Teaching Composition Quality at Scale

Human Judgment in the Age of Autograders

John DeNero & Stephen Martinis

Google & Berkeley UNIVERSITY OF CALIFORNIA

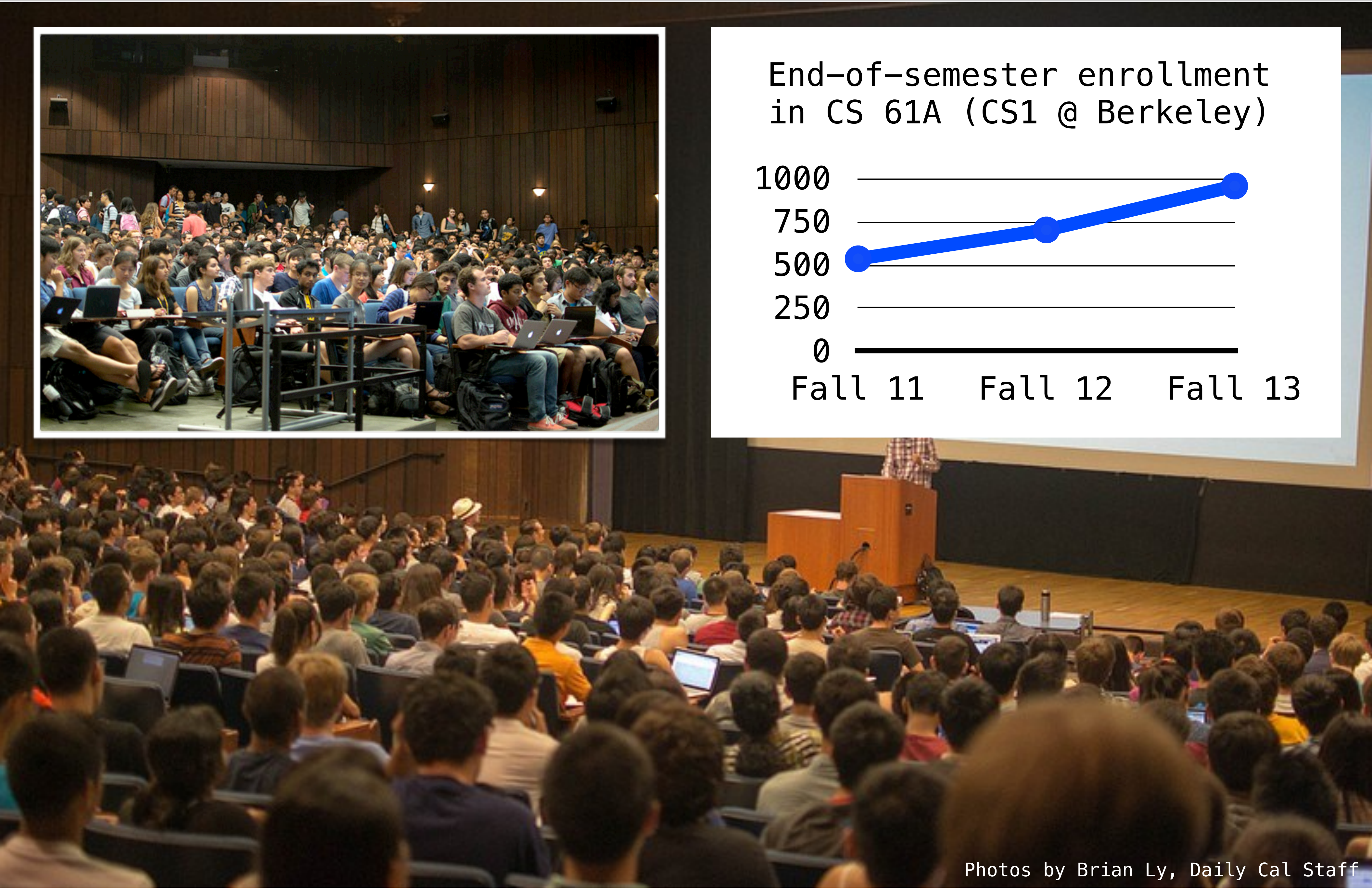
Computer Science Courses are Growing

Photos by Brian Ly, Daily Cal Staff

Computer Science Courses are Growing

Photos by Brian Ly, Daily Cal Staff

Computer Science Courses are Growing



End-of-semester enrollment in CS 61A (CS1 @ Berkeley)

Photos by Brian Ly, Daily Cal Staff

Teaching Composition Quality

Teaching Composition Quality

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

Teaching Composition Quality

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

- Guidelines for effective program composition

Teaching Composition Quality

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

- Guidelines for effective program composition
- Useful examples

Teaching Composition Quality

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

- Guidelines for effective program composition
- Useful examples
- Incentives

Teaching Composition Quality

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

- Guidelines for effective program composition
- Useful examples
- Incentives
- Feedback

Teaching Composition Quality

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

- Guidelines for effective program composition
- Useful examples
- Incentives
- Feedback

Correctness

Composition

Teaching Composition Quality

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

- Guidelines for effective program composition
- Useful examples
- Incentives
- Feedback

	Who?
Correctness	Computer
Composition	Humans

Teaching Composition Quality

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

- Guidelines for effective program composition
- Useful examples
- Incentives
- Feedback

	Who?	**How?**
Correctness	Computer	Script
Composition	Humans	Code Reviews

Teaching Composition Quality

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

- Guidelines for effective program composition
- Useful examples
- Incentives
- Feedback

	Who?	**How?**	**Why?**
Correctness	Computer	Script	78/90
Composition	Humans	Code Reviews	12/90

Teaching Composition Quality

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

- Guidelines for effective program composition
- Useful examples
- Incentives
- Feedback

	Who?	**How?**	**Why?**	**When?**
Correctness	Computer	Script	78/90	Instant
Composition	Humans	Code Reviews	12/90	~2 weeks

Teaching Composition Quality

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

- Guidelines for effective program composition
- Useful examples

1

- Incentives
- Feedback

	Who?	**How?**	**Why?**	**When?**
Correctness	Computer	Script	78/90	Instant
Composition	Humans	Code Reviews	12/90	~2 weeks

Teaching Composition Quality

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

- Guidelines for effective program composition
- Useful examples

1

- Incentives
- Feedback

2

	Who?	How?	Why?	When?
Correctness	Computer	Script	78/90	Instant
Composition	Humans	Code Reviews	12/90	~2 weeks

Guidelines & Useful Examples

Illustrating Composition is Challenging

Illustrating Composition is Challenging

- Demonstrations of programming language features often lack enough context for names to matter.

Illustrating Composition is Challenging

- Demonstrations of programming language features often lack enough context for names to matter.
  - My first non-trivial lecture example:
    `mul(add(2, mul(4, 6)), add(3, 5))`

Illustrating Composition is Challenging

- Demonstrations of programming language features often lack enough context for names to matter.
  - My first non-trivial lecture example: `mul(add(2, mul(4, 6)), add(3, 5))`
  - Placeholders: `foo`, `bar`, `baz`, `my_x`, `helper`, etc.

Illustrating Composition is Challenging

- Demonstrations of programming language features often lack enough context for names to matter.
  - My first non-trivial lecture example: `mul(add(2, mul(4, 6)), add(3, 5))`
  - Placeholders: `foo`, `bar`, `baz`, `my_x`, `helper`, etc.
- Some students claim that a program composed of many small functions is hard for them to follow.

Illustrating Composition is Challenging

- Demonstrations of programming language features often lack enough context for names to matter.
  - My first non-trivial lecture example: `mul(add(2, mul(4, 6)), add(3, 5))`
  - Placeholders: `foo`, `bar`, `baz`, `my_x`, `helper`, etc.
- Some students claim that a program composed of many small functions is hard for them to follow.
  - Tracing execution is easier without abstraction.

Illustrating Composition is Challenging

- Demonstrations of programming language features often lack enough context for names to matter.
  - My first non-trivial lecture example: `mul(add(2, mul(4, 6)), add(3, 5))`
  - Placeholders: `foo`, `bar`, `baz`, `my_x`, `helper`, etc.
- Some students claim that a program composed of many small functions is hard for them to follow.
  - Tracing execution is easier without abstraction.
- Many composition issues only arise in programs of sufficient complexity.

Illustrating Composition is Challenging

- Demonstrations of programming language features often lack enough context for names to matter.
  - My first non-trivial lecture example: `mul(add(2, mul(4, 6)), add(3, 5))`
  - Placeholders: `foo`, `bar`, `baz`, `my_x`, `helper`, etc.
- Some students claim that a program composed of many small functions is hard for them to follow.
  - Tracing execution is easier without abstraction.
- Many composition issues only arise in programs of sufficient complexity.
- *Our solution*: Emphasize composition in projects.

Focus on Human Judgment

Focus on Human Judgment

Composition is about communication among people.

Focus on Human Judgment

Composition is about communication among people.

Code Style

Rules of Thumb

Effective Communication

Focus on Human Judgment

Composition is about communication among people.

Code Style	**Rules of Thumb**	**Effective Communication**

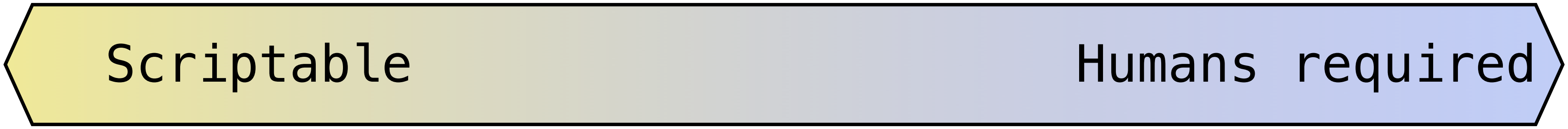
`sum_of(x, y)`
vs
`SumOf(x,y)`
	A function body should have at most k lines.	*Why is each variable name introduced?*

Focus on Human Judgment

Composition is about communication among people.

Code Style	**Rules of Thumb**	**Effective Communication**

sum_of(x, y)
vs
SumOf(x,y)
	A function body should have at most k lines.	*Why is each variable name introduced?*


Scriptable

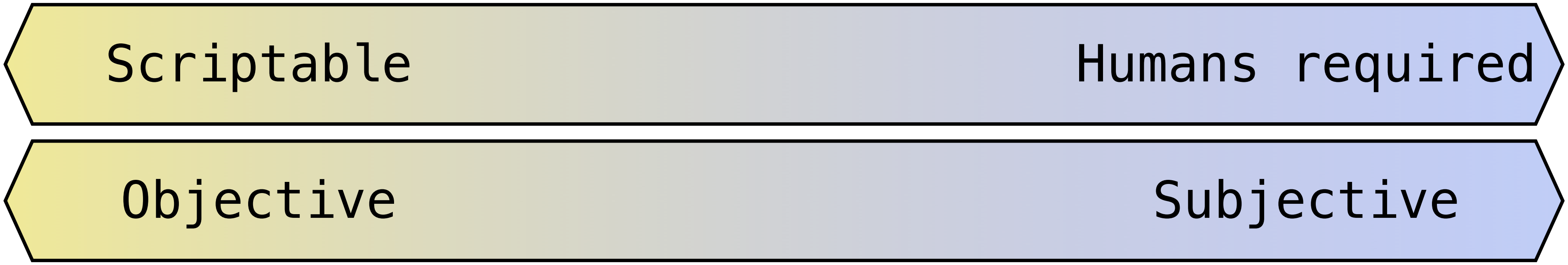
Humans required

Focus on Human Judgment

Composition is about communication among people.

Code Style	**Rules of Thumb**	**Effective Communication**

`sum_of(x, y)`
vs
`SumOf(x,y)`
	A function body should have at most k lines.	*Why is each variable name introduced?*



Scriptable — Humans required

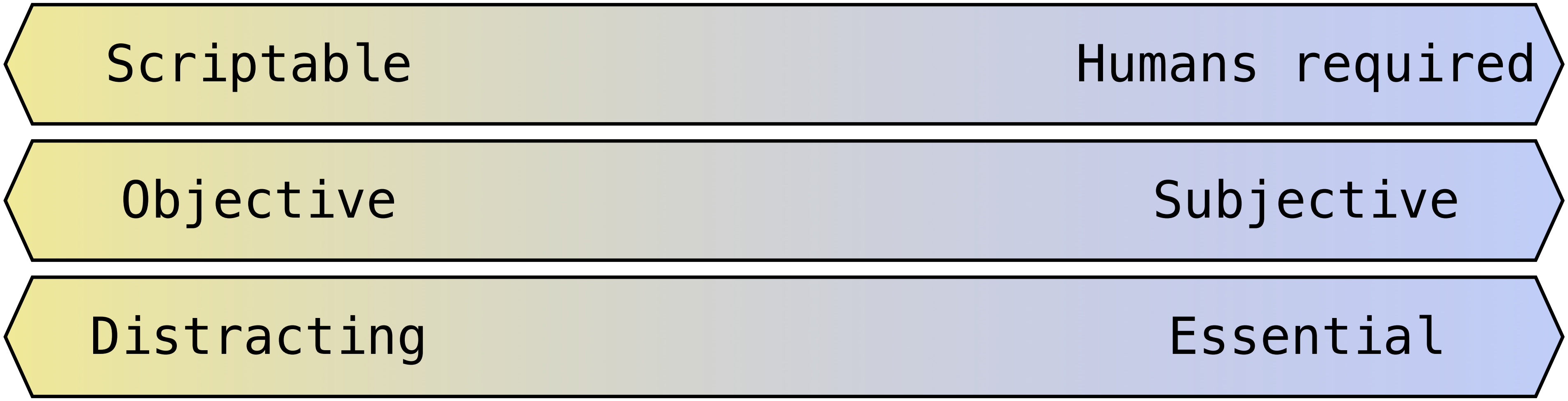
Objective — Subjective

Focus on Human Judgment

Composition is about communication among people.

