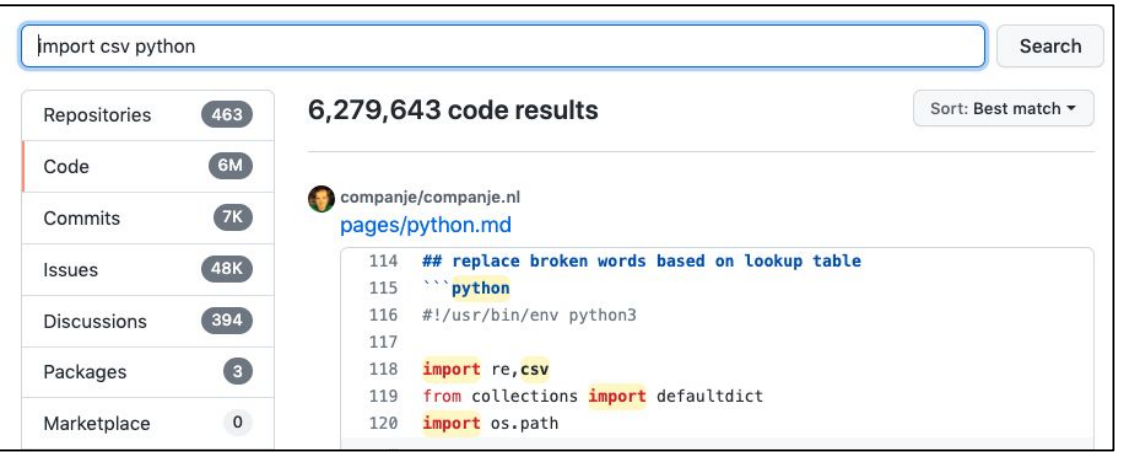
How Code Search Drives Software Engineering

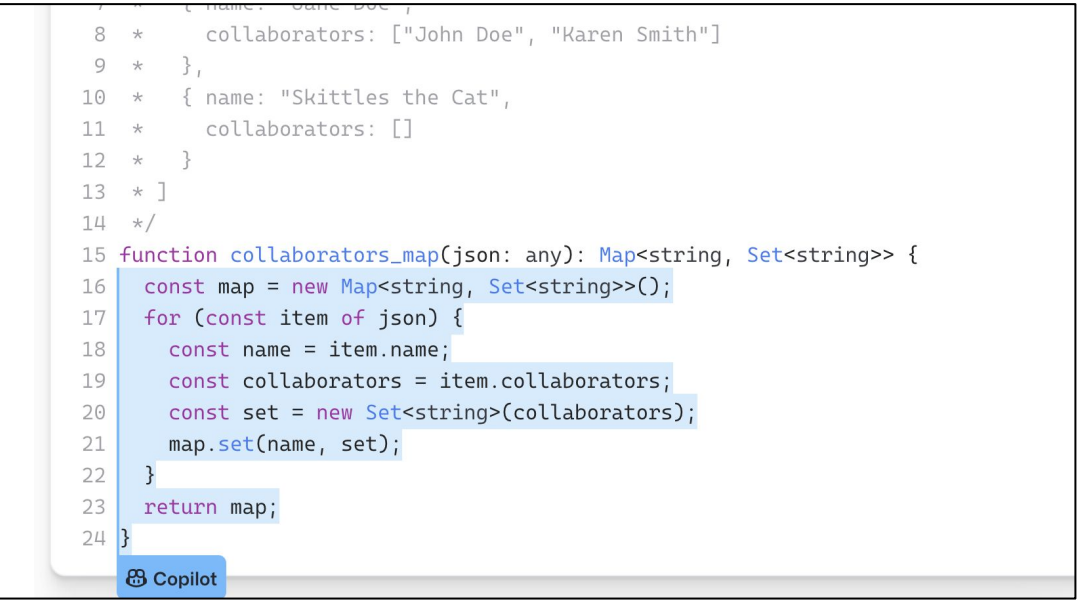
Dr. Kathryn (Katie) Stolee

Associate Professor

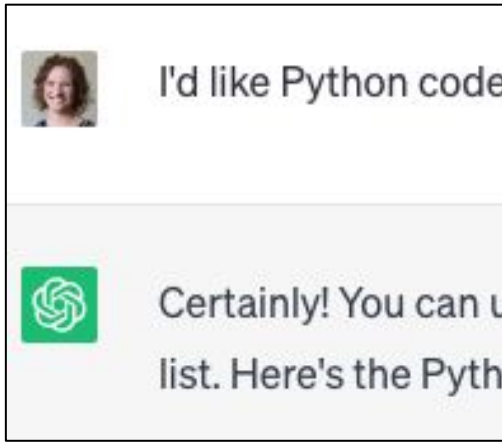
North Carolina State University

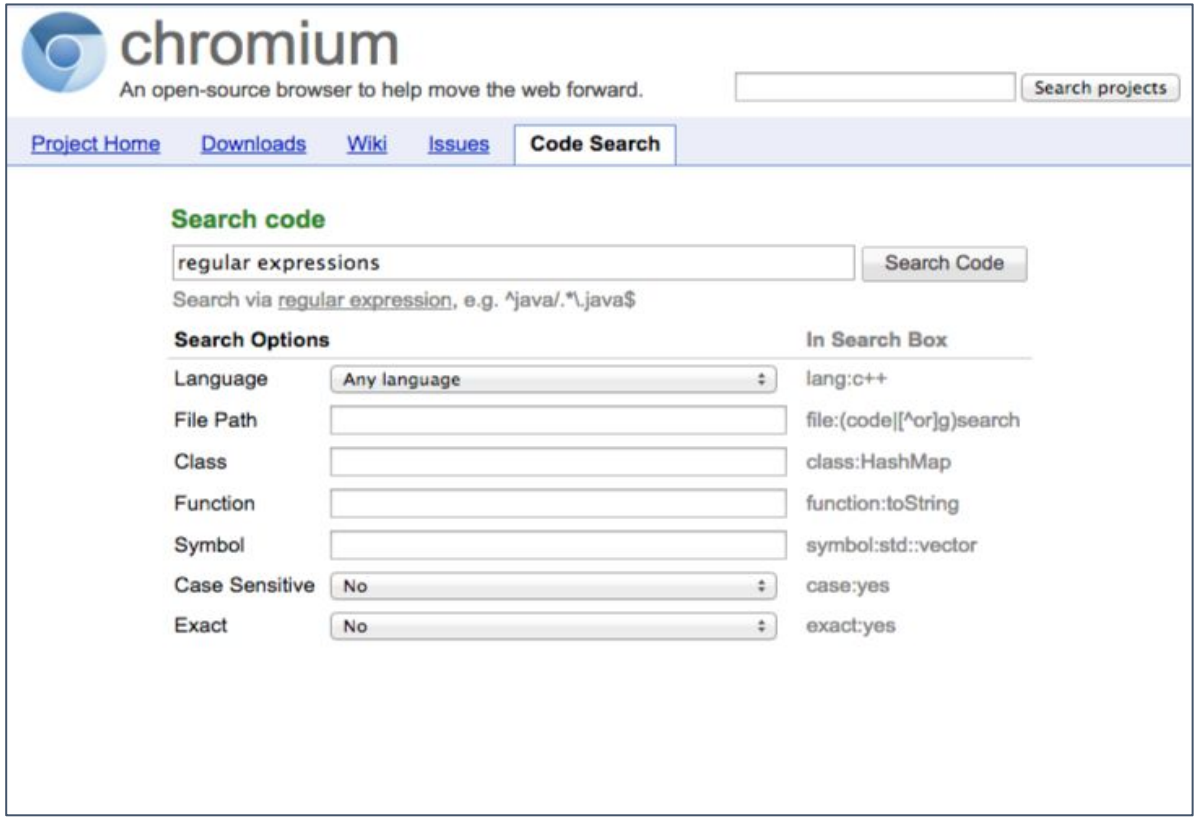
NC STATE UNIVERSITY


import csv python
Search
Repositories 463
Code 6M
Commits 7K
Issues 48K
Discussions 394
Packages 3
Marketplace 0
6,279,643 code results
Sort: Best match
companje/companje.nl
pages/python.md
114 ## replace broken words based on lookup table
115 ```python
116 #!/usr/bin/env python3
117
118 import re,csv
119 from collections import defaultdict
120 import os.path


8 * collaborators: ["John Doe", "Karen Smith"]
9 * },
10 * { name: "Skittles the Cat",
11 * collaborators: []
12 * }
13 *]
14 */
15 function collaborators_map(json: any): Map<string, Set<string>> {
16 const map = new Map<string, Set<string>>();
17 for (const item of json) {
18 const name = item.name;
19 const collaborators = item.collaborators;
20 const set = new Set<string>(collaborators);
21 map.set(name, set);
22 }
23 return map;
24 }
Copilot


Google
Google Search
I'm Feeling Lucky


I'd like Python code
Certainly! You can u
list. Here's the Pyth


chromium
An open-source browser to help move the web forward.
Search projects
Project Home
Downloads
Wiki
Issues
Code Search
Search code
regular expressions
Search Code
Search via regular expression, e.g. ^java/.*\.java$
Search Options
In Search Box
Language
Any language
lang:c++
File Path
file:(code|[^or]g)search
Class
class:HashMap
Function
function:toString
Symbol
symbol:std::vector
Case Sensitive
No
case:yes
Exact
No
exact:yes

We studied Google Developers in 2013

- Code search is performed throughout the development lifecycle
- Search queries happens ~12x per day

In 2022, we also found (yet unpublished observations):

- Searching for examples was **less successful** than searches for other purposes

FIND IT THINK IT TEST IT FIX IT

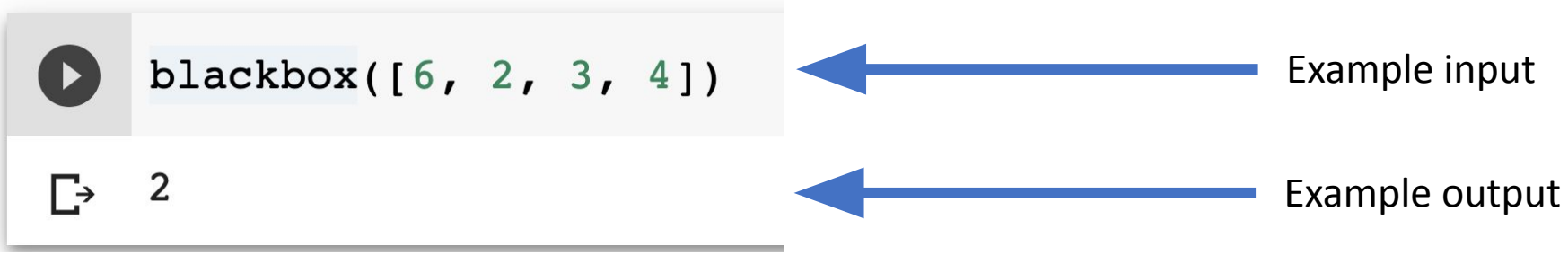
Finding Code Examples

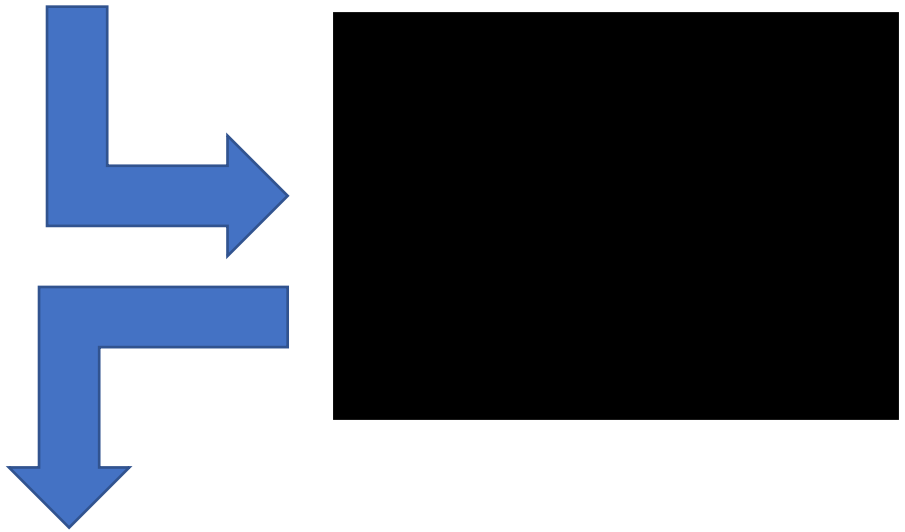
- Empirical Investigations into Developer Behavior
  - 85% of developers search for code at least weekly [TOSEM 2014]
  - Average of 12 queries per day [FSE 2015]
  - Code searches require more effort than information search [MSR 2018]

