Principles of a Debugging-First Puzzle Game for Computing Education

Michael J. Lee\textsuperscript{1}, Faezeh Bahmani\textsuperscript{2}, Irwin Kwan\textsuperscript{2}, Jillian LaFerte\textsuperscript{2}, Polina Charters\textsuperscript{1}, Amber Horvath\textsuperscript{2}, Fanny Luor\textsuperscript{1}, Jill Cao\textsuperscript{2}, Catherine Law\textsuperscript{3}, Michael Beswetherick\textsuperscript{1}, Sheridan Long\textsuperscript{2}, Margaret Burnett\textsuperscript{2}, Andrew J. Ko\textsuperscript{1}

University of Washington
The Information School\textsuperscript{1}
Seattle, Washington, USA

Oregon State University
School of EECS\textsuperscript{2} and STEM Academy\textsuperscript{3}
Corvallis, Oregon, USA

Summary By-Fardina Fathmiul Alam
SWE 795, Spring 2017
Principles of a Debugging-First Puzzle Game for Computing Education

Motivation

- Although many systems have been designed to help people acquire programming skills independently, but few explicitly teach the subject.

- These existing systems expect from learners to acquire the necessary skills on their own before they can begin creating their own programs from scratch.

Key Idea

- Based on all of these observations, this paper proposed a principled approach to teach programming using a debugging game called “Gidget” based on a unique set of seven design principles.

- Gidget is a game designed to teach computer programming concepts through debugging puzzles, which can help novice programmers to learn without requiring an instructor, teach them important program understanding and debugging skills, and lead to create their own programs.

- This paper has highlighted its seven design principles, design prototypes, and evaluated its use in both a laboratory think aloud study and two summer camps.
Seven Design Principles of Gidget

- **P1-debug. Debugging first:** debug existing programs before creating new programs.

- **P2-game. Game-oriented:** The environment should feel like a game to benefit learning.

- **P3-fallible. Computers as helpful but fallible:** Frame computers as helpful but fallible collaborators.

- **P4-goals. Embedded goals:** Focus to one specific game goal – debugging faulty code.

- **P5-instruction. Embedded instructions:** Provide embedded instruction including specific learning objectives, a planned curriculum, set of instructional materials and tasks.

- **P6-help. Scaffolded help:** Deliver, on request, in-game help: problem-solving strategies, higher-level programming concepts.

- **P7-gender. Gender inclusiveness:** Use a gender-neutral protagonist.
Gidget Approach (1/2)

- Gidget the robot was damaged on its way to clean up a chemical spill and save the animals, so it is the players’ job to fix Gidget’s problematic code to complete all the missions.

- Learners must debug faulty programs to progress through the game, which are set up in modules to teach specific computer programming concepts.

- Once all the levels are complete, learners are given the option to further engage in the game by creating their own levels that can be shared with their friends and family.

Figure 1. Gidget’s level design mode (the Gidget character is circled). In this mode, learners design their own levels for others to solve. Players write code (left) that can include graphics (right), and see animated results (middle), and graphics for the level are on the right.
Gidget Approach (2/2)
Experiment & Results

Conduction of two formative studies:

- **A laboratory think aloud study**: Record in-depth interactions with Gidget. (Think aloud data: The algorithm design barriers, the learning phase barriers)
- **Two summer camps**: observe participants play puzzles and create levels.

They calculated each team’s percent improvement per barrier type and found improvements in 15 out of 17 camp teams and an overall improvement of 45% from puzzle play to level design.
QUESTIONS FOR DISCUSSION

• Overall reactions

• Would you like to use this technique for computing education for novice learners?

• What limitations does this have?