Code Style	**Rules of Thumb**	**Effective Communication**

`sum_of(x, y)`
vs
`SumOf(x,y)`
	A function body should have at most k lines.	*Why is each variable name introduced?*



Composition Guidelines Given to Students

Composition Guidelines Given to Students

Names Choose names that evoke the purpose or meaning of the values to which they refer.

Composition Guidelines Given to Students

Names Choose names that evoke the purpose or meaning of the values to which they refer.

Functions Define functions with a clear role in the program that abstract the implementation of a behavior.

Composition Guidelines Given to Students

Names Choose names that evoke the purpose or meaning of the values to which they refer.

Functions Define functions with a clear role in the program that abstract the implementation of a behavior.

Purpose Remove all unused and unnecessary code and comments, leaving behind only code with a purpose.

Composition Guidelines Given to Students

Names Choose names that evoke the purpose or meaning of the values to which they refer.

Functions Define functions with a clear role in the program that abstract the implementation of a behavior.

Purpose Remove all unused and unnecessary code and comments, leaving behind only code with a purpose.

Brevity Prefer concise implementations to convoluted ones, without explicitly minimizing the line count.

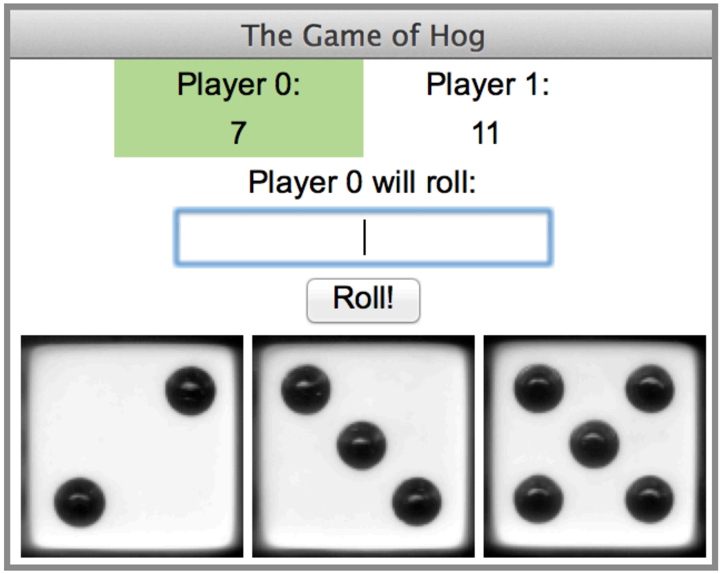
Scaffolded Programming Projects

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

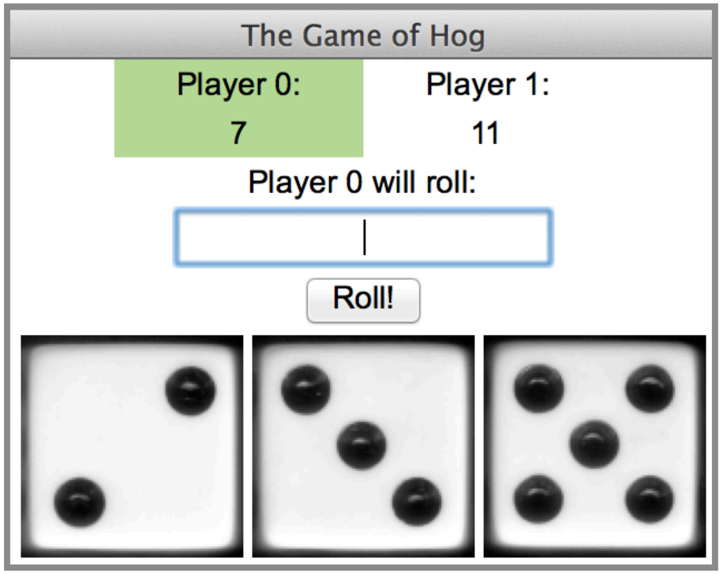


The "Hog" project is based on Todd Neller's "Pig" project, Nifty Assignments 2010.

Scaffolded Programming Projects

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

Composition can be taught by example in context.



The Game of Hog

Player 0: 7

Player 1: 11

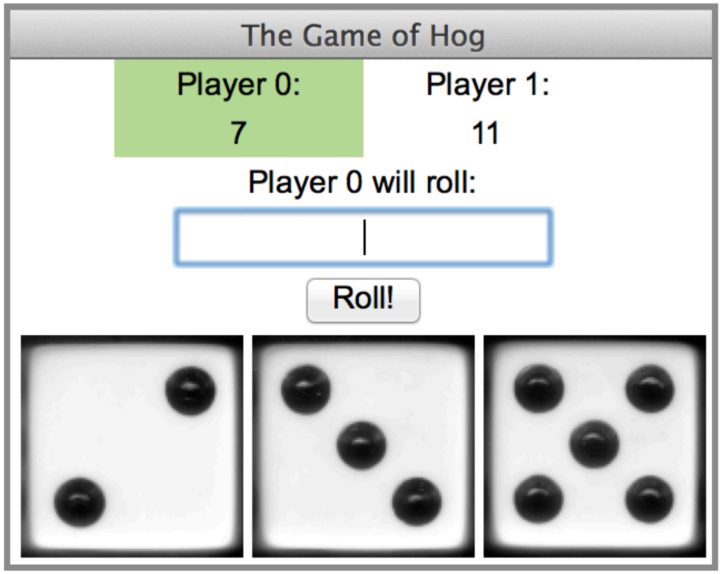
Player 0 will roll:

The "Hog" project is based on Todd Neller's "Pig" project, Nifty Assignments 2010.

Scaffolded Programming Projects

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

Composition can be taught by example in context.



The Game of Hog

Player 0: 7

Player 1: 11

Player 0 will roll:

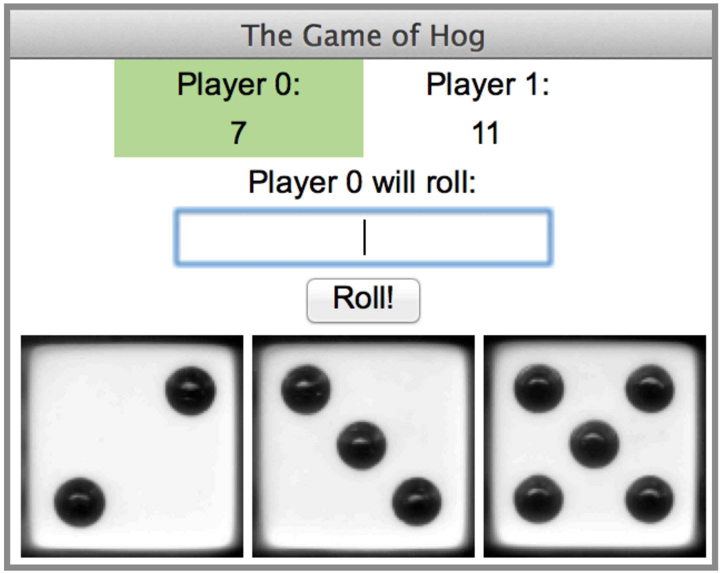
```
def make_averaged(fn, num_samples=1000):
    """Return a function that returns the average
    return value of FN called NUM_SAMPLES times.

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

The "Hog" project is based on Todd Neller's "Pig" project, Nifty Assignments 2010.

Scaffolded Programming Projects

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

Composition can be taught by example in context.



```
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 ***"
```

Shameless advertisement: All four course projects are freely available with comprehensive autograders.

(See one in the Nifty Assignments session tomorrow.)

The "Hog" project is based on Todd Neller's "Pig" project, Nifty Assignments 2010.

Online Composition Feedback System

Previous Work

Previous Work

- Assignments are managed and graded with an online tool. (Jones & Jamieson, 1997; Jackson & Usher, 1997)

Previous Work

- Assignments are managed and graded with an online tool. (Jones & Jamieson, 1997; Jackson & Usher, 1997)
- Summary statements and comments on specific lines. (Heaney & Daly, 2004)

Previous Work

- Assignments are managed and graded with an online tool. (Jones & Jamieson, 1997; Jackson & Usher, 1997)
- Summary statements and comments on specific lines. (Heaney & Daly, 2004)
- Text comments inlined with the submission source code. (MacWilliam & Malan, 2013)

Previous Work

```
def find_centroid(polygon):
    def get_x(p):
        return latitude(p)
    def get_y(p):
        return longitude(p)
```

- Assignments are managed and graded with an online tool. (Jones & Jamieson, 1997; Jackson & Usher, 1997)
- Summary statements and comments on specific lines. (Heaney & Daly, 2004)
- Text comments inlined with the submission source code. (MacWilliam & Malan, 2013)

```
    fsum = 0;
    def f( p1,p2):
        return get_x(p1)∗get_y(p2)−get_x(p2)∗get_y(p1)
    area = ...
    if(area == 0):
        return (latitude(polygon[0]),longitude(polygon[0]),0)
```

Previous Work

- Assignments are managed and graded with an online tool. (Jones & Jamieson, 1997; Jackson & Usher, 1997)
- Summary statements and comments on specific lines. (Heaney & Daly, 2004)
- Text comments inlined with the submission source code. (MacWilliam & Malan, 2013)

```
def find_centroid(polygon):
    def get_x(p):
        return latitude(p)
    def get_y(p):
        return longitude(p)
```

To give a new name to a function, you can use assignment:
get_y = longitude

```
    fsum = 0;
    def f( p1,p2):
        return get_x(p1)*get_y(p2)-get_x(p2)*get_y(p1)
    area = ...
    if(area == 0):
        return (latitude(polygon[0]),longitude(polygon[0]),0)
```

Previous Work

- Assignments are managed and graded with an online tool. (Jones & Jamieson, 1997; Jackson & Usher, 1997)
- Summary statements and comments on specific lines. (Heaney & Daly, 2004)
- Text comments inlined with the submission source code. (MacWilliam & Malan, 2013)

```
def find_centroid(polygon):
    def get_x(p):
        return latitude(p)
    def get_y(p):
        return longitude(p)
To give a new name to a function,
you can use assignment:
  get_y = longitude
    fsum = 0;
    def f( p1,p2):
        return get_x(p1)*get_y(p2)-get_x(p2)*get_y(p1)
    area = ...
    if(area == 0):
        return (latitude(polygon[0]),longitude(polygon[0]),0)
polygon[0] is repeated; perhaps give it the name "first" instead.
```

Previous Work

Nice work. Keep looking for more ways to eliminate repeated expressions by introducing names.

```
def find_centroid(polygon):
    def get_x(p):
        return latitude(p)
    def get_y(p):
        return longitude(p)
```

To give a new name to a function, you can use assignment:
get_y = longitude