- Innovations:
  - Behavior-based code search via static analysis [TOSEM 2014]
  - Behavior-based code search via dynamic analysis [ICSE 2020] [FSE 2021]

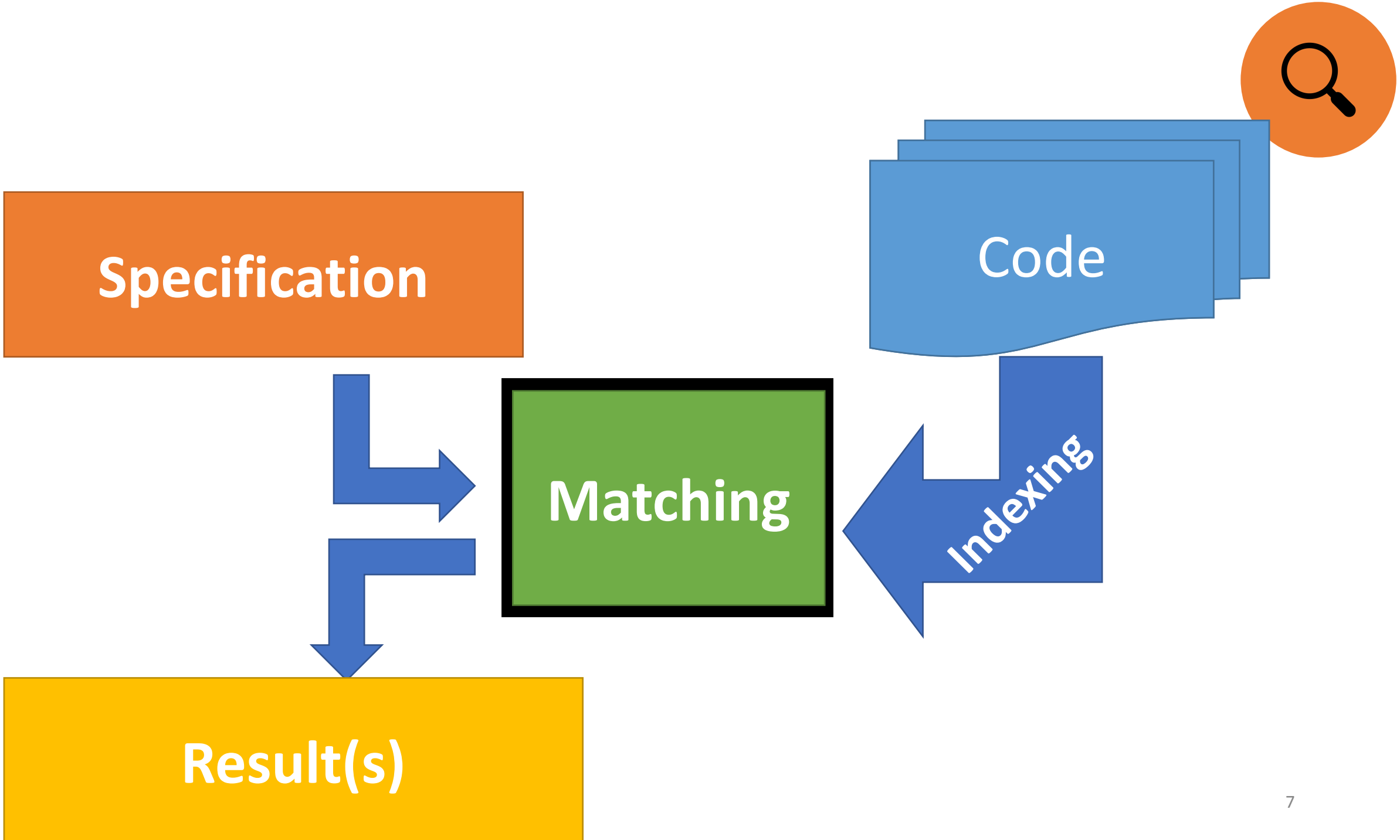
CAREER: On the Foundations of Semantic Code Search. $500k [active]

A different kind of search





```
def min_py2(a):
    return sorted(a)[0]
```

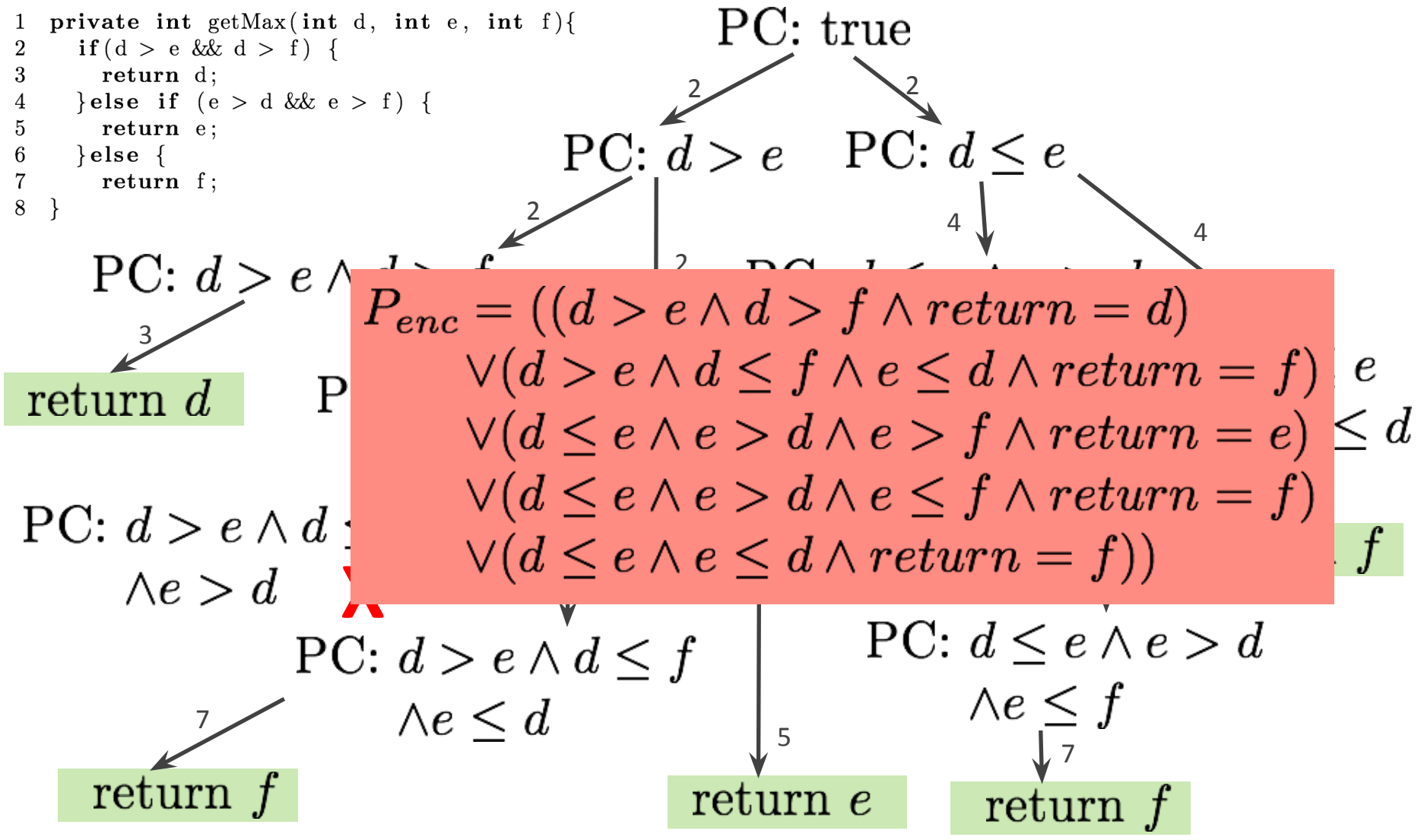

Specification
Code
Matching
Indexing
Result(s)

Code Search via Symbolic Execution

K. T. Stolee, S. Elbaum, M. B. Dwyer: Code search with input/output queries: Generalizing, ranking, and assessment. JSS 2016.

K. T. Stolee, S. Elbaum, D. Dobos: Solving the Search for Source Code. TOSEM 2014.

Symbolic Execution



SMT Solvers

Satisfiability **M**odulo **T**heory solvers determine if a logical formula is satisfiable

Facts	Assertions
`a >= 0`	`(assert (>= a 0))`
`b = 2`	`(assert (= b 2))`
`c = 2`	`(assert (= c 2))`
`c = a * b`	`(assert (= ( * a b) c))`

Result: **sat** $a \mapsto 1$

SMT Solvers

Satisfiability **M**odulo **T**heory solvers determine if a logical formula is satisfiable

Facts	Assertions
`a >= 0`	`(assert (>= a 0))`
`b = ?`	**`(assert (= b ?))`**
`c = 2`	`(assert (= c 2))`
`c = a * b`	`(assert (= ( * a b) c))`

Result: **sat** $a \mapsto 2 \wedge b \mapsto 2$

SMT Solvers

Satisfiability **M**odulo **T**heory solvers determine if a logical formula is satisfiable

Facts	Assertions
`a = 0`	`(assert (= a 0))`
`b = ?`	`(assert (= b ?))`
`c = 2`	`(assert (= c 2))`
`c = a * b`	`(assert (= ( * a b) c))`

Result: **unsat**

SMT Matching

```
1  private int getMax(int d, int e, int f){
2    if(d > e && d > f) {
3      return d;
4    }else if (e > d && e > f) {
5      return e;
6    }else {
7      return f;
8  }
```

$$P_{enc} = ((d > e \wedge d > f \wedge return = d) \vee (d > e \wedge d \leq f \wedge e \leq d \wedge return = f) \vee (d \leq e \wedge e > d \wedge e > f \wedge return = e) \vee (d \leq e \wedge e > d \wedge e \leq f \wedge return = f) \vee (d \leq e \wedge e \leq d \wedge return = f))$$

Query

Input	Output	Result
3, 4, 3	4	sa
3, 0, 0	3	ṣa
		ṭ

This is a result!

SMT Matching

Encoding

```
private int getsum(int a, int b, int c){
  return a + b + c;
}
```

Potential Search Result

```
(declare-fun a () Int)
(declare-fun b () Int)
(declare-fun c () Int)
(declare-fun return () Int)
(assert (= return (+ (+ a b) c)))
(assert (and (= a 3) (= b 4)
    (= c 0)))
(assert (= return 7))
```

Query

Input	Output	Result
3, 4, 3	7	unsa
3, 4, 0	7	sa
		t

Not a Result!

Why not just execute the code?

