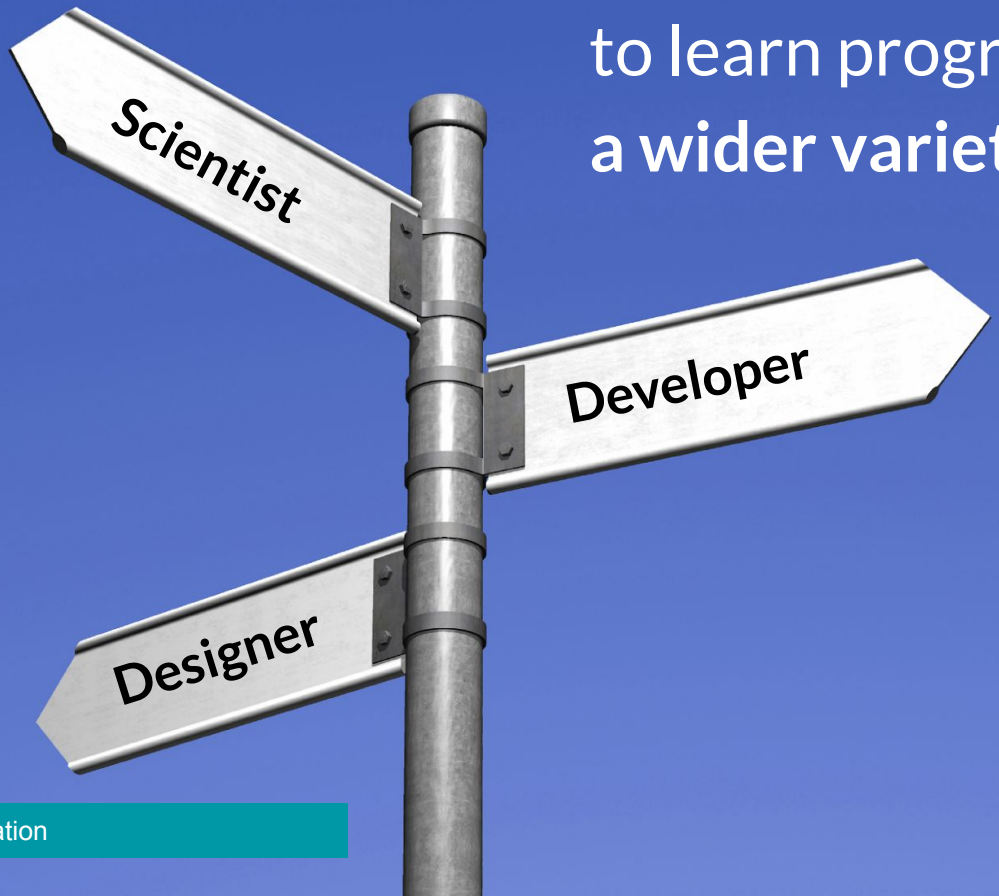
Purpose-first programming:

A programming learning approach for learners who care most about what code achieves

Kathryn Cunningham

Committee:
Mark Guzdial, CSE (co-chair)
Barbara Ericson, SI (co-chair)
Stephanie Teasley, SI
Chris Quintana, SoE (cognate)

Can we design new ways
to learn programming that consider
a wider variety of learners' goals?



A recent theory of programming instruction[1]

	semantics	template
read	**S1**: Given code, learner can determine the intermediate and final states of the code.	**S3**: Given code, learner can recognize a template and use it to understand what the code does.
write	**S2**: Given an unambiguous description, a learner can translate it to code.	**S4**: Given a problem description, a learner can devise a plan to use the template to solve the problem.

Skill #1:
Trace code execution
(low level)

Skill #3:
Describe code's purpose
(high level)

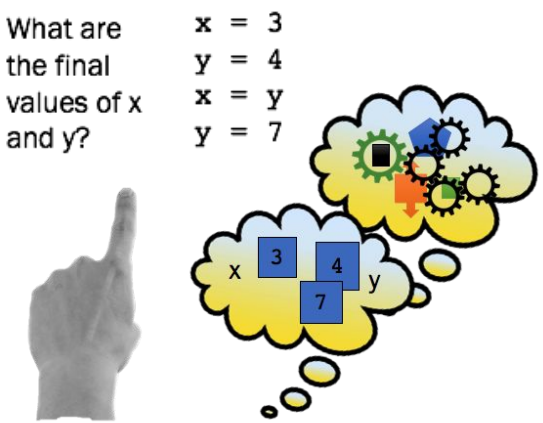
Skill #2:
Write code based on a description of a trace
(low level)

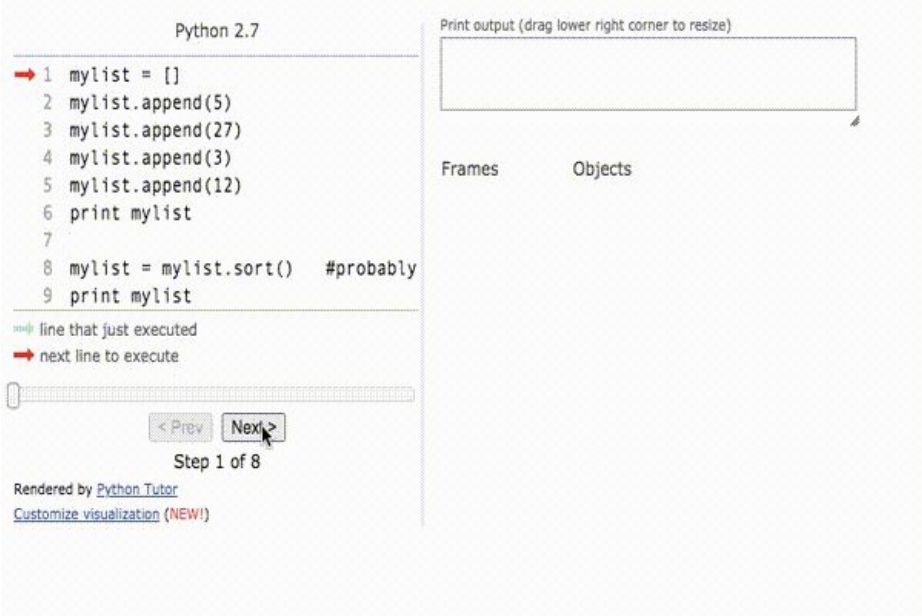
Skill #4:
Write code that achieves a goal
(high level)

[1] Xie et al. 2019.

What is code tracing?

Code tracing := Simulating program execution [1]





[1] Sorva. 2013.

Tracing activities often involve contextless problems with no “purpose”

Question 2.
Consider the following code fragment:

```
int[] x1 = {1, 2, 4, 7};
int[] x2 = {1, 2, 5, 7};
int i1 = x1.length-1;
int i2 = x2.length-1;
int count = 0;
while ((i1 > 0 ) && (i2 > 0 ))
{
     if ( x1[i1] == x2[i2] )
     {
          ++count;
          --i1;
          --i2;
     }
     else if (x1[i1] < x2[i2])
     {
          --i2;
     }
     else
     {    // x1[i1] > x2[i2]
          --i1;
     }
}
```

After the above while loop finishes, “count” contains what value?

a) 3
b) 2
c) 1
d) 0

Question 5.
Consider the following code fragment:

```
int[ ] x = {0, 1, 2, 3};
int temp;
int i = 0;
int j = x.length-1;

while (i < j)
{
     temp = x[i];
     x[i] = x[j];
     x[j] = 2*temp;
     i++;
     j--;
}
```

After this code is executed, array “x” contains the values:

a) {3, 2, 2, 0}
b) {0, 1, 2, 3}
c) {3, 2, 1, 0}
d) {0, 2, 4, 6}
e) {6, 4, 2, 0}

From Lister et al. 2004.

What happens if you don't think tracing is a fit for your needs?

	semantics	template
read	**S1**: Given code, learner can determine the intermediate and final states of the code.	**S3**: Given code, learner can recognize a template and use it to understand what the code does.
write	**S2**: Given an unambiguous description, a learner can translate it to code.	**S4**: Given a problem description, a learner can devise a plan to use the template to solve the problem.

Skill #1:
Trace code execution
(low level)

Skill #3:
Describe code's purpose
(high level)

Skill #2:
Write code based on a description of a trace
(low level)

Skill #4:
Write code that achieves a goal
(high level)

[1] Xie et al. 2019.

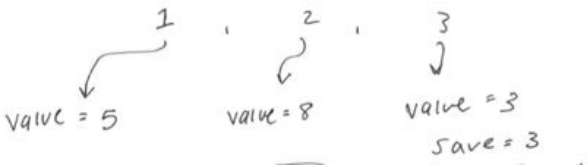
In my thesis, I propose an **alternative way to learn about code that prioritizes code's purpose**, so learners like the manager and the data scientist can understand and write meaningful code quickly.

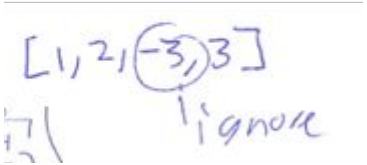
Contributions of my thesis

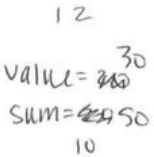
- **Analyzed existing programming skills hierarchies** (Ch 2)
- Understood affective and cognitive **reasons why some learners avoid code tracing** (Ch 3, 4)
- Designed **purpose-first programming**, an alternative learning approach that scaffolds code understanding in **larger, more meaningful pieces** (Ch 5)
- Developed a **proof-of-concept** purpose-first programming curriculum and evaluated how novices use it and how the approach **motivates** novice learners (Ch 5)

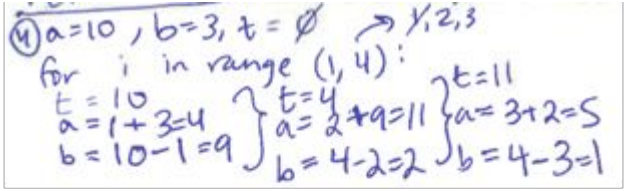
Part 1:
Why don’t novice programmers trace code?

I understand novice programmers’ code tracing activity by analyzing their *sketched traces*





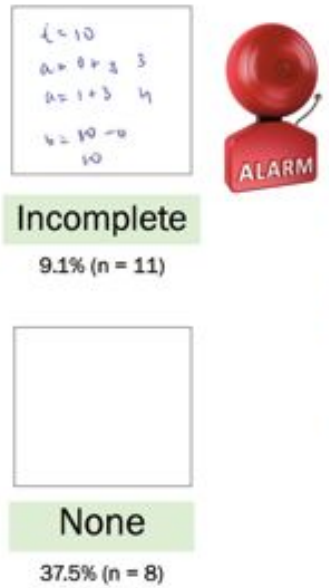




I found that performing tracing doesn't *always* lead to success...