```
    fsum = 0;
    def f( p1,p2):
        return get_x(p1)*get_y(p2)-get_x(p2)*get_y(p1)
    area = ...
    if(area == 0):
        return (latitude(polygon[0]),longitude(polygon[0]),0)
```

polygon[0] is repeated; perhaps give it the name "first" instead.

- Assignments are managed and graded with an online tool. (Jones & Jamieson, 1997; Jackson & Usher, 1997)
- Summary statements and comments on specific lines. (Heaney & Daly, 2004)
- Text comments inlined with the submission source code. (MacWilliam & Malan, 2013)

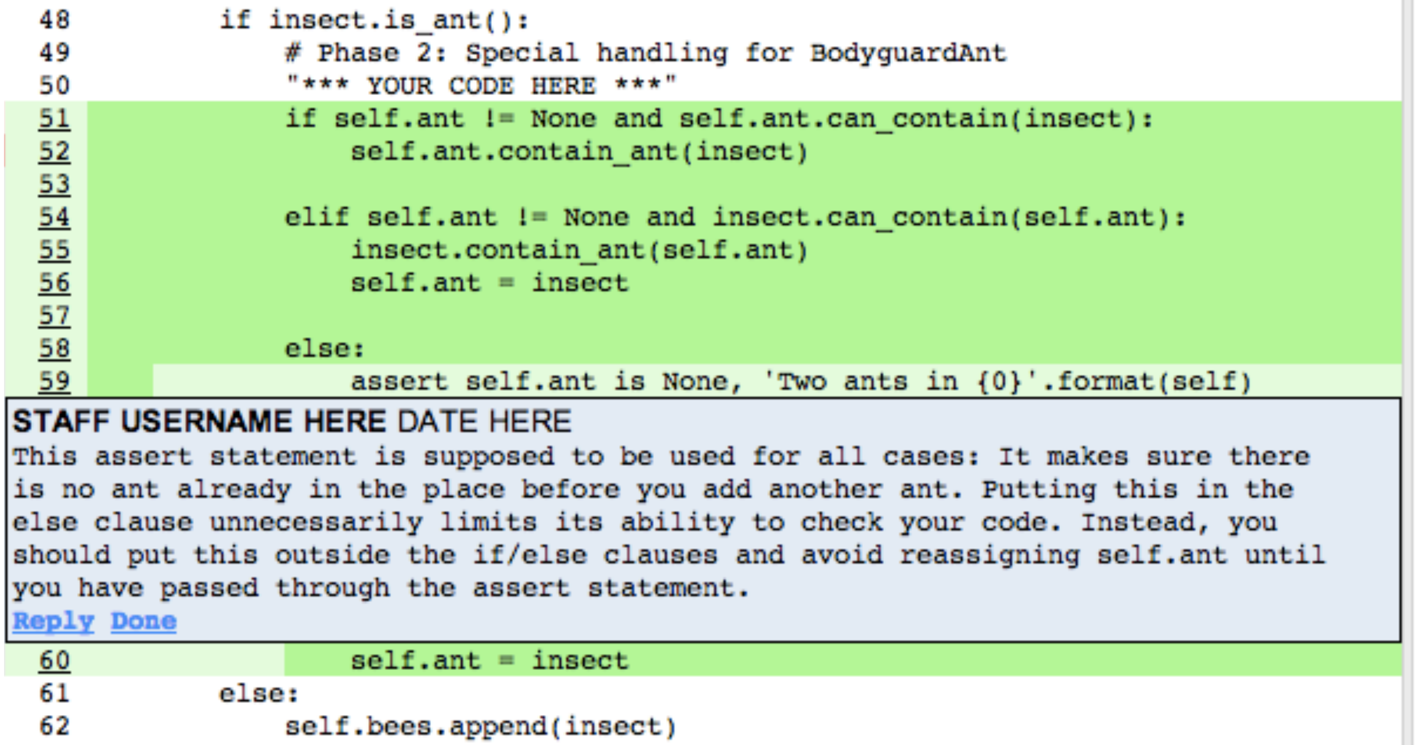
What's New About Our Code Review System?

What's New About Our Code Review System?

- Interface is adopted from the open source branch of Google's internal code review tool, Rietveld.

What's New About Our Code Review System?

- Interface is adopted from the open source branch of Google's internal code review tool, Rietveld.



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

            elif self.ant != None and insect.can_contain(self.ant):
                insect.contain_ant(self.ant)
                self.ant = insect

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

STAFF USERNAME HERE DATE HERE

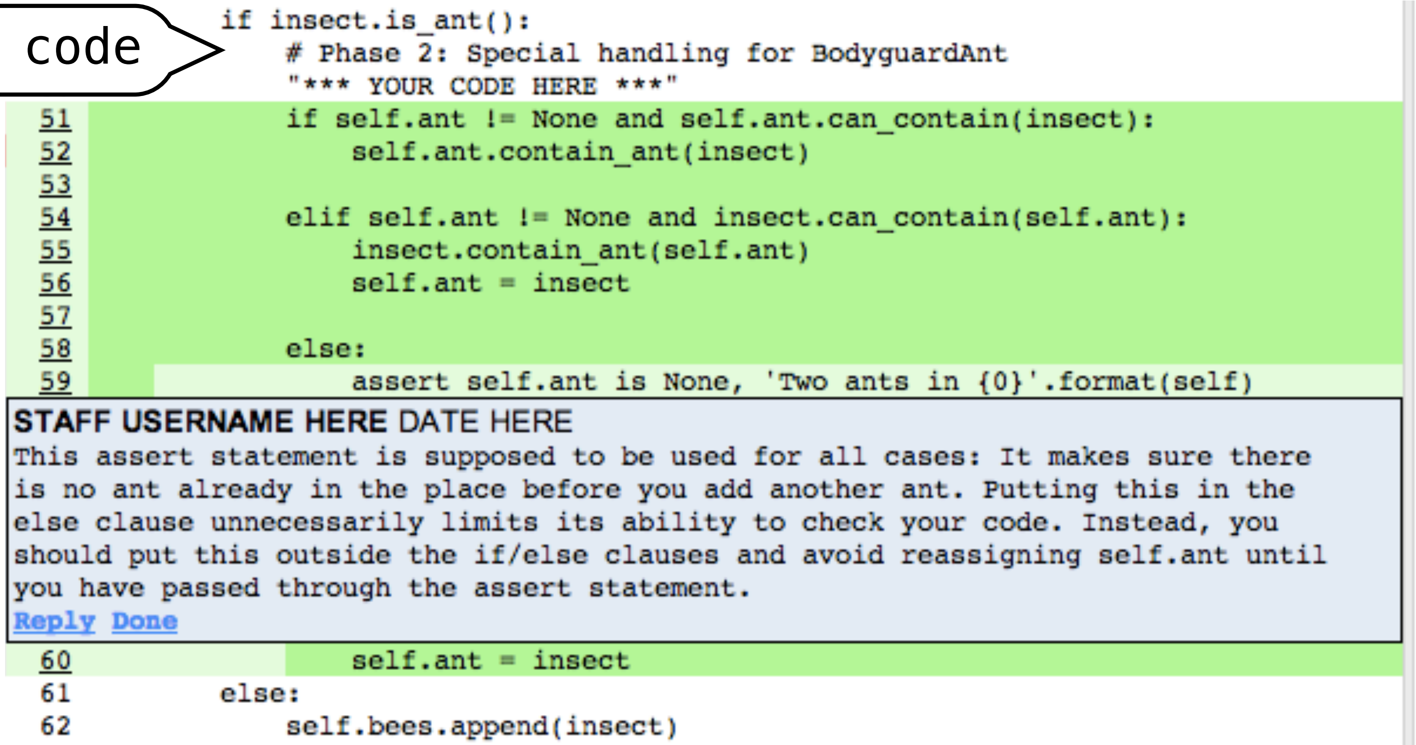
This assert statement is supposed to be used for all cases: It makes sure there is no ant already in the place before you add another ant. Putting this in the else clause unnecessarily limits its ability to check your code. Instead, you should put this outside the if/else clauses and avoid reassigning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

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

What's New About Our Code Review System?

- Interface is adopted from the open source branch of Google's internal code review tool, Rietveld.

Starter code



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

    elif self.ant != None and insect.can_contain(self.ant):
        insect.contain_ant(self.ant)
        self.ant = insect

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

STAFF USERNAME HERE DATE HERE

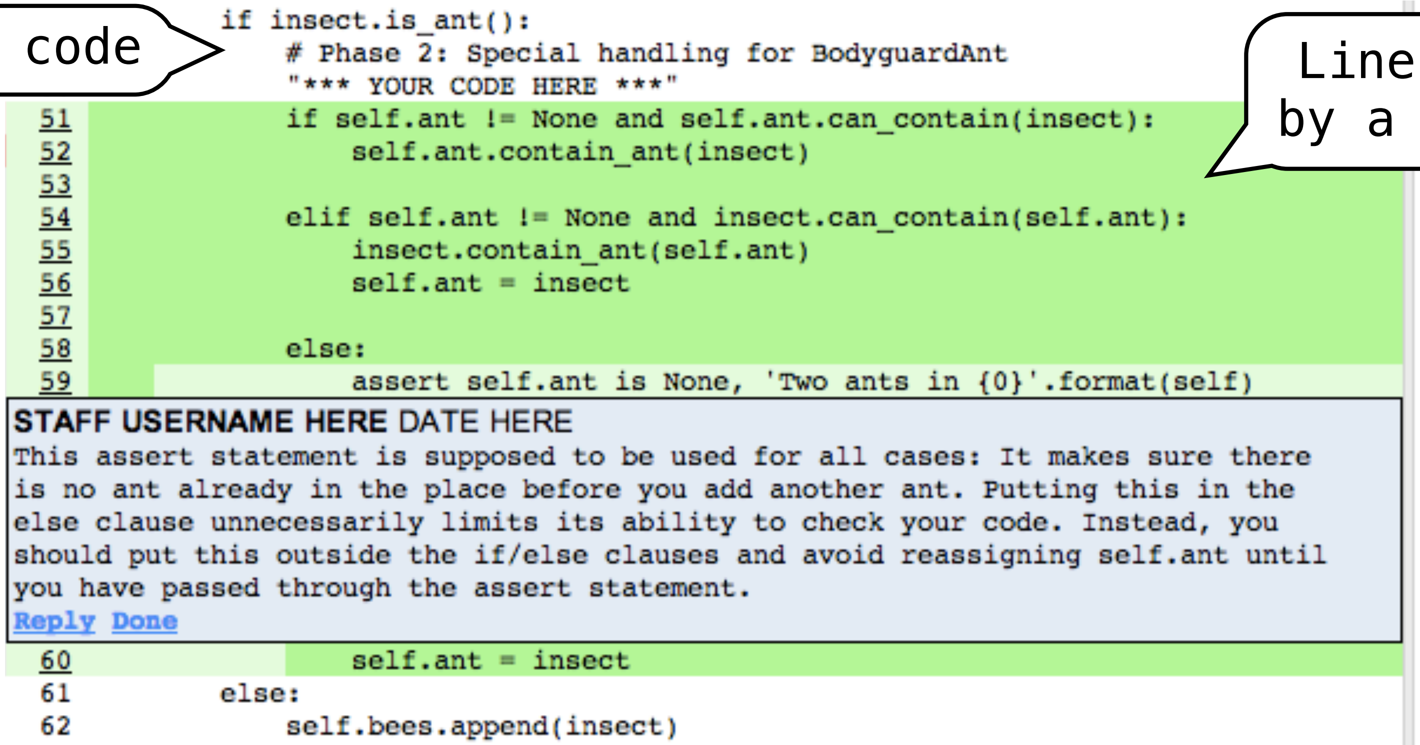
This assert statement is supposed to be used for all cases: It makes sure there is no ant already in the place before you add another ant. Putting this in the else clause unnecessarily limits its ability to check your code. Instead, you should put this outside the if/else clauses and avoid reassigning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

Reply Done

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

What's New About Our Code Review System?

- Interface is adopted from the open source branch of Google's internal code review tool, Rietveld.



Starter code