SMT Matching

Encoding

```
(declare-fun a () Int)
(declare-fun b () Int)
(declare-fun c () Int)
(declare-fun return () Int)
(assert (= return (+ (+ a b) ?)))
(assert (and (= a 3) (= b 4)
    (= c 0)))
(assert (= return 7))
```

Relax Encoding

```
private int getsum(int a, int b, int c){
  return a + b + c;
}
```

Potential Search Result

Query

Input	Output	Result
3, 4, 3	7	unsat
3, 4, 0	7	sat

This is a result (with modification)!

```
blackbox
2
```

This gets expensive!

```
] a) {

ength; i++) {
i] < m)
```

```
def min_py1(ar
    least = arr[
    for a in arr[1:]:
      if a < least:
        least = a
    return least
```

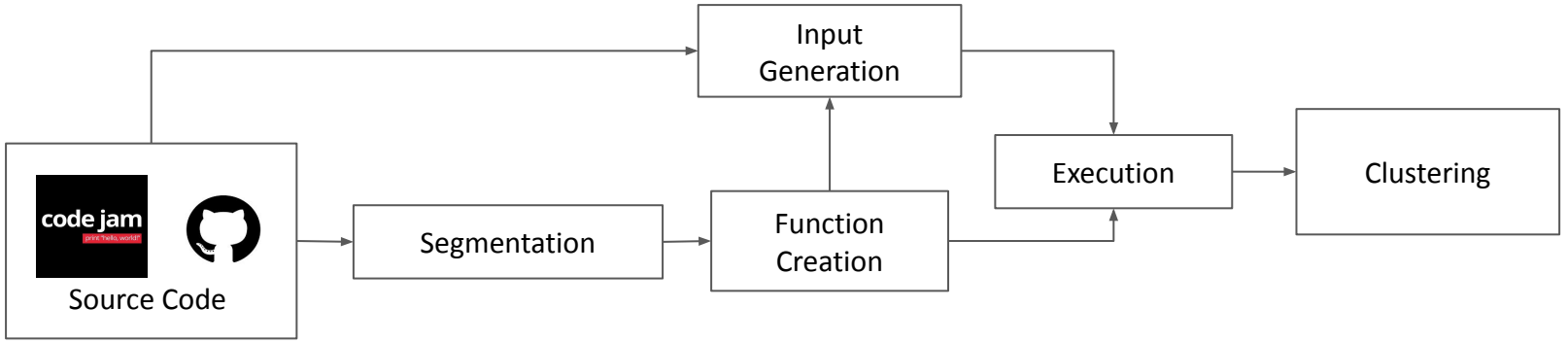
```
def min_py2(a):
    return sorted(a)[0]
```

Cross-Language Code Search

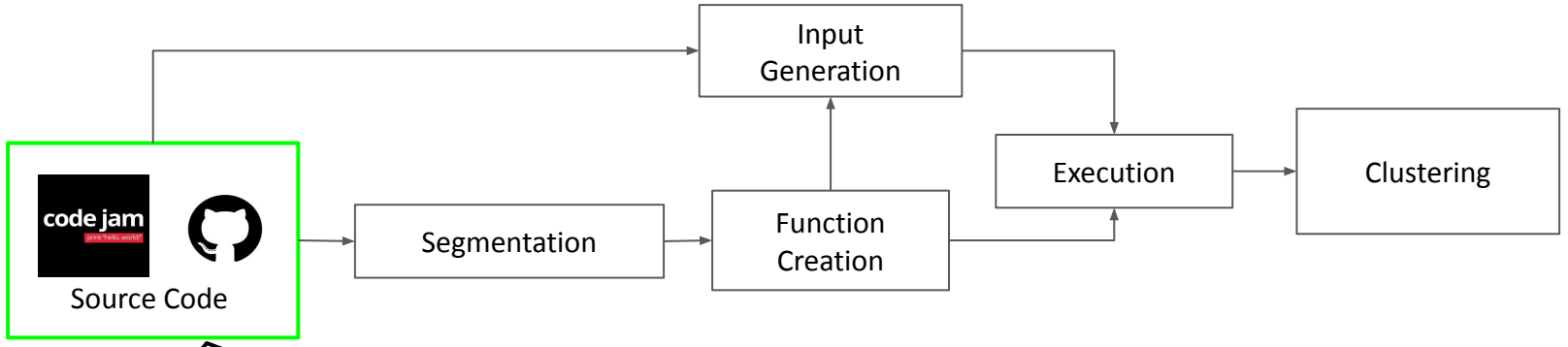
G. Mathew, K. T. Stolee: Cross-language code search using static and dynamic analyses. ESEC/SIGSOFT FSE 2021.

G. Mathew, C. Parnin, K. T. Stolee: SLACC: simion-based language agnostic code clones. ICSE 2020.

Cross-Language Behavioral Clones

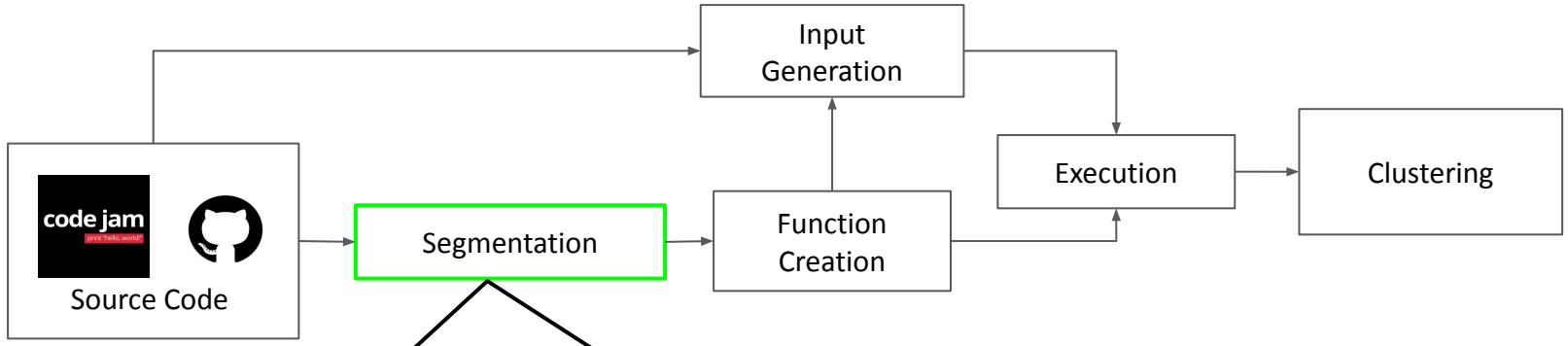


Cross-Language Behavioral Clones




```
public String interleave(int[] a, int[] b) {
  String result = "";
  int i = 0;
  for( i = 0; i < a.length && i < b.length; i++ ) {
    result += a[i];
    result += b[i];
  }
  int[] remaining = a.length < b.length ? b : a;
  for( int j = i; j < remaining.length; j++ ) {
    result += remaining[j];
  }
  return result;
}
```

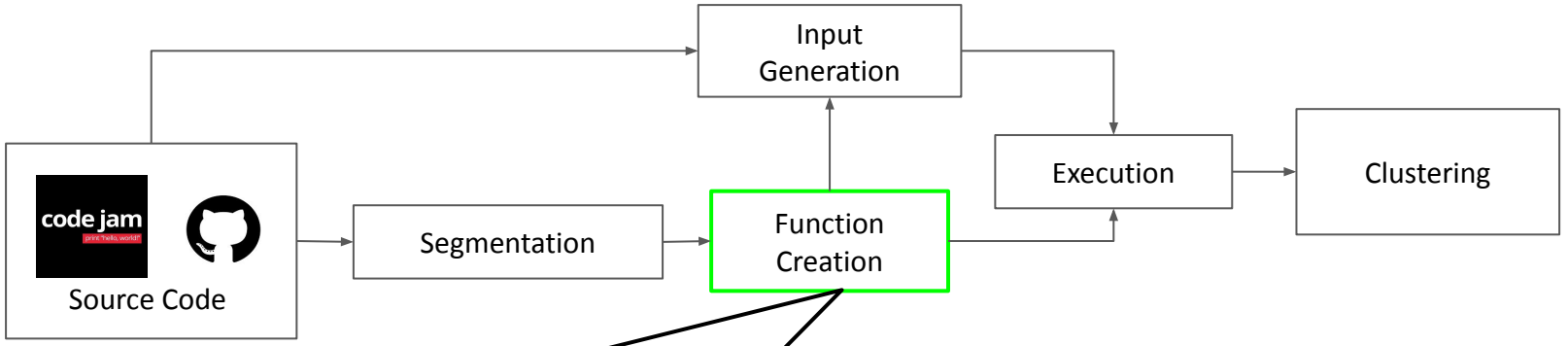
Cross-Language Behavioral Clones




```
String result = "";
int i = 0;
for( i = 0; i < a.length && i < b.length; i++ ) {
  result += a[i];
  result += b[i];
}
```

```
int[] remaining = a.length < b.length ? b : a;
for( int j = i; j < remaining.length; j++ ) {
  result += remaining[j];
}
return result;
```

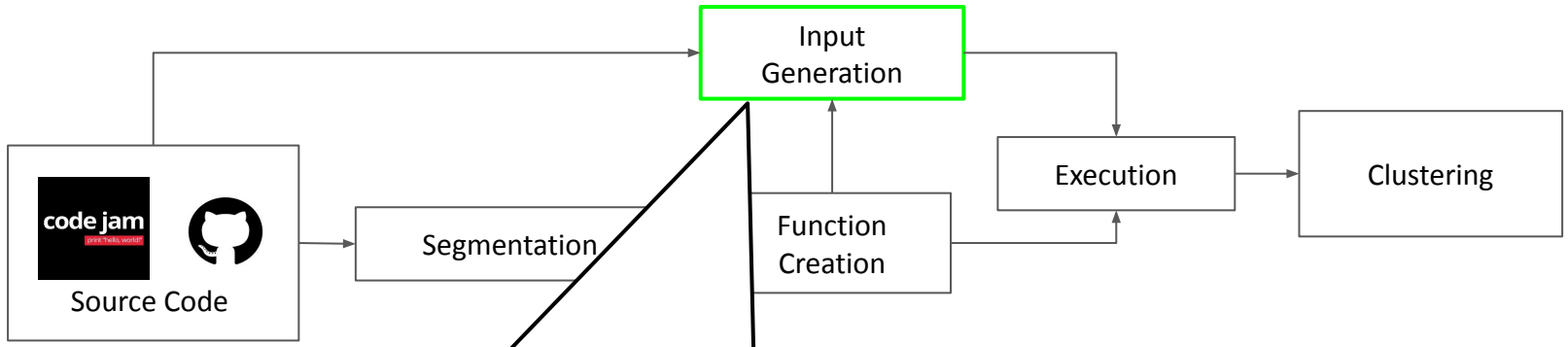
Cross-Language Behavioral Clones




```
public String func_b15f(int[] a, int[] b) {
  String result = "";
  int i = 0;
  for( i = 0; i < a.length && i < b.length; i++ ) {
    result += a[i];
    result += b[i];
  }
  return result;
}
```

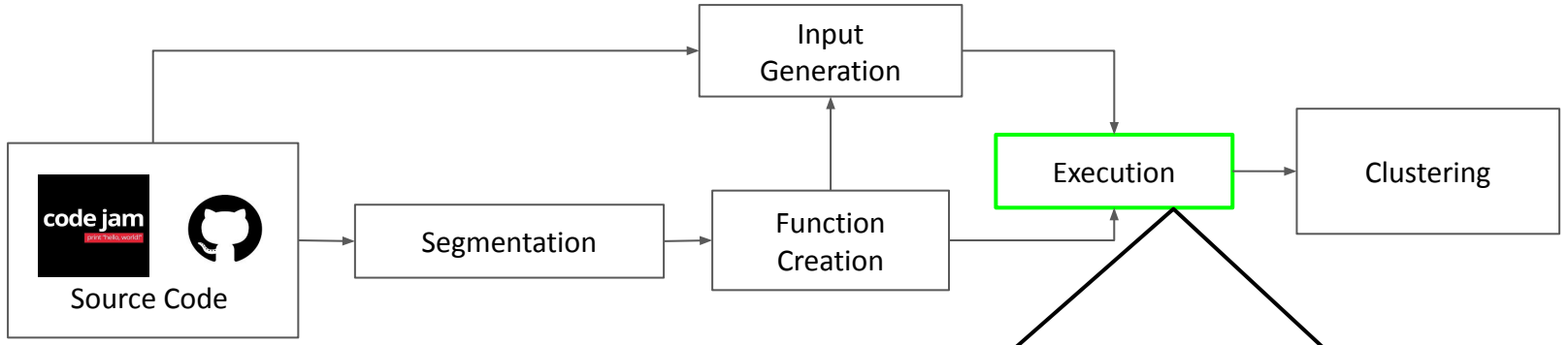
```
public String func_ea72(int[] a, int[] b, int i, String result) {
  int[] remaining = a.length < b.length ? b : a;
  for( int j = i; j < remaining.length; j++ ) {
    result += remaining[j];
  }
  return result;
}
```

