Classroom Applications of Question Formulation to Support Problem-Solving in Computer Science

James Quinlan
University of Southern Maine
james.quinlan@maine.edu

Michael Todd Edwards
Miami University
edwardm2@miamioh.edu

ABSTRACT

Questions are an integral part of the teaching and learning process. As students ask questions, they explore complex ideas, challenge assumptions, and confront contradictions. Too often, however, students do not know what questions to ask. They are hesitant to reveal their misunderstandings in front of their classmates. Other students don't participate because they are not invested in the course content. This paper presents the Question Formulation Technique (QFT), a teaching method designed to provide computer science students with targeted instruction on question-posing. As students learn to ask better questions, they become more confident in their abilities as learners. In this experience report, we provide a framework and implementation process, highlighting key steps and potential outcomes. Drawing on instructor observations, we report improvements in student engagement and critical thinking, while also discussing the limitations of anecdotal evidence and outlining directions for future research. Through in-class examples, we discuss the method's strengths and limitations while offering sample prompts that can be adapted for classroom use.

KEYWORDS

inquiry, student questioning, computer science education

1 INTRODUCTION

Undergraduate computer science departments experience high failure and dropout rates in introductory programming courses (IPCs) [14]. If students drop out of an IPC, they are unlikely to enroll in subsequent computer science courses [27]. Too often, instructors in IPCs emphasize syntax and semantics over problem-solving and collaboration [13], discouraging students who may be interested in computer science but need more programming experience. Students without such experience must learn syntax and problem-solving simultaneously, a daunting task for many undergraduates.

How can we better support such students? One approach is to provide them with more meaningful opportunities to learn from each other and by asking (and answering) questions. Indeed, asking questions is central to learning, in general, [1, 5, 6, 10, 15] and is a key component in problem-solving [4]. When students ask questions, they tend to experience improved learning outcomes [2]. Moreover, students develop a growth mindset [7] towards learning. Asking questions helps students become more confident in their ideas and abilities. Students begin to see themselves as knowledge creators rather than passive participants [19]. When students hone their questioning skills, they are better prepared outside the QFT context to ask questions, for instance, at professional meetings and in interviews [16, 20, 23]. Given the many benefits of questioning, it is surprising how little time is devoted to question-posing in computer science classrooms.

How might you encourage your students to ask more thoughtful, computer science-related questions in the classes you teach? In the sections that follow, we discuss the application of Rothstein and Santana's Question Formulation Technique (QFT) [22] in computer science classrooms. QFT is a pedagogical method to promote and advance students' questioning skills. The technique has been successfully applied in various subjects, including English [6], biology [10], electrical circuits [16], and mathematics [5]. QFT is a versatile method that can introduce students to a new topic, assess their knowledge and understanding of previously taught content, or wrap up a topic to evaluate their growth. Research across educational settings has demonstrated the efficacy of QFT in promoting student inquiry, curiosity, and engagement. In mathematics education, Mannion [18] reported measurable improvements in students' performance on open-ended, written-response questions after exposure to QFT. Similarly, Summers et al. [24] found that undergraduate students developed questioning, creativity, and collaboration skills through regular use of the technique, as evidenced by assessment data. LeBlanc et al. [16] showed that QFT stimulated curiosity and technical question formulation in engineering courses. Research in college classrooms also highlights increased agency and engagement, with students demonstrating greater ownership of their learning [11]. Comprehensive reviews further support QFT as a research-based practice to engage learners in various subjects [15, 17, 26].

These findings suggest that QFT is a robust, evidence-based approach for enhancing learning outcomes across various disciplines. While the present report relies on instructor observation, the broader literature provides a formal evaluation of QFT's effectiveness in promoting active learning and inquiry.