I found an **oddity**:

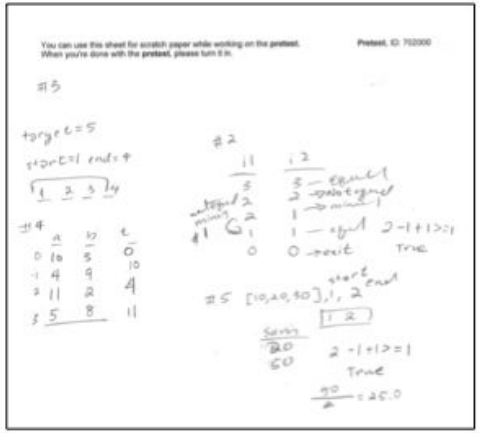
Those who started to trace but didn't finish did **very poorly**, even worse than those who didn't sketch at all!



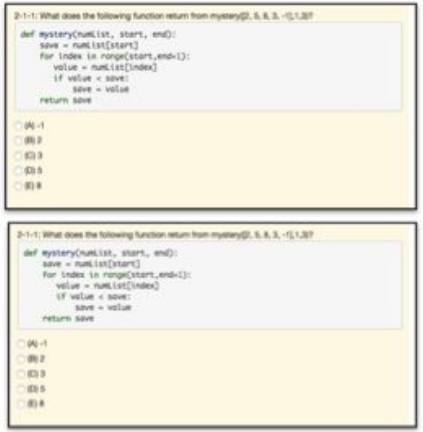
Published at ICER 2017

Why would novices **not complete a trace**? Not trace at all?

Retrospective artifact walkthroughs



Student sketches from exam



Problems from exam

(n=13)

I asked learners in a CS1 course about their choices to trace or not trace code.

Published at ITiCSE 2019

Search for goals and patterns is primary

Pattern matching → Interrupted tracing → Close tracing

"Once you look at the code and figure out what it's doing, then it's like, okay, I can compute an average without writing it down, especially if it's only two values."

"After I sort of got the hang of it, I just started to skip through writing it."

"You can also see the point where I realized like... and it clicked."

"When I saw it I wasn't like 'Oh, I think I have a hunch that this code does this.' I'm going to need to work through it."

Published at ITiCSE 2019

Takeaways

- **Tracing has a high cognitive load**. It's time-consuming and difficult, even for students in a for-majors intro course.
- Novice programmers **prefer to think with higher-level strategies**, rather than use fine-grained code tracing. They may need support to identify goals and patterns successfully.

Published at ITiCSE 2019

How might tracing be helpful when determining the purpose of code?

Problems from past "Explain in plain English" studies

Problem 1 - Tracing

Thursday, September 5, 2019 4:26 PM

Describe the purpose of the following code. Do not give a line-by-line description of what the code does. Instead, describe the code's purpose in one sentence.

```
def enigma(nums): # nums is a list of numbers
    for index in range(len(nums) - 1):
        if nums[index] > nums[index + 1]:
            return False
    return True
```

Trace through the following example until you figure it out. Please show your work.

```
enigma([3,4,2,5])
```

3 > 4 True

4 > 2 False

2 > 5 False

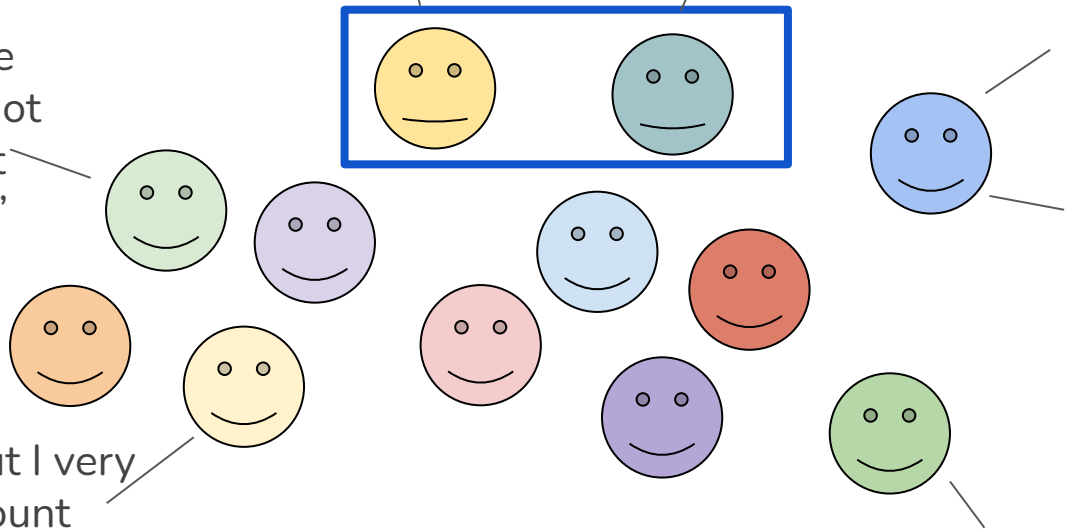
A participant's trace

But!

Participants had some unsolicited feedback about the <u>value</u> of the tracing tasks.

Published at ICLS 2020

Novice programmers, primarily from an Information major (n = 12)



"I hated this kind of work, honestly"

"I found that, like, the least helpful thing in my programming class was reading code"

"In [my course], they definitely had us draw a grid [to trace], and I definitely didn't do that"

"I'm like, objective oriented. So it's not necessarily about each line of code"

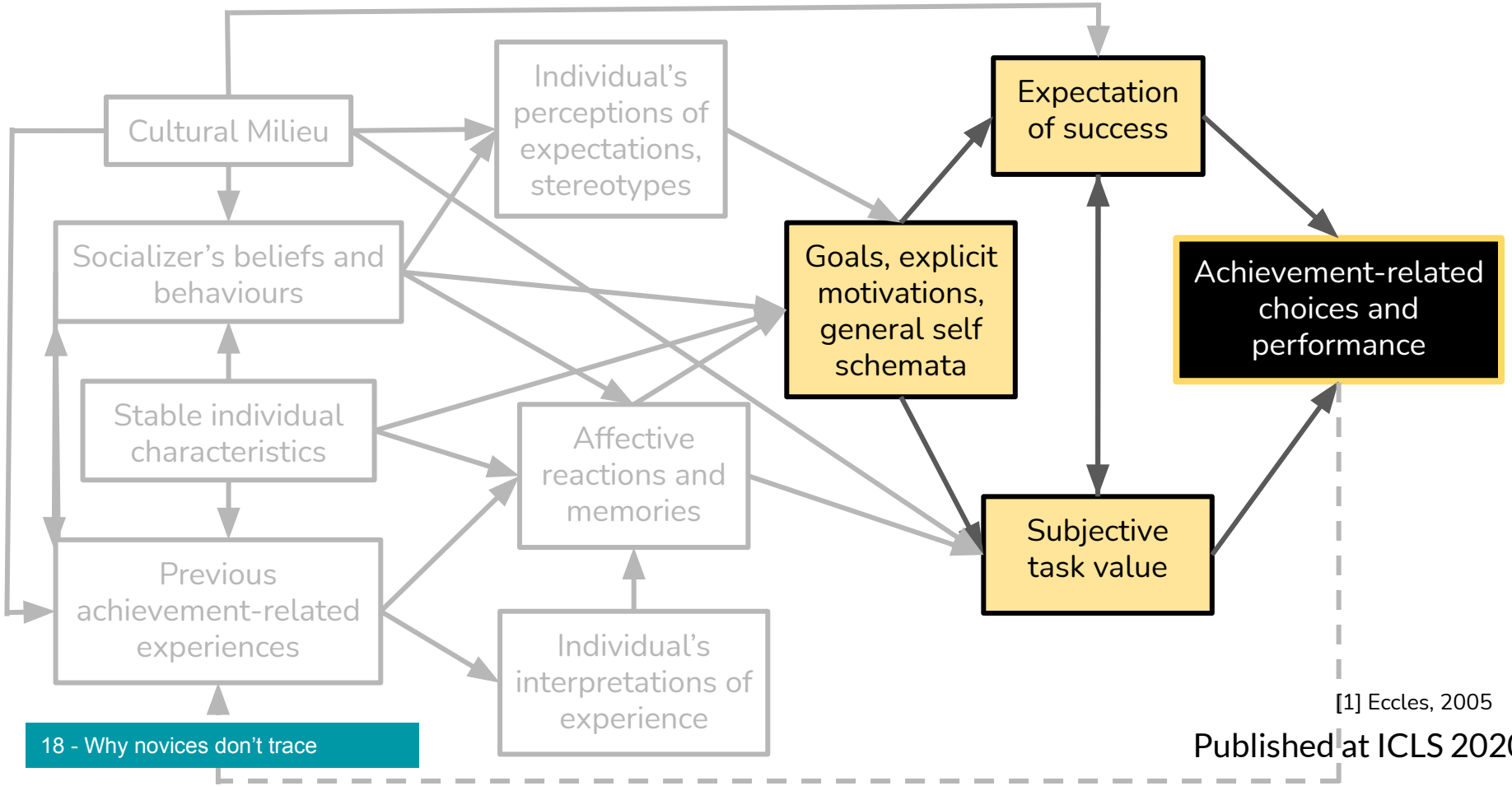
"You don't do much coding by hand, obviously"

"This is bad but I very infrequently count through the iterations"

"I should know this"

Published at ICLS 2020

Eccles Expectancy-Value Model of Achievement Choice[1]



[1] Eccles, 2005

Published at ICLS 2020

Charles
(undergraduate, Information major)

Goals and General Self-Schemata

"Yeah, I mean, it's just like, it makes me think like a computer. But I'm not a computer. And it's not that I can't work with the computer in tandem. I mean, that's why we have the computers."

I'm not a computer

Expectation for Success

"It seems like no matter how much I do it, I don't understand these things."

"If we were to do ten of these, I'm sure each one of them I would look over [overlook] a small component of the code."

I can't think like a computer does

Subjective Task Value

"It always seems like a really strange way to try and teach someone code when you could just execute it and see where it goes."

"Nowhere outside, I feel like, a college setting is ever gonna ask you that question."

The computer executes code, not me

Achievement-Related Choice

I don't trace

Published at ICLS 2020

Charles
(undergraduate, Information major)

Goals and General Self-Schemata

"Yeah, I mean, it's just like, it makes me think like a computer. But I'm not a computer. And it's not that I can't work with the computer in tandem. I mean, that's why we have the computers."