Lines added by a student

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

    elif self.ant != None and insect.can_contain(self.ant):
        insect.contain_ant(self.ant)
        self.ant = insect

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

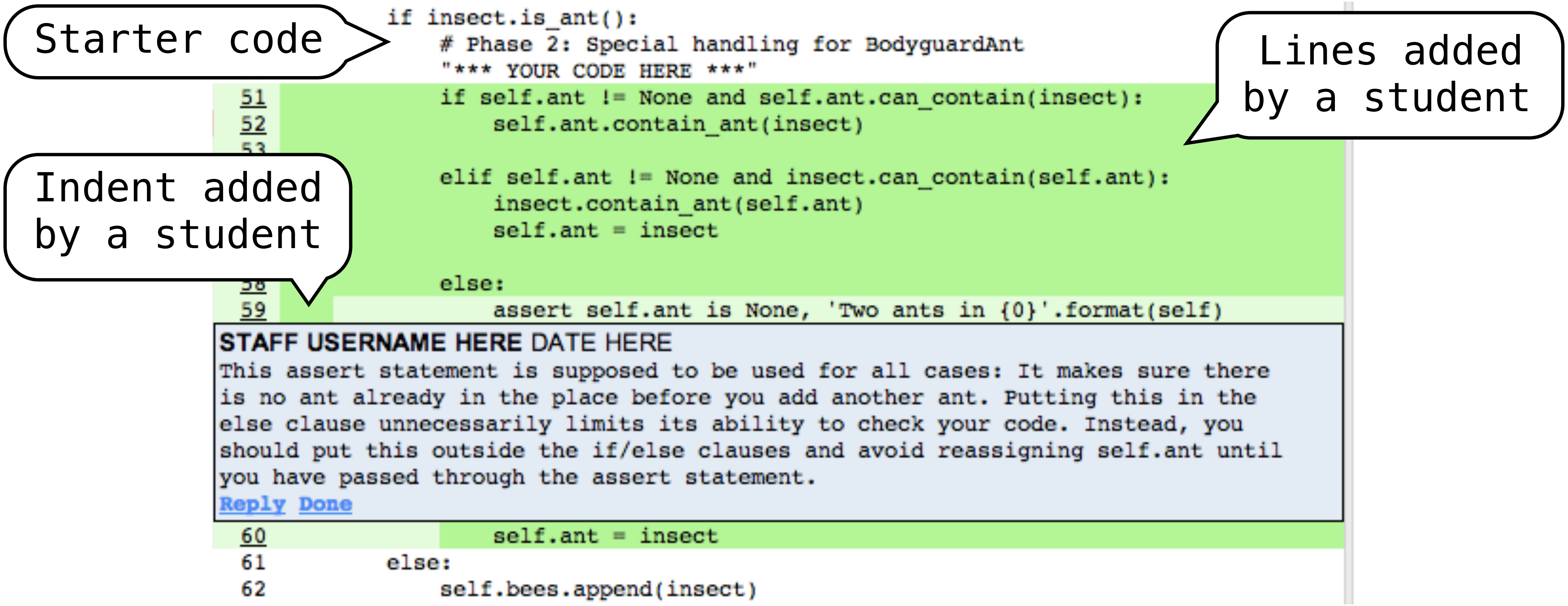
STAFF USERNAME HERE DATE HERE

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.

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

What's New About Our Code Review System?

- Interface is adopted from the open source branch of Google's internal code review tool, Rietveld.



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 ***"
    if self.ant != None and self.ant.can_contain(insect):
        self.ant.contain_ant(insect)

    elif self.ant != None and insect.can_contain(self.ant):
        insect.contain_ant(self.ant)
        self.ant = insect

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

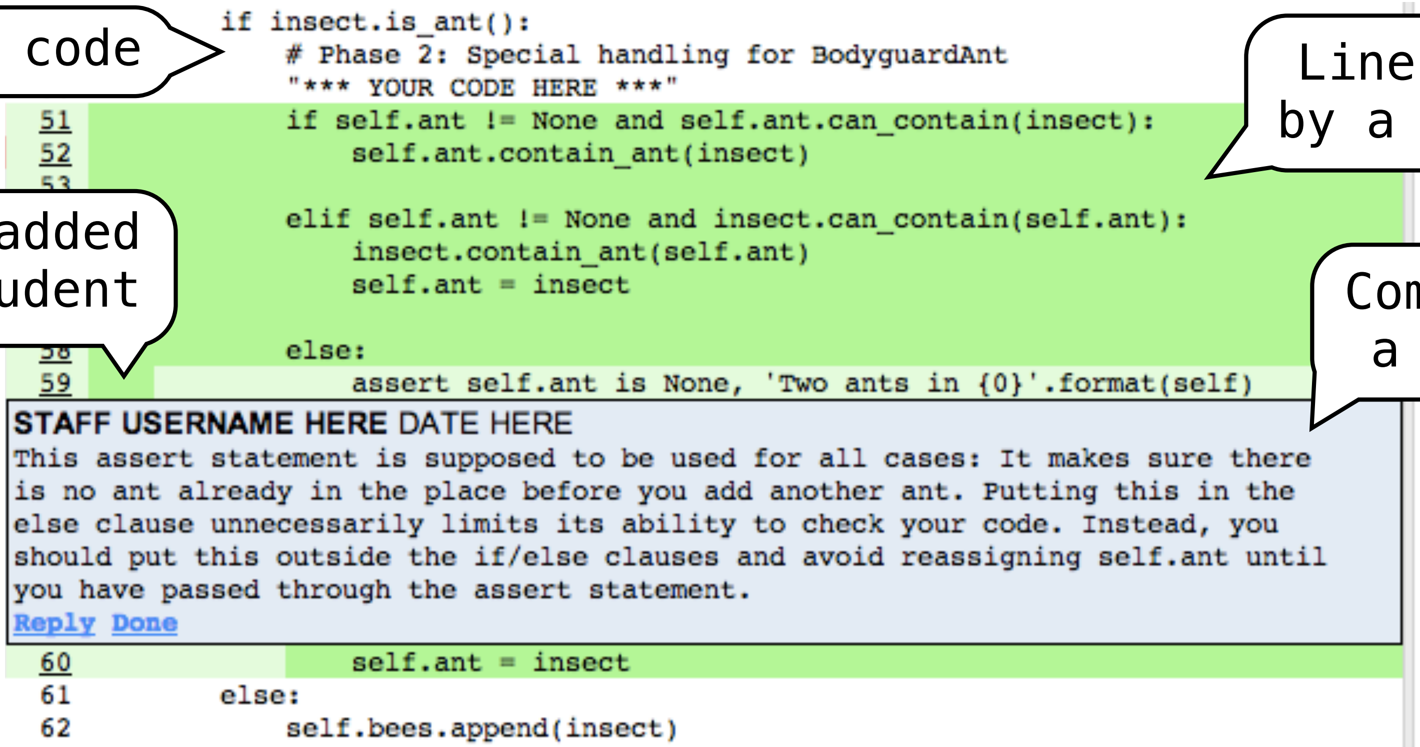
STAFF USERNAME HERE DATE HERE

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.

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

What's New About Our Code Review System?

- Interface is adopted from the open source branch of Google's internal code review tool, Rietveld.



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 ***"
    if self.ant != None and self.ant.can_contain(insect):
        self.ant.contain_ant(insect)

    elif self.ant != None and insect.can_contain(self.ant):
        insect.contain_ant(self.ant)
        self.ant = insect

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

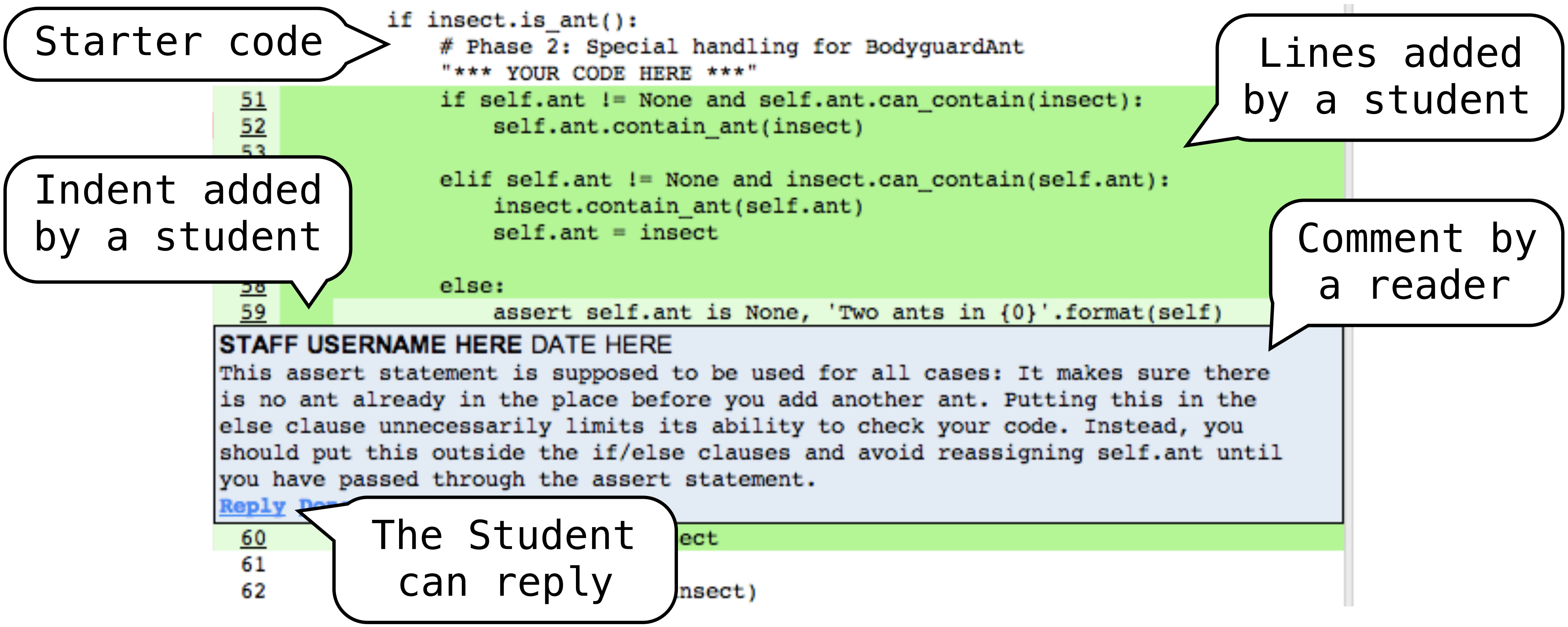
STAFF USERNAME HERE DATE HERE

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.

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

What's New About Our Code Review System?

- Interface is adopted from the open source branch of Google's internal code review tool, Rietveld.



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 ***"
    if self.ant != None and self.ant.can_contain(insect):
        self.ant.contain_ant(insect)

    elif self.ant != None and insect.can_contain(self.ant):
        insect.contain_ant(self.ant)
        self.ant = insect

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

STAFF USERNAME HERE DATE HERE

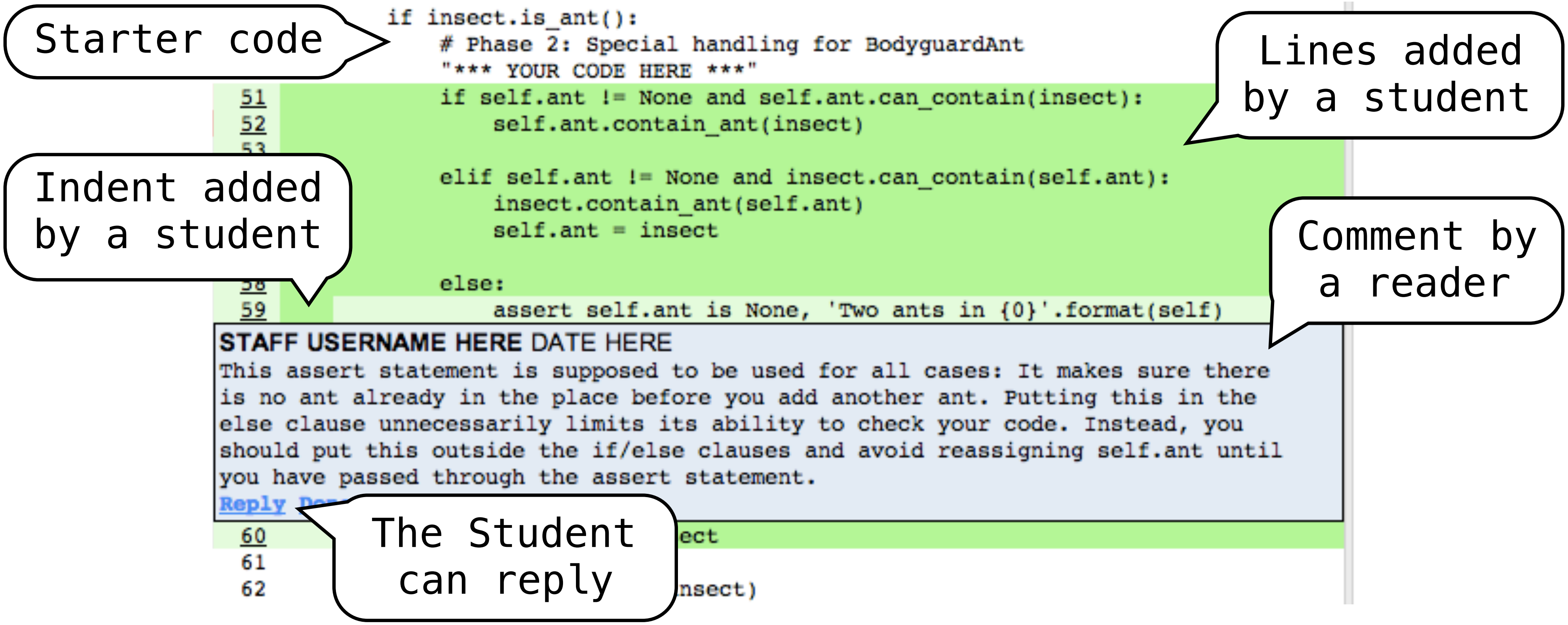
This assert statement is supposed to be used for all cases: It makes sure there is no ant already in the place before you add another ant. Putting this in the else clause unnecessarily limits its ability to check your code. Instead, you should put this outside the if/else clauses and avoid reassigning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

Reply

