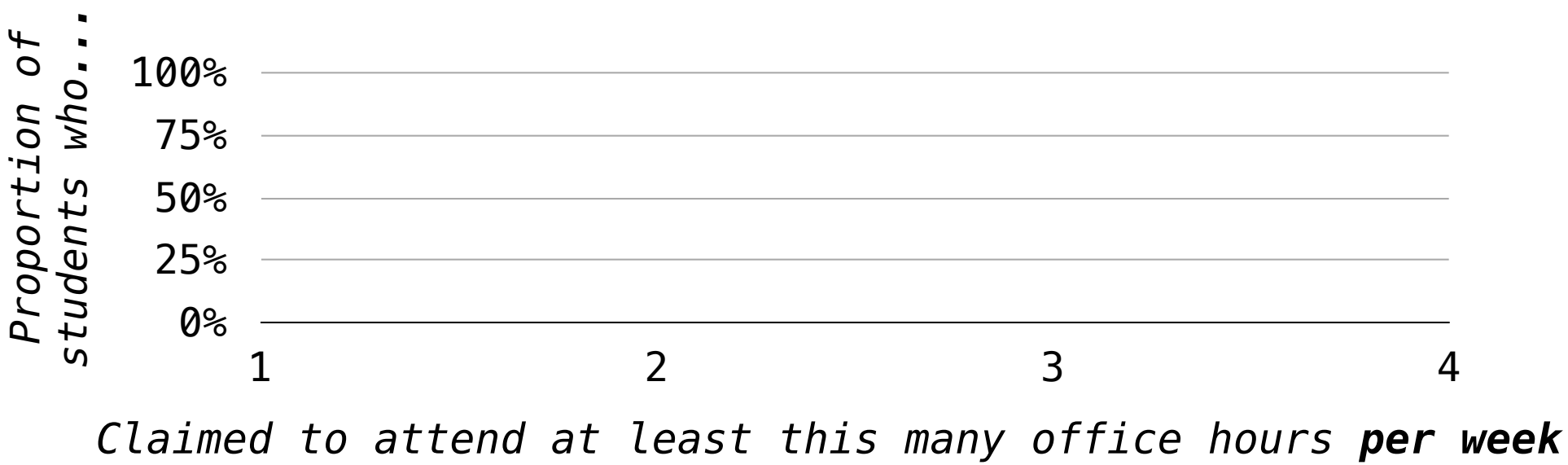
Computing is a collaborative, social discipline

I strongly encourage students to:

- Work with a partner on projects
- Discuss problems with classmates
- Attend office hours with questions

61% had a partner for all projects

True for **67%** of students with no prior programming experience



Community and Collaboration Guidelines

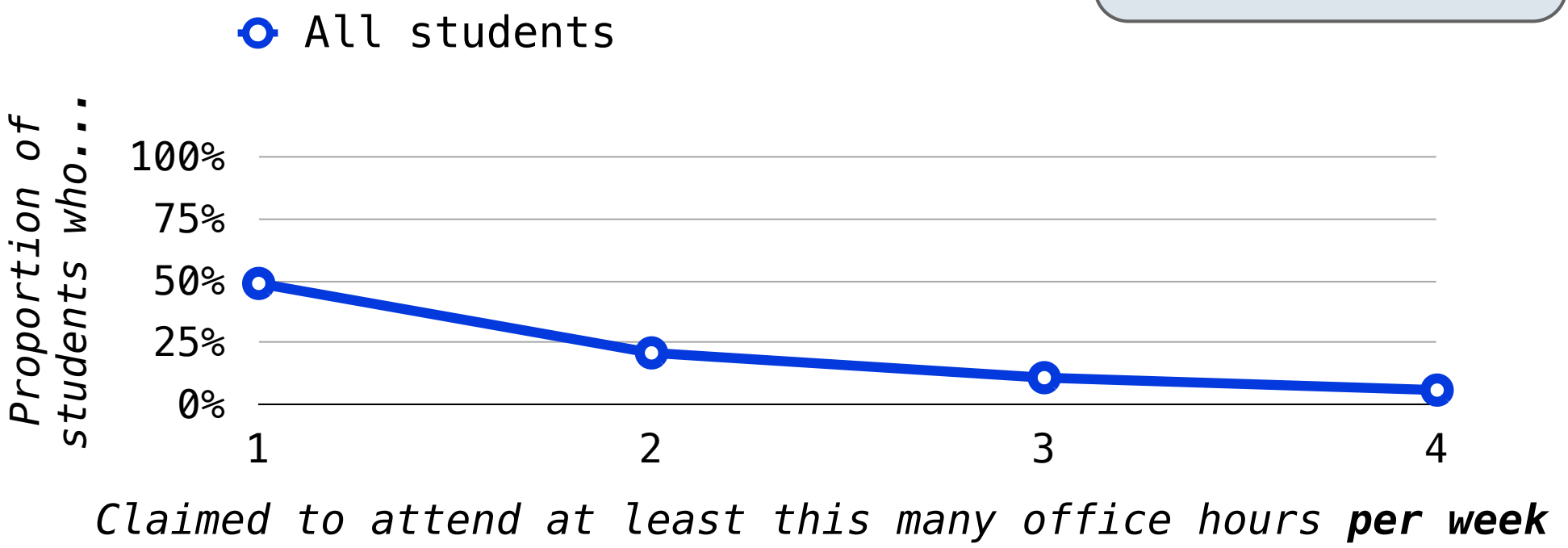
Computing is a collaborative, social discipline

I strongly encourage students to:

- Work with a partner on projects
- Discuss problems with classmates
- Attend office hours with questions

61% had a partner for all projects

True for **67%** of students with no prior programming experience



Community and Collaboration Guidelines

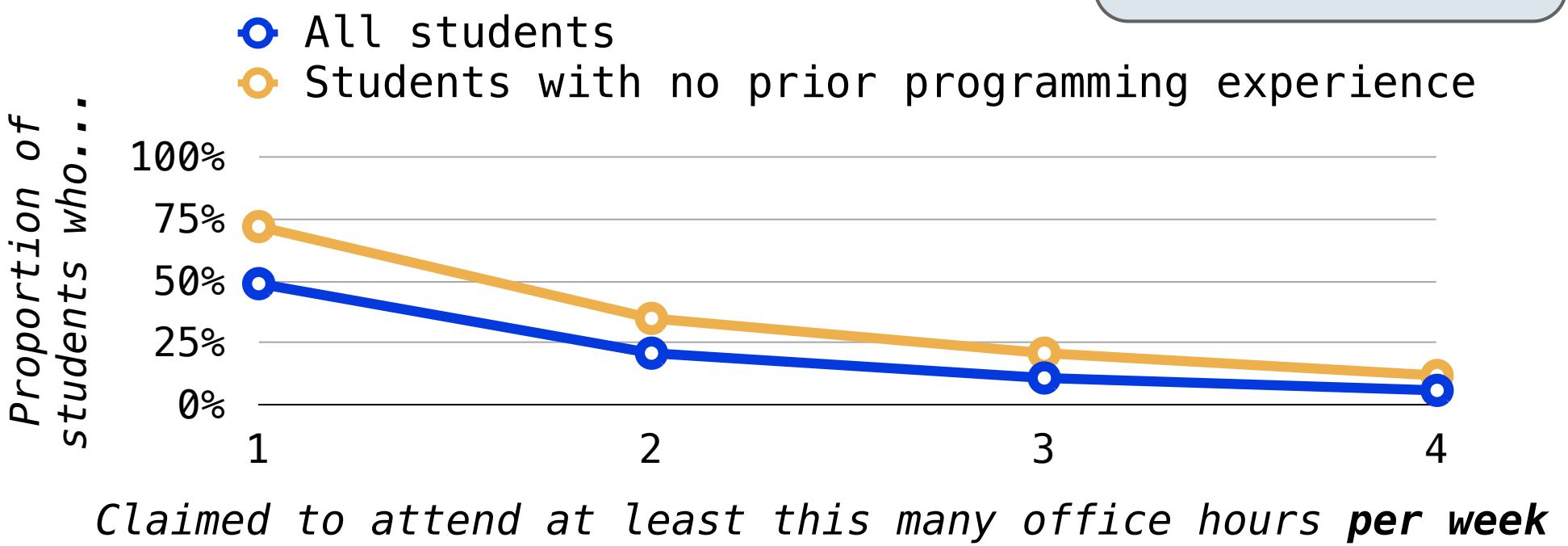
Computing is a collaborative, social discipline

I strongly encourage students to:

- Work with a partner on projects
- Discuss problems with classmates
- Attend office hours with questions

61% had a partner for all projects

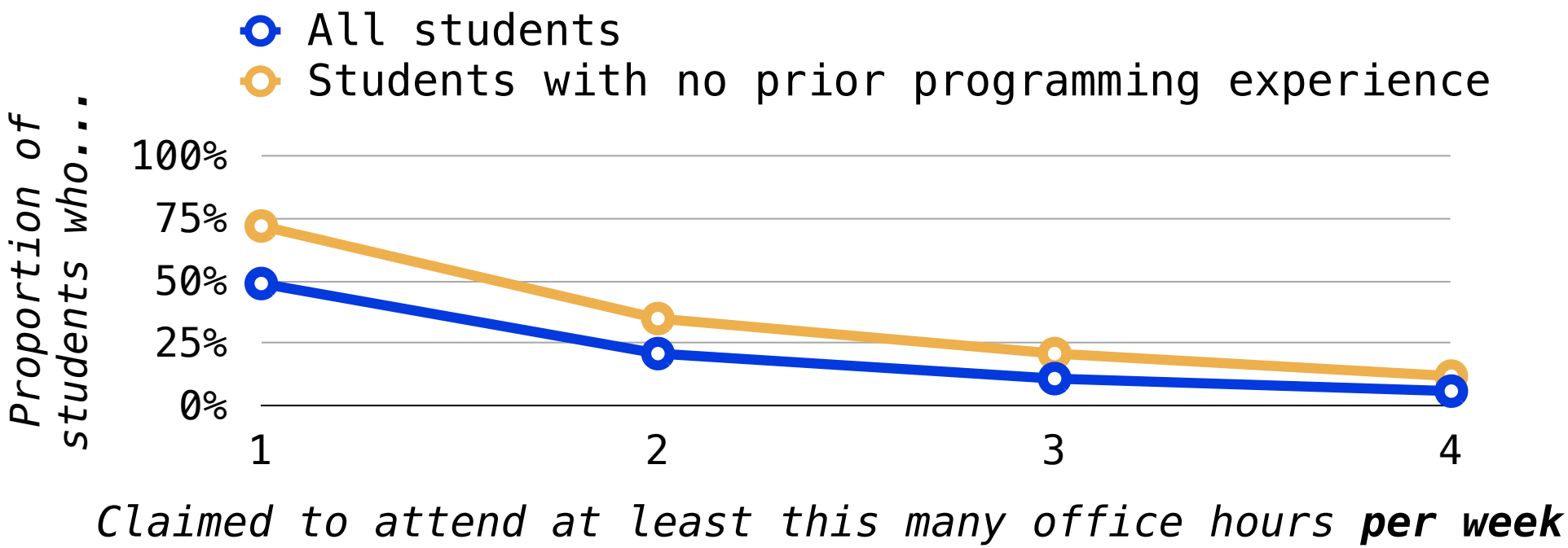
True for **67%** of students with no prior programming experience



Community and Collaboration Guidelines

Computing is a collaborative, social discipline

"I attended office hours religiously to get help with homework, projects, and key concepts of the class from all the TA's. Moreover, I worked with other students in office hours to further my understanding of the material by explaining concepts I already understood to them."