I'm not a computer

Expectation for Success

"It seems like no matter how much I do it, I don't understand these things."

"If we were to do ten of these, I'm sure each one of them I would look over [overlook] a small component of the code."

I can't think like a computer does

Subjective Task Value

"It always seems like a really strange way to try and teach someone code when you could just execute it and see where it goes."

"Nowhere outside, I feel like, a college setting is ever gonna ask you that question."

The computer executes code, not me

Achievement-Related Choice

I don't trace

Published at ICLS 2020

Charles
(undergraduate, Information major)

Goals and General Self-Schemata

"Yeah, I mean, it's just like, it makes me think like a computer. But I'm not a computer. And it's not that I can't work with the computer in tandem. I mean, that's why we have the computers."

I'm not a computer

Expectation for Success

"It seems like no matter how much I do it, I don't understand these things."

"If we were to do ten of these, I'm sure each one of them I would look over [overlook] a small component of the code."

I can't think like a computer does

Subjective Task Value

"It always seems like a really strange way to try and teach someone code when you could just execute it and see where it goes."

"Nowhere outside, I feel like, a college setting is ever gonna ask you that question."

The computer executes code, not me

Achievement-Related Choice

I don't trace

Published at ICLS 2020

Charles
(undergraduate, Information major)

Goals and General Self-Schemata

"Yeah, I mean, it's just like, it makes me think like a computer. But I'm not a computer. And it's not that I can't work with the computer in tandem. I mean, that's why we have the computers."

I'm not a computer

Expectation for Success

"It seems like no matter how much I do it, I don't understand these things."

"If we were to do ten of these, I'm sure each one of them I would look over [overlook] a small component of the code."

I can't think like a computer does

Subjective Task Value

"It always seems like a really strange way to try and teach someone code when you could just execute it and see where it goes."

"Nowhere outside, I feel like, a college setting is ever gonna ask you that question."

The computer executes code, not me

Achievement-Related Choice

I don't trace

Published at ICLS 2020

Luke (graduate student in Information)

Goals and General Self-Schemata

"If I was wanting to become a programmer, then perhaps it would be more interesting to me."

"I mean the purpose itself is not to code something....The code is just a means to an end, of creating an interaction, or creating a product or creating whatever else, right?"

I'm not a programmer

Expectations for Success

"It's literally like I learn it to where I need it, then I don't care to keep it."

"So like it's just to see what each part is doing, that's where I would get confused."

I try to forget programming details

Subjective Task Value

"This sort of stuff was blatantly to write Python. There's times I'm just like 'Why would I ever use this?'"

"It serves just sort of like a crossword puzzle that's not fun."

Tracing is only about learning a language

Achievement-Related Choice

I don't trace

Published at ICLS 2020

Luke (graduate student in Information)

Goals and General Self-Schemata

"If I was wanting to become a programmer, then perhaps it would be more interesting to me."

"I mean the purpose itself is not to code something....The code is just a means to an end, of creating an interaction, or creating a product or creating whatever else, right?"

I'm not a programmer

Expectations for Success

"It's literally like I learn it to where I need it, then I don't care to keep it."

"So like it's just to see what each part is doing, that's where I would get confused."

I try to forget programming details

Subjective Task Value

"This sort of stuff was blatantly to write Python. There's times I'm just like 'Why would I ever use this?'"

"It serves just sort of like a crossword puzzle that's not fun."

Tracing is only about learning a language

Achievement-Related Choice

I don't trace

Published at ICLS 2020

Luke (graduate student in Information)

Goals and General Self-Schemata

"If I was wanting to become a programmer, then perhaps it would be more interesting to me."

"I mean the purpose itself is not to code something....The code is just a means to an end, of creating an interaction, or creating a product or creating whatever else, right?"

I'm not a programmer

Expectations for Success

"It's literally like I learn it to where I need it, then I don't care to keep it."

"So like it's just to see what each part is doing, that's where I would get confused."

I try to forget programming details

Subjective Task Value

"This sort of stuff was blatantly to write Python. There's times I'm just like 'Why would I ever use this?'"

"It serves just sort of like a crossword puzzle that's not fun."

Tracing is only about learning a language

Achievement-Related Choice

I don't trace

Published at ICLS 2020

Luke (graduate student in Information)

Goals and General Self-Schemata

"If I was wanting to become a programmer, then perhaps it would be more interesting to me."

"I mean the purpose itself is not to code something....The code is just a means to an end, of creating an interaction, or creating a product or creating whatever else, right?"

I'm not a programmer

Expectations for Success

"It's literally like I learn it to where I need it, then I don't care to keep it."

"So like it's just to see what each part is doing, that's where I would get confused."

I try to forget programming details

Subjective Task Value

"This sort of stuff was blatantly to write Python. There's times I'm just like 'Why would I ever use this?'"

"It serves just sort of like a crossword puzzle that's not fun."

Tracing is only about learning a language

Achievement-Related Choice

I don't trace

Published at ICLS 2020

Takeaways

It's possible to connect self-beliefs to reasons to not trace code

- Contribute to low expectancy of success and low task value

Learners with the self-beliefs "I'm not a computer" and "I'm not a programmer" valued the *creation of code* more than understanding *how code works*.

These novices could benefit from an **alternative pathway** to build expertise in programming that doesn't prioritize code tracing.

- Purpose-oriented, in an authentic application context

Published at ICLS 2020

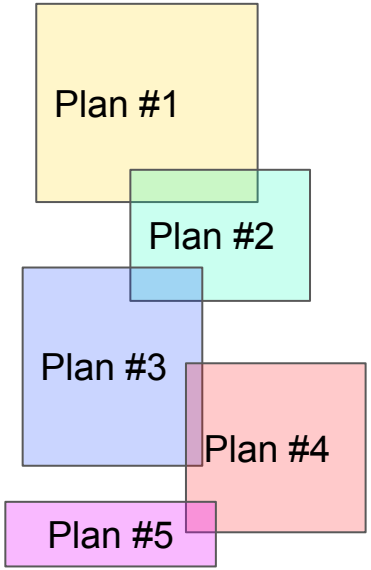
Part 2:
Designing to meet the need

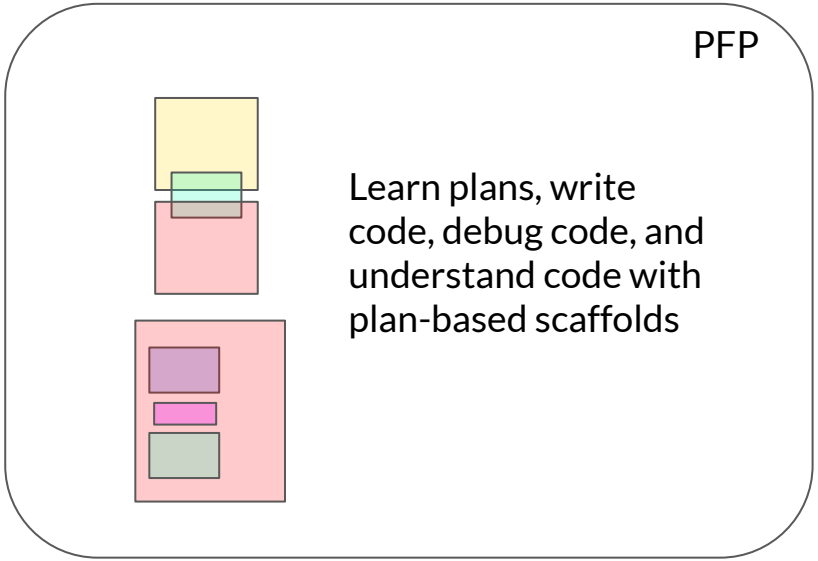
Emphasizing code's ***purpose*** rather than programming language semantics

Purpose-first programming

Using programming plans[1], chunks of code associated with a goal,
within a particular domain of interest,
to scaffold code understanding, writing, and debugging.

For a small set of plans authentic to a domain of interest (e.g. web scraping)





[1] Soloway & Ehrlich 1984

Plan-based scaffolds

Plan 2: Get a soup from multiple webpages

```
# Load libraries for web scraping
from bs4 import BeautifulSoup
import requests
# Get a soup from multiple URLs
base_url = [                                    ]
endings = [                                     ]
for ending in endings:
    url = base_url + ending
    r = requests.get(url)
    soup = BeautifulSoup(r.content)
```

Put the parts of the URL that are unique here.
E.g. `['steve-oney', 'paul-resnick']`

Plan 3: Get a single tag of a certain type

```
# Get first tag of a certain type from the soup
tag = soup.find([                               ])
# Get info from the tag
info = tag.[      ]
```

Plan 5: Print the info

```
# Print info
print([      ])
```

If you want a tag's text, put `text`
If you want a tag's link, put `get('href')`

- Group code as part of a plan
- Associate each plan with a goal
- Add subgoals to each plan's code
- Highlight which parts of a plan can be changed ("slots"), and which should stay the same
- Describe how slots should be filled, using domain-specific concepts

Plan-based scaffolds

Plan 2: Get a soup from multiple webpages

```
# Load libraries for web scraping
from bs4 import BeautifulSoup
import requests
# Get a soup from multiple URLs
base_url = 
endings = 
for ending in endings:
    url = base_url + ending
    r = requests.get(url)
    soup = BeautifulSoup(r.content)
```

Put the parts of the URL that are unique here.
E.g. `['steve-oney', 'paul-resnick']`

Plan 3: Get a single tag of a certain type

```
# Get first tag of a certain type from the soup
tag = soup.find( )
# Get info from the tag
info = tag.
```