```
ect

nsect)
```

What's New About Our Code Review System?

- Interface is adopted from the open source branch of Google's internal code review tool, Rietveld.



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 ***"
    if self.ant != None and self.ant.can_contain(insect):
        self.ant.contain_ant(insect)

    elif self.ant != None and insect.can_contain(self.ant):
        insect.contain_ant(self.ant)
        self.ant = insect

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

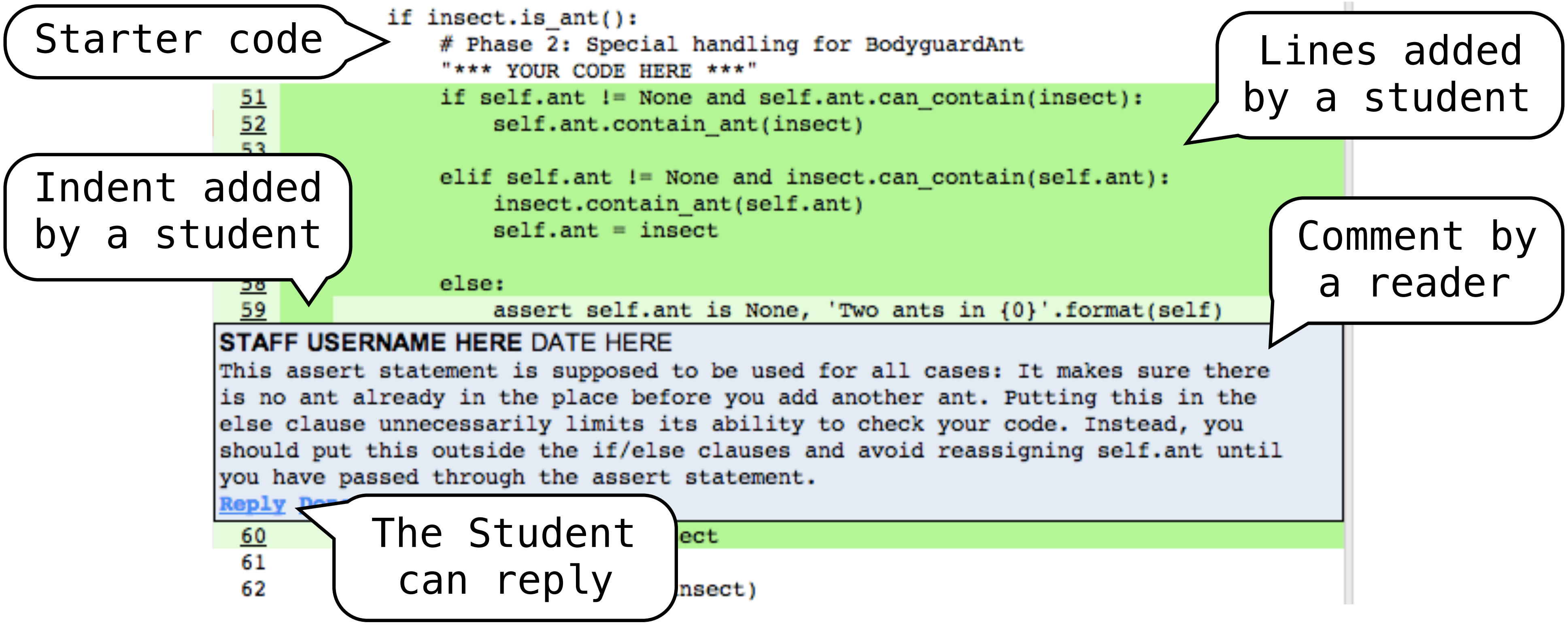
STAFF USERNAME HERE DATE HERE
This assert statement is supposed to be used for all cases: It makes sure there is no ant already in the place before you add another ant. Putting this in the else clause unnecessarily limits its ability to check your code. Instead, you should put this outside the if/else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

```
ect

nsect)
```

- Shared cache of comments among all graders.

What's New About Our Code Review System?

- Interface is adopted from the open source branch of Google's internal code review tool, Rietveld.



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 ***"
    if self.ant != None and self.ant.can_contain(insect):
        self.ant.contain_ant(insect)

    elif self.ant != None and insect.can_contain(self.ant):
        insect.contain_ant(self.ant)
        self.ant = insect

    else:
        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.

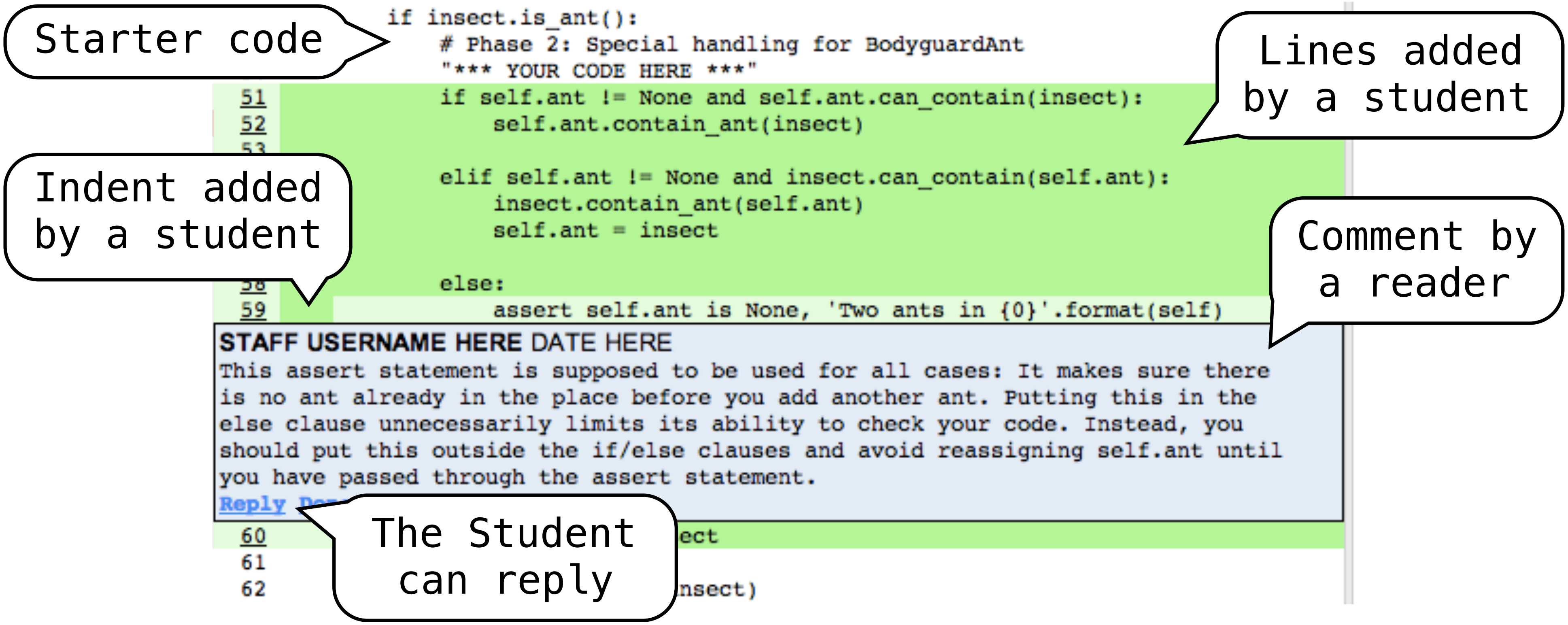
Reply

```
                                  ect

                                  nsect)
```

- Shared cache of comments among all graders.
- Integration with existing version control systems.

What's New About Our Code Review System?

- Interface is adopted from the open source branch of Google's internal code review tool, Rietveld.



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

    elif self.ant != None and insect.can_contain(self.ant):
        insect.contain_ant(self.ant)
        self.ant = insect

    else:
        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.