Cross-Language Behavioral Clones



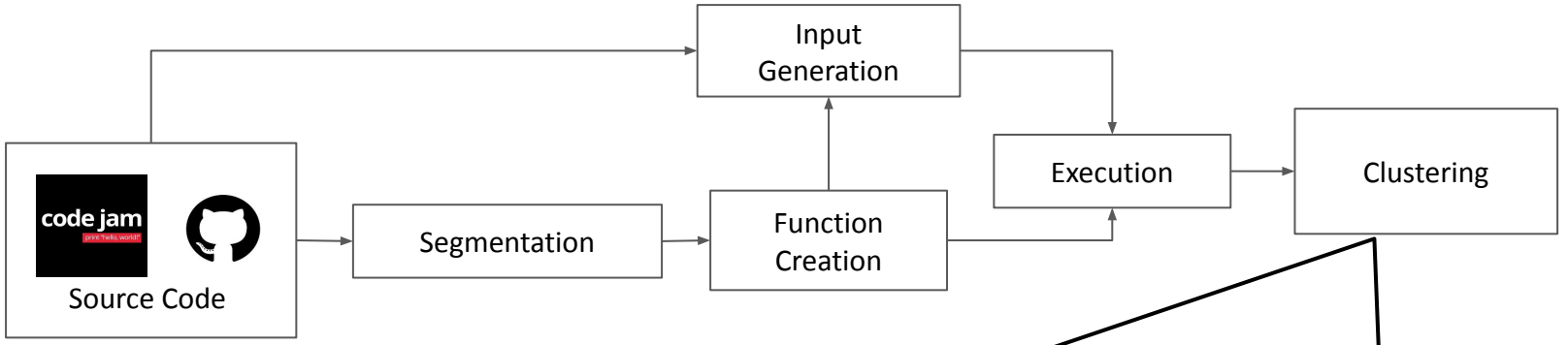

```
int[], int[]
• [1,2,3], [45, 16]
• [ ], [3,2,1]
  ⋮
  256 times
  ⋮
• [4, 5, 6, 7, 8, 99], [ ]
int[], int[], int, String
• [1,2,3], [45, 16], 0, “Hello
  World”
  ⋮
  256 times
  ⋮
```

Cross-Language Behavioral Clones




```
func_b15f(int[] a, int[] b)
• func_b15f([1,2,3], [45, 16]) ⇒ 1452163
• func_b15f([ ], [3,2,1]) ⇒ 321
  ⋮
  256 times
  ⋮
• func_b15f([4, 5, 6, 7, 8, 99], [ ]) ⇒ 4567899
• func_b15f([3], [2]) ⇒ 32
func_ea72(int[] a, int[] b, int i, String result)
• func_ea72([1,2,3], [45, 16], 0, "Hello World") ⇒
  Hello World123
  ⋮
  256 times
```

Cross-Language Behavioral Clones




```java
public String func_b15f(int[] a, int[] b) {
  String result = "";
  int i = 0;
  for( i = 0; i < a.length && i < b.length; i++ ) {
    result += a[i];
    result += b[i];
  }
  return result;
}
```

```python
def func_9f34(l1, l2):
    result = ""
    for (e1, e2) in zip(l1, l2):
        result += str(e1)
        result += str(e2)
    return result
```

```python
def func_15e8(l1, l2):
    from itertools import chain
    return "".join([str(x)
            for x in chain.from_iterable(zip(l1, l2))])
```

What does this mean for search?

1

```java
public Integer minJ(int[] a) {
    Integer m = null;
    for (int i=0; i < a.length; i++) {
      if (m == null || a[i] < m)
        m = a[i];
    }
    return m;
}
```

2

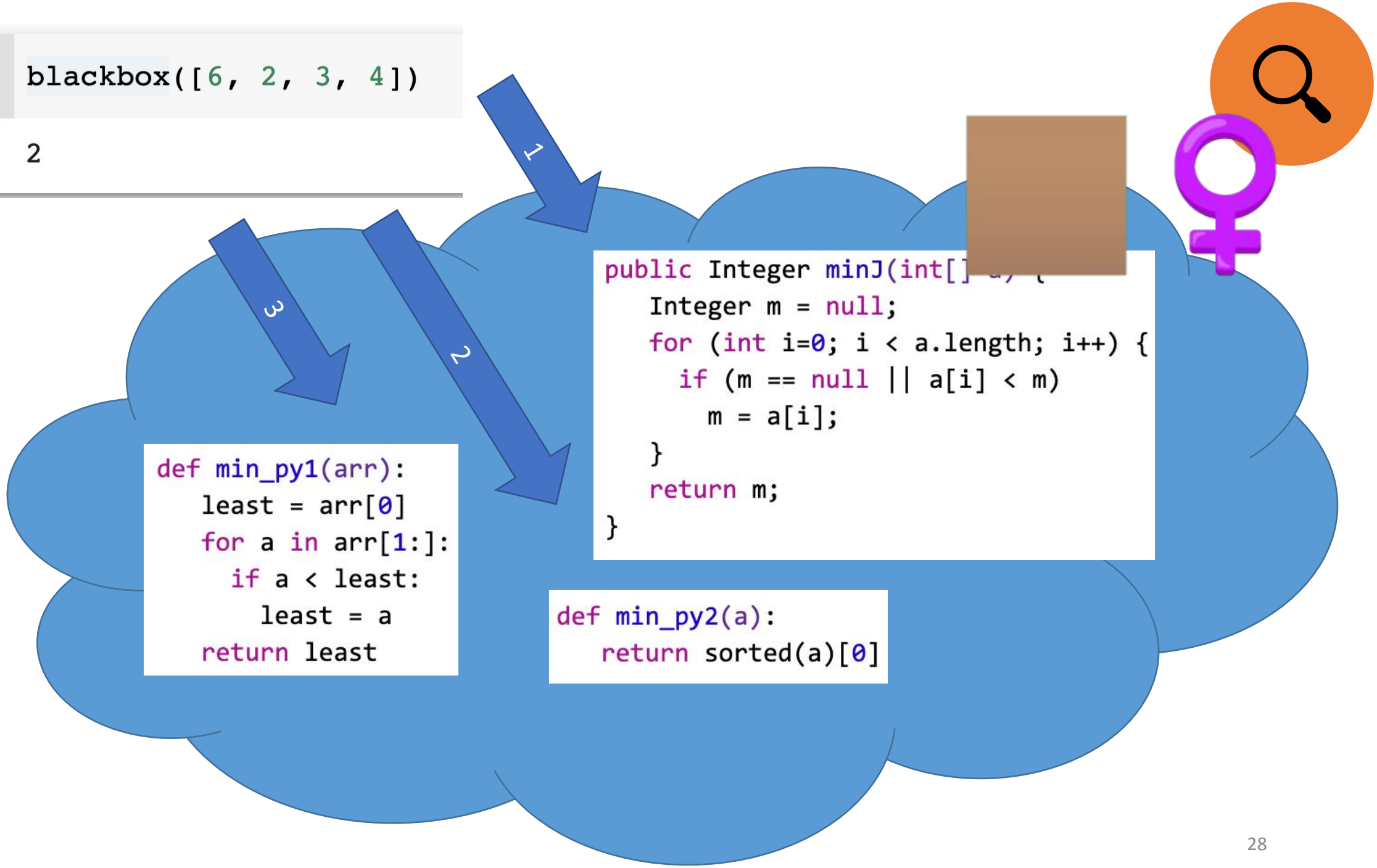
```python
def min_py2(a):
    return sorted(a)[0]
```

3

```python
def min_py1(arr):
    least = arr[0]
    for a in arr[1:]:
      if a < least:
        least = a
    return least
```

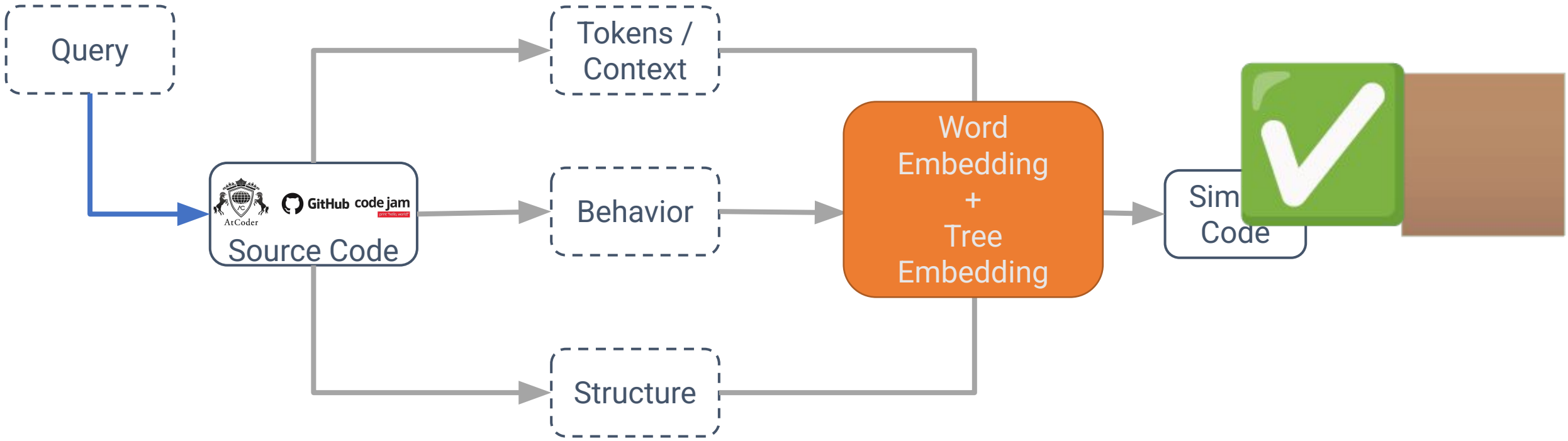
```
blackbox([6, 2, 3, 4])
```

```
2
```

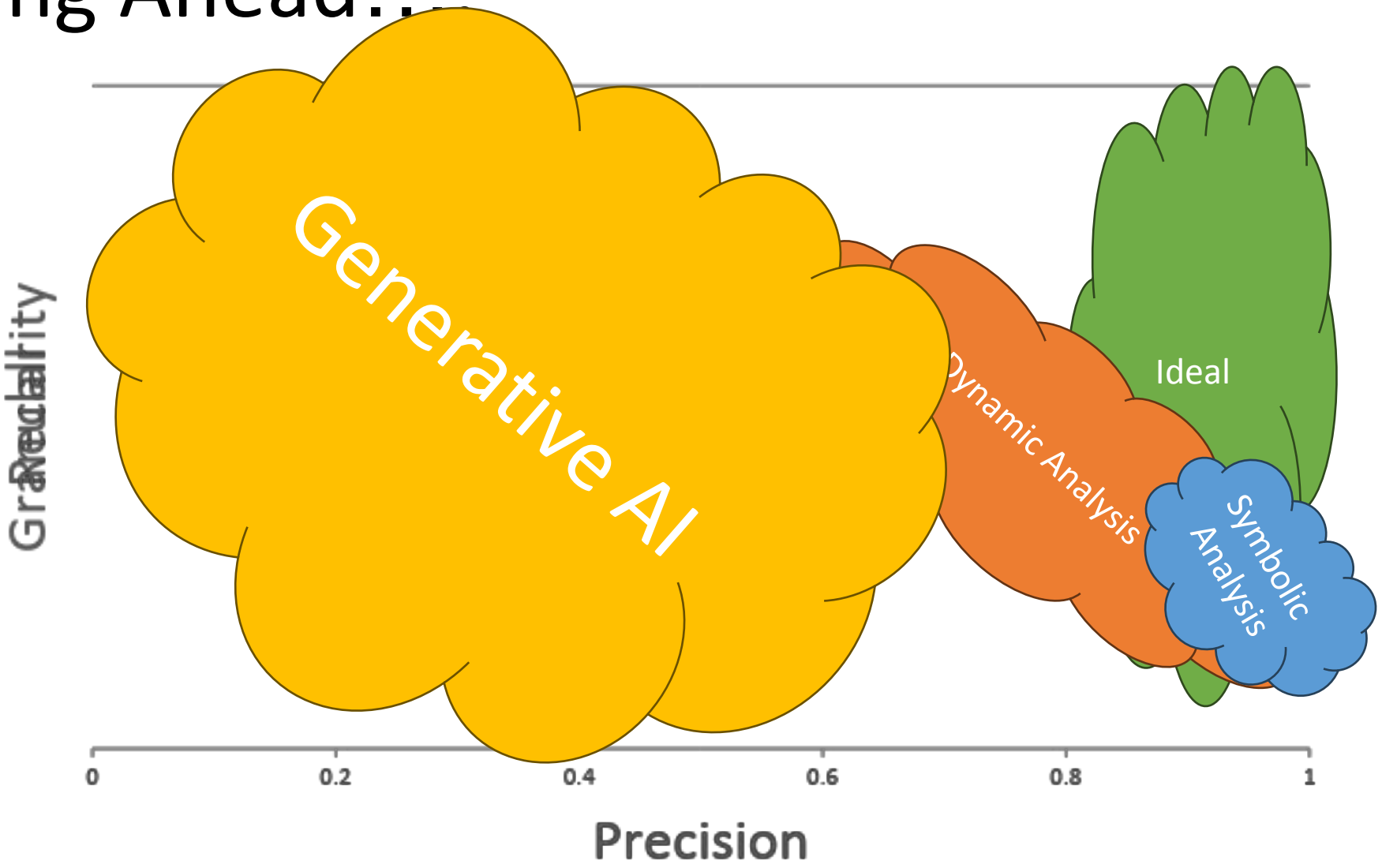


What about code-to-code search?

