LEARNING INNOVATION AND TEACHING ENHANCEMENT (LITE) GRANTS
Proposal Template for LITE Full Grants

a) Descriptive project title

Promoting Meaningful Learning in Computer Science Courses by Leveraging Concreteness Fading

b) Project summary (max 150 words):

Our project will investigate innovative approaches to enhancing teaching and learning in computer science courses by examining how to use concreteness fading--an instructional technique that has been proven to be effective in math and science education--to improve the teaching and learning in computer science (CS) courses. Concreteness fading has the potential to foster deep learning by helping learners make connections between what they are learning and what they already know. Our project will seek to realize the potential of this technique in real-world classroom settings through the design and evaluation of new tools that implement concreteness fading in computer science contexts. We anticipate that this will provide benefits to both students who are struggling to make sense of abstract computer science concepts, as well as instructors who are struggling to make abstract concepts relatable to students.

c) Project goals/outcomes and, where applicable, research question(s) to be investigated:

- Provide learning contents and tools, such as programming learning environment, that leverage concreteness fading as an instructional technique
- Present use cases describing how concreteness fading can be utilized for teaching particular computer science concepts
- Advance our understanding of how concreteness fading can influence various aspects of learning in computer science education, such as engagement and performance
- Offer design recommendations on how to implement concreteness fading in the teaching and curriculum design

d) Project rationale and description, including review of relevant literature (where applicable) and contextual information:

In recent years, computer science programs (CS) faced a huge growth in enrollment [1]. Despite the popularity and importance of this field, one-third of the students worldwide fail [2]. As a discipline, the computer science curriculum is typically fast-paced and courses tend to build on pre-requisite knowledge that students have to master in earlier courses. Therefore, it can be extremely challenging for students who struggle early on in introductory computer science (CS1/2) courses to catch up once they fall behind [3]; this makes success in

introductory computer science courses critical for the rest of their studies. In fact, studies show that high failure rates in CS1/2 courses lead to increased dropout rates from the computer science program [4].

Among several factors that are implicated as the causes for high failure rates in CS1/2 courses, many agree that the abstract nature of concepts and programing language makes it fundamentally difficult to approach and make sense of the materials [5]. In mathematics and science education, when instructors struggle with a similar problem of conveying its abstract concepts and language to learners, concreteness fading has shown to be an effective method [6, 7].

The theoretical framework of concreteness fading was first conceptualized by Bruner as an instructional technique to facilitate the learning and transfer of abstract ideas [8]. He believed that new concepts and procedures could be better delivered when presented in three progressive forms (Fig. 1)---enactive form, a physical or concrete model of the concept; iconic form, a graphic or pictorial model; and symbolic form, an abstract model of the concept. Bruner argued that grounding a concept in a form that is concrete and already familiar to learners can facilitate meaningful learning of foreign concepts [9, 10, 11].

While research continued to confirm the positive effect of the technique on students' performance in math and science education, in computer science, only recently a few works have examined the potential of using concreteness fading to teach abstract concepts [12, 13]. As a step towards understanding how to utilize concreteness fading for teaching abstract CS concepts, we have conducted an iterative design study, from which we developed a proof-of-concept programming learning environment called CodingStrip.[1]

Our design of this system is unique in two ways: (1) it establishes a concept in the form of comic and then strips away the details to transition into the next stage, visualization, and then to the final stage, code; (2) while the details are stripped away during the progression, each stage is designed to reference the previous stages to help learners easily recognize the connections across the stages. Preliminary findings from this study demonstrate that concreteness fading can help learners understand programming constructs by leveraging multiple representations, by fostering self-efficacy in novice learners, and by tailoring the interfaces to different learning styles. These findings showed its potential as well as the next step for our research, which is examining ways to tailor the interface to each learner based on their knowledge, preferences, and learning styles.

[1] https://codingstrip.github.io

As such, we believe that taking this instructional technique of concreteness fading and applying it to computer science contexts can greatly benefit students taking computer science courses at the University of Waterloo, as well as impact computer science teaching practices beyond the university.