```
ect

nsect)
```

- Shared cache of comments among all graders.
- Integration with existing version control systems.
- Easy administration through Google App Engine.

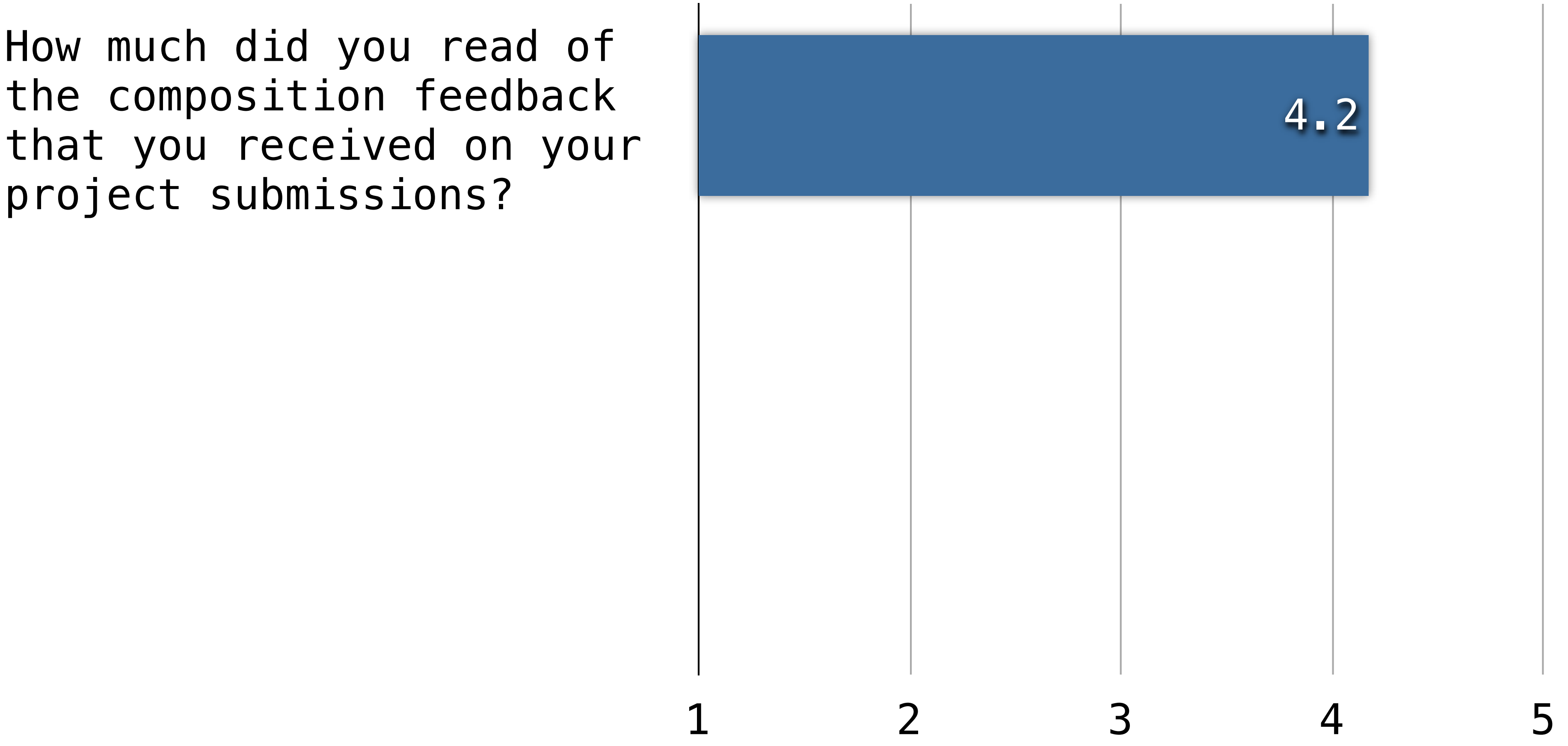
Evaluations

End-of-Semester Survey Results

Fall 2012 Average of 626 Responses
(All questions rated on a scale from 1 to 5)

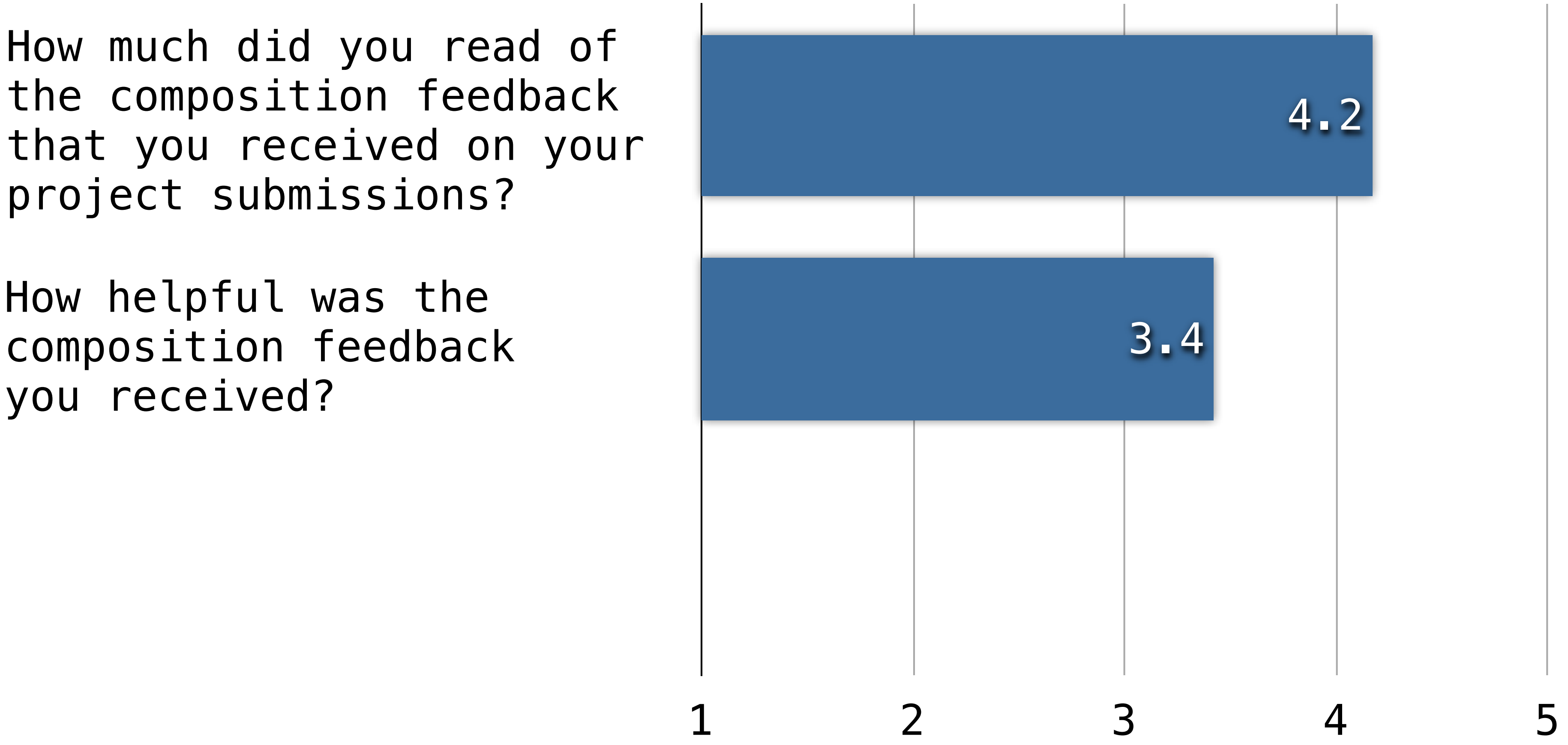
End-of-Semester Survey Results

Fall 2012 Average of 626 Responses
(All questions rated on a scale from 1 to 5)



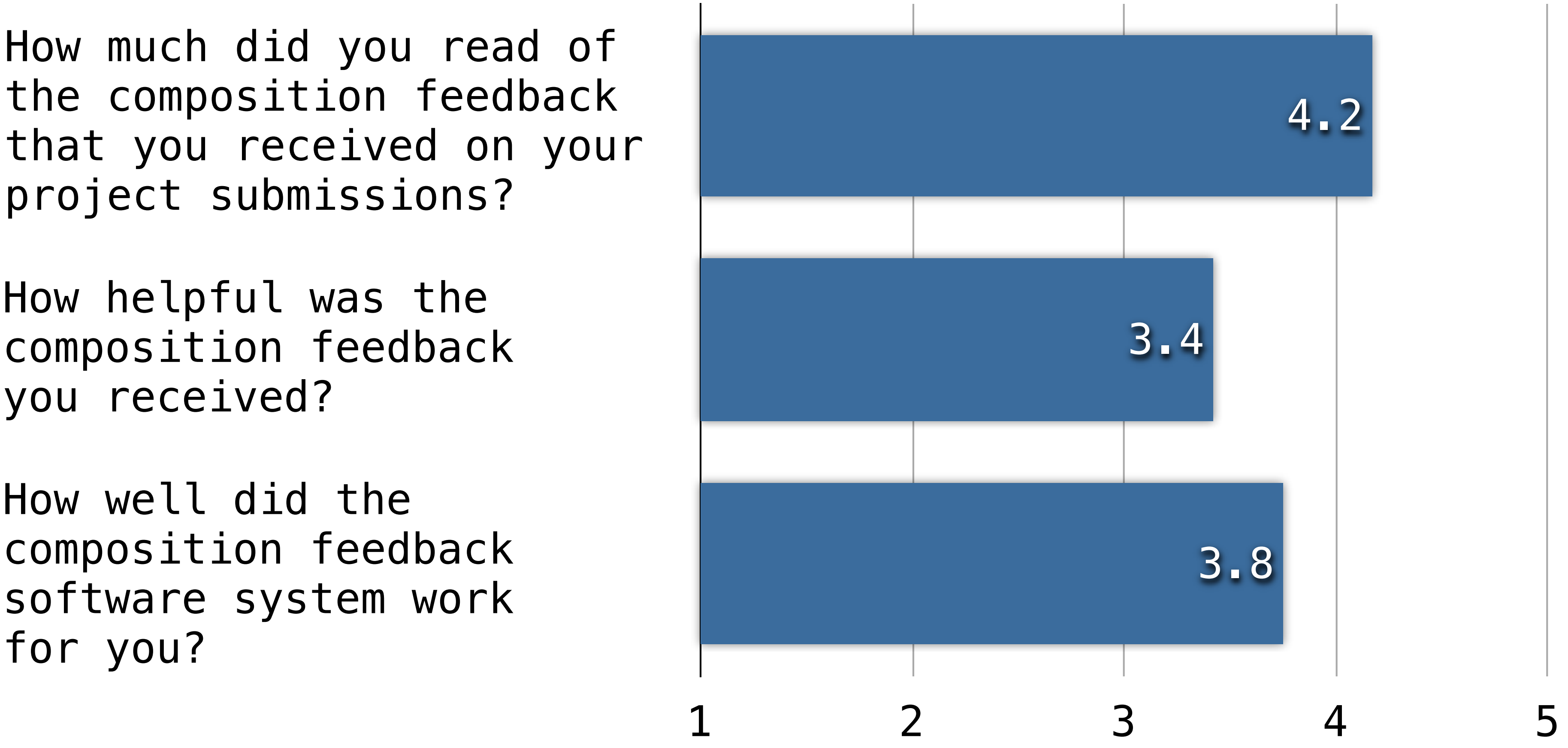
End-of-Semester Survey Results

Fall 2012 Average of 626 Responses
(All questions rated on a scale from 1 to 5)



End-of-Semester Survey Results

Fall 2012 Average of 626 Responses
(All questions rated on a scale from 1 to 5)



Blind Comparison of Student Submissions

Blind Comparison of Student Submissions

Experiment Design

Blind Comparison of Student Submissions

Experiment Design

- 16 student submissions were sampled without replacement from Fall 2011 (before composition focus) and Fall 2012 (after).

Blind Comparison of Student Submissions

Experiment Design

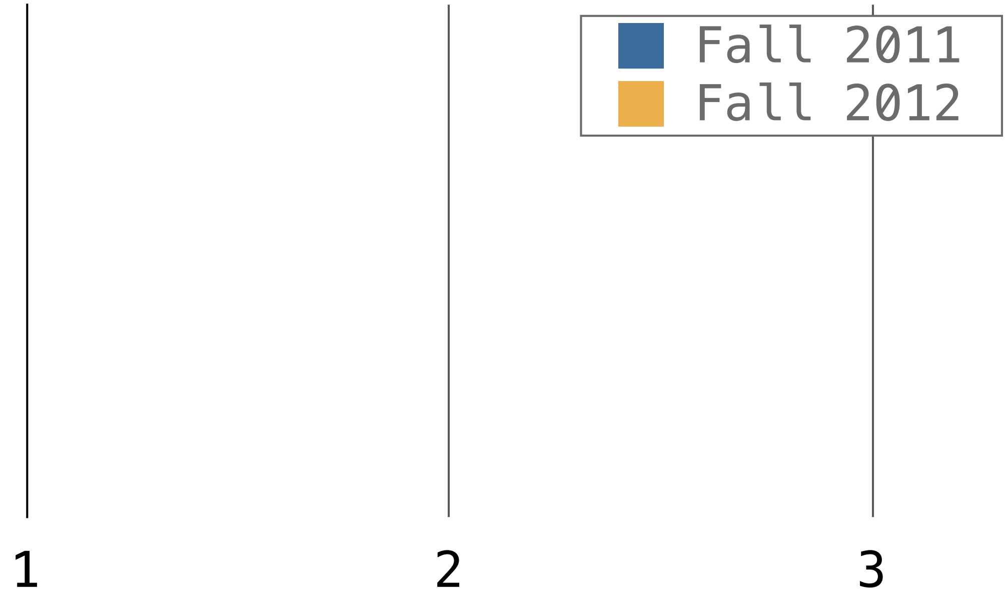
- 16 student submissions were sampled without replacement from Fall 2011 (before composition focus) and Fall 2012 (after).
- 6 course graders (experts) judged implementations of a single function, find_centroid, that didn't change significantly.

Blind Comparison of Student Submissions

Experiment Design

- 16 student submissions were sampled without replacement from Fall 2011 (before composition focus) and Fall 2012 (after).
- 6 course graders (experts) judged implementations of a single function, find_centroid, that didn't change significantly.

Average Responses by Graders (Scale from 1 to 3)



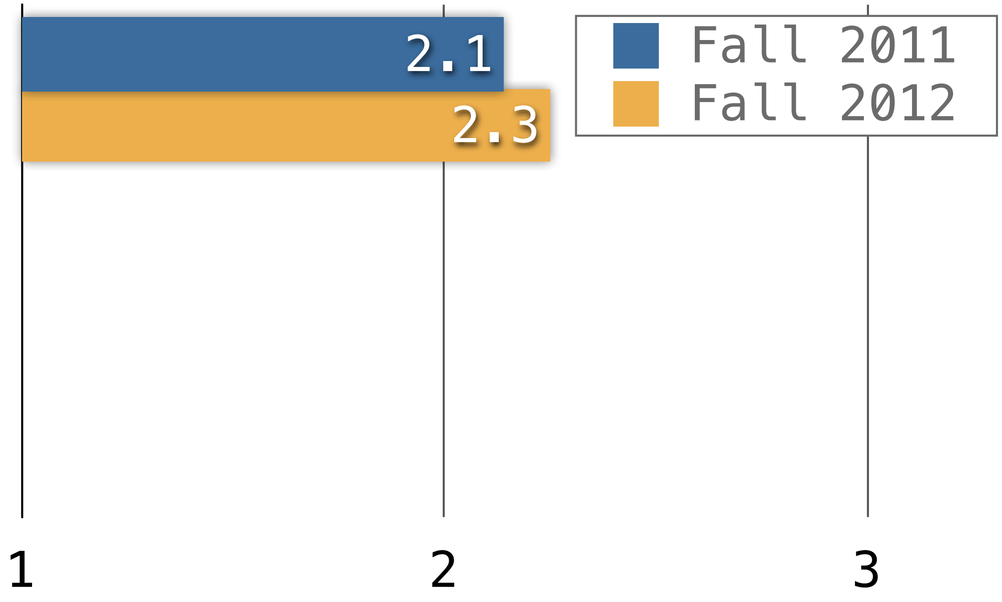
Blind Comparison of Student Submissions

Experiment Design

- 16 student submissions were sampled without replacement from Fall 2011 (before composition focus) and Fall 2012 (after).