Connecting Students to External Communities

Connecting Students to External Communities

Python is maintained and used by a large community of open-source developers.

Connecting Students to External Communities

Python is maintained and used by a large community of open-source developers.

Benefits to students:

Connecting Students to External Communities

Python is maintained and used by a large community of open-source developers.

Benefits to students:

- Targeted explanations of language behavior (40,000+ questions answered on Stack Overflow)

Connecting Students to External Communities

Python is maintained and used by a large community of open-source developers.

Benefits to students:

- Targeted explanations of language behavior (40,000+ questions answered on Stack Overflow)
- Online worked examples for many problem domains

Connecting Students to External Communities

Python is maintained and used by a large community of open-source developers.

Benefits to students:

- Targeted explanations of language behavior (40,000+ questions answered on Stack Overflow)
- Online worked examples for many problem domains
- Strong library support for extracurricular projects

python

Materials & Tools

Composing Programs: An Interactive Textbook

Composing Programs: An Interactive Textbook

Composing Programs is a free online introduction to programming and computer science.

Composing Programs: An Interactive Textbook

Composing Programs is a free online introduction to programming and computer science.

A product of public domain and open source content:

Composing Programs: An Interactive Textbook

Composing Programs is a free online introduction to programming and computer science.

A product of public domain and open source content:

- Derived from Structure and Interpretation of Computer Programs, the former CS 61A text

Composing Programs: An Interactive Textbook

Composing Programs is a free online introduction to programming and computer science.

A product of public domain and open source content:

- Derived from Structure and Interpretation of Computer Programs, the former CS 61A text
- Examples diagrammed by the Online Python Tutor

Composing Programs: An Interactive Textbook

Composing Programs is a free online introduction to programming and computer science.

A product of public domain and open source content:

- Derived from Structure and Interpretation of Computer Programs, the former CS 61A text
- Examples diagrammed by the Online Python Tutor

Demo: http://composingprograms.com/pages/16-higher-order-functions.html#functions-as-arguments

Demo: http://composingprograms.com/pages/23-sequences.html#recursive-lists

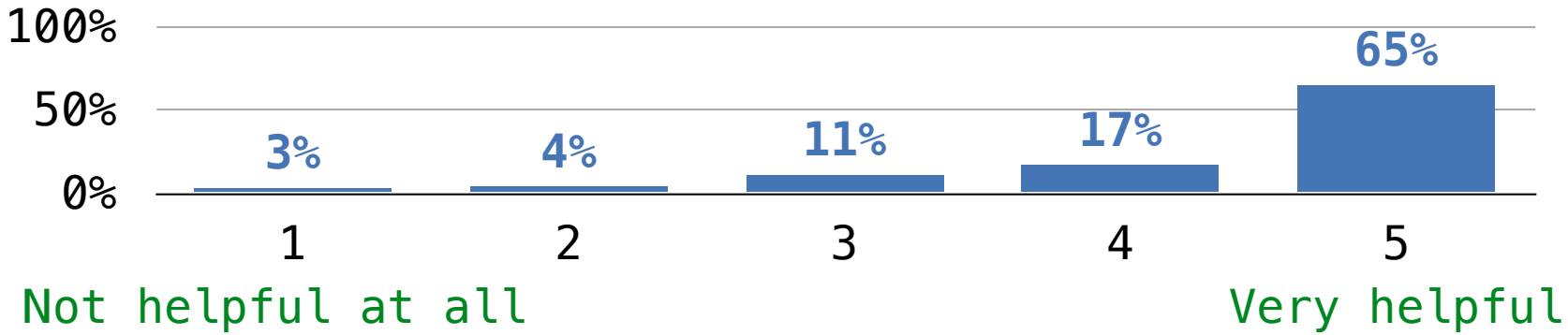
Composing Programs: An Interactive Textbook

Composing Programs is a free online introduction to programming and computer science.

A product of public domain and open source content:

- Derived from Structure and Interpretation of Computer Programs, the former CS 61A text
- Examples diagrammed by the Online Python Tutor

How helpful did you find the online tool for drawing environment diagrams in understanding course material?



Demo: http://composingprograms.com/pages/16-higher-order-functions.html#functions-as-arguments

Demo: http://composingprograms.com/pages/23-sequences.html#recursive-lists

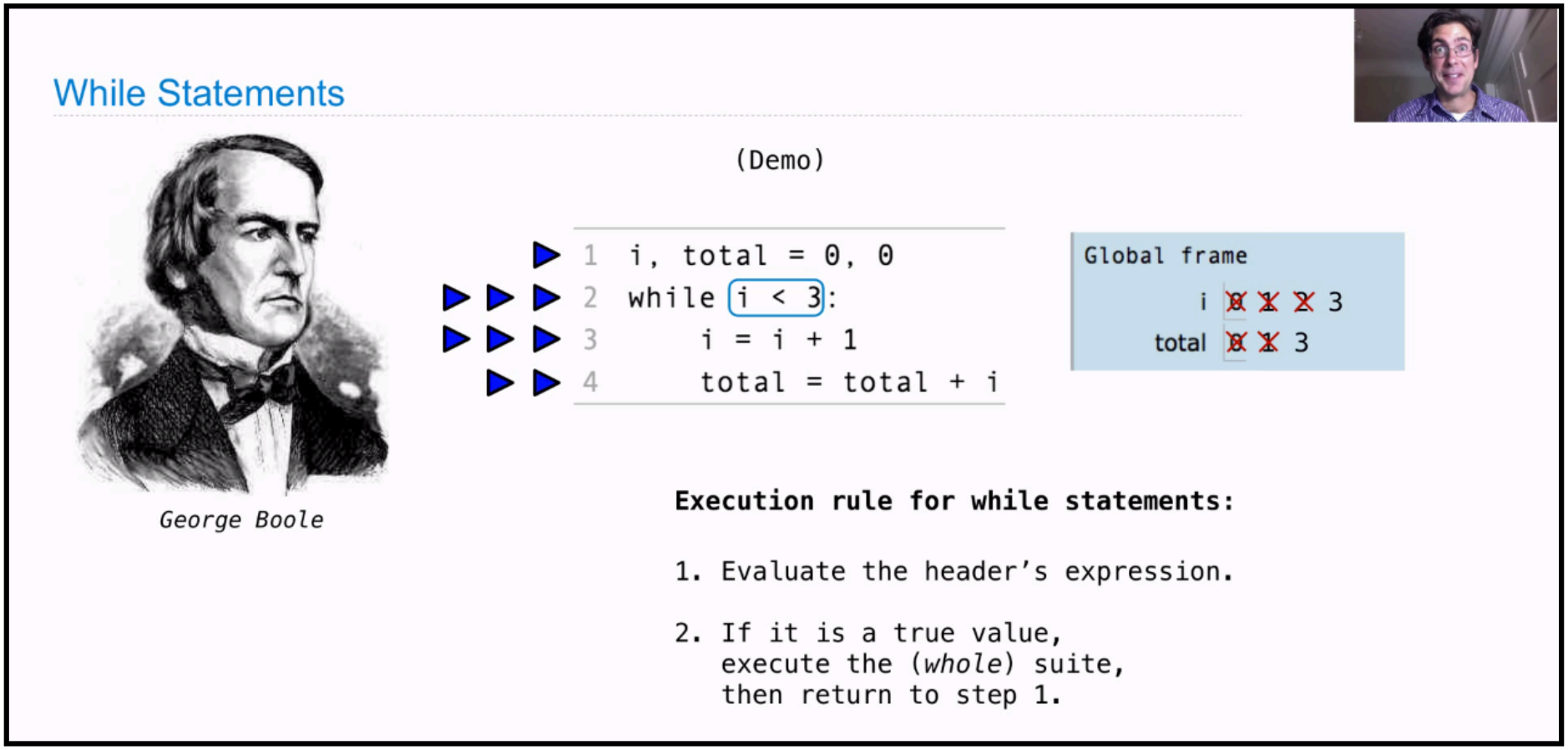
Expanding Beyond the Lecture Hall

Expanding Beyond the Lecture Hall

Video lectures allow students to pause & experiment.

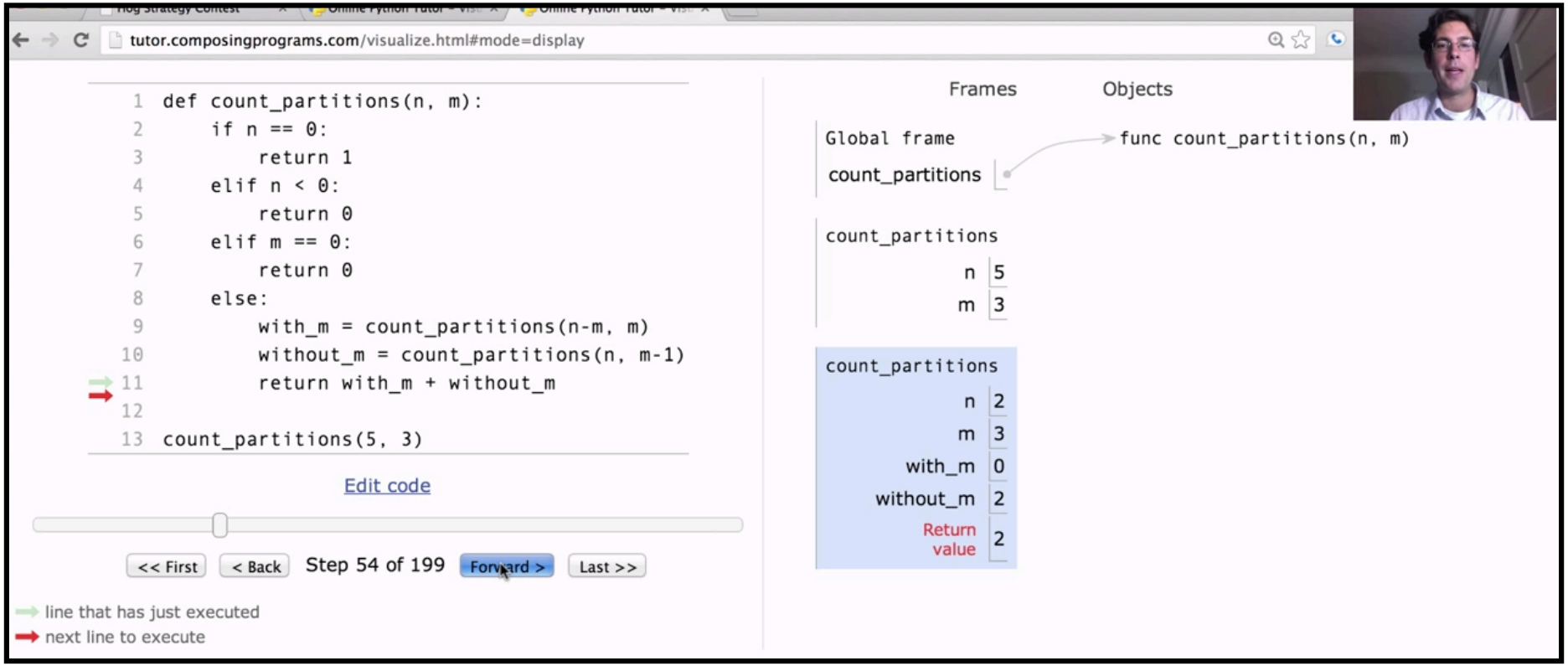
Expanding Beyond the Lecture Hall

Video lectures allow students to pause & experiment.