In the following sections, we share how we have used QFT with undergraduates enrolled in an IPC. In Section 2, we outline the steps of the QFT framework, with a significant emphasis on constructing high-quality prompts (see Subsection 2.1). Student-centered steps are covered in Subsections 2.2, 2.3, 2.4, 2.5, and 2.6. In Section 3, we provide a brief discussion of limitations and offer implementation tips (see Section 3.2). In addition to the limitations, Section 3.1 includes ideas for future work to address the limitations.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Copyright ©JOCSE, a supported publication of the Shodor Education Foundation Inc.

© 2025 Journal of Computational Science Education
https://doi.org/10.22369/issn.2153-4136/16/2/3

2 THE QFT FRAMEWORK

Six general steps comprise the QFT process [22, p. 4].

(1) *Design Question Focus*: The teacher designs a prompt for students. This is referred to as the "QFocus."
(2) *Generate Questions*: Students generate a list of questions about the QFocus.
(3) *Revise Questions*: Students revise questions, enhancing their readability and making them more conducive to further investigation.
(4) *Prioritize and Select Question*: Students discuss which questions seem most engaging.
(5) *Investigation/Implementation*: Students apply their question(s) to a classroom assignment such as a homework problem, a code experiment, or a lab project.
(6) *Reflection*: In small groups and whole-class presentations, students share their end products and discuss what they learned through the six steps of the QFT process.

Steps 2–6 may be shortened or altered to accommodate various needs or preferences. For example, students may work in groups instead of individually. Educational contexts, such as online distance learning situations or novel course scheduling, may also necessitate modifications. In the remainder of this section, we discuss the six steps in the QFT Process in greater detail, illustrating an implementation of QFT in an IPC classroom using a case study approach.

2.1 Design Question Focus

In the first step of the QFT process, the instructor designs and shares a question focus (QFocus) with students. As the Right Question Institute [21] notes:

> *The QFocus is a stimulus, a springboard, that students will use to ask questions. The QFocus can be a sentence, phrase, image, or situation that will be the "focus" for generating questions* (p. 3).

The QFocus is *not* a question itself—its purpose is to stimulate *student* questions. Since the QFocus sets the stage for the rest of the QFT experience, its importance cannot be overstated.

Early in our IPC courses a brief QFocus, "programming languages," helps us ascertain our students' pre-existing knowledge of programming languages. Later on, when we seek to deepen our students' understanding of more elaborate topics, we provide a QFocus that requires students to consider ideas from different vantage points. For instance, when our students study recursion, we present the QFocus "recursion and iteration" to students to encourage them to think more deeply about the relationships between recursion and iteration.

The QFocus should be narrowly focused to draw students' attention in specific directions. Vague or broad prompts make it difficult for students to formulate meaningful questions. Developing a suitable QFocus is challenging, so it's helpful to keep the following criteria in mind when designing them:

(1) The QFocus should not be a question.
(2) The QFocus should produce different lines of questioning.
(3) The QFocus should be simple, yet not overly simplistic [12].

Note that the QFocus can be graphical rather than purely textual. We've provided objects, images, word clouds, hashtags, and animated GIFs as QFoci. Consider, for instance, the flowchart depicted in Figure 1.