Code-to-code Search



Looking Ahead…



Looking Ahead…

I'd like Python code that, given an input of [6,2,3,4], produces the output 2

Certainl
list. Her

```
python
def f
    s
    r

input
outpu
print
```

What is another way to do this?

Another way to find the smallest number in a list is by using a simple loop. Here's an alternative implementation using a loop:

python Copy code

```
def find_smallest_number(numbers):
    smallest = numbers[0]  # Initialize smallest as the first element of the

    for num in numbers:
        if num < smallest:
            smallest = num
```

FIND IT THINK IT TEST IT FIX IT

Understanding Code

- Empirical Investigations into Code Comprehension
  - Regular expression representation significantly impacts understandability [ASE 2017]
  - Comparing similar code algorithms is difficult and error-prone for developers [VL/HCC 2022]
  - Code review of refactorings is *very hard* for students [under review]

SHF: SMALL: Automated Discovery of Cross-Language Program Behavior Inconsistency $250k (Lead PI, $500k total grant) [active]

Understanding Code

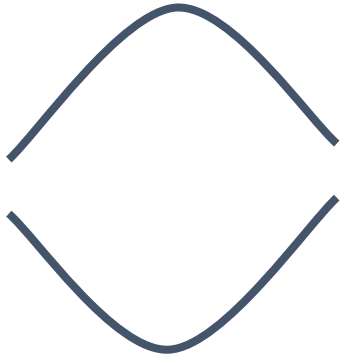
- Empirical Investigations into Code Comprehension
  - Regular expression representation significantly impacts understandability [ASE 2017]
  - **Comparing similar code algorithms is difficult and error-prone for developers [VL/HCC 2022]**
  - Code review of refactorings is *very hard* for students [under review]

SHF: SMALL: Automated Discovery of Cross-Language Program Behavior Inconsistency $250k (Lead PI, $500k total grant) [active]

Comparative Comprehension

The cognitive activity of understanding how algorithms behave relative to each other

```
def sumup(numbers):
  accumulator = 0
  for value in numbers:
    accumulator += value
  return accumulator
```



```
def sumup(x):
  s = 0
  i = 0
  while i < len(x):
    s += i
    i += 1
  return s
```

Controlled Experiment

```java
public static boolean isAnagram(String str1,
                                String str2) {
  if (str1.length() != str2.length())
    return false;

  int[] count1 = new int[256];
  int[] count2 = new int[256];

  for (int i = 0; i < str1.length(); i++) {
    ++count1[str1.charAt(i)];
    ++count2[str2.charAt(i)];
  }

  for (int i = 0; i < 256; i++)
    if (count1[i] != count2[i])
      return false;

  return true;
}
```

```python
def isAnagram(s, t):
  hash1 = [0]*256
  hash2 = [0]*256

  for char in s:
    hash1[ord(char)] += 1
  for char in t:
    hash2[ord(char)] += 1

  return hash1 == hash2
```

- 4 independent dimensions of variation
  - Behavior (same or not)
  - Language (same or not)
  - Structures (similar AST or not)
  - Meaningful names (original or obfuscated)

Controlled Experiment

Interviews

n=16

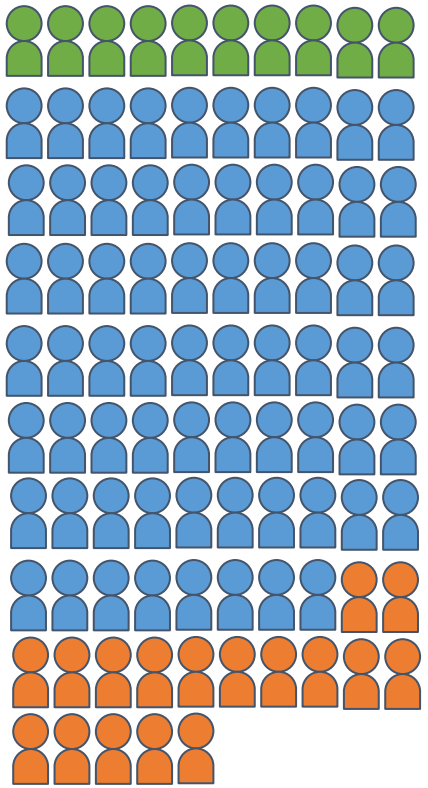
Undergraduate students

Graduate students

Professionals

Survey

n=95



Unknown

Graduate students

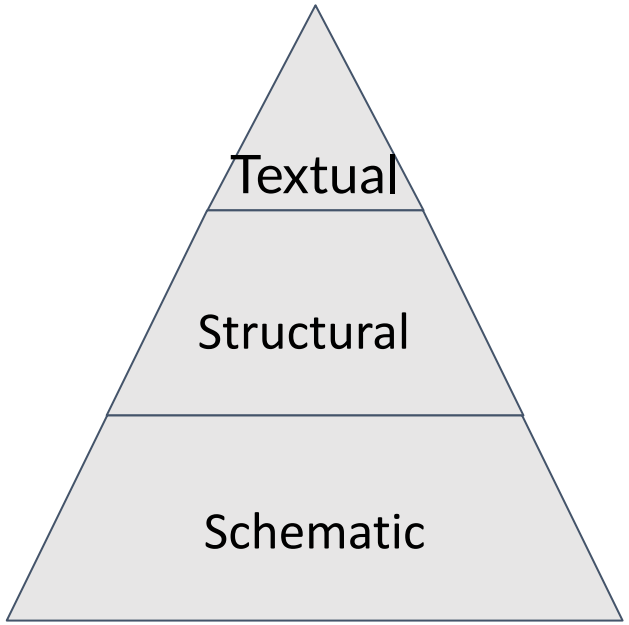
Professionals

Comparison Accuracy

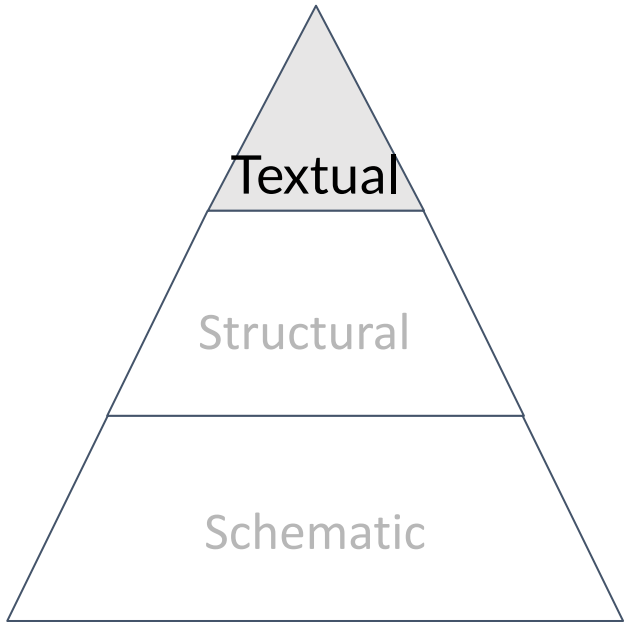
Overall correctness: 292 of 439 — 66.5%

	Correctness (%) for…		
	Similarity	Dissimilarity	
Clone Truth	85.3	46.7	***
Language	70.9	62.7	*
Structure	75.0	59.9	*
Names (Meaningful|Obf.)	66.8	66.2	

Comparison Strategies



Comparison Strategies

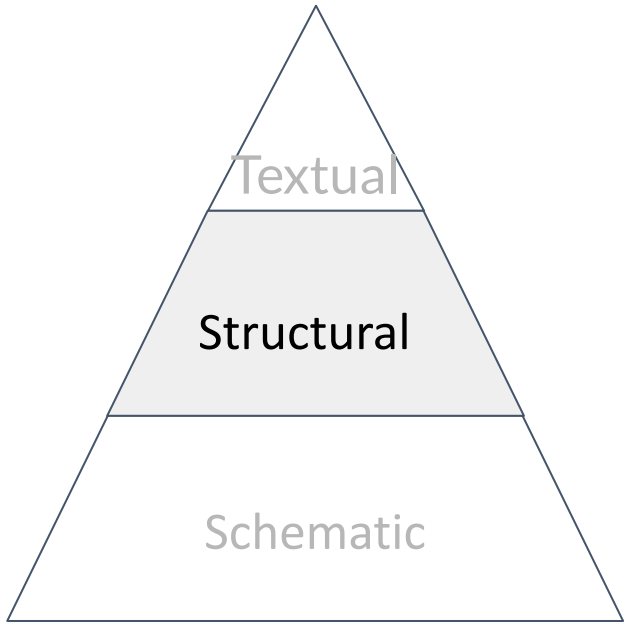



```python
def camel_case(string):
  a = list(string)
  for i in range(0, len(a)):
    if i==0 or a[i-1]=='_':
      a[i] = a[i].upper()
  return ''.join(a).replace('_','')
```

```python
def camel_case(string):
  a = list(string)
  for i in range(len(a)):
    if i==0 or a[i-1]=='_':
      a[i] = a[i].upper()
  return ''.join([c for c in a if c != "_"])
```

“I didn't even need to [understand the logic] because they were so similar.” - P4

Comparison Strategies

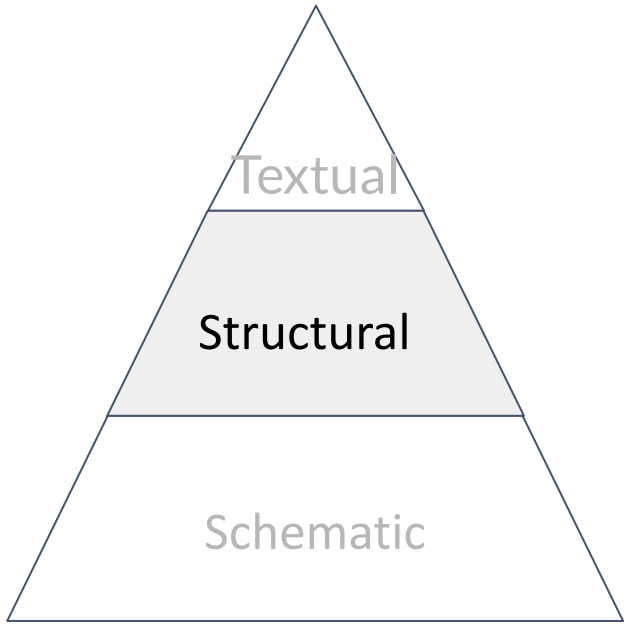



```java
public static String removeDuplicates(String orig)
{
  if (orig.length() == 0) return "";
  String result = "" + orig.charAt(0);
  for (int i = 1; i < orig.length(); i++)
    if (orig.charAt(i-1) != orig.charAt(i))
      result+=orig.charAt(i);
  return result;
}
```

```python
def removeadjacent(orig):
  if len(orig) == 0: return orig
  res = ""
  for i in range(1, len(orig)):
    if orig[i-1] != orig[i]:
      res += orig[i]
  return orig[0] + res
```

U4 on cross-language deduplicators

Comparison Strategies

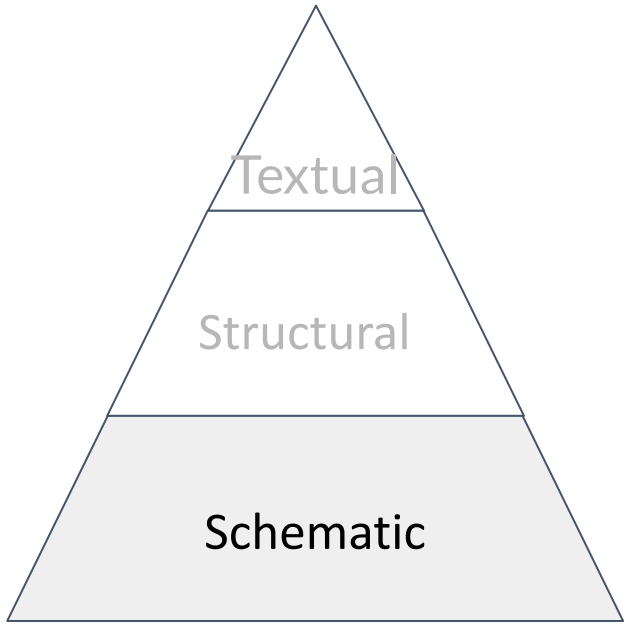



```java
public static String removeDuplicates(String orig)
{
  if (orig.length() == 0) return "";
  String result = "" + orig.charAt(0);
  for (int i = 1; i < orig.length(); i++)
    if (orig.charAt(i-1) != orig.charAt(i))
      result+=orig.charAt(i);
  return result;
}
```

```python
def removeadjacent(orig):
  if len(orig) == 0: return orig
  res = ""
  for i in range(1, len(orig)):
    if orig[i-1] != orig[i]:
      res += orig[i]
  return orig[0] + res
```