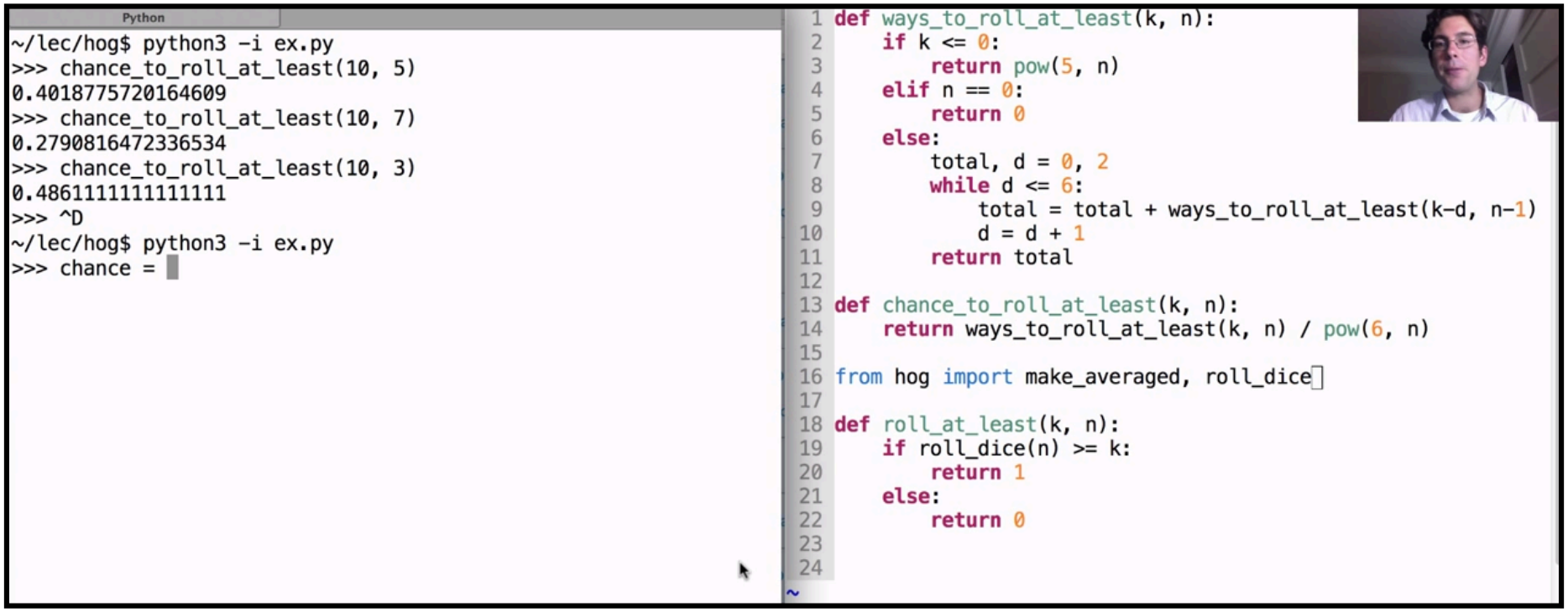
Expanding Beyond the Lecture Hall

Video lectures allow students to pause & experiment.



Expanding Beyond the Lecture Hall

Video lectures allow students to pause & experiment.



Expanding Beyond the Lecture Hall

Video lectures allow students to pause & experiment.

"I watched most of the videos at home where I was able to pause when I didn't understand a concept. I thought that being able to do so really made it so that I could learn at my own pace and thoroughly understand something before moving on."

Expanding Beyond the Lecture Hall

Video lectures allow students to pause & experiment.

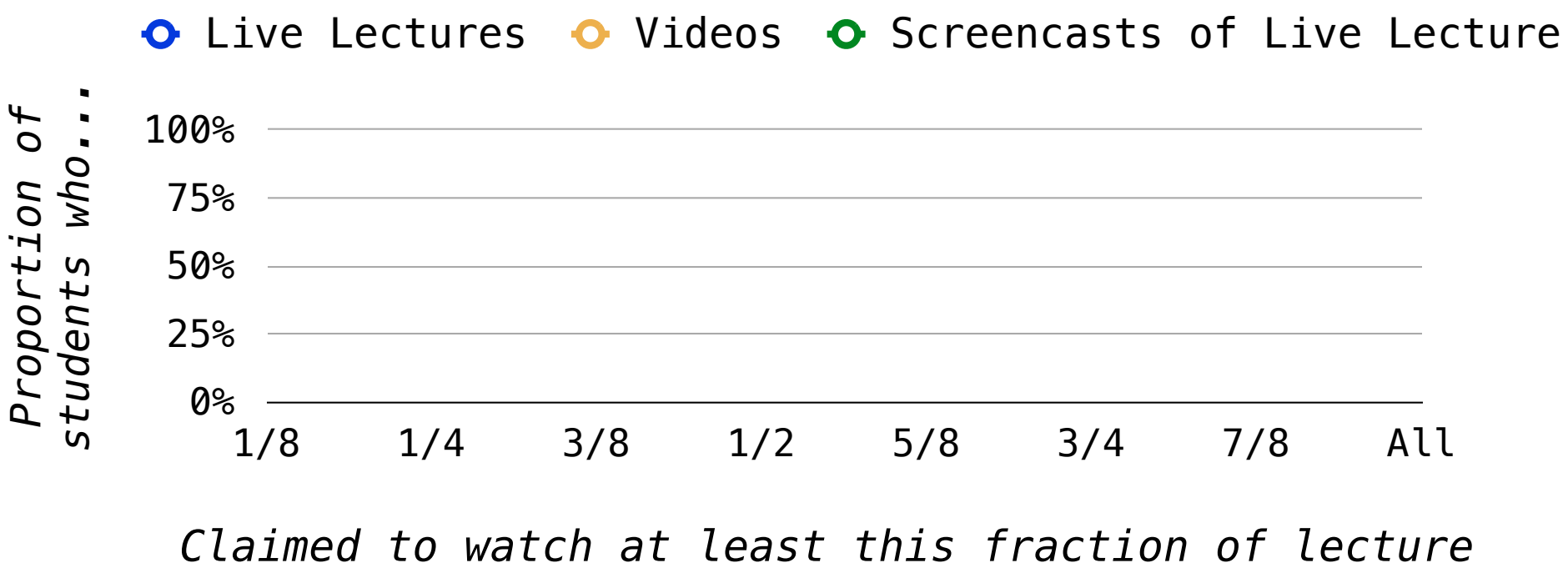
"I watched most of the videos at home where I was able to pause when I didn't understand a concept. I thought that being able to do so really made it so that I could learn at my own pace and thoroughly understand something before moving on."

Live Lectures Videos Screencasts of Live Lecture

Expanding Beyond the Lecture Hall

Video lectures allow students to pause & experiment.

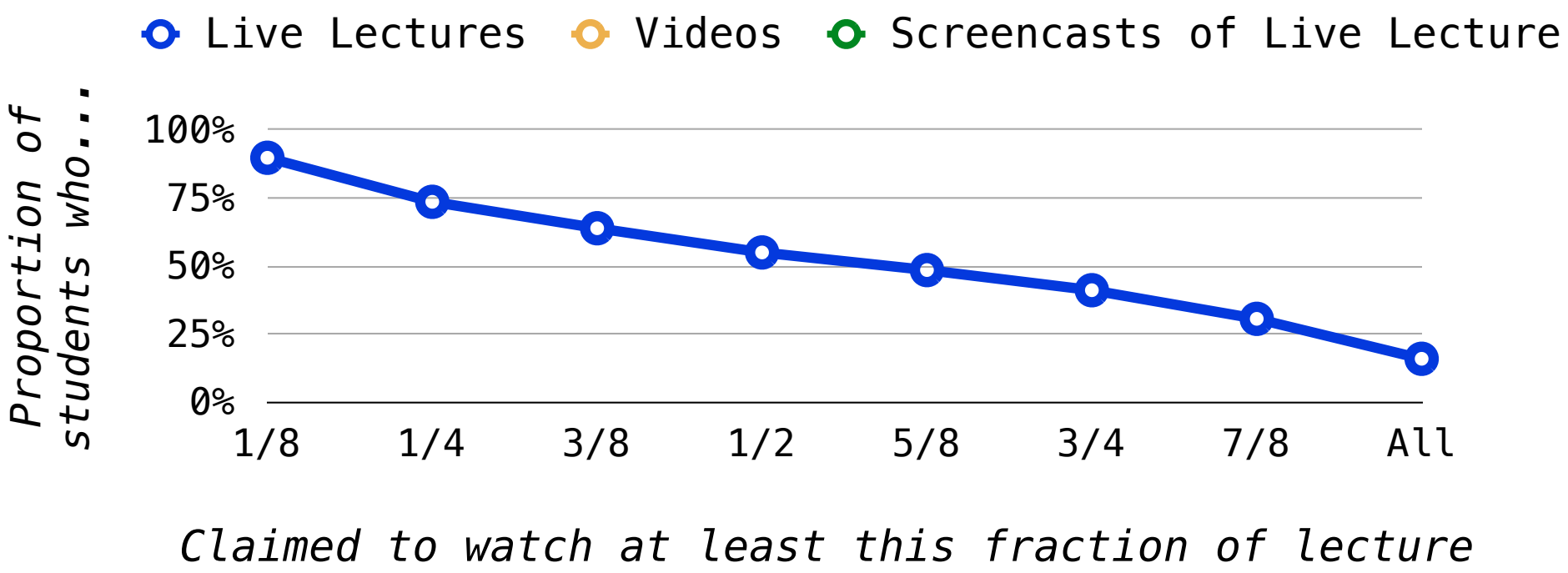
"I watched most of the videos at home where I was able to pause when I didn't understand a concept. I thought that being able to do so really made it so that I could learn at my own pace and thoroughly understand something before moving on."



Expanding Beyond the Lecture Hall

Video lectures allow students to pause & experiment.

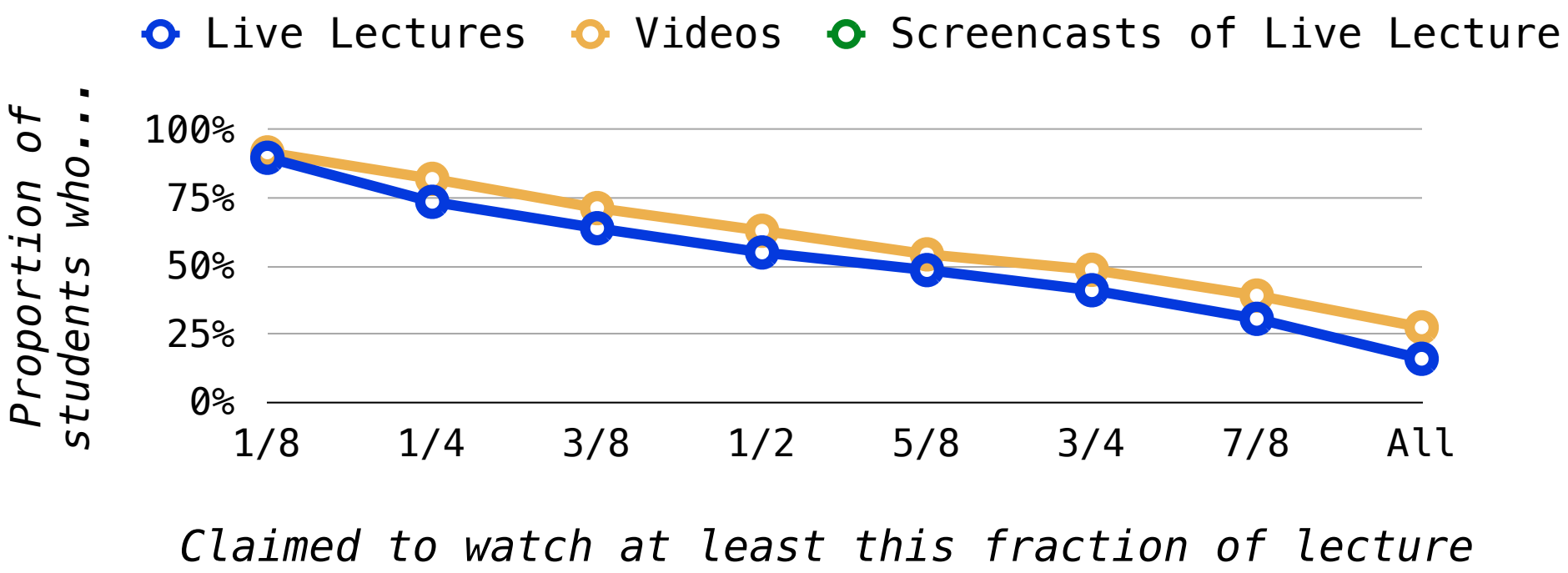
"I watched most of the videos at home where I was able to pause when I didn't understand a concept. I thought that being able to do so really made it so that I could learn at my own pace and thoroughly understand something before moving on."