Plan 5: Print the info

```
# Print info
print( )
```

If you want a tag's text, put `text`
If you want a tag's link, put `get('href')`

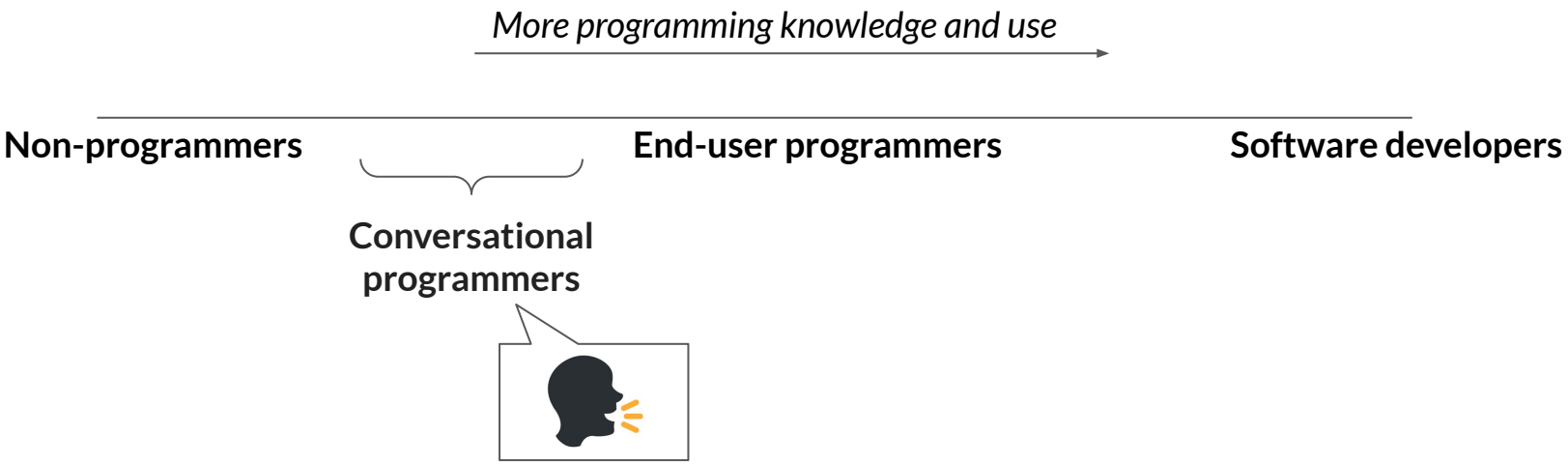
Write code by **choosing, ordering and tailoring plans**

Understand code by **reading goals and subgoals**

Debug by **focusing on plan slots and how they should be filled**

Understanding responses of novices to purpose-first scaffolding

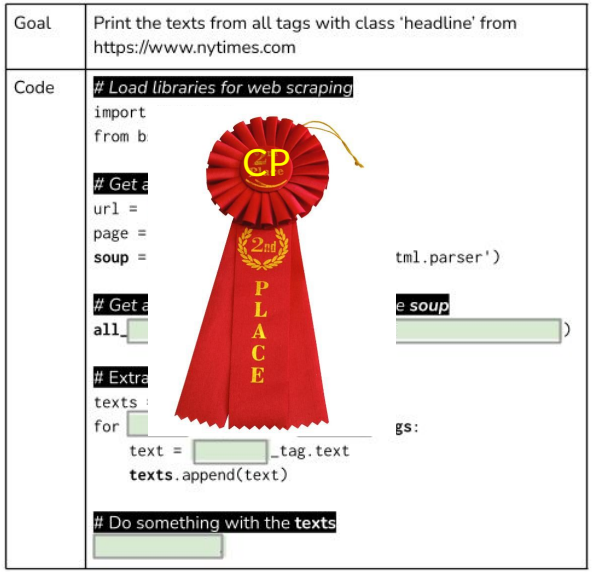
Preferences for learning tasks *differ* between **conversational programmers** and other novice programmers



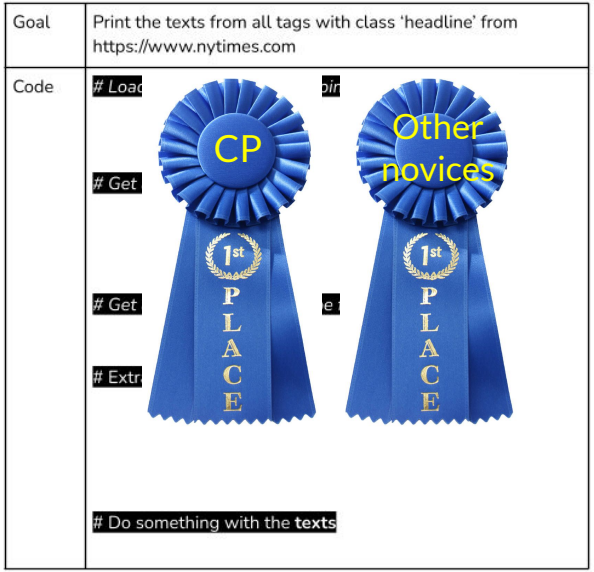
Survey study

"Rank these 3 activities as far as their usefulness in preparing you for your future"

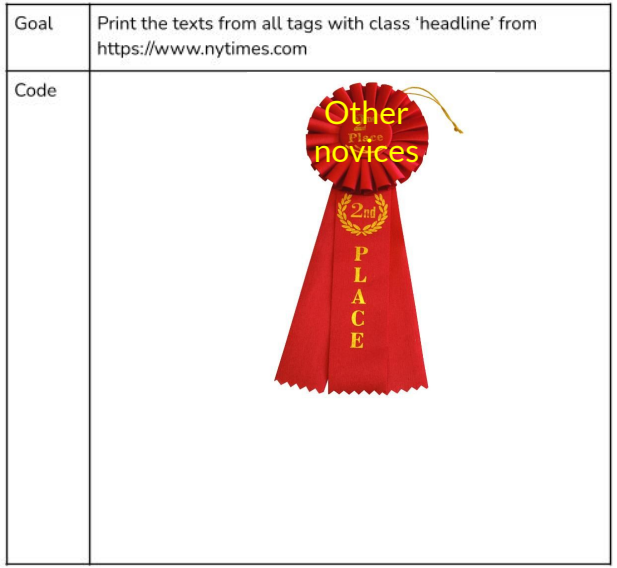
(n=33; 16 conversational programmers, 17 other novice programmers)



High scaffolding
(fill in code)



Medium scaffolding
(subgoal labels)



No scaffolding
(write code from scratch)

Developing
a “proof of concept” purpose-first programming curriculum

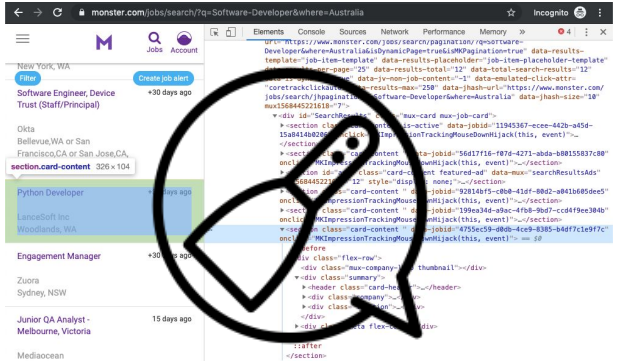
Developing the “proof-of-concept” curriculum

1. Choosing a domain
2. Identifying authentic plans
3. Develop worked examples + practice
    a. Innovative technical supports
4. Create code writing, debugging, and explanation problems
    a. Innovative technical supports

Developing the “proof-of-concept” curriculum

1. Choosing a domain

Web scraping with BeautifulSoup



Reason #1: Disciplinary authenticity:
In focus groups, I found that Information students found web scraping to be valuable to their future goals.

Reason #2: Semantically complex but “planfully” simple:
Use of libraries and “complex” language semantics like objects. A few patterns are repeated often.

Developing the “proof-of-concept” curriculum

1. Choosing a domain
2. **Identifying authentic plans**

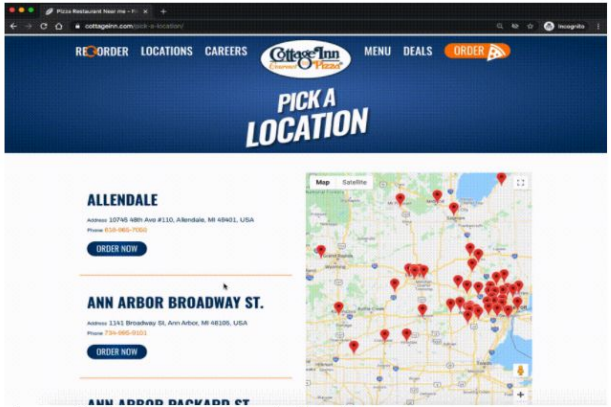
Scraped files from Github that use BeautifulSoup for web scraping → *Identified 5 common patterns from subset of 50 files* → *Annotated patterns with slots, goals and subgoals* → *Two experts who use BeautifulSoup professionally review plans*

Developing the “proof-of-concept” curriculum

1. Choosing a domain
2. I
3. D - pr
a po

Scrape all the Cottage Inn Pizza locations

Let’s say that you want to make a list of all the Cottage Inn Pizza locations. When you go to their website, it turns out that there are a *lot* of locations.



If only you could write a little Python to easily collect them all…

It turns out that you can! Run the code below to see what it collects.

Run

Original - 1 of 1

```
#Get the webpage
# Load libraries for web scraping
from bs4 import BeautifulSoup
import requests
# Get a soup from a URL
url = 'https://web.archive.org/web/20200427175705/https://cottageinn.com/pick-a-lo
r = requests.get(url)
soup = BeautifulSoup(r.content, 'html.parser')

#Extract info from the page
# Get all tags of a certain type from the soup
tags = soup.find_all('h3')
# Collect info from the tags
collect_info = []
for tag in tags:
```

```
['Allendale', 'Ann Arbor Broadway St.', 'Ann Arbor Packard St.', 'Ann Arbor Stadium Blvd.',
```

This code probably seems a bit complicated. In this ebook, we will break down web scraping into a few common "plans". This example is made up of three plans. Click on each of them to learn more.