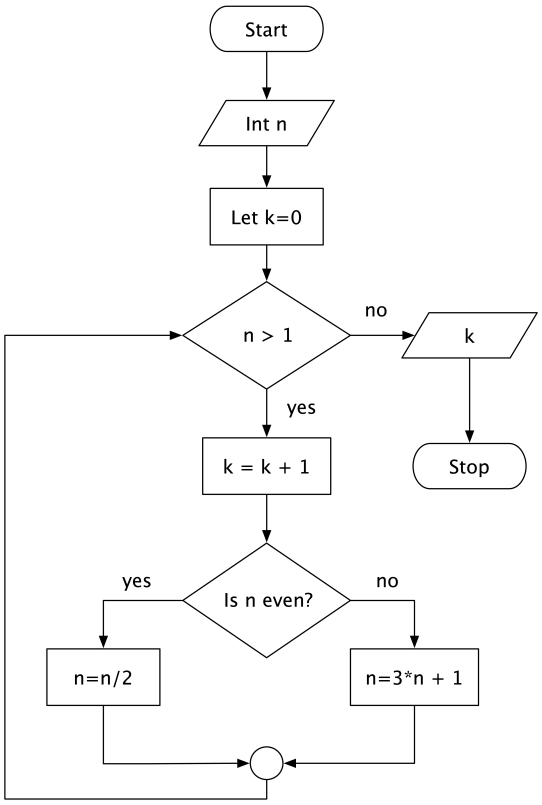

Figure 1: A flowchart of the Collatz conjecture as a visual QFocus.

This visually engaging representation is designed to capture students' attention and motivate them to think about coding and syntax, as well as the functionality of the code, its potential applications, and the process of revision.

Moreover, note the mathematical context embedded within the flowchart—namely, the Collatz conjecture. We find it beneficial to include mathematics in introductory computer science courses, similar to, but opposite of Friend et al. [9], who advocates for including computer science ideas in mathematics coursework. In general, we find it helpful to "hide" mathematical topics in our QFoci as this approach connects computer science ideas to content (e.g., mathematics), which is arguably more familiar to IPC students. Additionally, the method provides robust topics that lend themselves to student questioning.

Figure 2 illustrates a different visual QFocus designed to engage students in recursive thinking through the mathematics of the Tower of Hanoi puzzle. The prompt encourages students to consider how the problem might be solved using code. With this prompt, students have asked questions about loops, decision structures, syntax, recursion, and code efficiency. This is a great prompt for small groups of 3 to 4 students.

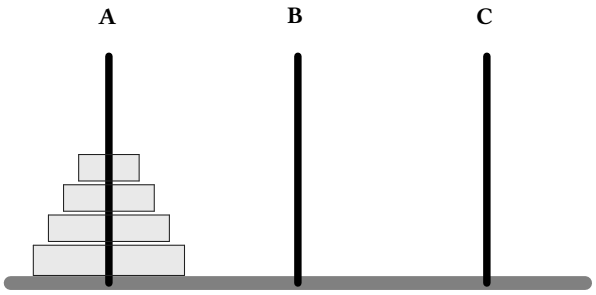

Figure 2: Move all disks from A to C without letting a larger disk be on top of a smaller disk, moving only one disk at a time.

2.1.1 More Examples of QFoci Prompts. The following is a list of QFoci we have presented to our students. The QFoci are presented in *italics* with brief commentary.

- *Increase in surface area of a sphere to increased radius.* This QFocus could be used to develop problem-solving and reinforce arithmetic operations in a specific programming language.
- *The Harmonic series,*

$$\sum_{k=1}^{\infty} \frac{1}{n}.$$

  Students are confronted with how to implement a summation with infinite terms and infinite loops. This might lead to questions regarding convergence, as well as round-off errors in floating-point systems.
- *The limit,*

$$\lim_{x \to 0} \frac{\sin(x)}{x}.$$

  This prompt encourages students to consider loops and syntax more carefully. Moreover, depending on the programming language (e.g., Python), students may need to import standard math libraries to implement their code.
- *Primes of the form* $4k + 1$ *and* $4k + 3$. In 1853, Russian mathematician Pafnuty Chebyshev hypothesized that there are more primes of the form $4k + 3$ than of the form $4k + 1$, up to the same limit [3]. We've used Chebyshev's conjecture to launch student discussions about modular arithmetic, cardinality, branching, and looping.
- *Sorting large datasets.* This QFocus can prompt students to formulate questions about algorithmic complexity, runtime efficiency, memory usage, and the practical trade-offs between different sorting methods.
- *Facial recognition software.* Although slightly outside the typical scope of an IPC, this prompt effectively engages students, as most have existing opinions or experiences, thus encouraging questions that extend beyond purely technical aspects of computer science.