Expanding Beyond the Lecture Hall

Video lectures allow students to pause & experiment.

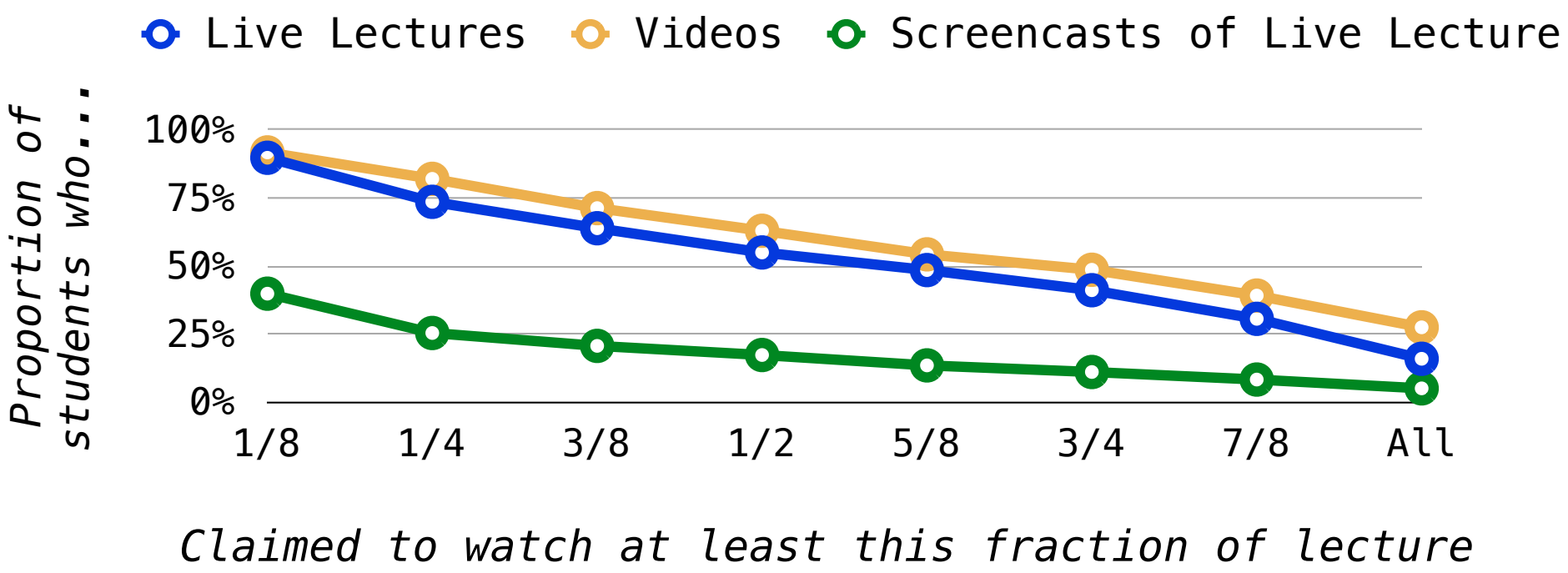
"I watched most of the videos at home where I was able to pause when I didn't understand a concept. I thought that being able to do so really made it so that I could learn at my own pace and thoroughly understand something before moving on."



Expanding Beyond the Lecture Hall

Video lectures allow students to pause & experiment.

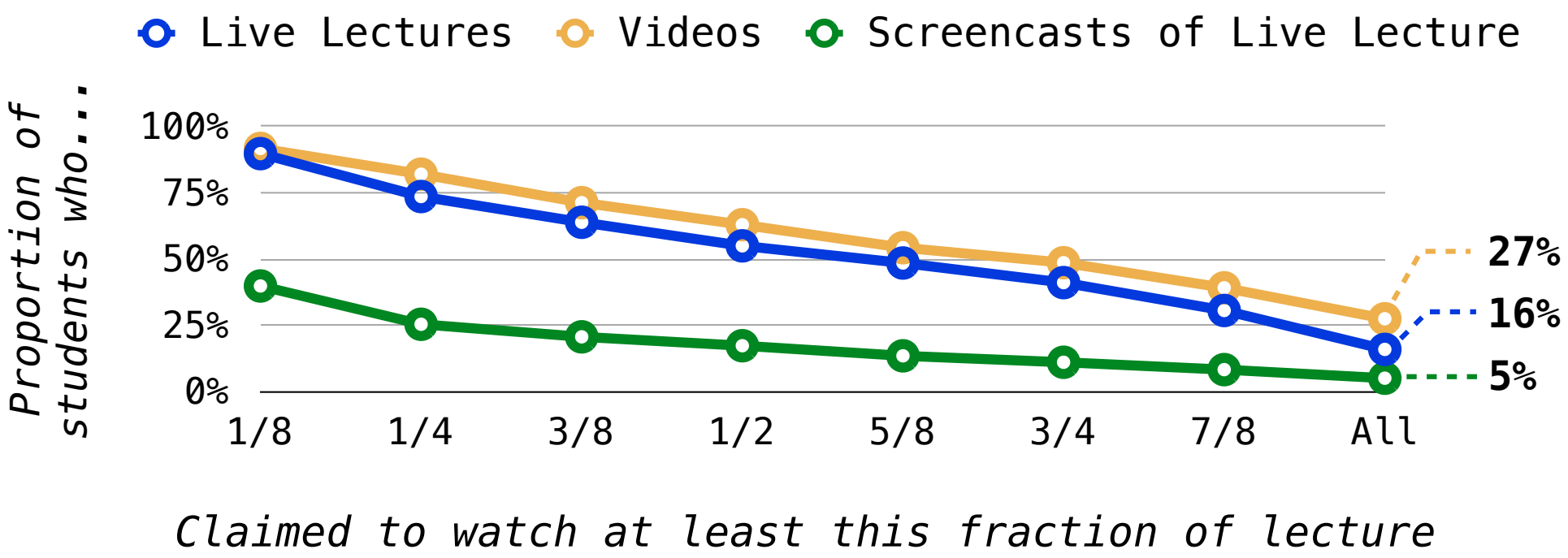
"I watched most of the videos at home where I was able to pause when I didn't understand a concept. I thought that being able to do so really made it so that I could learn at my own pace and thoroughly understand something before moving on."



Expanding Beyond the Lecture Hall

Video lectures allow students to pause & experiment.

"I watched most of the videos at home where I was able to pause when I didn't understand a concept. I thought that being able to do so really made it so that I could learn at my own pace and thoroughly understand something before moving on."



Online Code Review for Education

Online Code Review for Education

CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

Online Code Review for Education

CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

"Programs must be written for people to read,
and only incidentally for machines to execute."
(Structure and Interpretation of Computer Programs)

Online Code Review for Education

CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

"Programs must be written for people to read, and only incidentally for machines to execute."
(Structure and Interpretation of Computer Programs)

```
48          if insect.is_ant():
49              # Phase 2: Special handling for BodyguardAnt
50              "*** YOUR CODE HERE ***"
51              if self.ant != None and self.ant.can_contain(insect):
52                  self.ant.contain_ant(insect)
53
54              elif self.ant != None and insect.can_contain(self.ant):
55                  insect.contain_ant(self.ant)
56                  self.ant = insect
57
58              else:
59                  assert self.ant is None, 'Two ants in {0}'.format(self)
```

STAFF USERNAME HERE DATE HERE
This assert statement is supposed to be used for all cases: It makes sure there is no ant already in the place before you add another ant. Putting this in the else clause unnecessarily limits its ability to check your code. Instead, you should put this outside the if/else clauses and avoid reassigning self.ant until you have passed through the assert statement.
Reply Done

```
60                  self.ant = insect
61          else:
62              self.bees.append(insect)
```

Online Code Review for Education

CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

"Programs must be written for people to read, and only incidentally for machines to execute."
(Structure and Interpretation of Computer Programs)

Starter code

```
if insect.is_ant():
    # Phase 2: Special handling for BodyguardAnt
    "*** YOUR CODE HERE ***"
51          if self.ant != None and self.ant.can_contain(insect):
52              self.ant.contain_ant(insect)
53
54          elif self.ant != None and insect.can_contain(self.ant):
55              insect.contain_ant(self.ant)
56              self.ant = insect
57
58          else:
59              assert self.ant is None, 'Two ants in {0}'.format(self)
```

STAFF USERNAME HERE DATE HERE
This assert statement is supposed to be used for all cases: It makes sure there is no ant already in the place before you add another ant. Putting this in the else clause unnecessarily limits its ability to check your code. Instead, you should put this outside the if/else clauses and avoid reassigning self.ant until you have passed through the assert statement.
Reply Done

```
60              self.ant = insect
61      else:
62          self.bees.append(insect)
```

Online Code Review for Education

CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

"Programs must be written for people to read, and only incidentally for machines to execute."
(Structure and Interpretation of Computer Programs)

Starter code

Lines added by a student

```
if insect.is_ant():
    # Phase 2: Special handling for BodyguardAnt
    "*** YOUR CODE HERE ***"
51      if self.ant != None and self.ant.can_contain(insect):
52          self.ant.contain_ant(insect)
53
54      elif self.ant != None and insect.can_contain(self.ant):
55          insect.contain_ant(self.ant)
56          self.ant = insect
57
58      else:
59          assert self.ant is None, 'Two ants in {0}'.format(self)
```

STAFF USERNAME HERE DATE HERE
This assert statement is supposed to be used for all cases: It makes sure there is no ant already in the place before you add another ant. Putting this in the else clause unnecessarily limits its ability to check your code. Instead, you should put this outside the if/else clauses and avoid reassigning self.ant until you have passed through the assert statement.
Reply Done

```
60          self.ant = insect
61      else:
62          self.bees.append(insect)
```

Online Code Review for Education

CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

"Programs must be written for people to read, and only incidentally for machines to execute."
(Structure and Interpretation of Computer Programs)

Starter code

Lines added by a student

Indent added by a student

```
if insect.is_ant():
    # Phase 2: Special handling for BodyguardAnt
    "*** YOUR CODE HERE ***"
51      if self.ant != None and self.ant.can_contain(insect):
52          self.ant.contain_ant(insect)
53
        elif self.ant != None and insect.can_contain(self.ant):
            insect.contain_ant(self.ant)
            self.ant = insect

        else:
59          assert self.ant is None, 'Two ants in {0}'.format(self)
```

STAFF USERNAME HERE DATE HERE
This assert statement is supposed to be used for all cases: It makes sure there is no ant already in the place before you add another ant. Putting this in the else clause unnecessarily limits its ability to check your code. Instead, you should put this outside the if/else clauses and avoid reassigning self.ant until you have passed through the assert statement.
Reply Done

```
60          self.ant = insect
61      else:
62          self.bees.append(insect)
```