U4 on cross-language deduplicators

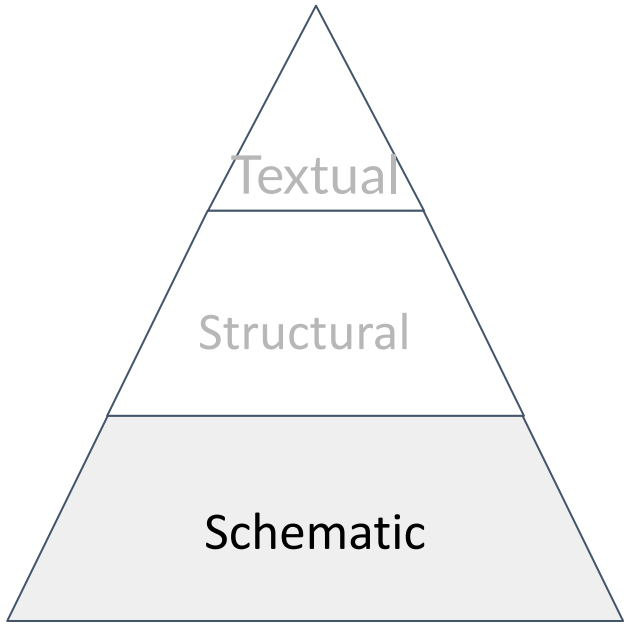
Comparison Strategies




```python
def to_camel_case(text):
  cap = True
  newText = ''
  for t in text:
    if t == '_':
      cap = True
      continue
    else:
      if cap == True:
        t = t.upper()
      newText = newText + t
      cap = False
  return newText
```

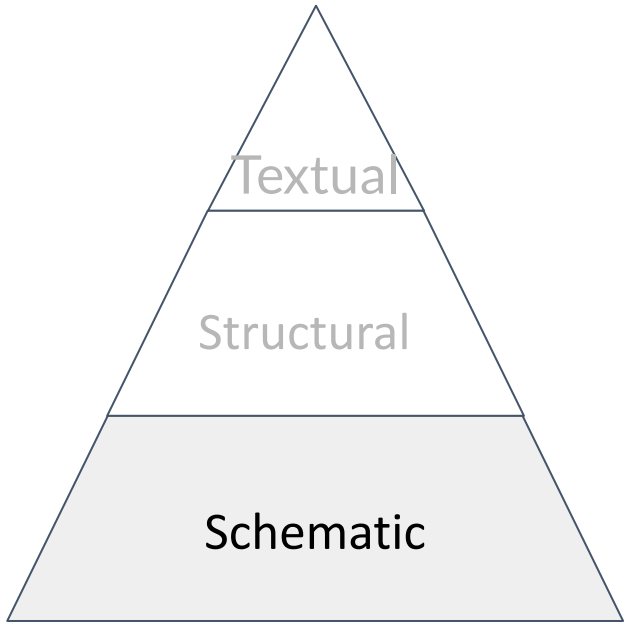
```python
def UnderscoreToCamelCase(under_score):
  segments = under_score.split('_')
  return segments[0] + ''.join([s[0].upper() + s[1:] \
    for s in segments[1:] if len(s) > 0])
```

Comparison Strategies



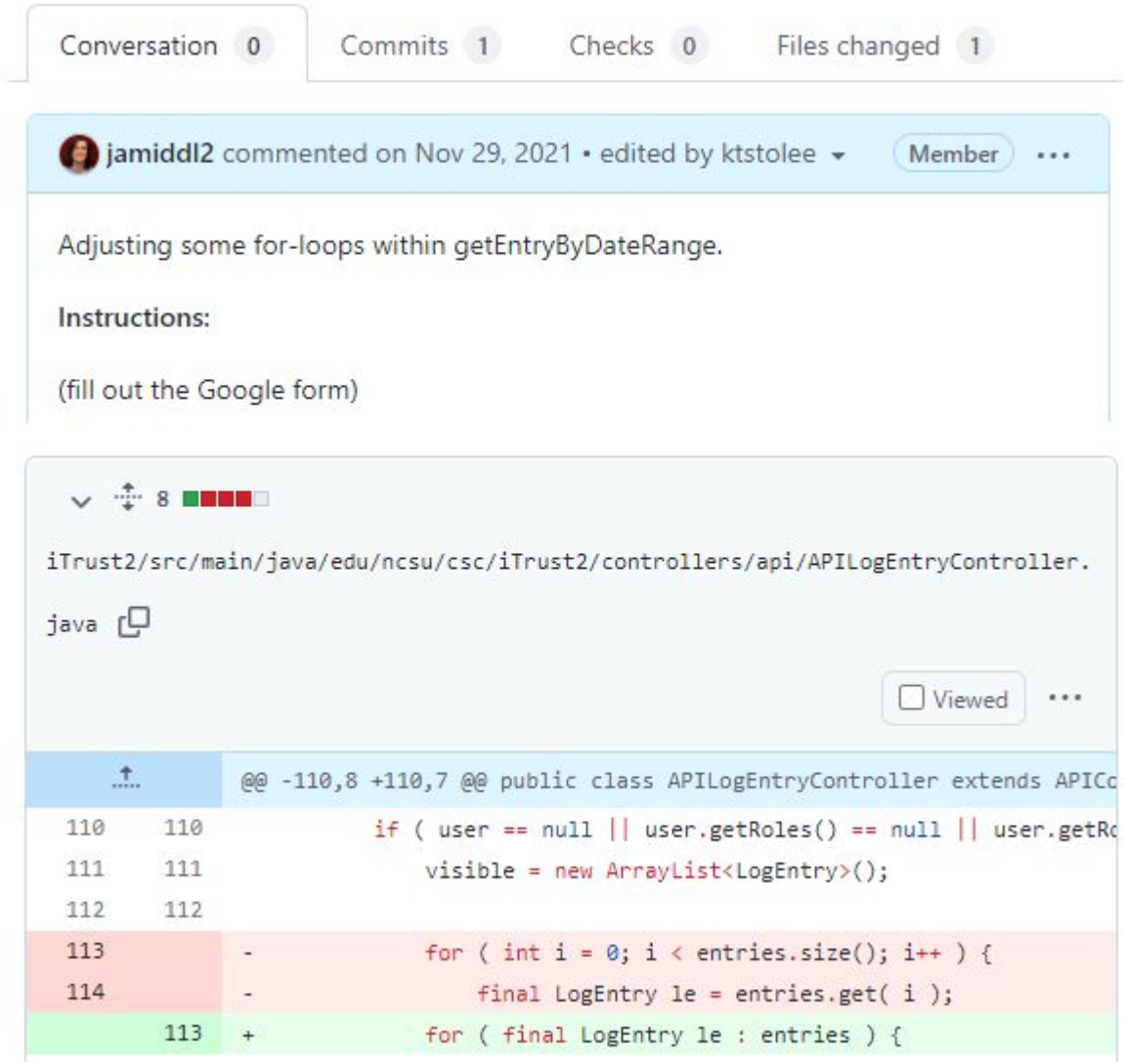

```
def UnderscoreToCamelCase(under_score):
  segments = under_score.split('_')
  return segments[0] + ''.join([s[0].upper() + s[1:] \
    for s in segments[1:] if len(s) > 0])
```

Comparison Strategies

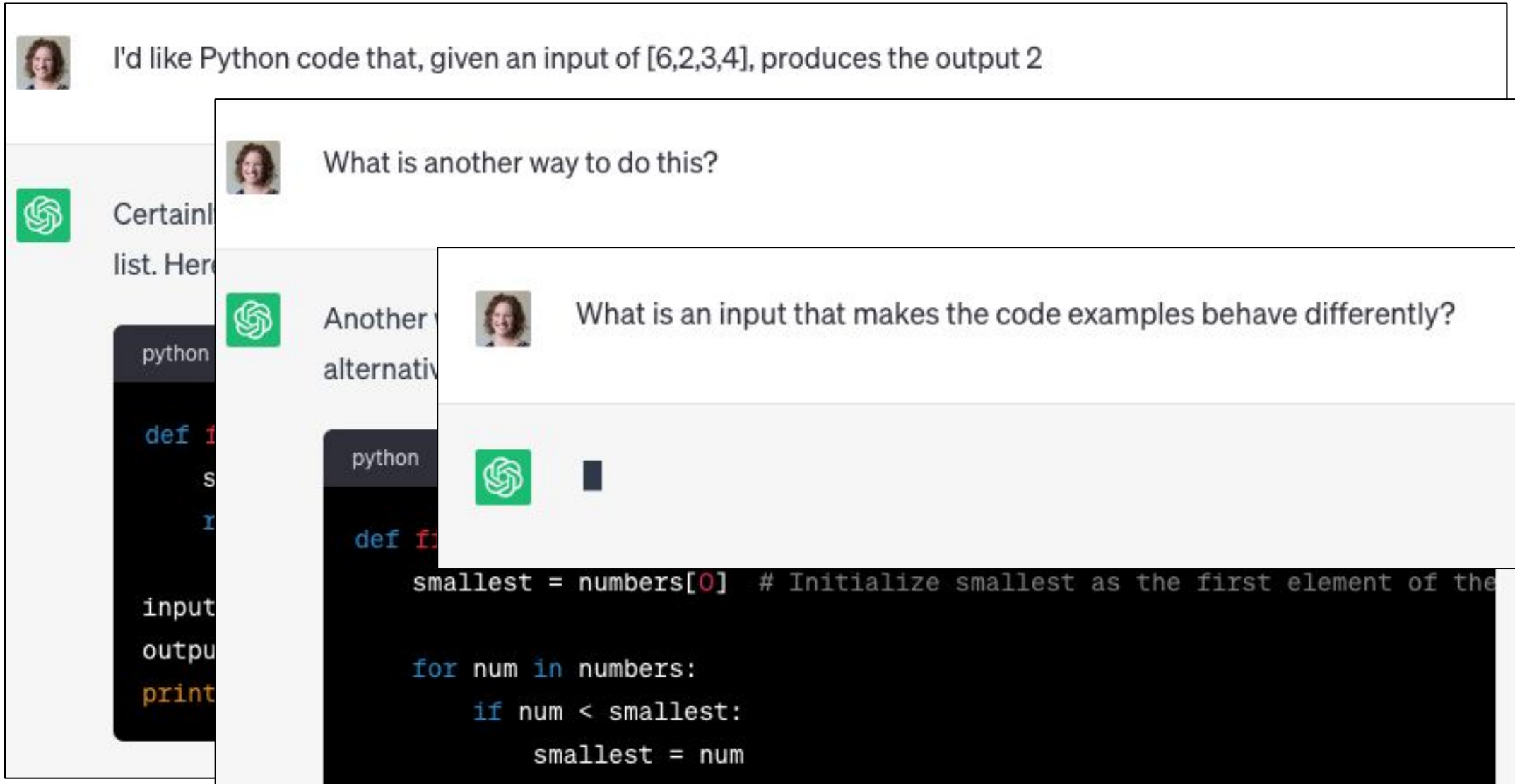



```python
def to_camel_case(text):
    cap = True
    newText = ''
    for t in text:
      if t == '_':
        cap = True
        continue
      else:
        if cap == True:
          t = t.upper()
        newText = newText + t
        cap = False
    return newText
```

Looking Ahead…



Looking Ahead…



FIND IT THINK IT TEST IT FIX IT

Testing Code

- Empirical Investigations into Code Repositories
  - Only 17% of regular expressions are tested at all [FSE 2018]
  - Students believe code coverage is the most important outcome for test suites [ITiCSE 2021]

- Innovations:
  - A static checklist for testing is as effective as coverage tools for second-year students [ITiCSE 2022]

SHF: Small: Supporting Regular Expression Testing, Search, Repair, Comprehension, and Maintenance $500k [completed]

IUSE: EHR: Improving Software Testing Education through Lightweight Explicit Testing Strategies and Feedback $150k (lead PI, $300k total) [active]

Testing Code

- Empirical Investigations into Code Repositories
  - Only 17% of regular expressions are tested at all [FSE 2018]
  - **Students believe code coverage is the most important outcome for test suites [ITiCSE 2021]**

- Innovations:
  - **A static checklist for testing is as effective as coverage tools for second-year students [ITiCSE 2022]**

SHF: Small: Supporting Regular Expression Testing, Search, Repair, Comprehension, and Maintenance $500k [completed]

IUSE: EHR: Improving Software Testing Education through Lightweight Explicit Testing Strategies and Feedback $150k (lead PI, $300k total) [active]

README.md

Test Case Checklist

Each test case *should*:

- [ ] be executable (i.e., it has an **@Test** annotation and can be run via "Run as JUnit Test")
- [ ] have at least one assert statement or assert an exception is thrown. Example assert statements include: **assertTrue**, **asse**
  assertEquals (click for tutorials). For asserting an exception is thrown, there are different approaches: **try{...; fail();} catc**
  e){assertThat...;}, **@Test(expected = exception.class)** in JUnit 4, or **assertThrows** in JUnit 5 (click for tutorials).
- [ ] evaluate/test only one method

Each test case *could*:

- [ ] be descriptively named and commented
- [ ] If there is redundant setup code in multiple test cases, extract it into a common method (e.g., using **@Before**)
- [ ] If there are too many assert statements in a single test case (e.g., more than 5), you might split it up so each test evaluate
  behavior.