2.2 Generate Questions

After presenting the QFocus, students typically work individually or break into small groups to generate questions. This step is the most crucial for students.

We provide 10 minutes to generate questions for the first few QFT exercises. After repeated practice, this step requires less time. Assigning this step as homework is another possibility, providing more time for students to generate questions.

We provide the following "rules of engagement" to our students *each* time we employ QFT, following [22].

(1) Ask lots of questions.
(2) Do not discuss, judge, or answer questions.
(3) Record questions precisely as initially stated.
(4) Change statements into questions.

We have found it beneficial to model words or behaviors that are "judging" in a whole-group setting before breaking students into smaller groups. Additionally, depending on your class, providing students with time to generate questions individually may be valuable before sharing them with others, as it can support equitable participation. This is particularly practical for students with limited English proficiency or less computer science experience.

We ask groups to generate at least ten questions. In our experience, the number of questions is somewhat inversely proportional to their quality. When we require too many, students tend to "pad" their list with trivial questions to meet the minimum requirement. We also encourage students to ask any questions that come to mind and remind them that the "best" questions will be selected from their list later in the process (see Sections 2.3 and 2.4).

Ultimately, students sift through the questions they generate and select a handful for further investigation. For instance, after examining the flowchart of the Collatz conjecture from Figure 1, students generate questions such as the following:

(1) Why study the Collatz conjecture?
(2) What is the Collatz conjecture, and how is it formulated?
(3) How can we store intermediate iterates?
(4) Which loop is appropriate for implementing the Collatz conjecture as code?
(5) Does the input need to be a positive integer?
(6) What if we do not input an integer?
(7) What is the code corresponding to the logic of this flowchart?
(8) What are the longest-known sequences?
(9) Can all flowcharts be reduced to code? Can a flowchart represent all code?
(10) What happens if the update formulas for n are changed?
(11) What if we made more than two update conditions? For example, what happens if we change from mod 2 to mod 3?

Note the variation in the questions. Some indicate previous experience with computer science concepts, while others aren't specific to computer science at all. Some will require significant work to answer, while others require only a one-word response, such as "Yes" or "No."

In the next section, we discuss the next step of the QFT Process—namely, revising questions. We provide tips to help your students assess and adjust first-draft questions for maximum impact. Ultimately, in subsequent phases of the QFT, each student will construct and select a question to explore in more detail.

2.3 Revise Questions

Revision leads to higher-quality questions. After the brainstorming phase, students revisit their questions for clarity. Additionally, we ask students to transform any "closed" questions into "open" ones. Using the following definitions, we explain that open questions are better suited for exploration and research.

- *Closed questions* have a single correct answer, such as 'yes' or 'no', or other factual information. For example, "Do we have to use a loop?", "Is there only one way to implement branching?"
- *Open questions* have multiple right answers and come in two flavors: interpretive and evaluative. *Interpretive questions* must be supported with evidence. For example, "Which code has the fastest runtime, and why?" *Evaluative questions* ask for opinions, beliefs, or points of view and have no wrong answers. "Do particular integers have common sequence characteristics?"

Open questions are preferable for teaching and learning because they elicit expanded thinking and processing of information [8]. We have our students classify their questions by placing an "O" next to their open questions and a "C" next to their closed questions [21].

Next, we model the transformation of a closed question into an open one. For instance, closed questions may be "opened" by adding 'how' or 'why' to the beginning. We ask students to brainstorm other techniques and post their list on the classroom whiteboard for reference.

At this point, students revise their questions. We collect students' revisions to understand their thinking process better and assess their understanding of open and closed questions. Consider, for instance, the fourth question about the Collatz conjecture.