Online Code Review for Education

CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

"Programs must be written for people to read, and only incidentally for machines to execute."
(Structure and Interpretation of Computer Programs)

Starter code

Lines added by a student

Indent added by a student

Comment by a reader

```
if insect.is_ant():
    # Phase 2: Special handling for BodyguardAnt
    "*** YOUR CODE HERE ***"
51      if self.ant != None and self.ant.can_contain(insect):
52          self.ant.contain_ant(insect)
53
        elif self.ant != None and insect.can_contain(self.ant):
            insect.contain_ant(self.ant)
            self.ant = insect

58      else:
59          assert self.ant is None, 'Two ants in {0}'.format(self)
```

STAFF USERNAME HERE DATE HERE
This assert statement is supposed to be used for all cases: It makes sure there is no ant already in the place before you add another ant. Putting this in the else clause unnecessarily limits its ability to check your code. Instead, you should put this outside the if/else clauses and avoid reassigning self.ant until you have passed through the assert statement.
Reply Done

```
60          self.ant = insect
61      else:
62          self.bees.append(insect)
```

Online Code Review for Education

CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

"Programs must be written for people to read, and only incidentally for machines to execute."
(Structure and Interpretation of Computer Programs)

Starter code

Lines added by a student

Indent added by a student

Comment by a reader

The Student can reply

```
if insect.is_ant():
    # Phase 2: Special handling for BodyguardAnt
    "*** YOUR CODE HERE ***"
51  if self.ant != None and self.ant.can_contain(insect):
52      self.ant.contain_ant(insect)
53
    elif self.ant != None and insect.can_contain(self.ant):
        insect.contain_ant(self.ant)
        self.ant = insect

    else:
59      assert self.ant is None, 'Two ants in {0}'.format(self)
```

STAFF USERNAME HERE DATE HERE
This assert statement is supposed to be used for all cases: It makes sure there is no ant already in the place before you add another ant. Putting this in the else clause unnecessarily limits its ability to check your code. Instead, you should put this outside the if/else clauses and avoid reassigning self.ant until you have passed through the assert statement.
Reply

```
60  ect
61
62  nsect)
```

Online Code Review for Education

CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

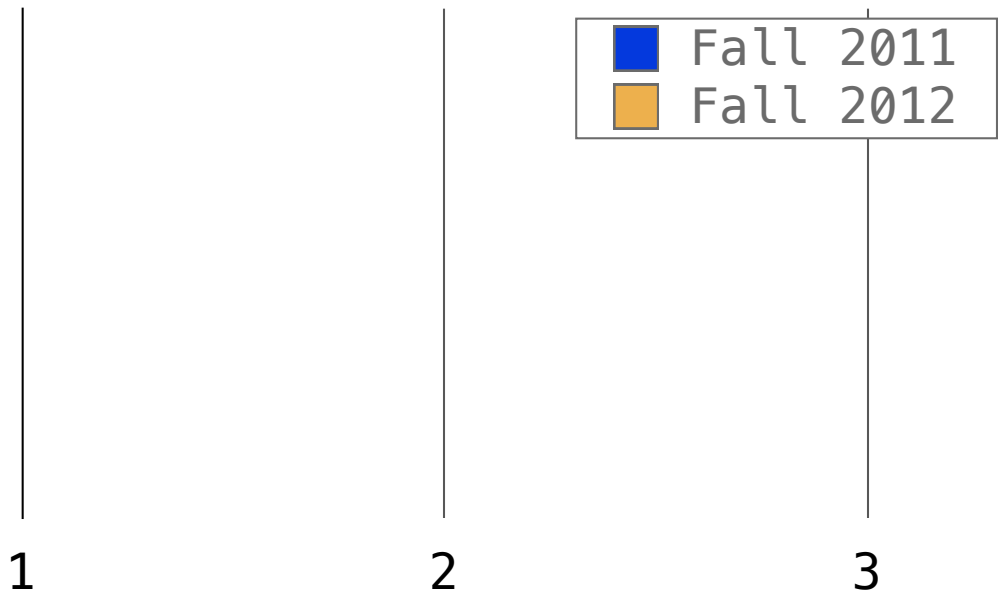
"Programs must be written for people to read,
and only incidentally for machines to execute."
(Structure and Interpretation of Computer Programs)

Online Code Review for Education

CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

"Programs must be written for people to read,
and only incidentally for machines to execute."
(Structure and Interpretation of Computer Programs)

Blind evaluation of a sample of submissions



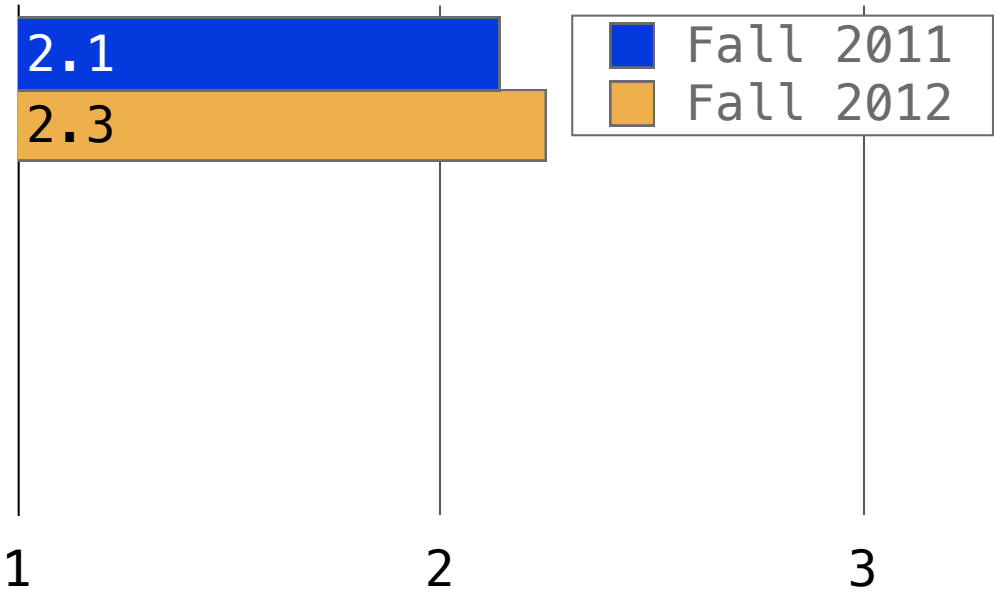
Online Code Review for Education

CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

"Programs must be written for people to read,
and only incidentally for machines to execute."
(Structure and Interpretation of Computer Programs)

Blind evaluation of a sample of submissions

Do names convey the meaning & purpose of their values?



Online Code Review for Education

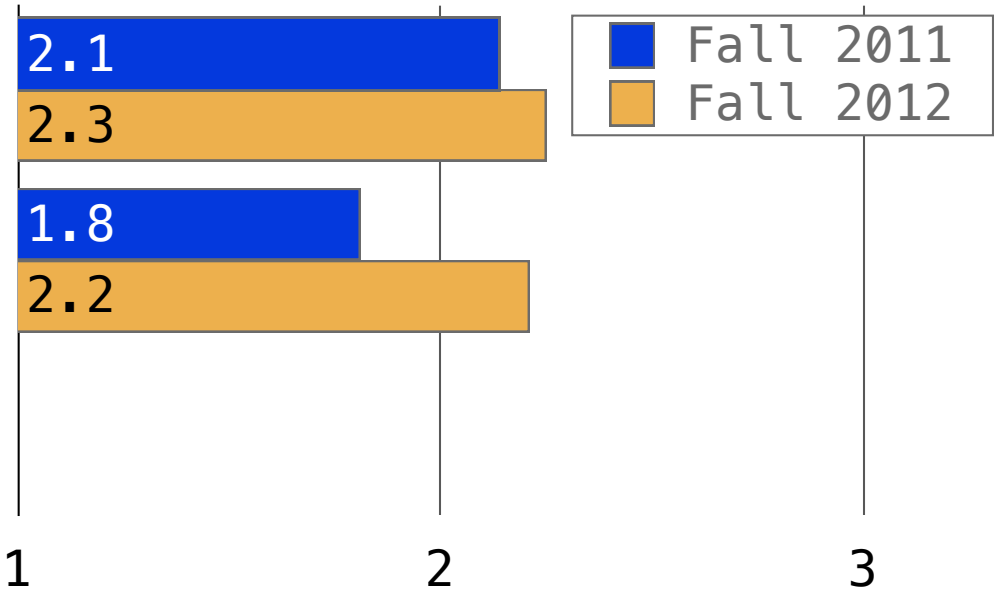
CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

"Programs must be written for people to read,
and only incidentally for machines to execute."
(Structure and Interpretation of Computer Programs)

Blind evaluation of a sample of submissions

Do names convey the meaning & purpose of their values?

Is the implementation concise?

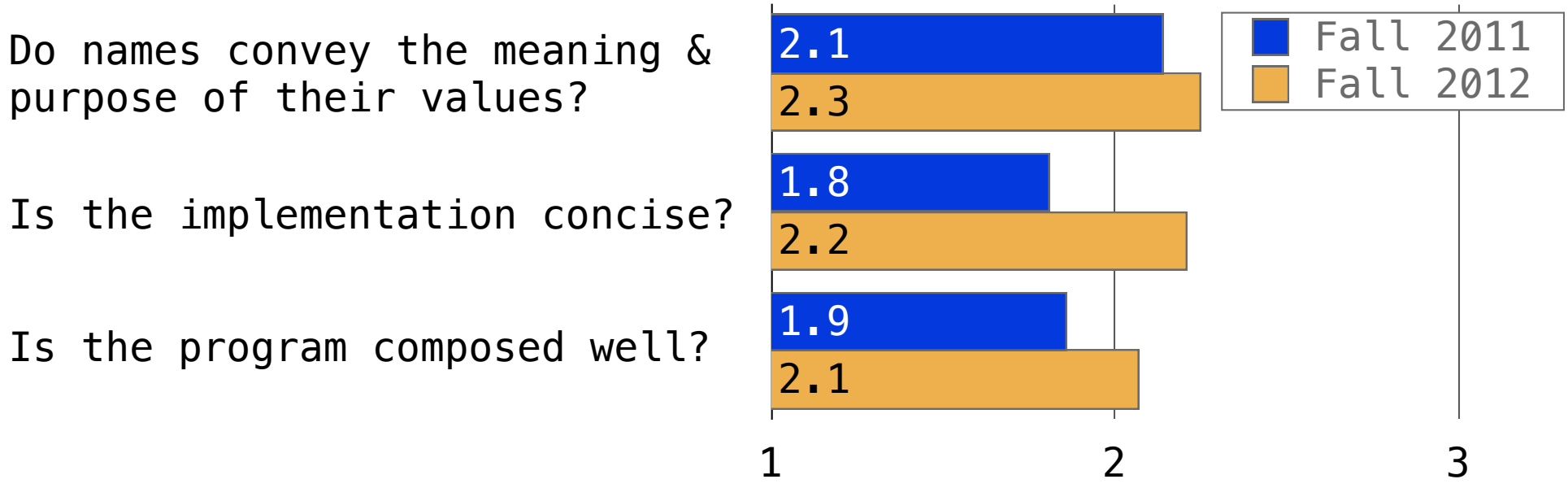


Online Code Review for Education

CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

"Programs must be written for people to read,
and only incidentally for machines to execute."
(Structure and Interpretation of Computer Programs)