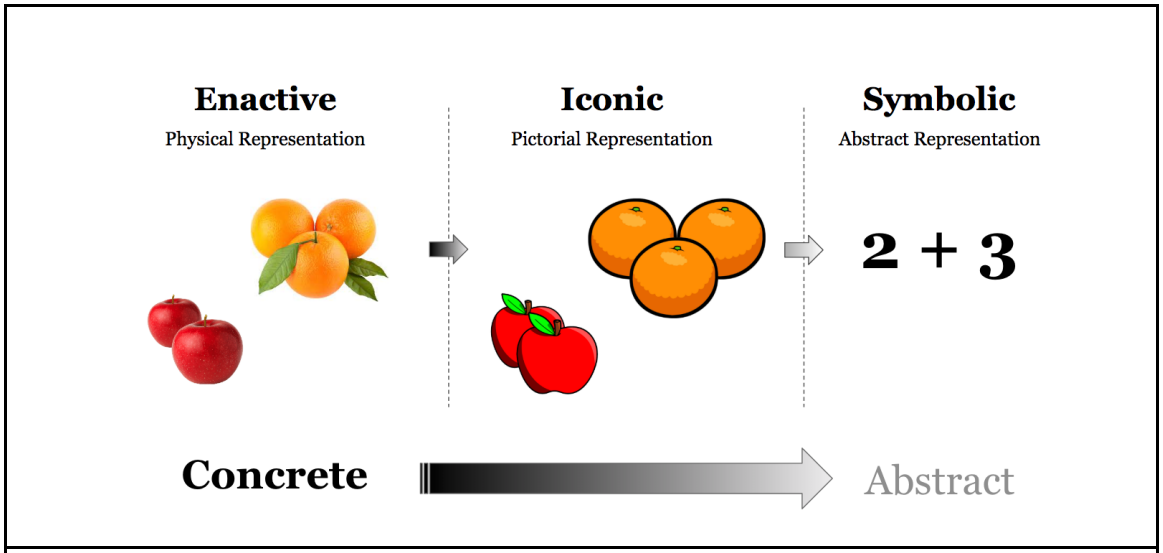

Figure 1. Bruner's theoretical framework of concreteness fading

e) Plan/methods/procedures for carrying out and assessing the project:

There are 4 studies in this project. The first study focuses on examining the challenges in introductory computer science courses and the need for concreteness fading techniques; this will involve interview and participatory design studies with first-year computer science instructors as well as students with experience in the CS1/2 courses. The second and fourth studies involve conceptualization of new tools to concretely implement concreteness fading to teach abstract computer science concepts (e.g., recursion). This would involve lab experiments testing different interfaces of the concreteness fading tools. Finally, the third study will focus on working with first-year computer science instructors to develop a module that implements concreteness fading to teach a computer science concept, and evaluate its effectiveness compared to traditional teaching methods. The metrics for evaluation may include, but not limited to, the following:

- programming test (reading & writing computer programs)
- memory test (e.g., short-term and long-term recall)
- engagement
- confidence

- interest in learning more
- transfer test
- explanation test
- metacognitive accuracy, etc.

f) Statement regarding areas of expertise of project applicant(s)

We have background in Human-Computer Interaction, Artificial Intelligence, Machine Learning, Data Mining, as well as research experience with a number of learning-related projects, along with development experience in building educational applications and learning content. Our background and experience qualifies us to conduct a project of this kind, in which we will need to conduct both qualitative and quantitative studies, as well as develop the relevant learning content and tools. Most importantly, this project is the extension of a recent research project of ours, in which findings from a qualitative study showed promising potential for future work. Our background, experience, and prior work will help us lead this project to success.

g) Outline of project's broader impact -- contribution to UWaterloo community and beyond:

Although the project focuses on improving CS courses, the results will inform effective use of concreteness fading as a general technique for teaching abstract concepts in other disciplines.
We plan to share our experience and findings within the University of Waterloo through seminar talks and invite instructors from multiple faculties/departments to attend. We will also aim to publish our findings at several top-tier computer science education conferences, including SIGCSE, in order to describe use cases and design guidelines that help others adopt concreteness fading techniques in their teaching.

h) Plan for dissemination:

i. At the individual level:
   a. Share our research with potential collaborators or researchers working on relevant projects.
ii. At the Department level:
   a. Present our experience and findings at a Ph.D. seminar, introductory CS course (e.g., CS105/115/135) meetings.
   b. Advertise information about our developed tools to faculty members in the CS department, and share them for use of any kind, if interested.

iii. At the university (uWaterloo) level:
   a. Present our experience and findings at 2020 Teaching and Learning Conference.
   b. Develop a module in an introductory CS course based on concreteness fading

iv. At the national and/or international levels:
   a. Make our learning contents and tools available on the Internet
   b. Provide open source repositories for anyone to use, modify, and refine the learning contents and tools from the project
   c. Present our experience and findings at relevant international conferences

i) Budget:

We would like to hire a Graduate Research Student in the spring term of first and second year to maintain momentum and to develop resources in a timely manner. This student will be tasked with the development of a tool and module for use in our studies (this will take place in our lab and through a uWaterloo CS course). We also need to use Amazon Web Services (AWS) to host an online programming learning environment (for experiments in the first year and for running a CS course in the second year). Moreover, we would like to use part of the budget to pay the participants for our studies.