Plan 1: Get a soup from a URL

```
# Load libraries for web scraping
from bs4 import BeautifulSoup
import requests
# Get a soup from a URL
url = 'https://pizzatown.com/pick-a-location/'
r = requests.get(url)
soup = BeautifulSoup(r.content, 'html.parser')
```

Plan 3: Get info from all tags of a certain type

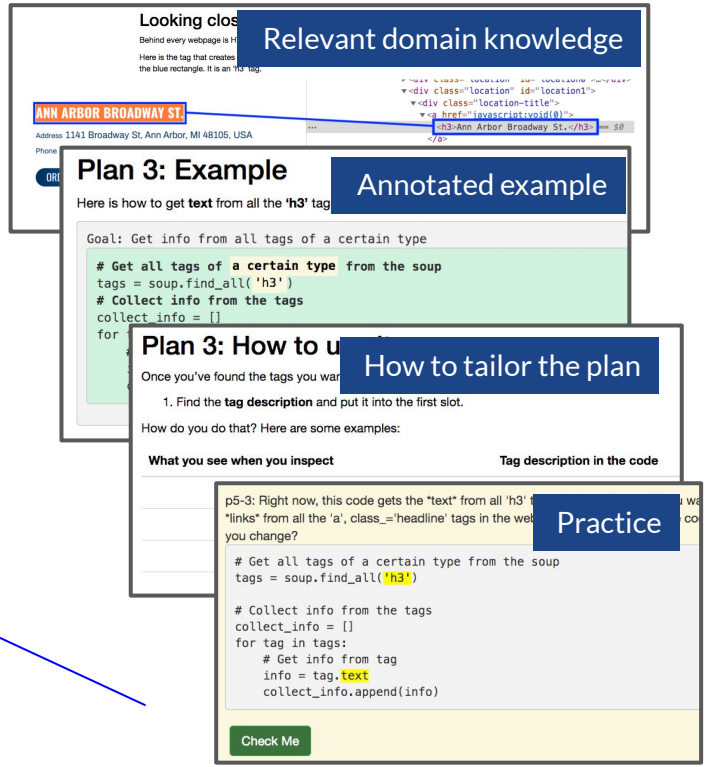
```
# Get all tags of a certain type from the soup
tags = soup.find_all('h3')
# Collect info from the tags
collect_info = []
for tag in tags:
    # Get info from tag
    info = tag.text
    collect_info.append(info)
```

Plan 5: Print the info

```
# Print the info
print(collect_info)
```

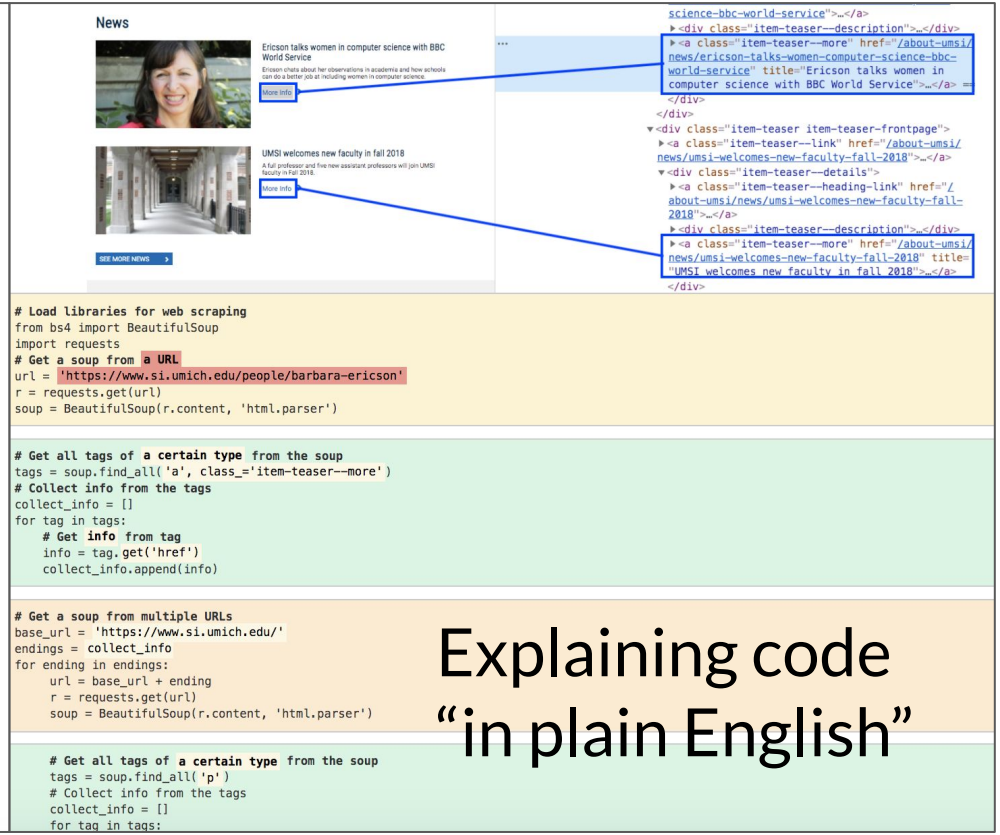
link

return



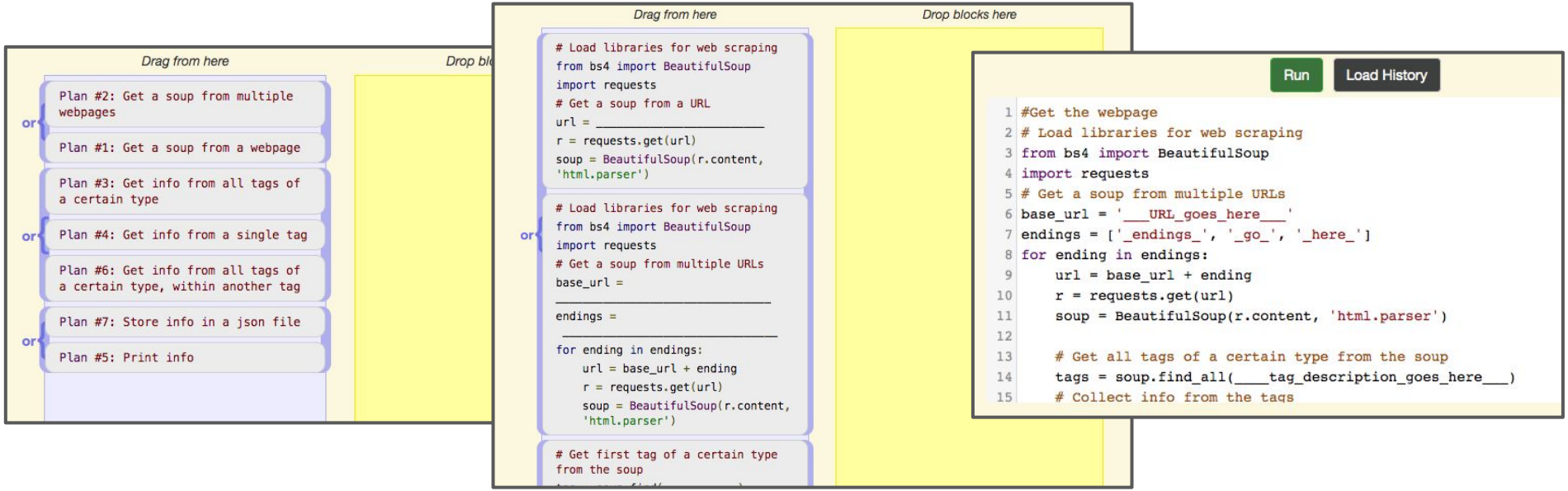
Developing the “proof-of-concept” curriculum

Debugging code



Explaining code “in plain English”

Writing code



#1: Order plan goals → ***#2: Order plan code*** → ***#3: Fill plan slots***

Evaluating
the “proof of concept” purpose-first programming curriculum

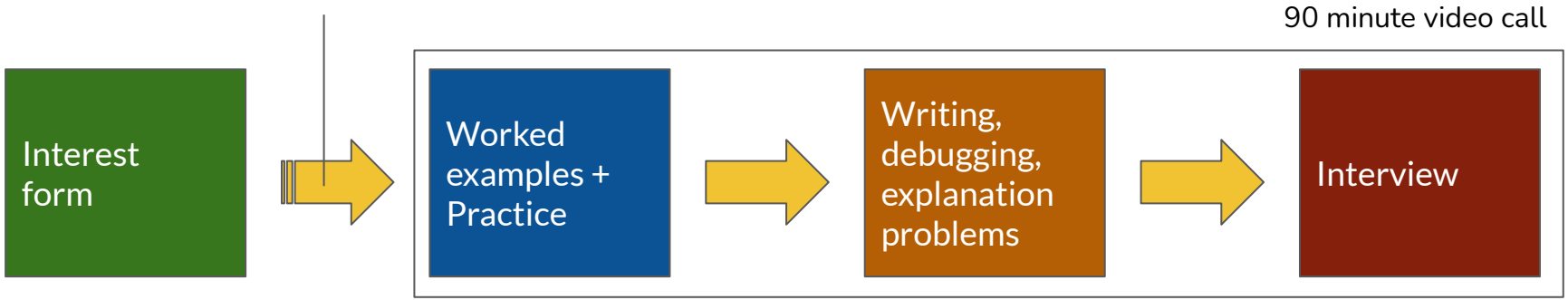
Research questions

1. Can novice programmers **read, write, and debug** complex code using the scaffolds provided by purpose-first programming?
    a. How do they complete these tasks?
2. Is purpose-first programming **motivating** for learners who are more likely to reject code tracing?
    a. How do novice programmers describe their *expectancy of success* and *subjective task value* for purpose-first programming activities?

Study design

Recruited 9 participants (and 1 pilot) with

- No prior BeautifulSoup experience
- No advanced programming experience
- Interest in web scraping
- Lower expectancy or value for code tracing



21 students planning to take a non-majors programming course completed interest form.

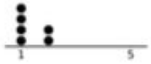
Participant completed instructional content, with assistance if needed.

Participant attempted novel problems that combined plans in new ways. Participant performed concurrent thinkaloud.

I performed a semi-structured interview focusing on aspects of expectancy and value for purpose-first programming.

*RQ1: Participants **could complete** scaffolded problems*

After 31 minutes of instructional time

Activity	Succeeded without researcher hint	Mean time to completion (min)	Frequency distribution of number of attempts
Writing 1 (order plan goals)	100%	1:31 (sd=0:40)	
Writing 2 (order plan code)	100%	2:00 (sd=1:09)	
Writing 3 (fill plan slots)	67%	7:58 (sd=2:13)	
Debugging	89%	2:19 (sd=1:36)	
Explanation	56 %	6:07 (sd=2:49)	

Involved 5 plans, while other problems involved 3

writing2-1: Choose which of the following plans you will use, and put them in the correct order.