Blind evaluation of a sample of submissions

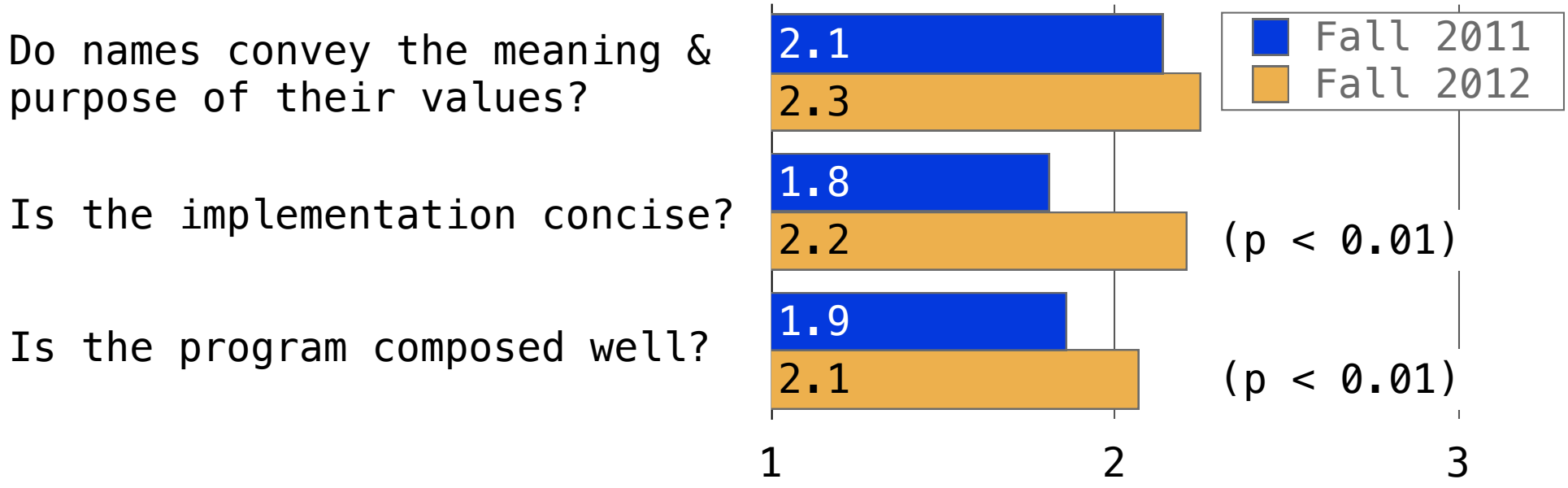


Online Code Review for Education

CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

"Programs must be written for people to read, and only incidentally for machines to execute."
(Structure and Interpretation of Computer Programs)

Blind evaluation of a sample of submissions

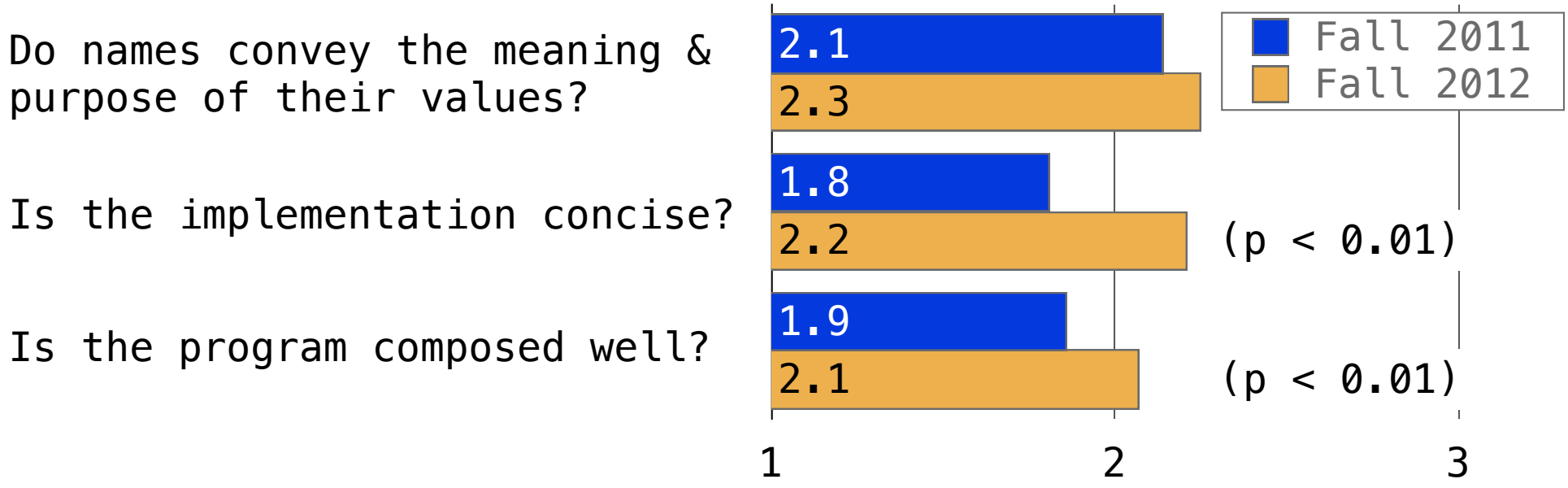


Online Code Review for Education

CS 61A uses a custom version of Google's (former) code review tool to give feedback about composition.

"Programs must be written for people to read, and only incidentally for machines to execute."
(Structure and Interpretation of Computer Programs)