Items	Amount Year 1	Amount Year 2	Justification
AWS Usage Fee	$1,200	$4,800	Year 1: $100/month * 12 months Year 2: $100/month * 12 months
Study#1 remuneration	$900		$15/participant * 60 participants
Study#2 remuneration		$1,950	$15/participant * 130 participants
Study#4 remuneration		$1,500	$30/participant * 50 participants
TOTAL	$2,100	$8,250	**$10,350** (= Year1 + Year2)

Research Assistant(s)	Rate	# hours	Amount Year 1	Amount Year 2	Justification
PhD Research Assistant	$48.01/hour	10 hours/ week	$7,681	$7,681	Year 1: $7,681 * 1 term Year 2: $7,681 * 1 term
Vacation pay	4%	N/A	$307.24	$307.24	Year 1: $307.24 Year 2: $307.24
TOTAL			$7,988.24	$7,988.24	**$15,976.48** (= Year1 + Year2)

	Items	Research Assistant(s)	Justification
TOTAL	**$10,350**	**$15,976.48**	**$26,326.48** **(= 10,350 + 15,976.48)**

j) Sustainability:

Sustainability of this project will be ensured by the sharing of our resources over the Internet. Specifically, the code base used to build our tools will be stored in open-source repositories for anyone to use, modify, and refine. In the case that we are unable to maintain a server due to cost, the code and learning contents

can be made publicly available through services, such as Github,[2] at no cost. As a result, our learning contents and tools will be available for use and further development even after the LITE grant funding has expired. One of our applicants will be tasked with maintaining the repository, and as a point of contact.

k) Timeline

Activities and Milestones	**2019**											
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Project begins	x											
[**Study #1**] Interview / participatory design study to understand the challenges of intro computer science courses and ways to incorporate concreteness fading	x	x	x	x	x							
[**Study #2**] Work with Research Assistant to design prototypes of the programming learning environment, and conduct lab experiments evaluating various aspects of the new tool.						x	x	x				
Prepare publication for an HCI conference								x	x			
Develop a module for 2020 Winter term CS course that makes use of the previously developed tools for concreteness fading										x	x	x
Submit progress report												x

Activities and Milestones	**2020**											
	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec

[2] https://github.com/

[**Study #3**] Work with instructor;s to deploy the modules in one or more CS courses, with students as research participants	x	x	x	x								
[**Study #4**] Enhance tool for concreteness fading based on classroom findings, run additional experiments					x	x	x					
Present findings at Teaching and Learning Conference at uWaterloo					x							
Prepare publications for an HCI and/or computer science education conference.								x	x			
Prepare final report and present findings at various talks and departmental seminars										x	x	x

Activities and Milestones	**2021**
	Jan
Submit final report	x

References

[1] Zweben, S. and Bizot, B., 2015. CRA Taulbee Survey Report 2014: Relentless Growth in Undergraduate CS Enrollment; Doctoral Degree Production Remains Strong but No New Record. Computing Research News.

[2] Watson, C. and Li, F.W., 2014, June. Failure rates in introductory programming revisited. In *Proceedings of the 2014 conference on Innovation & technology in computer science education* (pp. 39-44). ACM.

[3] Ahadi, A., Lister, R. and Teague, D., 2014, June. Falling behind early and staying behind when learning to program. In *Proceedings of the 25th Psychology of Programming Conference, PPIG*(Vol. 14).

[4] Dasuki, S. and Quaye, A., 2016. Undergraduate students' failure in programming courses in institutions of higher education in developing countries:

A Nigerian perspective. *The Electronic Journal of Information Systems in Developing Countries*, *76*(1), pp.1-18.

[5] Butler, M. and Morgan, M., 2007. Learning challenges faced by novice programming students studying high level and low feedback concepts. *Proceedings ascilite Singapore*, (99-107).

[6] Sousa, D.A., 2008. Recognizing and addressing mathematics difficulties: In How the brain learns mathematics (pp. 186-188).

[7]] Emily R Fyfe, Nicole M McNeil, and Stephanie Borjas. 2015. Benefits of “concreteness fading” for children’s mathematics understanding. Learning and Instruction 35 (2015), 104–120.

[8] Bruner, J.S., 1966. *Toward a theory of instruction*(Vol. 59). Harvard University Press.

[9] David Paul Ausubel, Joseph Donald Novak, Helen Hanesian, et al. 1968. Educational psychology: A cognitive view. Vol. 6. Holt, Rinehart and Winston New York.

[10] Jean Piaget. 2005. The psychology of intelligence. Routledge.

[11] David E Rumelhart and Donald A Norman. 1981. Analogical processes in learning. Cognitive skills and their acquisition (1981), 335–359.

[12] Ian Arawjo, Cheng-Yao Wang, Andrew C Myers, Erik Andersen, and François Guimbretière. 2017. Teaching programming with gamified semantics. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems. ACM, 4911–4923

[13] Anthony Trory, Kate Howland, and Judith Good. 2018. Designing for concreteness fading in primary computing. In Proceedings of the 17th ACM Conference on Interaction Design and Children. ACM, 278–288.