Drag from here | *Drop blocks here*

```
# Get first tag of a certain type
from the soup
first_tag = soup.find(__________)
# Get all tags of a certain type
from the first tag
tags =
first_tag.find_all(___________)
# Collect info from the tags
collect_info = []
for tag in tags:
    info = tag.___________
    collect_info.append(info)
```

or

```
# Get all tags of a certain type
from the soup
tags = soup.find_all(__________)
# Collect info from the tags
collect_info = []
for tag in tags:
    info = tag.___________
    collect_info.append(info)
```

```
# Get first tag of a certain type
from the soup
tag = soup.find(__________)
# Get info from the tag
info = tag.___________
```

*"Okay. Um... First we want to start with the **getting tags from the soup**. Choose between these three. **Get first tag of certain type from soup**, **get all tags of a certain type**, **get first tag of a certain type**. I think it's this one, we want **all tags**. Okay." -P2*

P2's code writing (order plan code)

Okay. So let's start with base_url, url goes here. So that's the part of the URL that doesn't really change.

```
# Load libraries for web scraping
from bs4 import BeautifulSoup
import requests
# Get a soup from multiple URLs
base_url = '___URL_goes_here___'
endings = ['_endings_', '_go_', '_here_']
for ending in endings:
    url = base_url + ending
    r = requests.get(url)
    soup = BeautifulSoup(r.content, 'html.parser')

    # Get all tags of a certain type from the soup
    tags = soup.find_all(____tag_description_goes_here___)
    # Collect info from the tags
    collect_info = []
    for tag in tags:
        # Get info from tag
        info = tag._____
        collect_info.appe

    # Print the info
    print(_____)
```

*`find_all`, tag description goes here. So it would be the div tag, div tag, comments. And I think **div** is also here, **div class**.*

*Okay, so this would be **text** because I'm trying to find a comment. So this would be `tag`.**text**. It's not the link so I wouldn't do the **href** thing.*

*So let me see what print is like. Oh I just put the **info**. `print`(**info**). Oh for one tag, okay. But it's multiple tags so it'll be `print`(**collect_info**). Right, okay.*

Domain-specific language

P9's code writing (fill plan slots)

RQ2:
*All participants said they **wanted to learn with plans in the future***

Why?

I performed deductive reflexive thematic analysis[1] to understand the reasoning behind this motivation

[1] Braun & Clarke. 2006.

Themes

Learners perceived purpose-first programming as having **low cognitive load**

Participants felt **success** and **enjoyment**, which came from **understanding** and **completing** problems

Participants felt that purpose-first programming was for **beginners** and those who need **extra help**

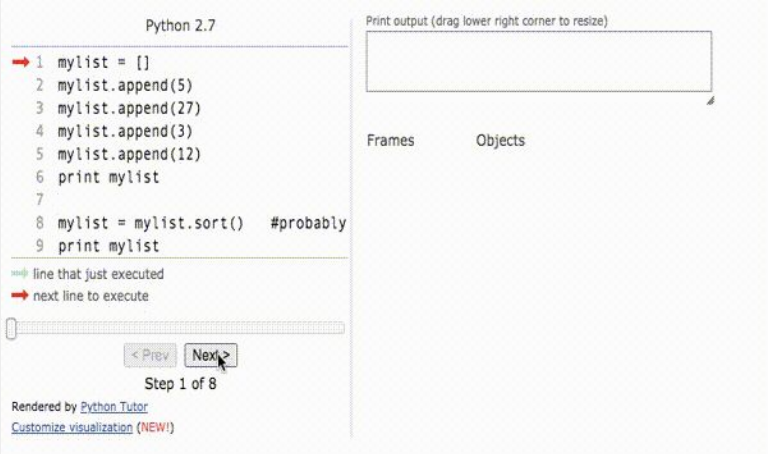
Participants believed purpose-first programming gave them **conceptual**, **high-level knowledge**

Participants found curricular content **realistic** and **applicable**

Some participants found code tracing visualizations difficult to understand and apply, and preferred plans.

*"I mean I guess in theory CodeLens [a PythonTutor variant] is useful but sometimes it gets so **confusing**." - P2.*

*"Just from my experience CodeLens was helpful **for people who knew programming really well**, so it kind of feels like the plans might be for more beginners." - P5*



Others found code tracing visualizations helpful for gaining deeper knowledge beyond plans

*"I think plans helped me in the very beginning, but once I understand the concept, debugging is what really helps me **further my learning** past the plans into more real life scenarios." - P9*

Conversational Programmers' Goals and General Self-Schemata

- I'm not good at programming
- I won't be a "hard coder" who writes code unassisted
- Learning to code is practical, but I just want the basics

Expectations for Success

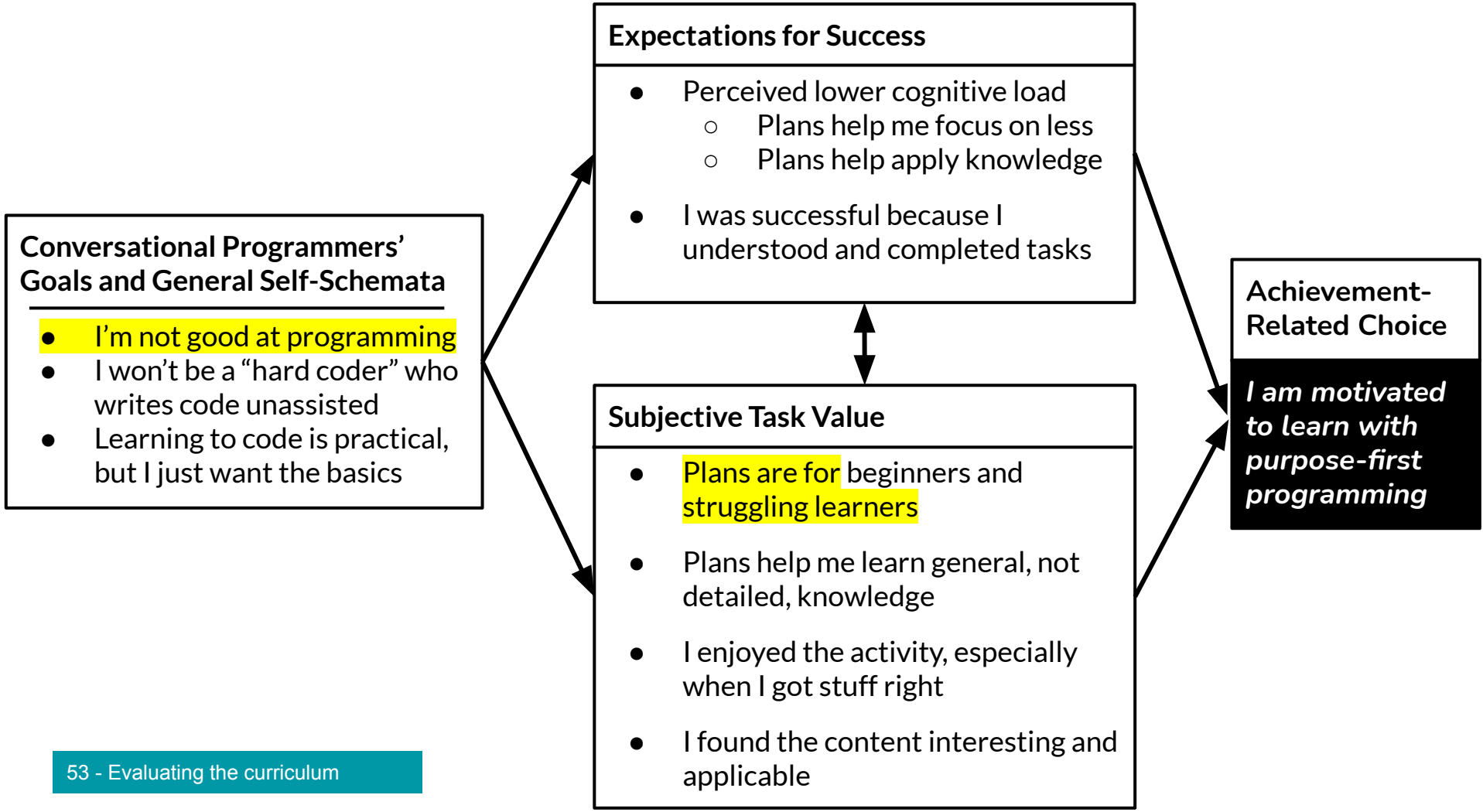
- Perceived lower cognitive load
  - Plans help me focus on less
  - Plans help apply knowledge
- I was successful because I understood and completed tasks