Blind evaluation of a sample of submissions

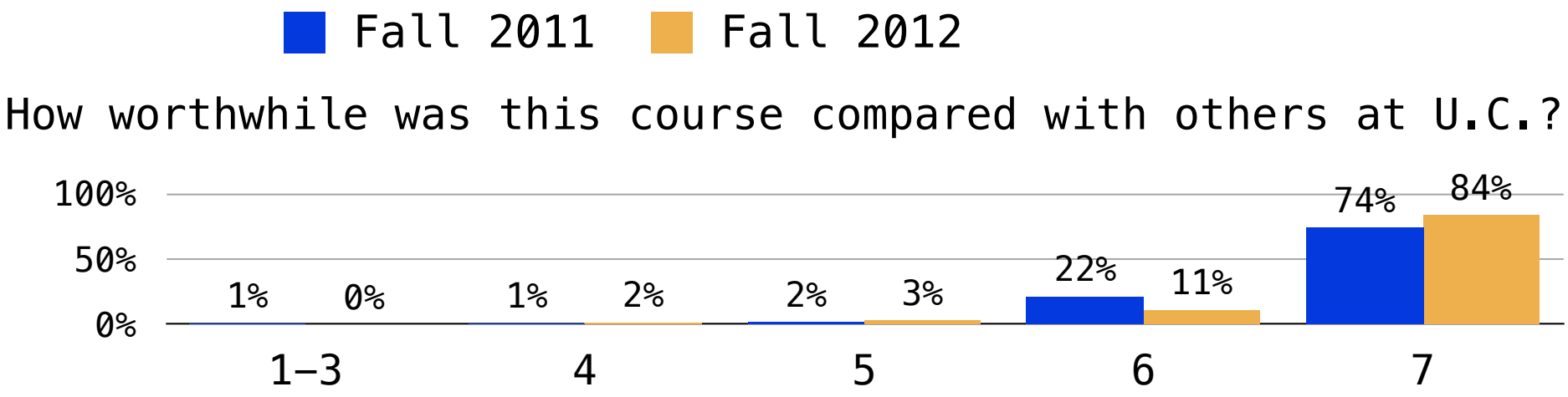


Teaching Composition Quality at Scale, DeNero and Martinis, SIGCSE 2014

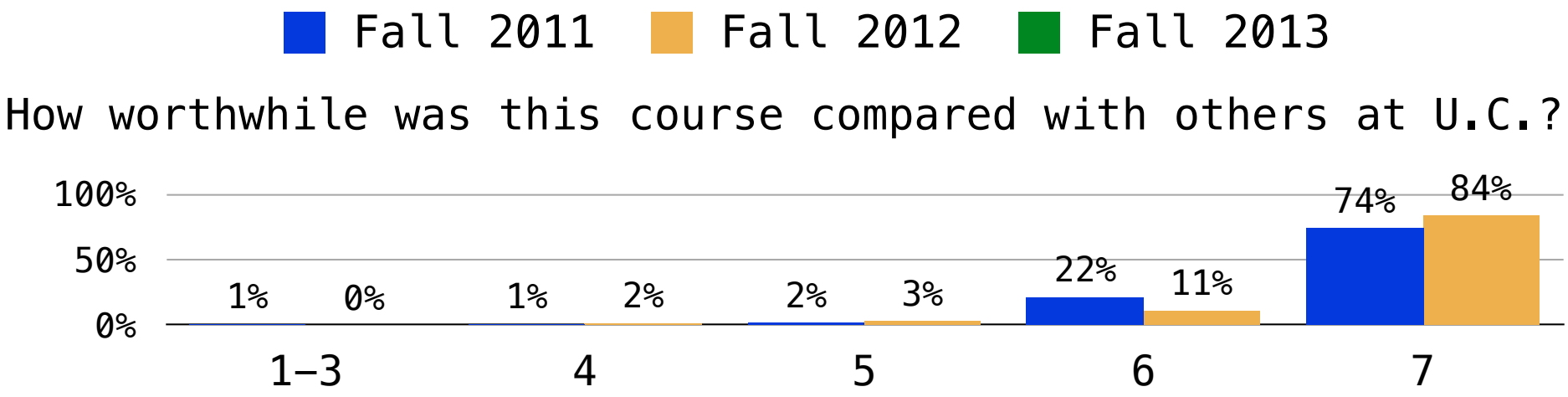
Course Trends

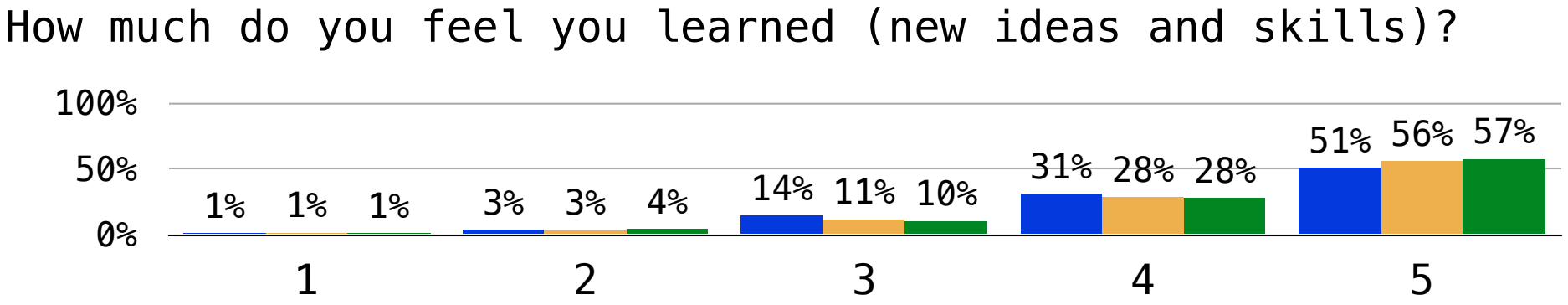
Overall Student Experience

Overall Student Experience



Overall Student Experience

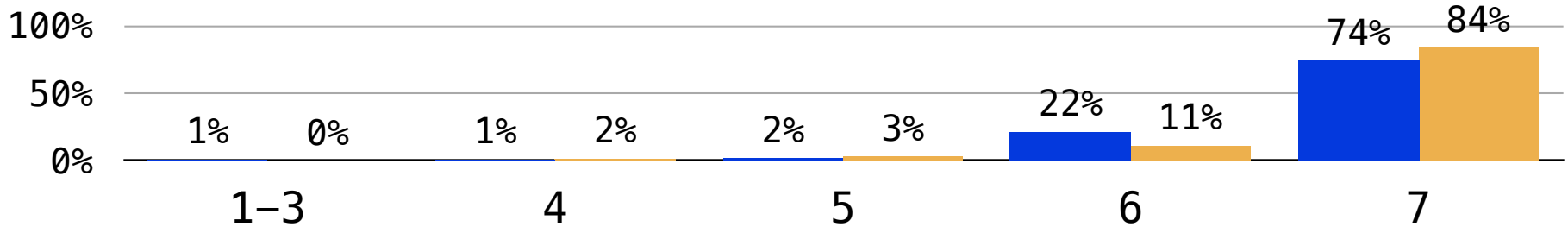




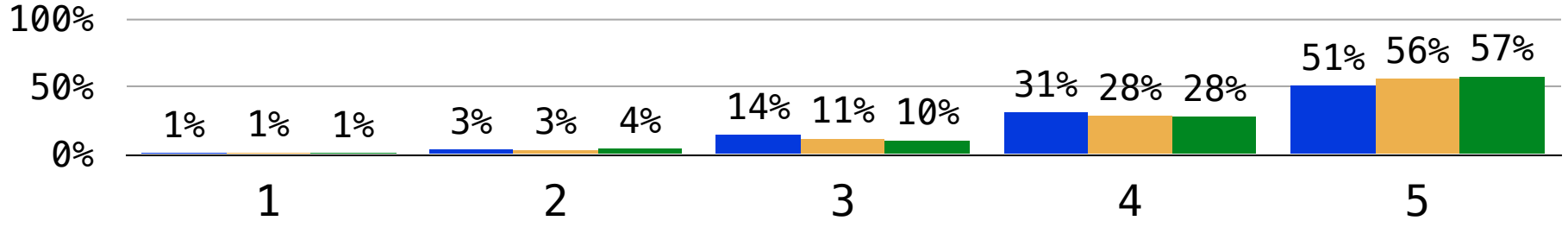
Overall Student Experience

Fall 2011 Fall 2012 Fall 2013

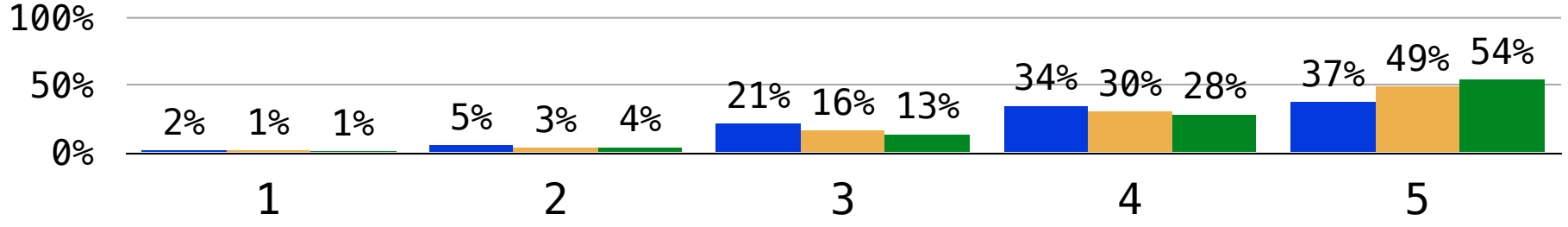
How worthwhile was this course compared with others at U.C.?



How much do you feel you learned (new ideas and skills)?



Relative to your expectations, how much do you feel you learned?



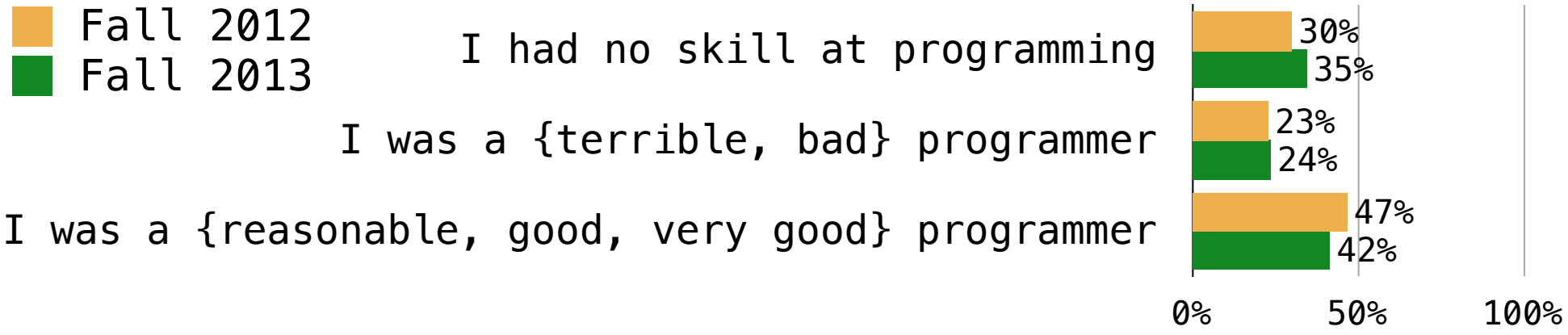
No Programming Experience Required

No Programming Experience Required

Before taking this course this semester, how good a programmer did you consider yourself to be?

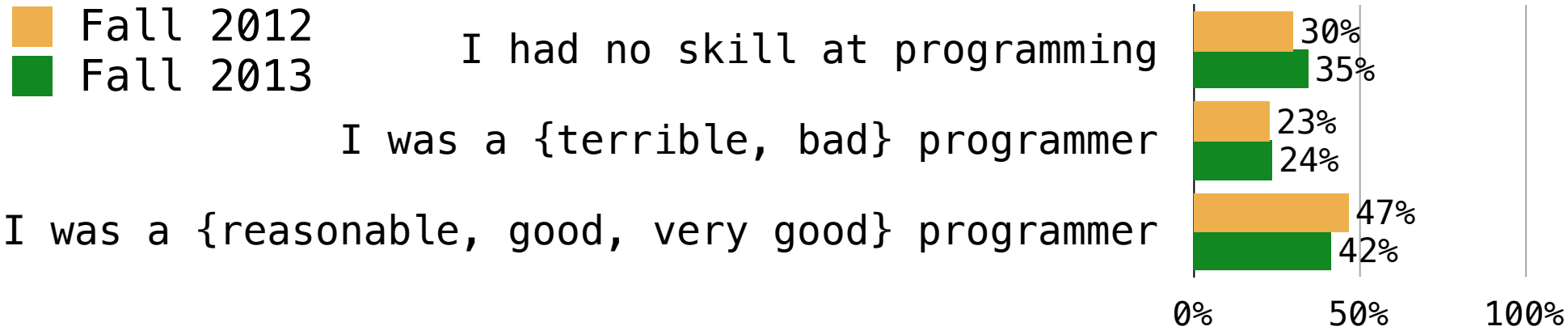
No Programming Experience Required

Before taking this course this semester, how good a programmer did you consider yourself to be?

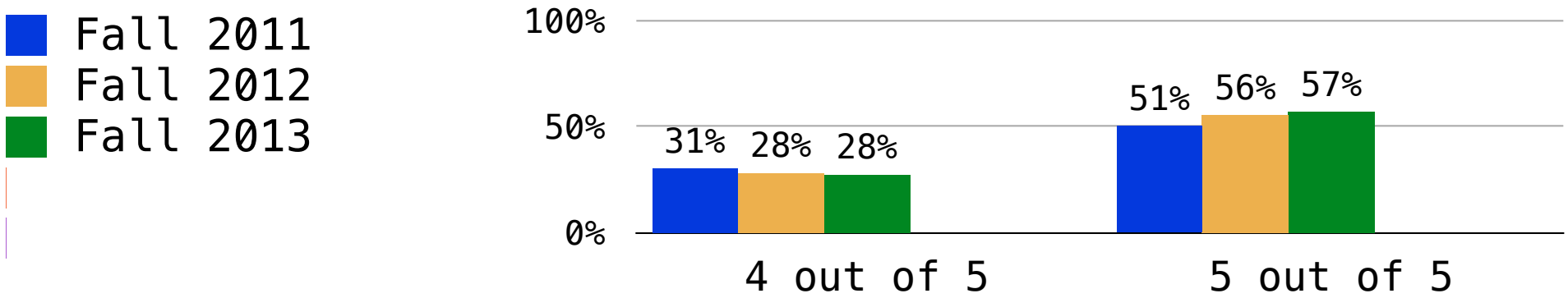


No Programming Experience Required

Before taking this course this semester, how good a programmer did you consider yourself to be?

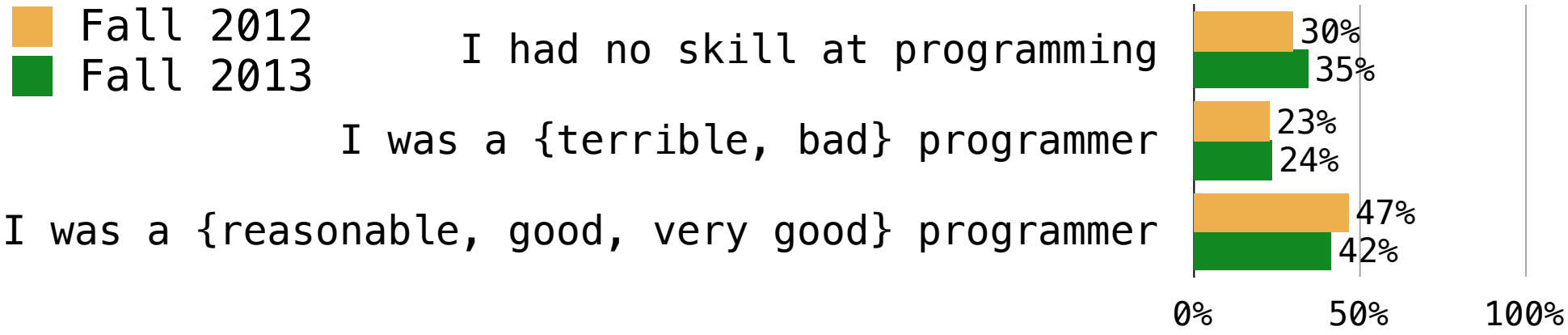


How much do you feel you learned (new ideas and skills)?

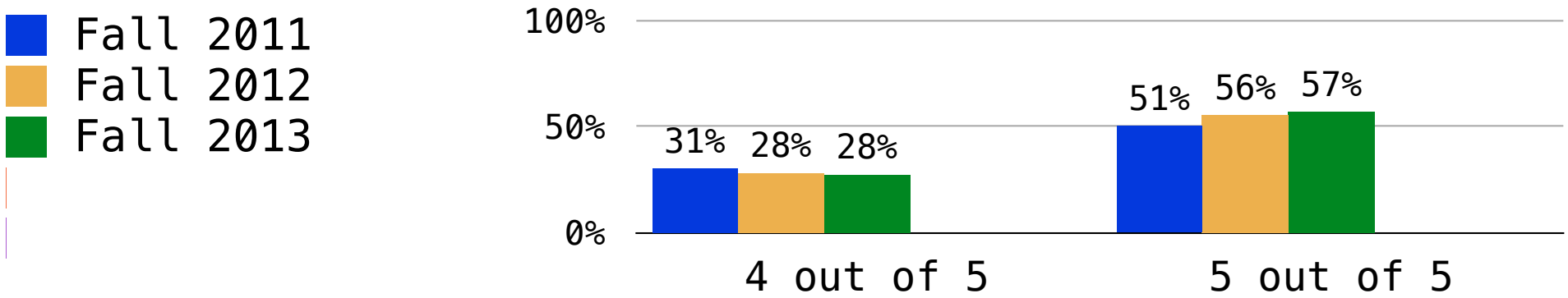


No Programming Experience Required

Before taking this course this semester, how good a programmer did you consider yourself to be?