- 6 course graders (experts) judged implementations of a single function, find_centroid, that didn't change significantly.

Average Responses by Graders (Scale from 1 to 3)

Do names convey the meaning & purpose of their values?

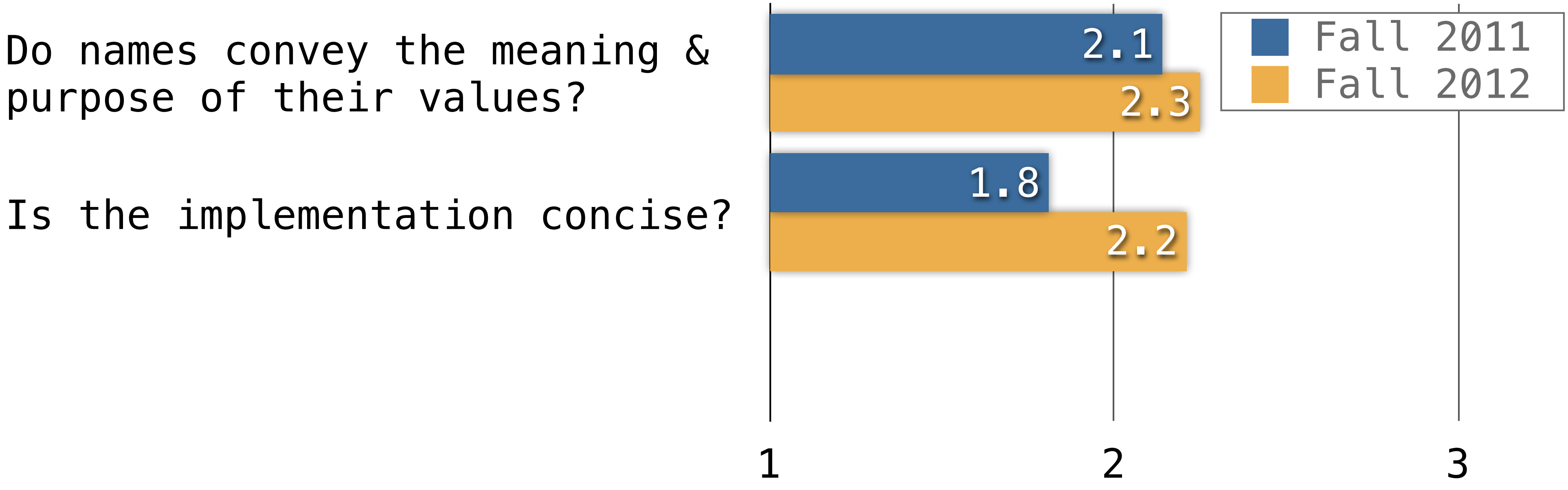


Blind Comparison of Student Submissions

Experiment Design

- 16 student submissions were sampled without replacement from Fall 2011 (before composition focus) and Fall 2012 (after).
- 6 course graders (experts) judged implementations of a single function, find_centroid, that didn't change significantly.

Average Responses by Graders (Scale from 1 to 3)

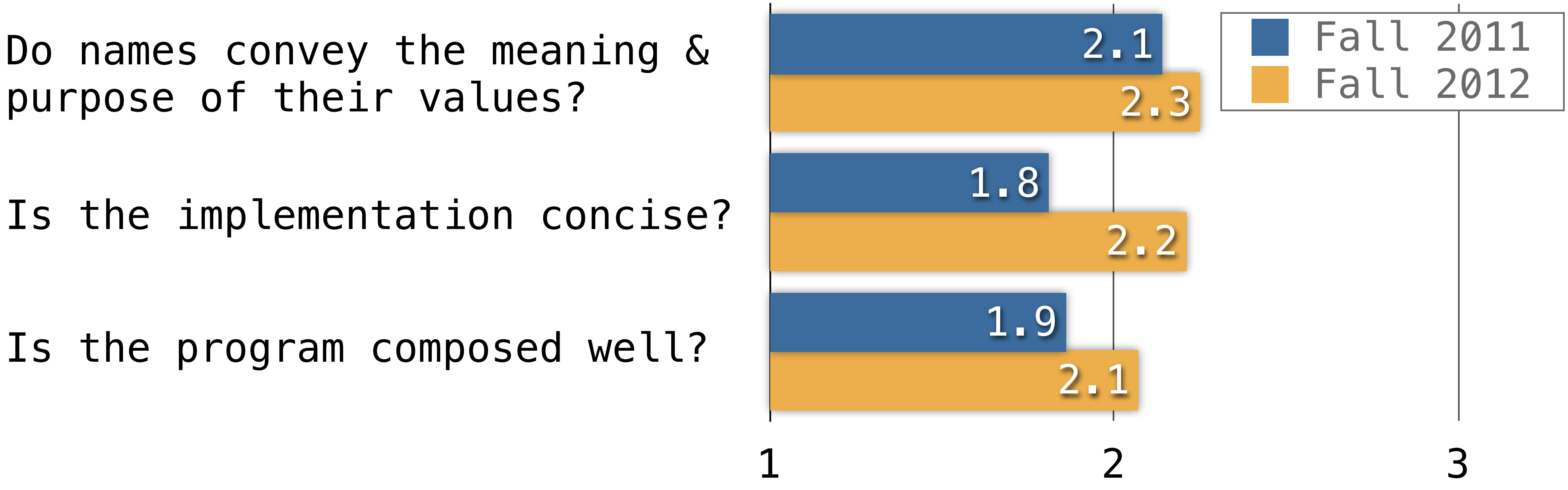


Blind Comparison of Student Submissions

Experiment Design

- 16 student submissions were sampled without replacement from Fall 2011 (before composition focus) and Fall 2012 (after).
- 6 course graders (experts) judged implementations of a single function, find_centroid, that didn't change significantly.

Average Responses by Graders (Scale from 1 to 3)

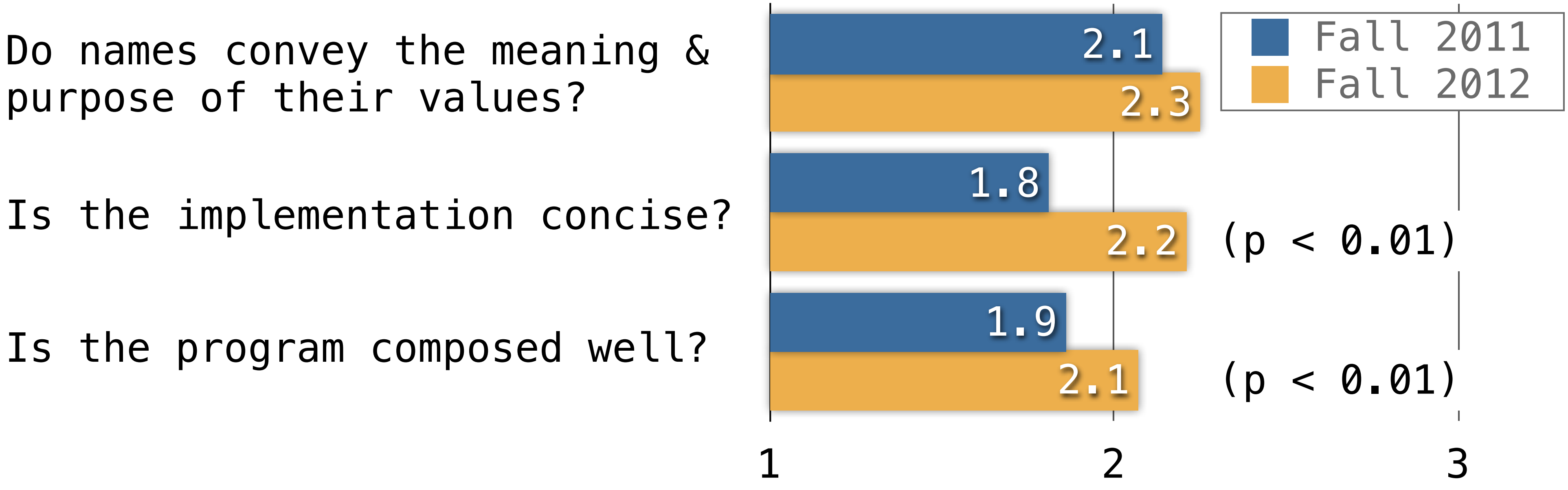


Blind Comparison of Student Submissions

Experiment Design

- 16 student submissions were sampled without replacement from Fall 2011 (before composition focus) and Fall 2012 (after).
- 6 course graders (experts) judged implementations of a single function, find_centroid, that didn't change significantly.

Average Responses by Graders (Scale from 1 to 3)



Another Shameless Advertisement

Composing Programs: An Interactive Textbook

ComposingPrograms.com hosts a free online introduction to programming and computer science.

Composing Programs: An Interactive Textbook

ComposingPrograms.com hosts a free online introduction to programming and computer science.

A product of public domain and open source content:

Composing Programs: An Interactive Textbook

ComposingPrograms.com hosts 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

