Design Principles behind the Beauty and Joy of Computing Curriculum and Its Influence on Snap*!*

Mary Fries (EDC), Paul Goldenberg (EDC), Brian Harvey (UCB), June Mark (EDC)

A major goal of the Beauty and Joy of Computing AP Computer Science Principles curriculum is broadening participation in computer science. To this end, we the BJC writers at University of California, Berkeley and Education Development Center, Inc. developed a set of design principles to guide the writing of the course. We believe that students must feel invited to use their own creativity and logic and to enjoy the power of their logic and the beauty and elegance of the code by which they express it. All students need genuine challenge and sensible support so all can have the joy of making—seeing themselves as creators, not just consumers, and seeing that their own intellect, not just our instructions, is the source of that making. The curriculum design principles we developed to support these goals have also guided the development of Snap!; learners should focus on the logic and structure of their thinking—not on misplaced semicolons, because attention to such syntactic detail is antithetical to broadening participation.

Pedagogical Principles

All kids can do challenging things. BJC aims to:

- Keep text light and limit prerequisite knowledge without limiting challenge or depth.
- Provide approaches that are genuine challenges to all kids.
- Provide sensible support so all students can experience the joy of making and see their own intellect as the source of that making.

Experience before formality. BJC aims to:

- Give students experience with a concept before any explicit teaching, either in the curriculum pages or in class. We postpone presenting details like vocabulary or a formal definition so that students can build the basic concept before focusing on technicalities.
- Present new ideas with the use-modify-create paradigm, allowing students to first experiment with existing code, then modify it with their own content, and finally extend the idea to their own purposes.

Organizing around big ideas. BJC aims to:

- Weave each AP CSP standard into a consistent social or intellectual storyline.
- Focus on the logic, power, and intellectual beauty of programming and not on syntactic detail by using the Snap! visual programming language, which allows us to focus on coding and debugging over syntax.

Learning by doing. BJC aims to:

- Offer a creative, hands-on experience of making things.
- Provide projects not as the objective of the lessons but as the motivating context for learning important tools and techniques to support students in improvising as they create their own projects.
- Help students experience "I can create," "I can solve problems," and "I can program."

Helping students recognize and enjoy their own logic and creativity. BJC aims to:

- Offer more programming experience than the College Board requires.
- Help students see their code as "poetry" with structure, elegance and power.

Design for diversity by offering diversity and personalization. BJC aims to:

- Attract and actively recruit the most underrepresented groups in computer science.
- Appeal to a breadth of personal, social and intellectual interests that cross race, gender, and economic lines (mathematics, language, games, art, science, etc.).
- Leave room for students to put their own stamp on their work.

Programming Content Principles

BJC aims to help students see that they can do and enjoy computer science. We invite students beyond the entry points to experience recursion and higher-order functions and the powerful and beautiful way they exemplify abstraction, a key idea in CS and in the AP CSP framework. Such "advanced topics" are often seen by others as too difficult for students, but Snap!'s explicit visual representations make them more accessible.

The power of recursion. BJC students should experience the beauty and simplicity of recursion (when a procedure calls itself) through "aha!" moments as they see the outcome of recursive procedures.

Functions as data. BJC students should use functions as inputs to procedures and learn that a function can be treated just like any other piece of data.

Mathematics as a tool. BJC students should appreciate mathematical thinking (basic algebra, use of coordinates, modular arithmetic, etc.) as a valuable tool in programming.

Social Implications of Computing Content Principles

BJC aims to build students' sense of agency, so we balance a fundamental optimism about the future of computer technology with a critical stance toward each specific use, moving past either/or questions to ask how a technology might be redesigned to keep the benefits and avoid the harms.

Social implications differ for different groups. BJC students consider more than just the "benefits and harms" of new technologies and focus on who benefits and who is harmed.

Everyone can participate in developing technology policy.

- BJC students learn that they can control the development of new technologies simply by being aware voters and consumers.
- They read, discuss, and write about issues of computing in society using the Blown to Bits book and regularly engage in "Computing in the News" activities, in which they present a recent article about technology, and the class asks clarifying questions and discusses implications.

Teaching social implications is not "teaching ethics." Rather than telling students what to do (e.g., don't cyberbully) and what to think (e.g., downloading music illegally is bad), BJC respects students as thoughtful social agents and aims to inform them about the implications and uses of technology for different stakeholders.

Influence on Snap*!*

- All of Snap*!* comes out of BJC: the connection between UC, Berkeley and Jens Mönig formed around developing a visual language for teaching recursion and higher order functions; even the name Snap*!* came from BJC teacher input!
- The development of unevaluated data types ("any" and "Boolean") came out of creating the reporter IF block in response to BJC teacher feedback.
- The FIND FIRST block came out of the desire to simplify BJC projects that were more complicated and slower than they needed to be.
- Table view for lists of lists was created in response to a desire to represent databases visually.

More About Our Design Principles

Learn more about the BJC design principles by reading our SIGCSE 2020 paper: Design Principles behind Beauty and Joy of Computing (https://doi.org/10.1145/3328778.3366794).

 The development of this site is funded by the National Science Foundation under grant number 1441075. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.