> *Which loop is appropriate for implementing the Collatz conjecture as code?*

It is closed since this question can be answered with a simple response (e.g., "for loop" or "while loop"). However, a slight modification makes it *open*—namely, adding "how" to the query: "*How does one determine which type of decision structure is best for implementing the Collatz conjecture as code?*" Similarly, question 5, "*Does the input need to be a positive integer?*", can be converted to an open question by rephrasing it to be, "*why does the input need to be a positive integer?*"

2.4 Selecting Questions

Once students have revised their questions, they determine which will generate the most productive exploration of course concepts. We stipulate that individuals, or groups, generate a "top 5 list" of their best questions. For each selected question, students write a brief (1-2 sentence) rationale justifying its inclusion in the "top 5."

The following prompts may help students prioritize questions for possible exploration:

(1) *Is the question suitably narrow?* In other words, is the topic narrow enough that one could reasonably be expected to answer the question within the timeframe of the assignment?
(2) *Is the topic properly connected to the QFocus and the instructional objectives of our course?*
(3) *Will the topic help you strengthen your understanding of a particular area of computer science?* In other words, what capacity does the question have to push your learning and understanding of computer science?

We have students answer the prompts for each question in their "top 5 list" to help them make a final selection. Once they've selected a question, students are ready to begin the next stage of the process, namely Investigation/Implementation.

2.5 Investigation/Implementation

As University of Michigan notes [25], students could use their questions as a starting point to:

- Develop a lab experiment.
- Design a product or process.
- Write a research paper.
- Deliver a presentation.
- Prepare for an in-class discussion or debate.

In general, we've found it advantageous to introduce QFT with relatively short, low-stakes projects—such as writing pseudo-code for a short practice program or preparing for an in-class topic discussion. Using QFT in the first few weeks of class allows students to learn the process without worrying about a class grade.

For the Collatz conjecture, many of our introductory programming students focus on writing code to explore the topic (e.g., finding sequences of a particular length). Those with more coding experience often opt to learn more about the conjecture or compare various code implementations (e.g., "bit hacks").

In one class, we asked students to summarize the findings from their question in a "one-pager" (e.g., one page of code or a short research report of 250-500 words) and include a bibliography and inline citations from peer-reviewed sources. We've found that restricting their writing to one page encourages students to concentrate on the selected question without diverging into unrelated topics.

At the next class meeting, students share their one-pagers, highlighting key ideas and solutions they uncovered as part of the research process. As students share their findings, they gain confidence in their capabilities as self-directed learners and benefit from considering content from multiple points of view—rather than from the instructor's view alone.

2.6 Reflection

In the Reflection phase, students can produce a summary document showcasing what they (or their group) learned through the QFT process. Although our students typically present their results in writing, findings can be disseminated through oral presentations or coding demonstrations. In addition to sharing content with classmates, students often share thoughts about the QFT process. Typically, sentiments include recognition of classroom engagement and confidence in posing questions. As students share their questions and corresponding answers or solutions, they demonstrate their grasp of the topic, their ability to think critically, and their proficiency in applying programming concepts.

During the reflection phase, it's essential to consider group dynamics if working in a group. To ensure that all students are actively engaged, it is important to assign each group member a defined

role. This promotes balanced participation, encourages equitable contribution, and supports individual accountability. For instance, we assign a leader for each group, who submits a link to a single document (e.g., a Google Doc) for their group. This document consists of a cover page and one additional page for each group member (i.e., a one-pager for each group member). The cover page includes the following:

- The group's initial list of questions, along with revisions.
- Markings indicating questions that were initially open ("O") or closed ("C").
- A summary paragraph that discusses what the group collectively learned from the QFT experience.

Other roles have included a scribe to document group decisions and a presenter to share findings with the class. Colleagues who frequently incorporate group work into their instruction have also employed roles such as 'devil's advocate', 'prioritizer', 'diverger' (to encourage divergent thinking), and 'converger' (to support convergent thinking).

3 DISCUSSION AND LIMITATIONS

We observed several notable improvements in student learning and engagement after implementing the QFT in our introductory programming courses. Moreover, we also observed that students' ability to formulate questions extended beyond the QFT session to regular class discussions and problem-solving activities.