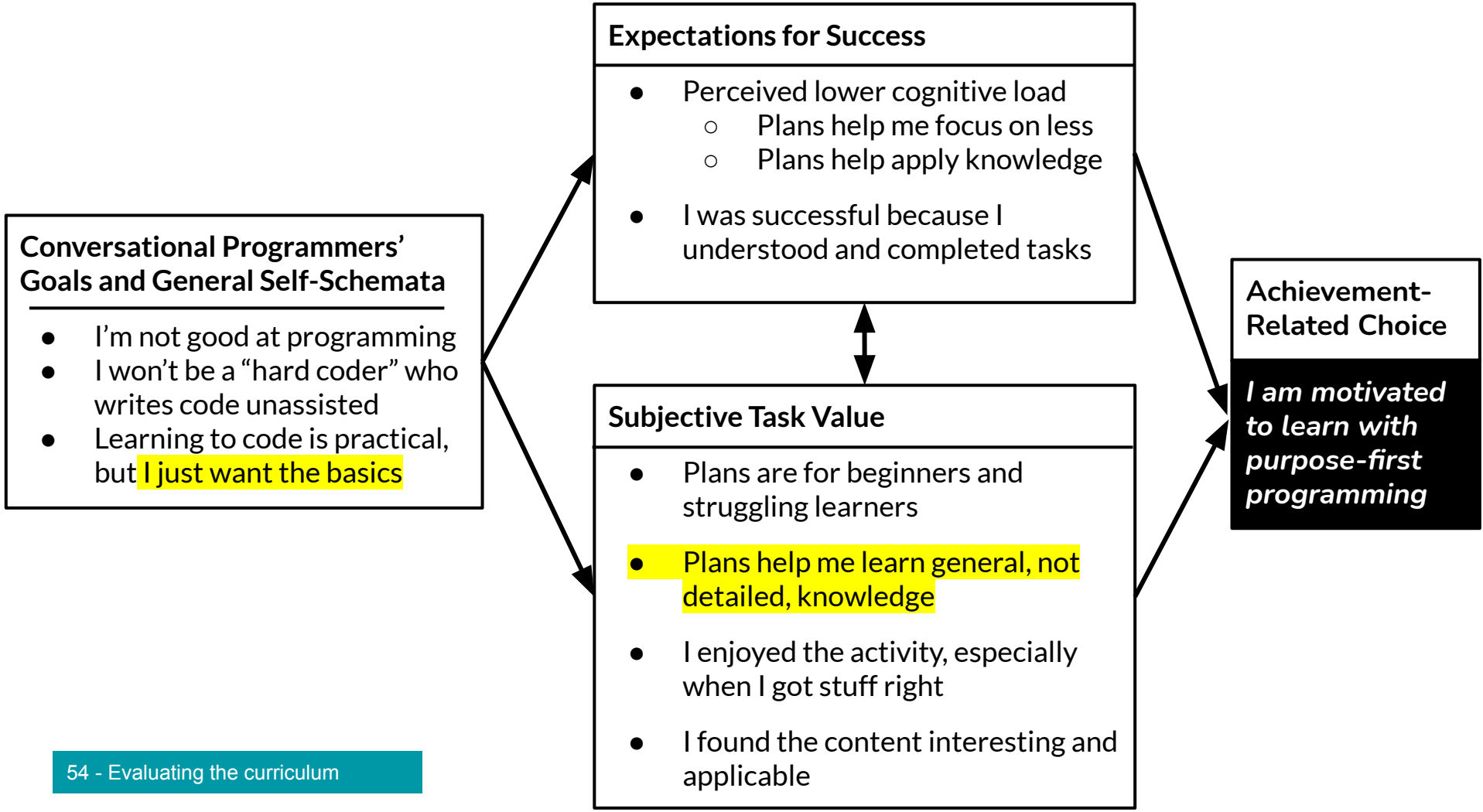
Subjective Task Value

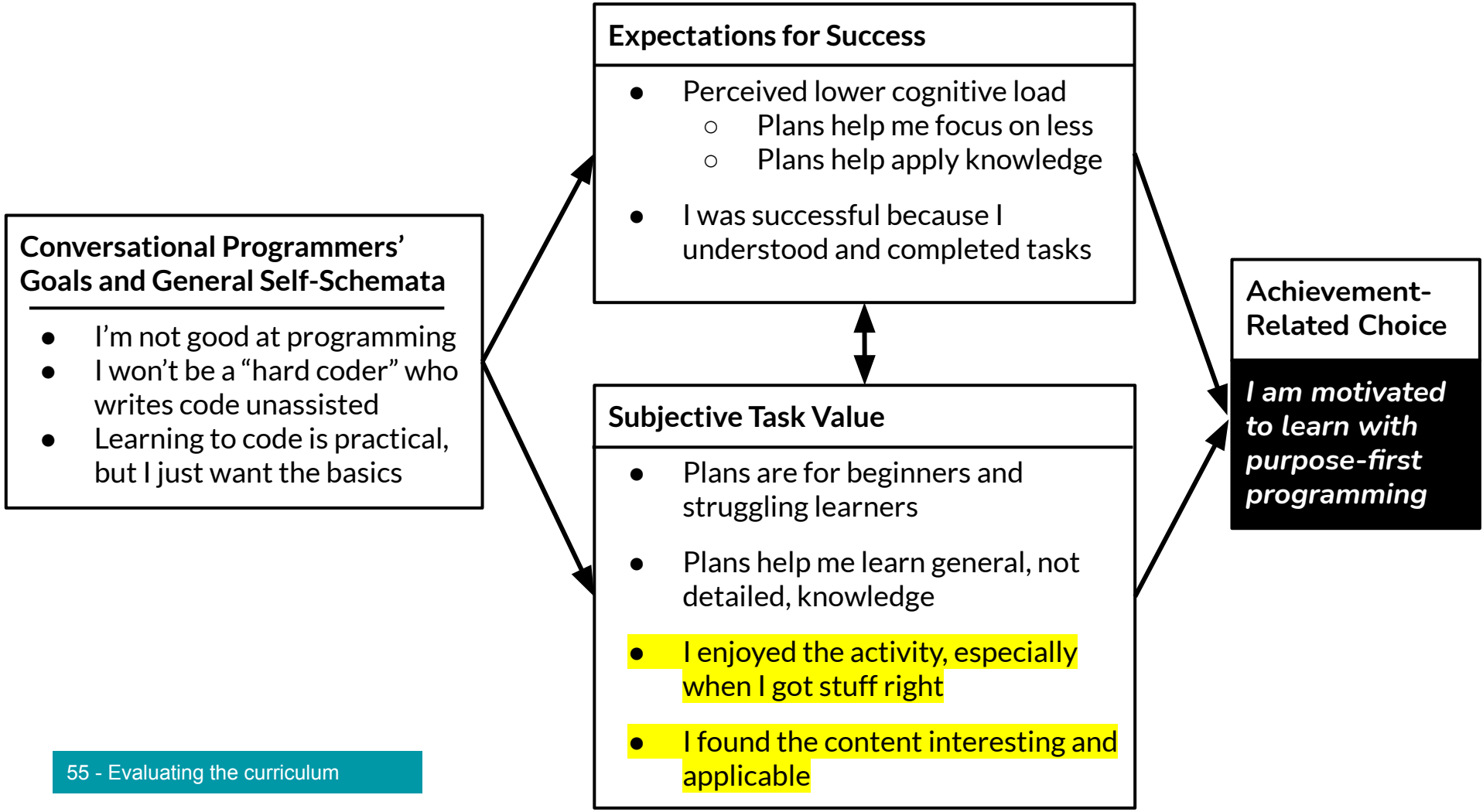
- Plans are for beginners and struggling learners
- Plans help me learn general, not detailed, knowledge
- I enjoyed the activity, especially when I got stuff right
- I found the content interesting and applicable