Composing Programs: An Interactive Textbook

ComposingPrograms.com hosts 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
- Examples diagrammed by the Online Python Tutor

Composing Programs: An Interactive Textbook

ComposingPrograms.com hosts 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
- 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

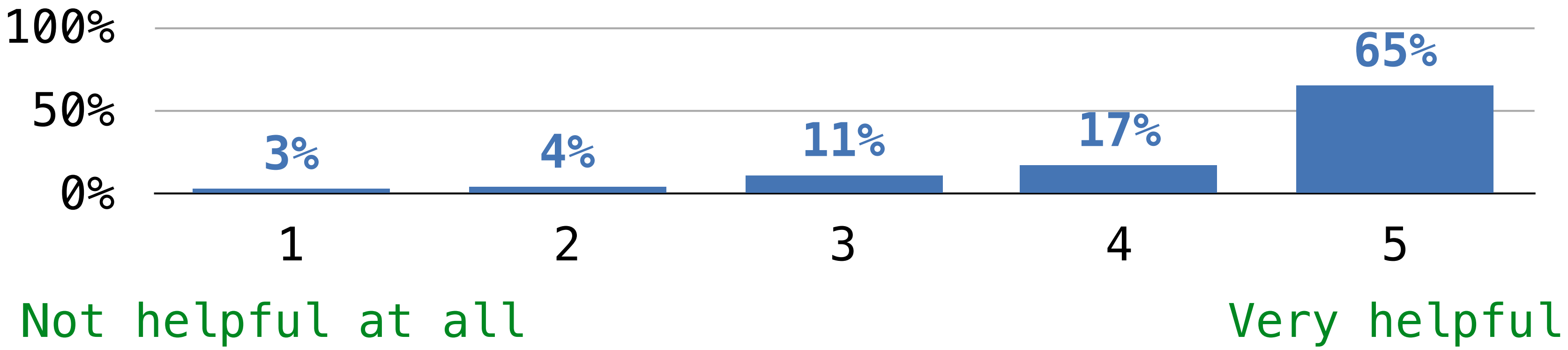
Composing Programs: An Interactive Textbook

ComposingPrograms.com hosts 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
- 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

Thanks!

Source Code: github.com/moowiz/ucb-codereview

Description: composingprograms.com/codereview.html

Interested in the Composing Programs textbook?
Please fill out the survey on composingprograms.com!

Extra

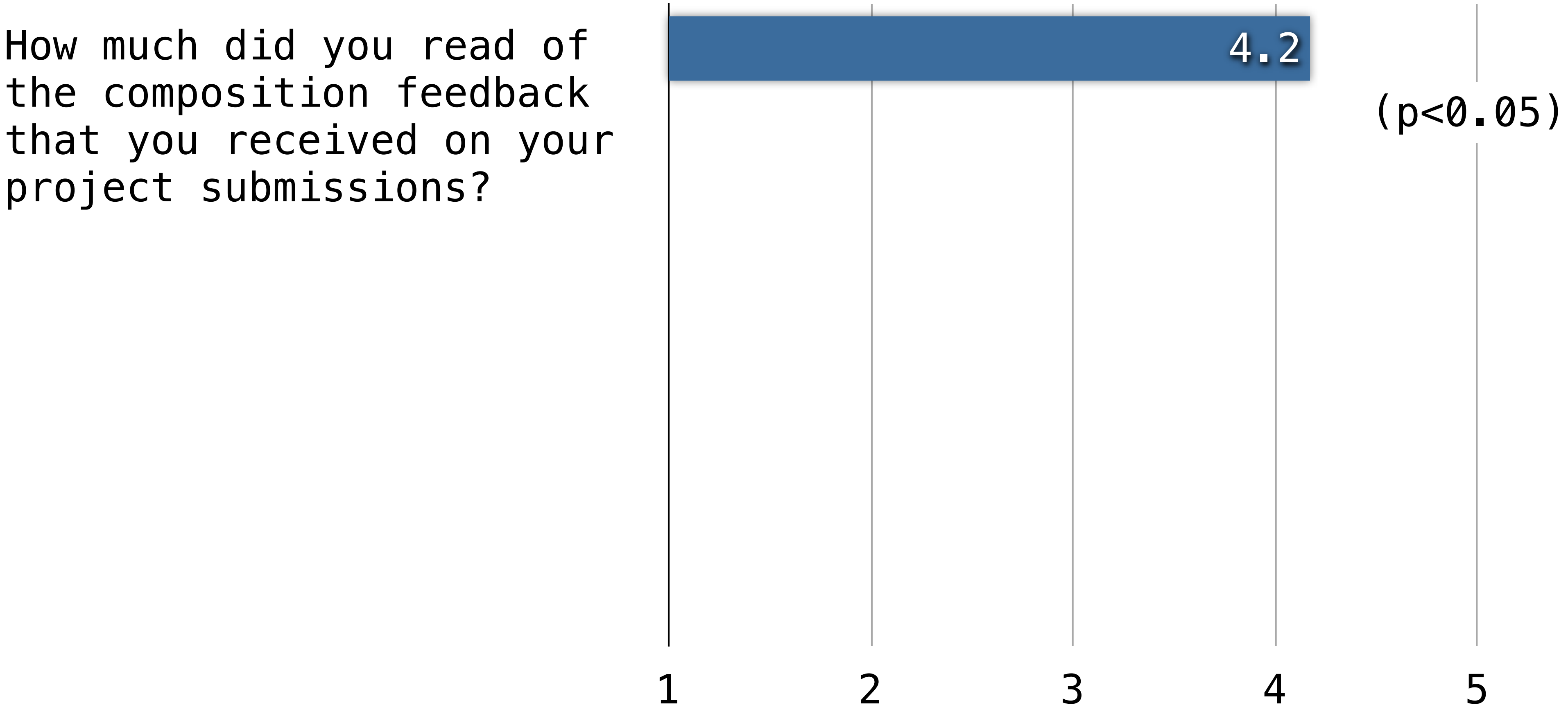
End-of-Semester Survey Results

Fall 2012 Average Responses (Scale from 1 to 5)

($p<0.05$)

End-of-Semester Survey Results

Fall 2012 Average Responses (Scale from 1 to 5)

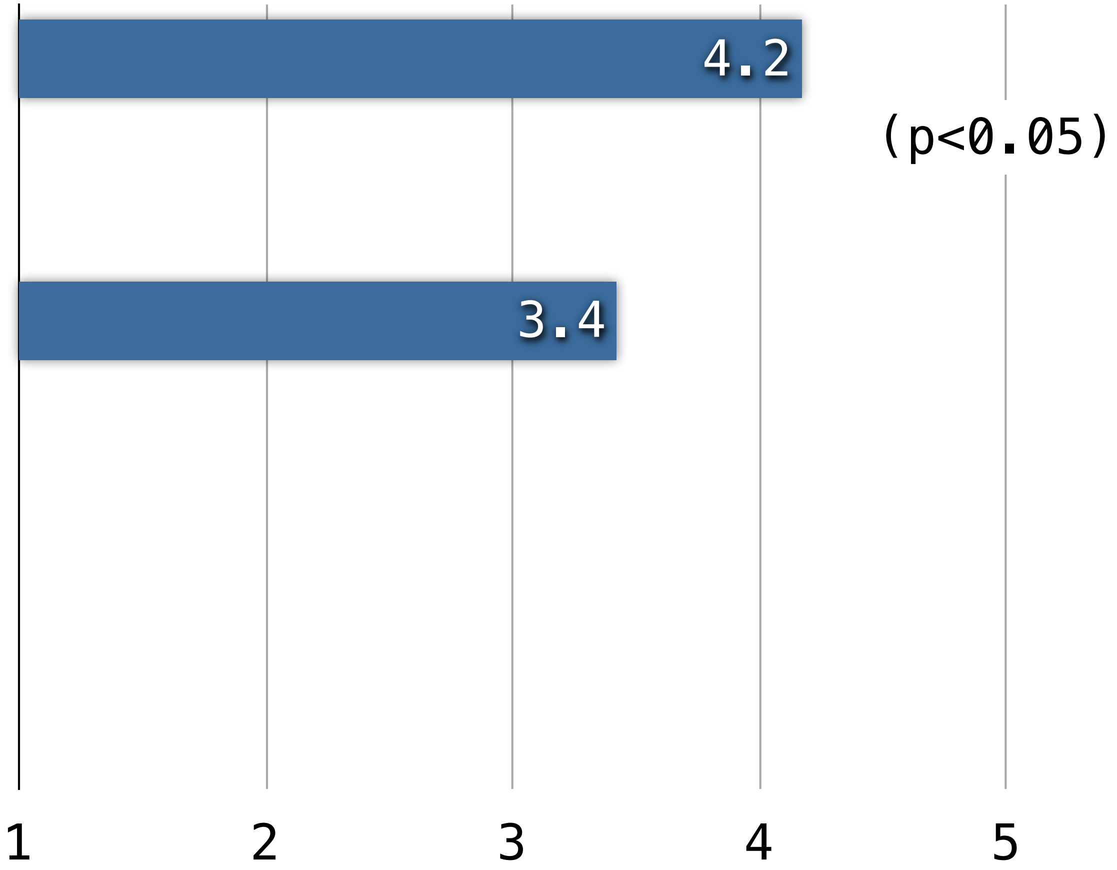


End-of-Semester Survey Results

Fall 2012 Average Responses (Scale from 1 to 5)

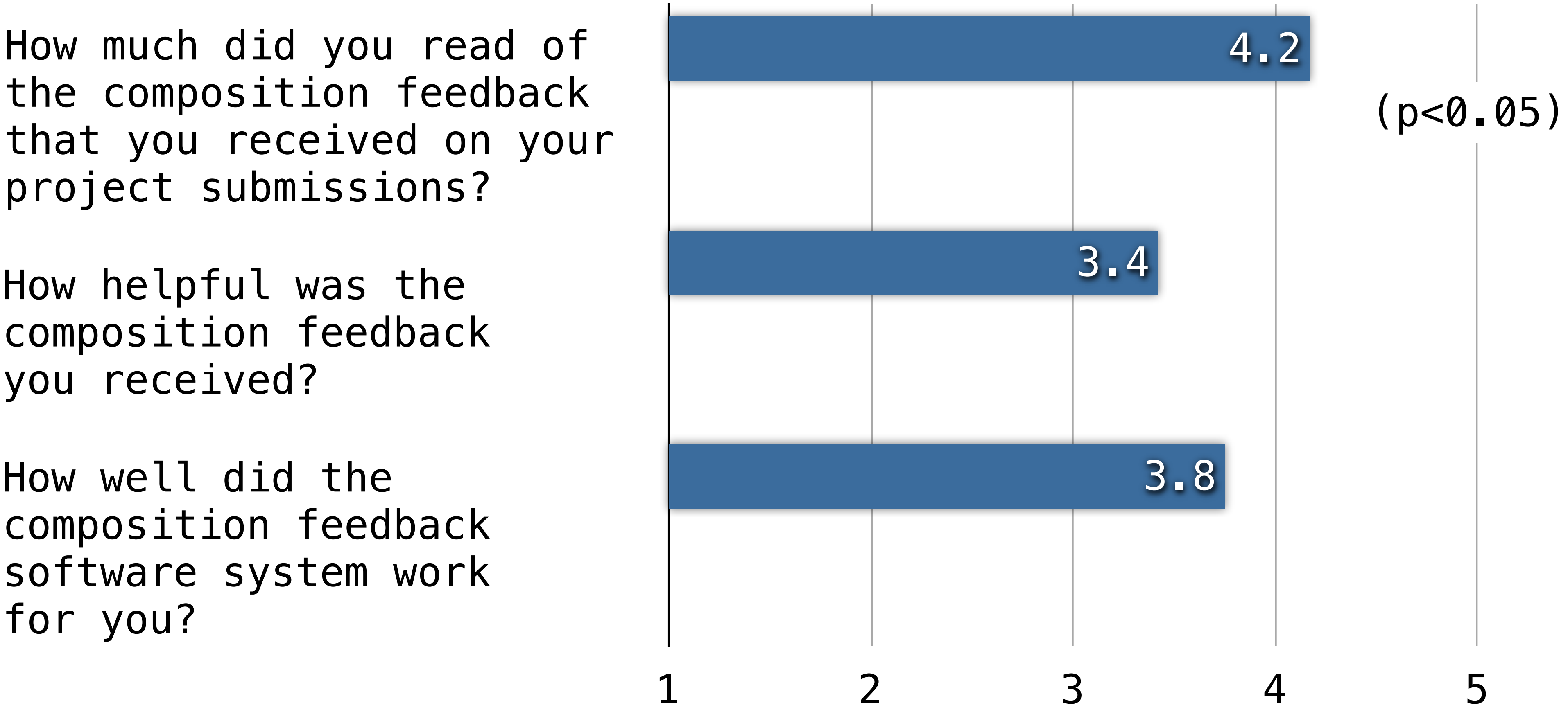
How much did you read of the composition feedback that you received on your project submissions?

How helpful was the composition feedback you received?



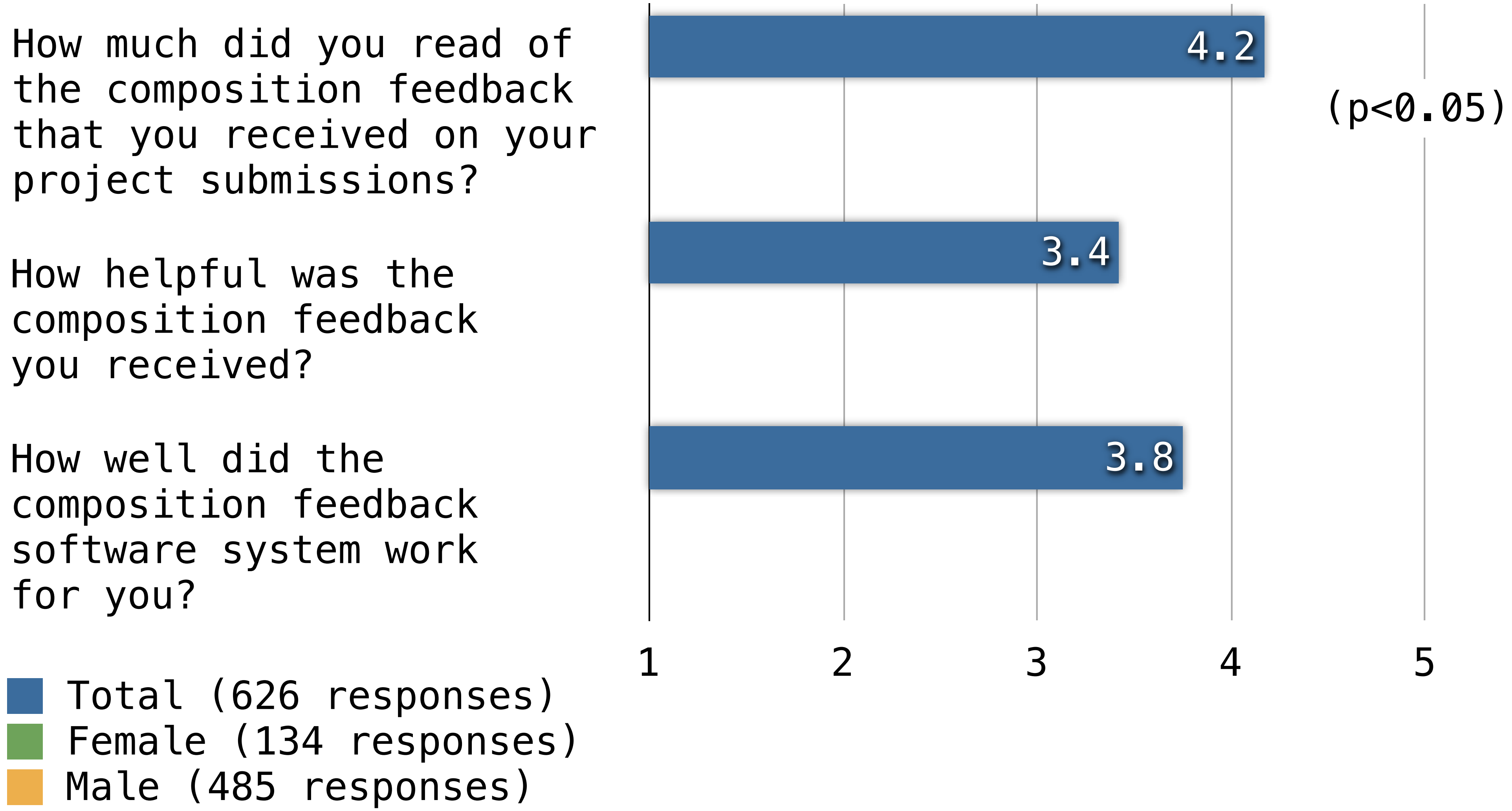
End-of-Semester Survey Results

Fall 2012 Average Responses (Scale from 1 to 5)



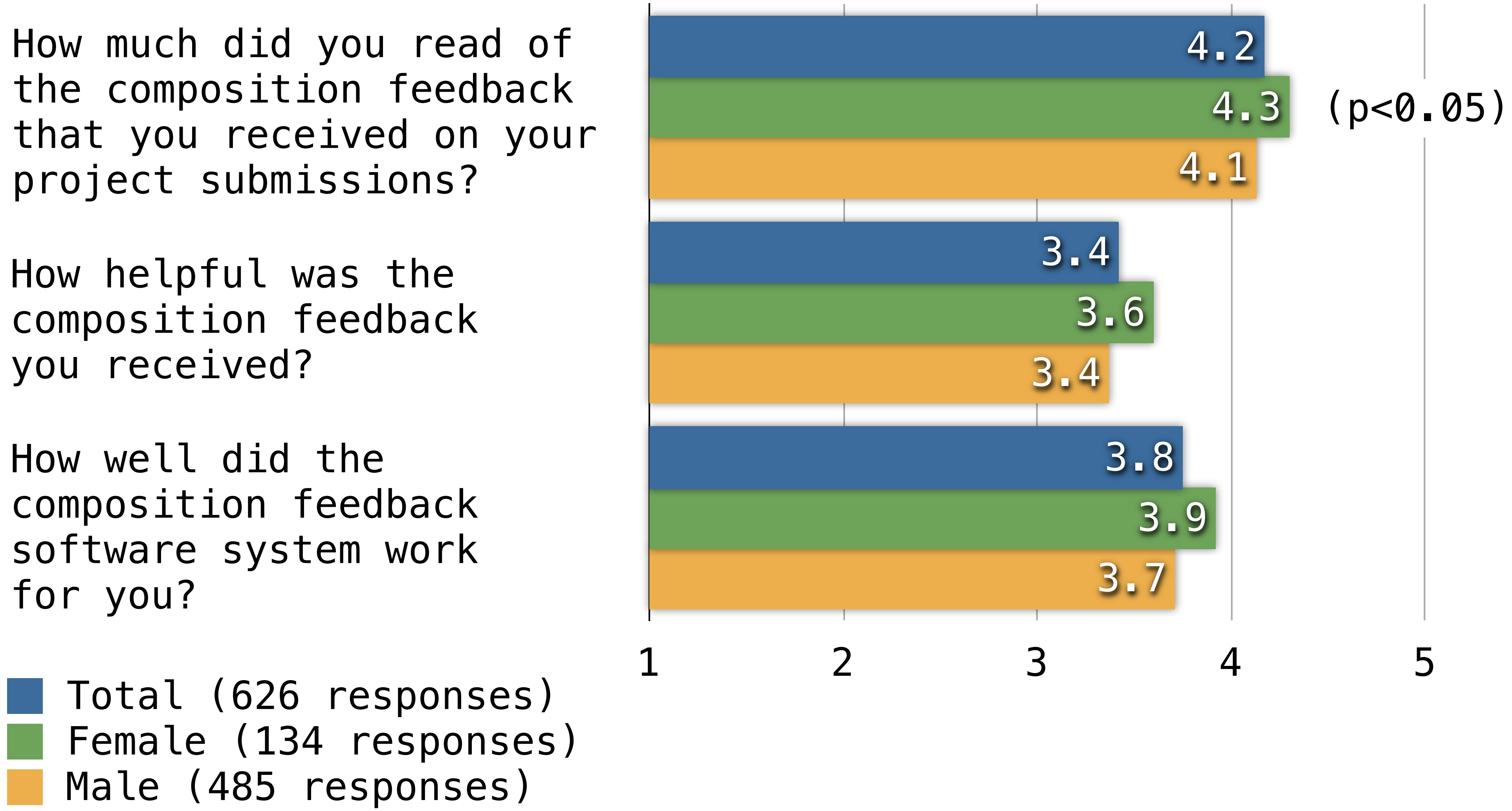
End-of-Semester Survey Results

Fall 2012 Average Responses (Scale from 1 to 5)



End-of-Semester Survey Results

Fall 2012 Average Responses (Scale from 1 to 5)


How much did you read of the composition feedback that you received on your project submissions?
(p<0.05)
How helpful was the composition feedback you received?
How well did the composition feedback software system work for you?
🟦 Total (626 responses)
🟩 Female (134 responses)
🟧 Male (485 responses)