Active Learning and Engagement. Based on instructor observations, students responded positively to QFT sessions, showing increased engagement, frequently volunteering to participate, and expressing enthusiasm during group activities. This was evident in the volume and quality of student participation, as students took ownership by generating, refining, and exploring their questions about course topics. While some students offered positive comments informally, we did not implement a systematic evaluation or survey regarding the technique.

Development of Critical Thinking. Instructor observations revealed that students moved beyond superficial or factual queries to ask more analytical and open-ended questions. For example, students were observed debating the efficiency of different coding strategies, seeking to understand not only "what" but "why" a particular approach worked. This deepened their exploration of programming concepts and promoted higher-order thinking skills.

Curiosity and Intellectual Exploration. Students' curiosity was demonstrated by their willingness to pose follow-up questions and explore "what if" scenarios beyond the initial prompt. For instance, some students asked about alternative ways to solve a problem or proposed modifying existing code to test new hypotheses. These behaviors reflected a genuine interest in understanding the material and a desire to experiment with their learning.

Teamwork and Communication. QFT sessions fostered a collaborative classroom environment. Students worked together to develop questions, compare perspectives, and present their findings. Through group discussions and peer presentations, students practiced articulating their ideas clearly and responding to feedback, thereby improving their communication and collaborative problem-solving skills.

3.1 Limitations and Future Work

We acknowledge that the findings reported in this study are based solely on instructor observation and informal classroom feedback; no formal surveys, quantitative assessments, or systematic data collection were conducted during the implementation of QFT in our courses. As a result, while we observed improvements in student engagement, critical thinking, curiosity, and teamwork, these observations are anecdotal and should be interpreted with caution.

To more rigorously assess the impact of QFT, future research should employ comprehensive evaluation methods. For example, controlled experiments could compare students exposed to QFT with those who are not, using pre- and post-assessments to measure changes in questioning skills, critical thinking, and programming knowledge. Qualitative data from interviews, focus groups, or open-ended surveys can offer deeper insights into students' experiences and perceptions of QFT. Another promising direction for future research would be to replicate the analysis conducted by Summers et al. [24], which examined students' questions, feedback, and reflections to trace the evolution of their questioning throughout the QFT process.

By conducting such systematic assessments, we can more precisely determine the effects of QFT and further refine strategies to promote active, student-centered learning in computer science education.

Additional avenues for future work include exploring optimal QFT implementation in online or hybrid environments, investigating its effectiveness across programming languages and course levels, and integrating QFT with other active learning strategies. Another promising direction involves exploring how generative AI tools might assist instructors in creating QFoci or help students refine their questions during the revision phase.

3.2 Tips for Implementing

Based on our experience, we provide several implementation suggestions.

- *Allocate specific class times*: Dedicate regular periods or portions for QFT sessions. This can help establish a routine and signal to students the importance of question formulation. In a CS1 course, we allocated 15-20 minutes *every* class period for Steps 2, 3, and 4. Consider adopting a flipped classroom model if the time spent on QFT activities limits the coverage of course content.
- *Assign parts as homework*: Steps 5 and 6 of the QFT process can be assigned as homework. This can also maximize in-class time for more collaborative aspects, like improving or prioritizing questions.
- *Gradual integration*: Instead of implementing the full QFT process immediately, consider introducing it gradually throughout the course. Start with simpler QFT activities and build up to more complex ones as students become more comfortable with the technique.
- *Model the process*: Demonstrate the QFT process yourself by modeling how to formulate, improve, and prioritize questions related to the course content.

- *Encourage self-evaluation*: Incorporate opportunities for students to self-evaluate their questions, reflect on their questioning skills, and set goals for improvement.
- *Collaborate and share*: Encourage students to collaborate in small groups during QFT sessions, allowing them to share and build upon each other's questions.
- *Provide feedback*: Offer constructive feedback on students' questions, highlighting strengths and areas for improvement in their question formulation abilities.