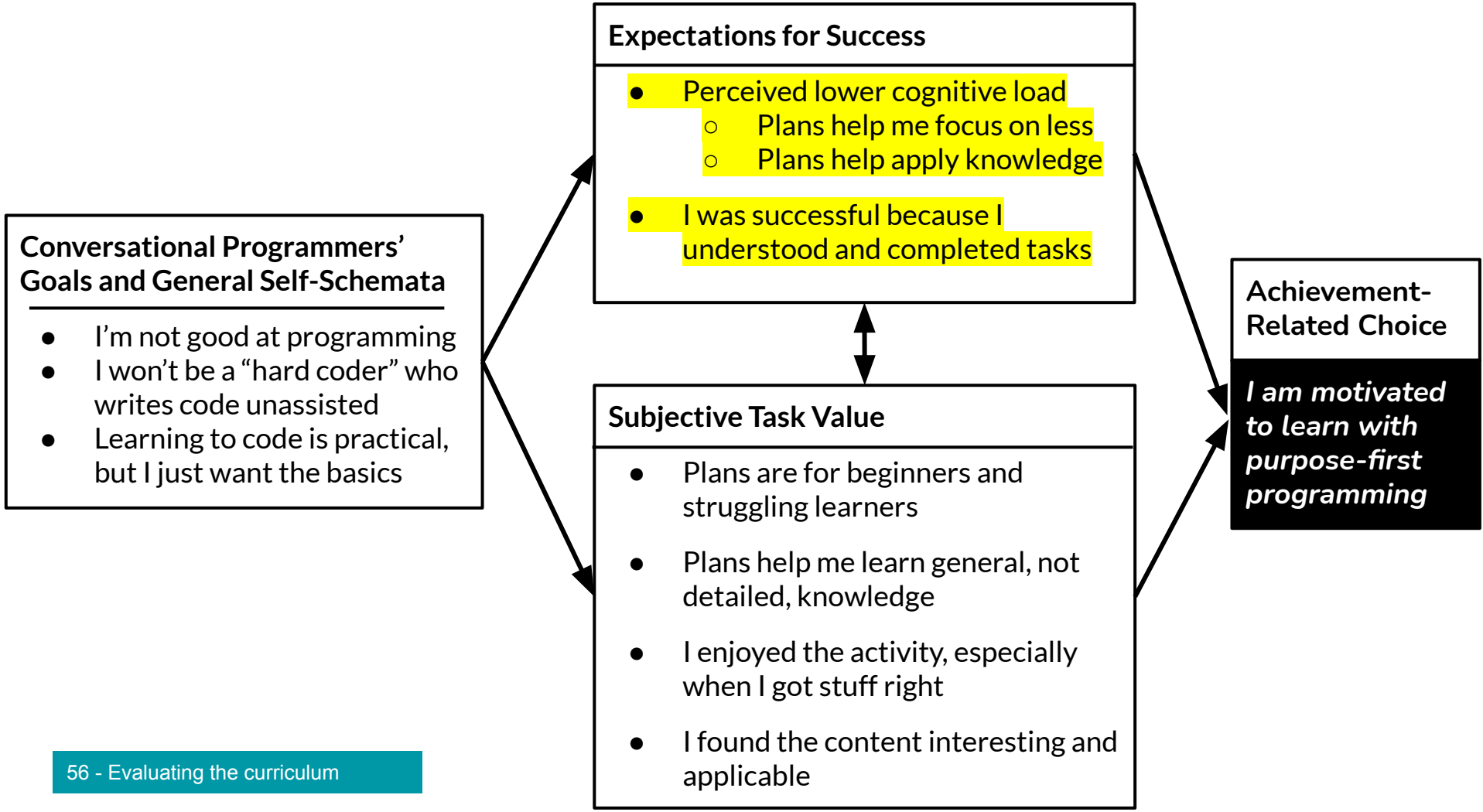
Achievement-Related Choice

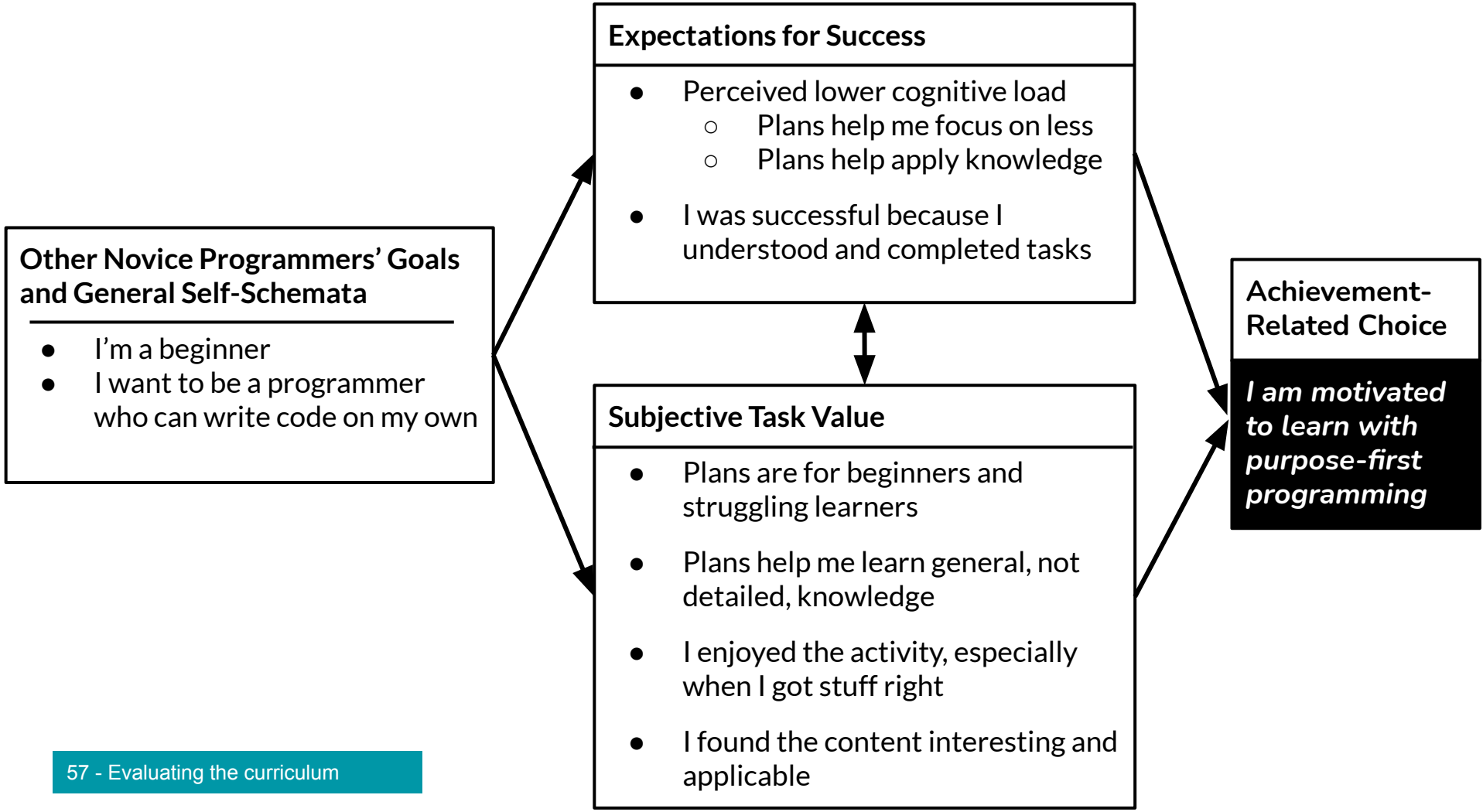
I am motivated to learn with purpose-first programming

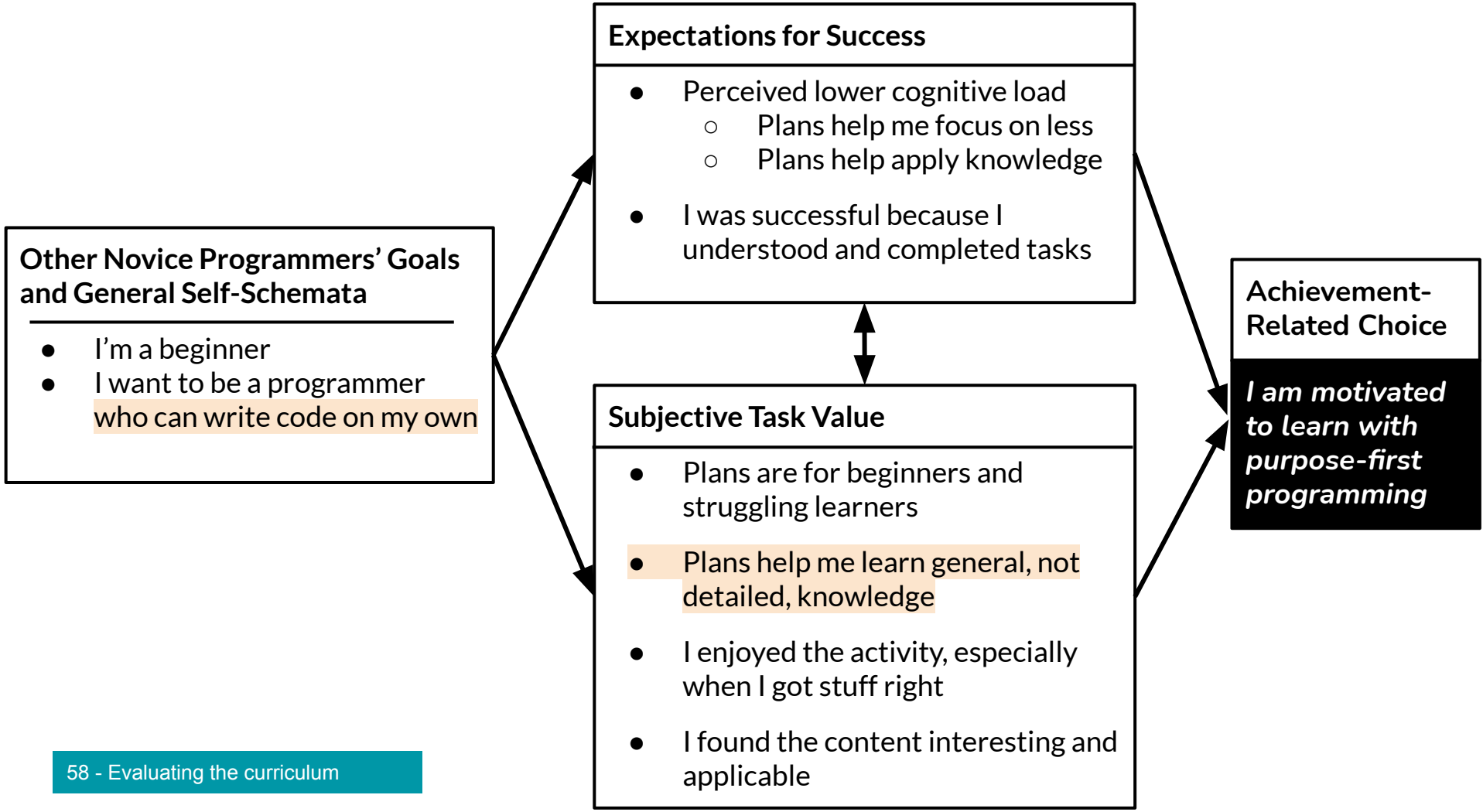

Conversational Programmers' Goals and General Self-Schemata
I'm not good at programming
I won't be a "hard coder" who writes code unassisted
Learning to code is practical, but I just want the basics
Expectations for Success
Perceived lower cognitive load
Plans help me focus on less
Plans help apply knowledge
I was successful because I understood and completed tasks
Subjective Task Value
Plans are for beginners and struggling learners
Plans help me learn general, not detailed, knowledge
I enjoyed the activity, especially when I got stuff right
I found the content interesting and applicable
Achievement-Related Choice
I am motivated to learn with purpose-first programming


Conversational Programmers' Goals and General Self-Schemata
I'm not good at programming
I won't be a "hard coder" who writes code unassisted
Learning to code is practical, but I just want the basics
Expectations for Success
Perceived lower cognitive load
Plans help me focus on less
Plans help apply knowledge
I was successful because I understood and completed tasks
Subjective Task Value
Plans are for beginners and struggling learners
Plans help me learn general, not detailed, knowledge
I enjoyed the activity, especially when I got stuff right
I found the content interesting and applicable
Achievement-Related Choice
I am motivated to learn with purpose-first programming


Conversational Programmers' Goals and General Self-Schemata
I'm not good at programming
I won't be a "hard coder" who writes code unassisted
Learning to code is practical, but I just want the basics
Expectations for Success
Perceived lower cognitive load
Plans help me focus on less
Plans help apply knowledge
I was successful because I understood and completed tasks
Subjective Task Value
Plans are for beginners and struggling learners
Plans help me learn general, not detailed, knowledge
I enjoyed the activity, especially when I got stuff right
I found the content interesting and applicable
Achievement-Related Choice
I am motivated to learn with purpose-first programming

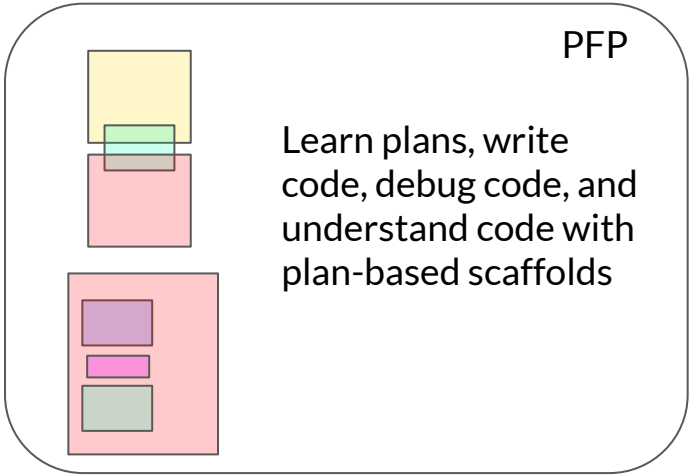

Conversational Programmers' Goals and General Self-Schemata
I'm not good at programming
I won't be a "hard coder" who writes code unassisted
Learning to code is practical, but I just want the basics
Expectations for Success
Perceived lower cognitive load
Plans help me focus on less
Plans help apply knowledge
I was successful because I understood and completed tasks
Subjective Task Value
Plans are for beginners and struggling learners
Plans help me learn general, not detailed, knowledge
I enjoyed the activity, especially when I got stuff right
I found the content interesting and applicable
Achievement-Related Choice
I am motivated to learn with purpose-first programming


Other Novice Programmers' Goals and General Self-Schemata
I'm a beginner
I want to be a programmer who can write code on my own
Expectations for Success
Perceived lower cognitive load
Plans help me focus on less
Plans help apply knowledge
I was successful because I understood and completed tasks
Subjective Task Value
Plans are for beginners and struggling learners
Plans help me learn general, not detailed, knowledge
I enjoyed the activity, especially when I got stuff right
I found the content interesting and applicable
Achievement-Related Choice
I am motivated to learn with purpose-first programming


Other Novice Programmers' Goals and General Self-Schemata
I'm a beginner
I want to be a programmer who can write code on my own
Expectations for Success
Perceived lower cognitive load
Plans help me focus on less
Plans help apply knowledge
I was successful because I understood and completed tasks
Subjective Task Value
Plans are for beginners and struggling learners
Plans help me learn general, not detailed, knowledge
I enjoyed the activity, especially when I got stuff right
I found the content interesting and applicable
Achievement-Related Choice
I am motivated to learn with purpose-first programming

Conclusions

Based on their goals, learners value programming learning tasks differently



Meets many learning needs of conversational programmers

Could be a motivating starting place for other programming learners, if scaffolding fades

Changing the way learners *think* about code can result in increased *motivation*

Where do we go from here?

Can purpose-first programming increase programming **self-efficacy**?

I focused on motivation, but questions of **learning** are next.

Technical supports could work more efficiently. How can they **fade**?

A **platform** could can support the creation of purpose-first modules for a variety of domain areas.

Thanks!