Test Suite Checklist

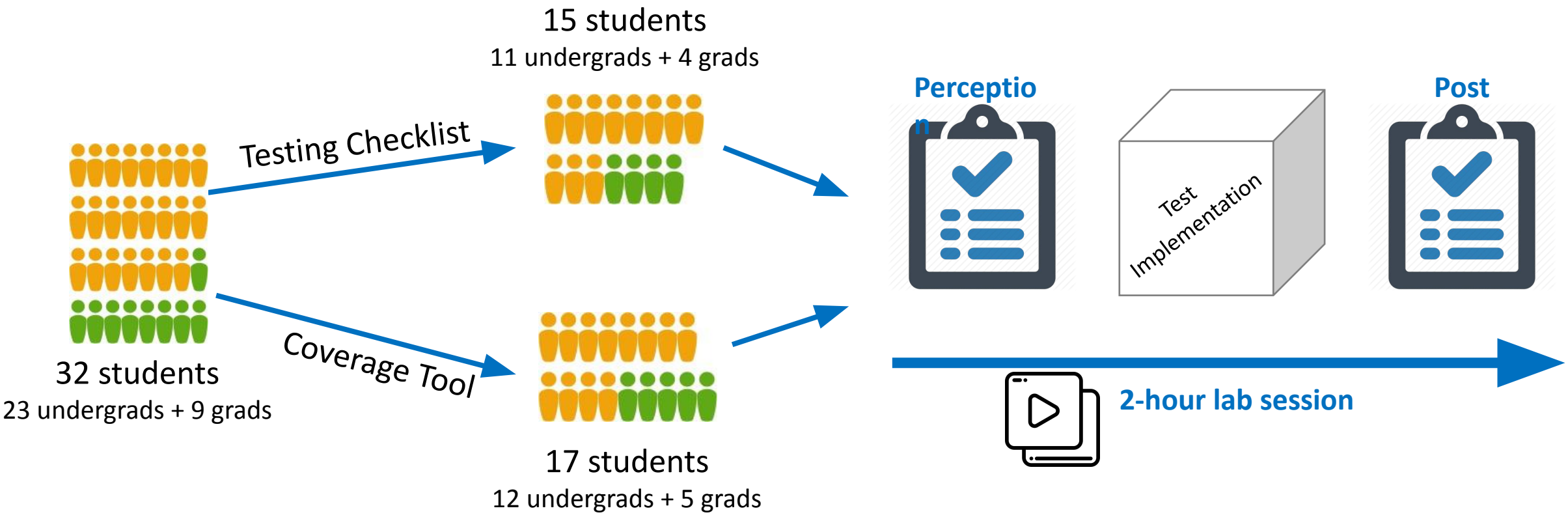
The test suite *should*:

- [ ] have at least one test for each requirement
- [ ] appropriately use the setup and teardown code (e.g., **@Before**, which runs before each **@Test**)
- [ ] contain a fault-revealing test for each bug in the code (i.e., a test that fails)
- [ ] For each requirement, contain test cases for:
  - [ ] Valid inputs
  - [ ] Boundary cases
  - [ ] Invalid inputs
  - [ ] Expected exceptions

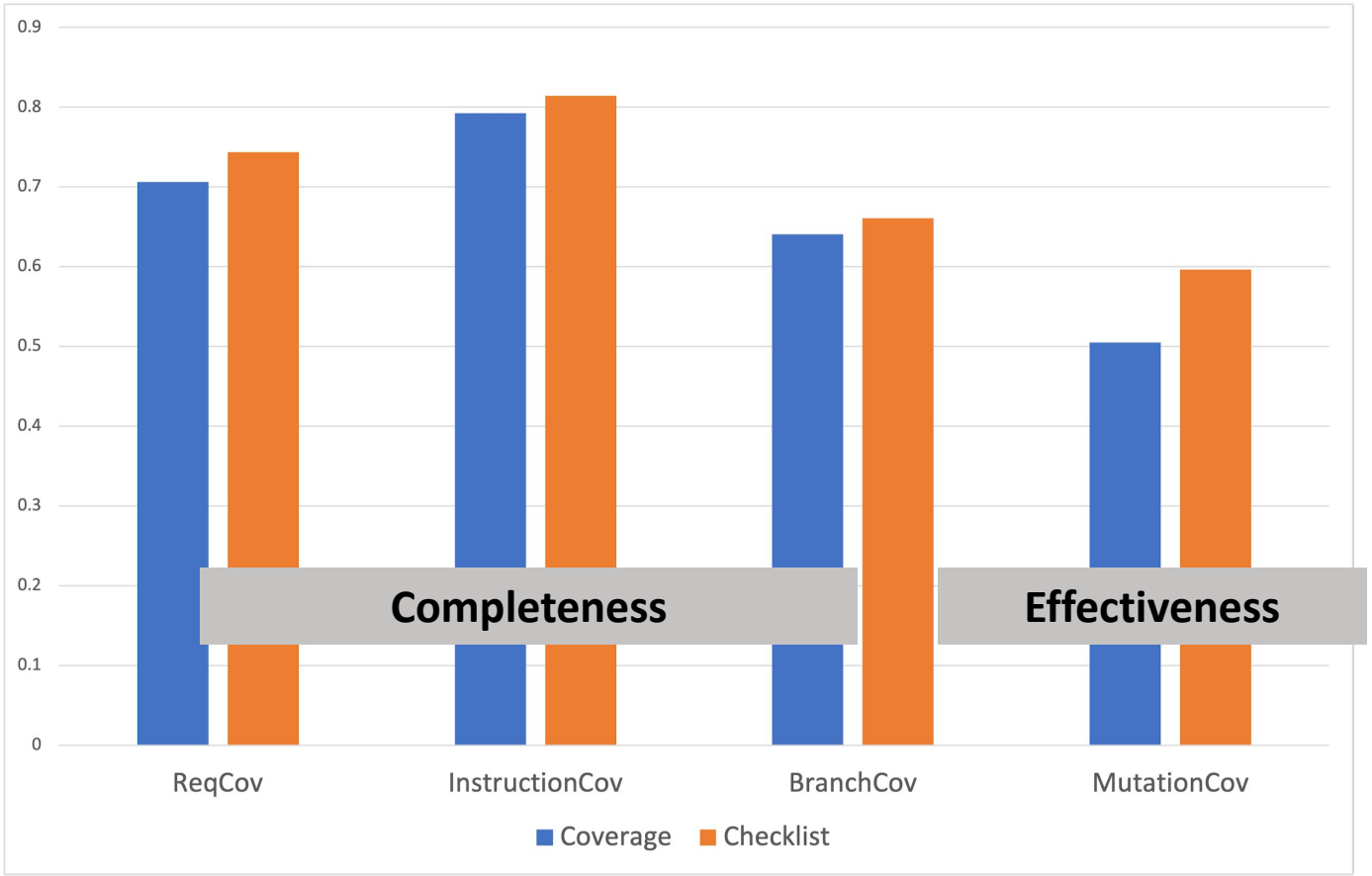
To improve the test suite, you *could*:

- [ ] measure code coverage using an appropriate tool, such as EclEmma (installation, tutorial). Inspect uncovered code and w
  appropriate.

Methodology

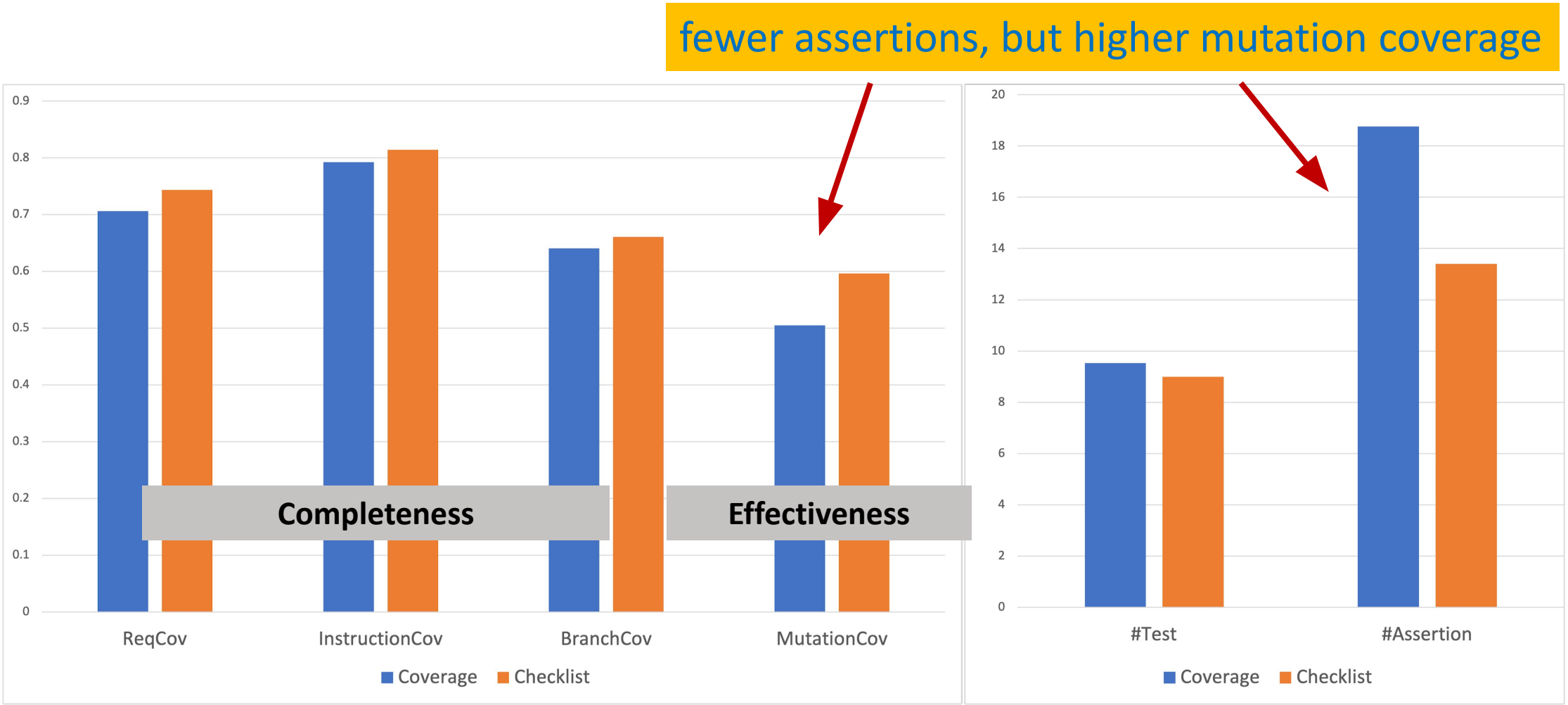


Checklists vs. Coverage Tools



Tool support does not need to be sophisticated to be effective!