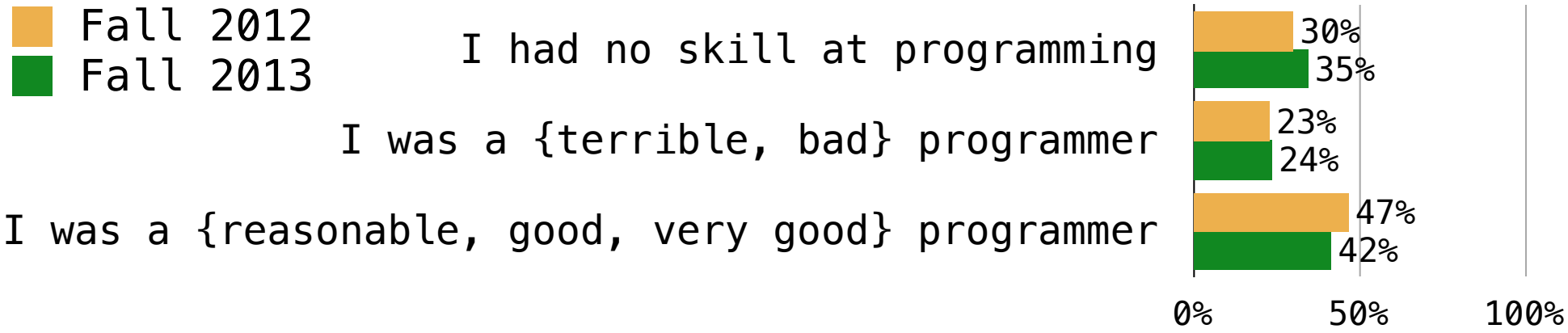
How much do you feel you learned (new ideas and skills)?



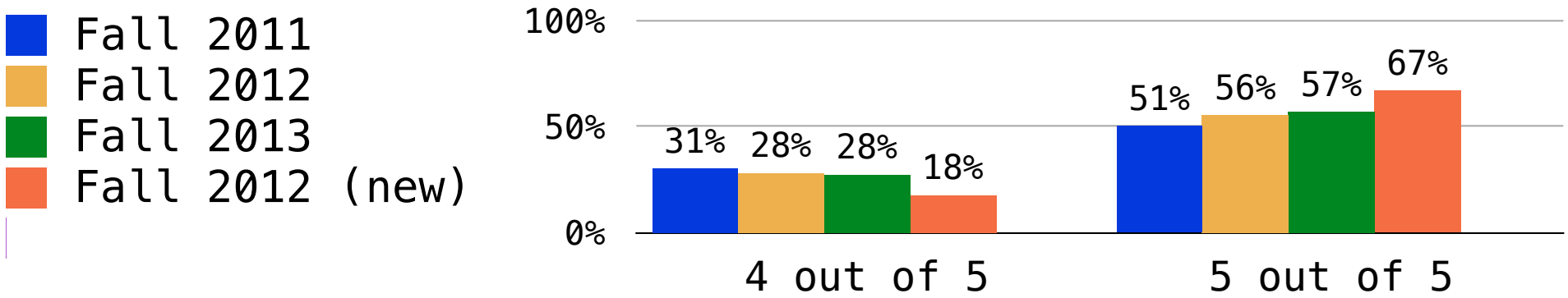
(new): "I had no prior skill at programming"

No Programming Experience Required

Before taking this course this semester, how good a programmer did you consider yourself to be?



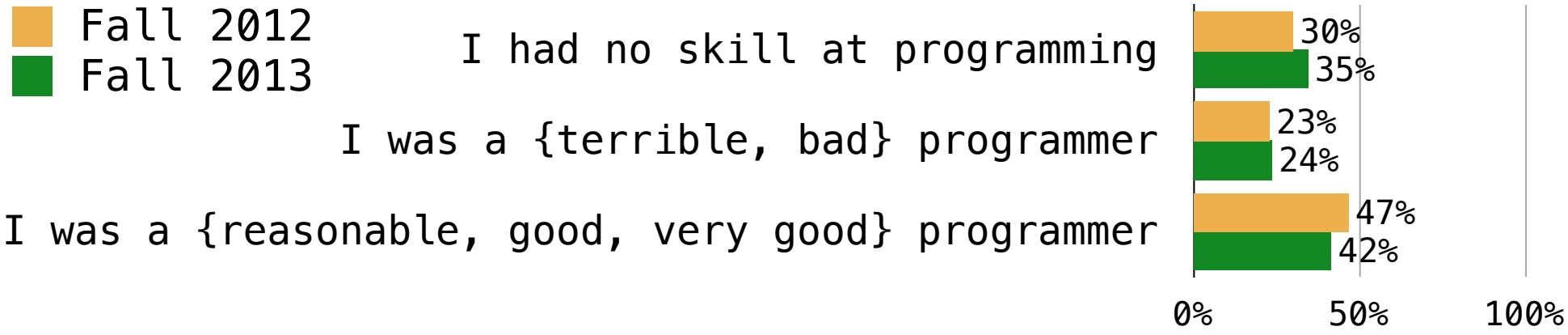
How much do you feel you learned (new ideas and skills)?



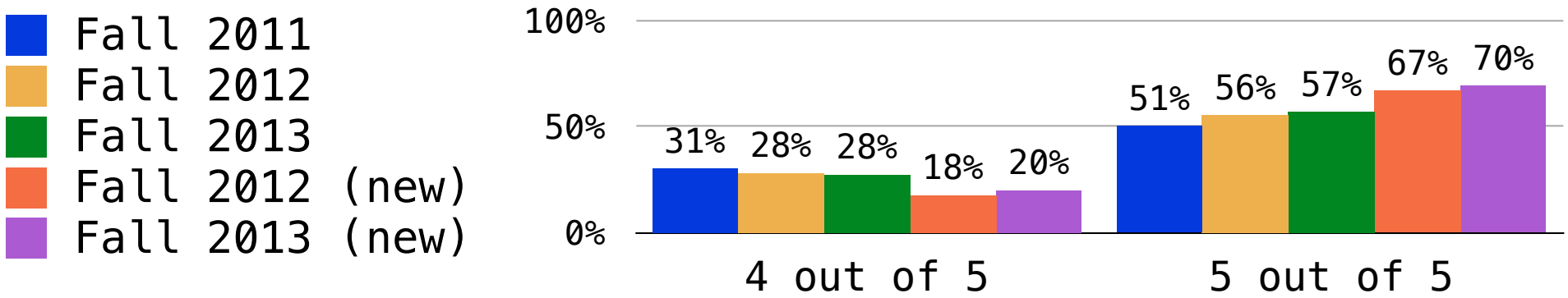
(new): "I had no prior skill at programming"

No Programming Experience Required

Before taking this course this semester, how good a programmer did you consider yourself to be?



How much do you feel you learned (new ideas and skills)?



(new): "I had no prior skill at programming"

Women in Computer Science 61A

Women in Computer Science 61A

The total number of women in CS 61A increased by **59%** from Fall 2012 to Fall 2013.

Women in Computer Science 61A

The total number of women in CS 61A increased by **59%** from Fall 2012 to Fall 2013.

46% of women in Fall 2013 did not have prior programming experience.

Women in Computer Science 61A

The total number of women in CS 61A increased by **59%** from Fall 2012 to Fall 2013.

46% of women in Fall 2013 did not have prior programming experience.

33% of students without prior experience were women.

Women in Computer Science 61A

The total number of women in CS 61A increased by **59%** from Fall 2012 to Fall 2013.

46% of women in Fall 2013 did not have prior programming experience.

33% of students without prior experience were women.

Relative to before you took CS 61A, how interested are you in pursuing computer science as a major?

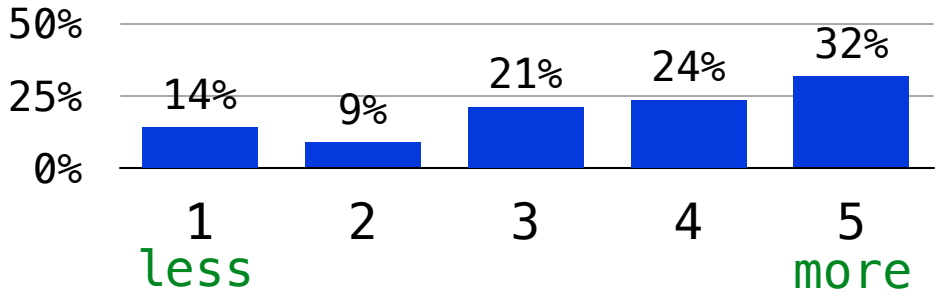
Women in Computer Science 61A

The total number of women in CS 61A increased by **59%** from Fall 2012 to Fall 2013.

46% of women in Fall 2013 did not have prior programming experience.

33% of students without prior experience were women.

Relative to before you took CS 61A, how interested are you in pursuing computer science as a major?



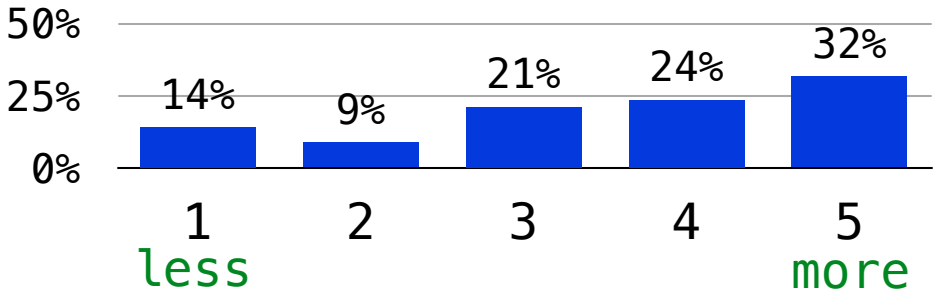
Women in Computer Science 61A

The total number of women in CS 61A increased by **59%** from Fall 2012 to Fall 2013.

46% of women in Fall 2013 did not have prior programming experience.

33% of students without prior experience were women.

Relative to before you took CS 61A, how interested are you in pursuing computer science as a major?



How happy are you about your decision to take CS 61A?

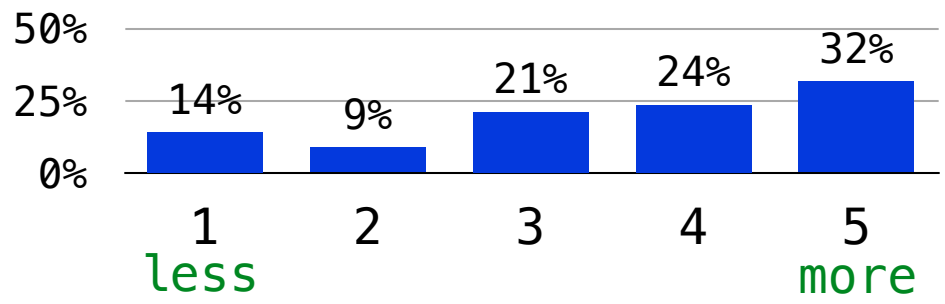
Women in Computer Science 61A

The total number of women in CS 61A increased by **59%** from Fall 2012 to Fall 2013.

46% of women in Fall 2013 did not have prior programming experience.

33% of students without prior experience were women.

Relative to before you took CS 61A, how interested are you in pursuing computer science as a major?



How happy are you about your decision to take CS 61A?