Checklists vs. Coverage Tools



Looking Ahead…

generate tests for a bowling scorekeeper

1. Test: Calculate total score
   - Input: Player A rolls [5, 3, 8, 2, 10, 4, 6, 10, 9, 0, 5, 5, 10, 8, 1]
   - Expected Output: Total score = 142
2. Test: Calculate total score with spare bonus
   - Input: Player A rolls [5, 3, 8, 2, 10, 4, 6, 10, 9, 1, 5, 5, 10, 8, 1]
   - Expected Output: Total score = 152
3. Test: Calculate total score with strike bonus
   - Input: Player A rolls [5, 3, 8, 2, 10, 4, 6, 10, 10, 10, 5, 5, 10, 8, 1]
   - Expected Output: Total score = 201
4. Test: Calculate total score with consecutive strikes (turkey)
   - Input: Player A rolls [10, 10, 10, 5, 3, 8, 2, 10, 4, 6, 10, 9, 1, 5, 5, 10]

- Are the tests correct?
- Are the tests complete?
- What happens if the requirements change?

FIND IT THINK IT TEST IT FIX IT

Automated Program Repair

- Innovations:
  - Program Repair fueled by semantic search creates patches that are less prone to over-fitting [ASE 2015]
  - Semantic search is promising for producing high-quality real-world defect repairs [TSE 2021]

SHF: Medium: Collaborative Research: Semi and Fully Automated Program Repair and Synthesis via Semantic Code Search $387k (co-PI, $1.2m total) [completed]

SHF: EAGER: Collaborative Research: Demonstrating the Feasibility of Automatic Program Repair Guided by Code Search. $87k (co-PI, $287; total) [completed]

Automated Program Repair

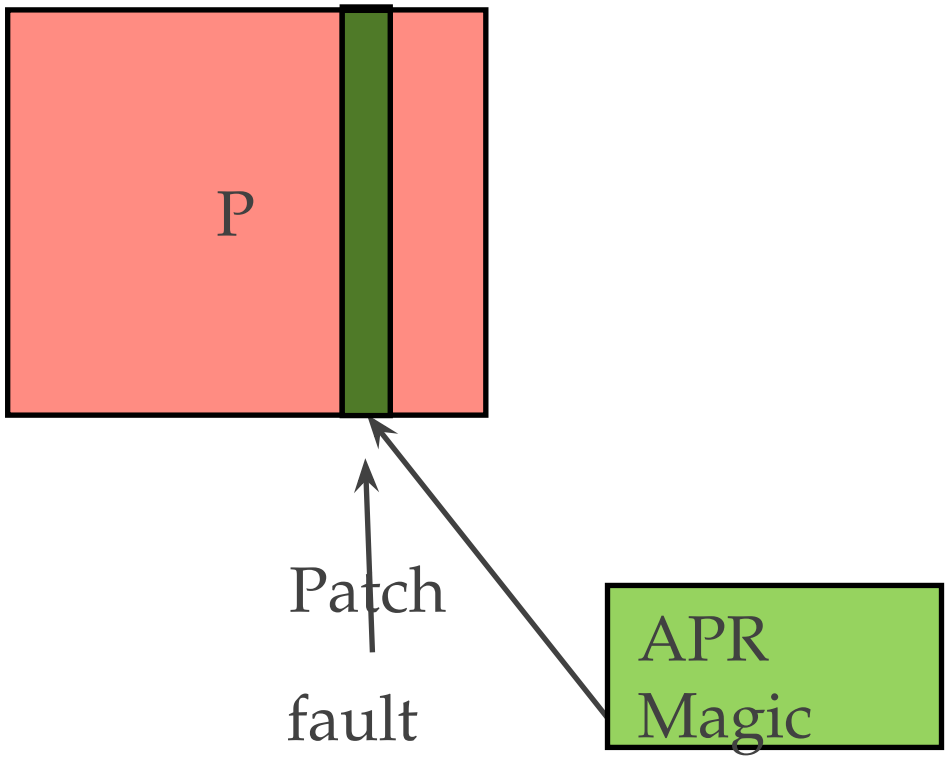
- Innovations:
  - **Program Repair fueled by semantic search creates patches that are less prone to over-fitting [ASE 2015]**
  - **Semantic search is promising for producing high-quality real-world defect repairs [TSE 2021]**

SHF: Medium: Collaborative Research: Semi and Fully Automated Program Repair and Synthesis via Semantic Code Search $387k (co-PI, $1.2m total) [completed]

SHF: EAGER: Collaborative Research: Demonstrating the Feasibility of Automatic Program Repair Guided by Code Search. $87k (co-PI, $287; total) [completed]

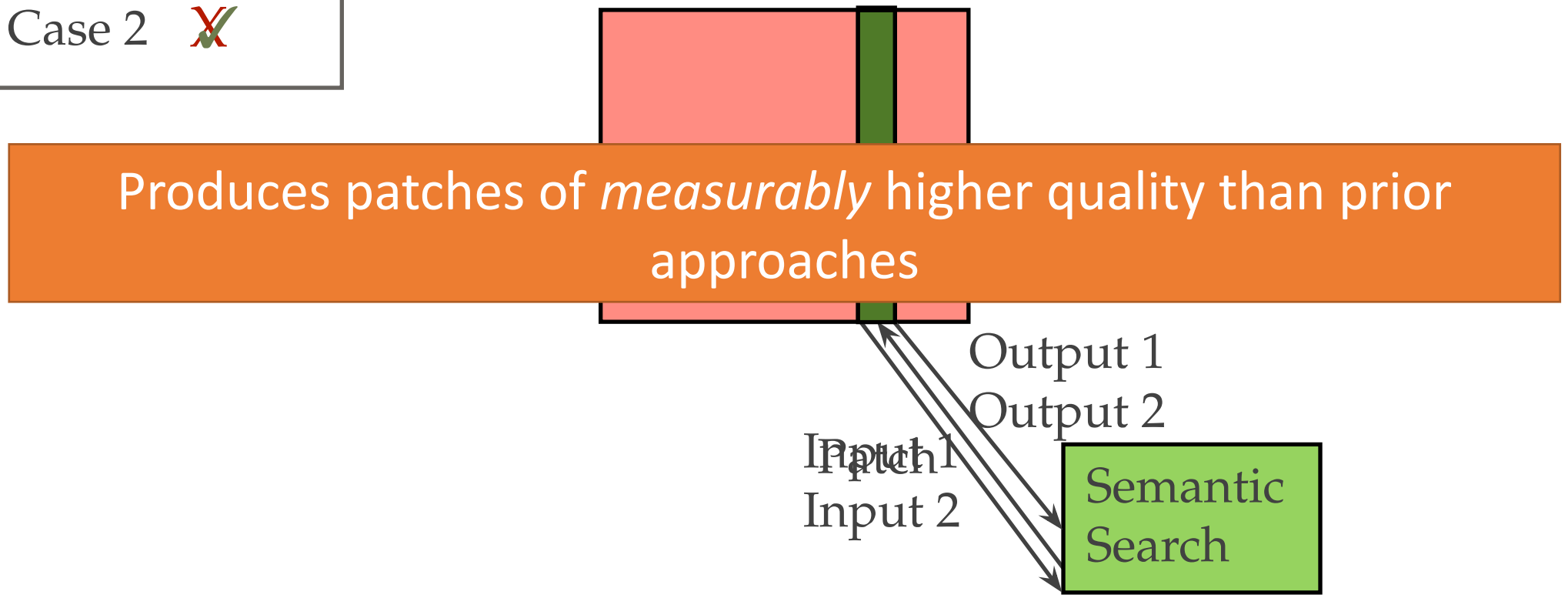
Automated Program Repair

Test Suite	
Test Case 1	✓
Test Case 2	✗✓



My Automated Program Repair

Test Suite	
Test Case 1	✓
Test Case 2	✗✓



Can it patch real bugs?

Python bug #69223

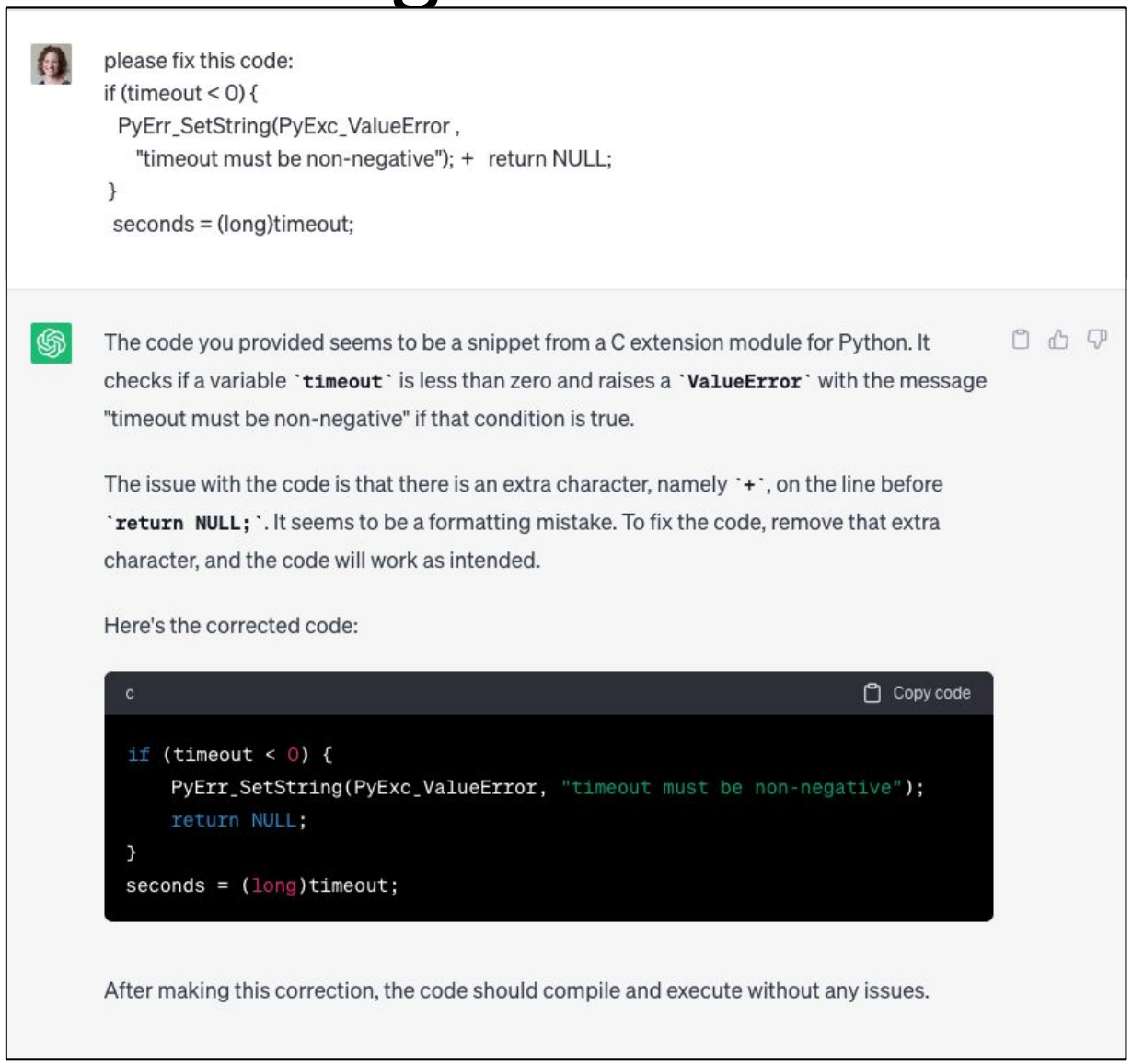
Developer Patch

```
  }
+ if (timeout < 0) {
+   PyErr_SetString(PyExc_ValueError ,
+       "timeout must be non-negative");
+   return NULL;
+ }
  seconds = (long)timeout;
```

SearchRepair Patch

```
if ( timeout < 0
    PyErr_SetString(PyExc_ValueError ,
       "read length must be positive");
    return NULL;
}
seconds = (long)timeout;
```

Looking Ahead…



FIND IT THINK IT TEST IT FIX IT

Big Results


Code search via analysis leads to high precision and has promise for a variety of applications


Program Repair via code search leads to patches that are higher quality


Comparative comprehension is challenging when behaviors are close but not exact


Testing via checklist is as effective as coverage tools for early students

Teamwork makes it happen.



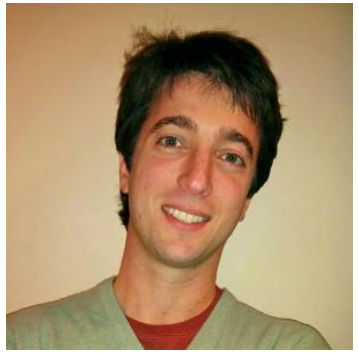
















…. And more!

Thanks!

ktstolee@ncsu.edu

Thanks!

ktstolee@ncsu.edu