REFERENCES

[1] Ester Aflalo. 2021. Students generating questions as a way of learning. *Active Learning in Higher Education* 22, 1 (2021), 63–75.

[2] William S. Carlsen. 1991. Questioning in Classrooms: A Sociolinguistic Perspective. *Review of Educational Research* 61, 2 (1991), 157–178. http://www.jstor.org/stable/1170533

[3] Pafnuty Lvovich Chebyshev. 1853. Lettre de M. le Professeur Tchébychev á M. Fuss sur un nouveaux théorème relatif aux nombres premiers contenus dans les formes $4n + 1$ et $4n + 3$. *Bull. Classe Phys. Acad. Imp. Sci. St. Petersburg* 11 (1853), 208.

[4] Christine Chin and Jonathan Osborne. 2008. Students' questions: a potential resource for teaching and learning science. *Studies in science education* 44, 1 (2008), 1–39.

[5] Paul Davis. 1994. Asking Good Questions about Differential Equations. *The College Mathematics Journal* 25, 5 (1994), 394–400. http://www.jstor.org/stable/2687504

[6] Ken Donelson. 2008. The Art of Asking Questions: Two Classes That Changed My Teaching Life. *The English Journal* 97, 6 (2008), 75–78. http://www.jstor.org/stable/40503416

[7] Carol Dweck. 2015. Carol Dweck revisits the growth mindset. *Education week* 35, 5 (2015), 20–24.

[8] Cornell University Center for Teaching Innovation. 2020. Using effective questions: Center for Teaching Innovation. https://teaching.cornell.edu/fall-2020-course-preparation/engaging-students/using-effective-questions

[9] Michelle Friend, Andrew W Swift, Betty Love, and Victor Winter. 2023. A Wolf in Lamb's Clothing: Computer Science in a Mathematics Course. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*. ACM, 256–262.

[10] L. Goodman and G. Berntson. 2000. The Art of Asking Questions: Using Directed Inquiry in the Classroom. *The American Biology Teacher* 62, 7 (2000), 473–476. http://www.jstor.org/stable/4450954

[11] Martha Higginbotham. 2023. *Teaching students to ask questions: The role of question formulation technique in building agency and student engagement in the college classroom.* Ph.D. Dissertation. Rider University.

[12] The Right Question Institute. [n.d.]. Designing the question focus (QFOCUS) - right question institute. https://rightquestion.org/wp-content/uploads/2019/05/RQI-Resource-An-Introduction-to-QFocus-Design-PDF.pdf

[13] Sohail Iqbal and Om Kumar Harsh. 2013. A self review and external review model for teaching and assessing novice programmers. *International Journal of Information and Education Technology* 3, 2 (2013), 120.

[14] Sohail Iqbal Malik and Jo Coldwell-Neilson. 2017. Impact of a new teaching and learning approach in an introductory programming course. *Journal of Educational Computing Research* 55, 6 (2017), 789–819.

[15] Todd Larsen. 2020. Using Student-Generated Questions to Promote Curiosity and Student Learning. (2020).

[16] Heath J LeBlanc, Kundan Nepal, and Greg S Mowry. 2017. Stimulating curiosity and the ability to formulate technical questions in an electric circuits course using the question formulation technique (QFT). In *2017 IEEE Frontiers in Education Conference (FIE)*. IEEE, IEEE, New York, NY, 1–6.

[17] Cristo Leon. 2024. Empowering Inquiry: The Transformative Power of the Question Formulation Technique in Education. (2024).

[18] Jessica M Mannion. 2019. The effectiveness of the question formulation technique on open-ended, written response questions in mathematics. (2019).

[19] Barbara L McCombs and Robert J Marzano. 1990. Putting the self in self-regulated learning: The self as agent in integrating will and skill. *Educational Psychologist* 25, 1 (1990), 51–69.

[20] Kaitie O'Bryan. 2017. Using QFT to prepare students for new experiences. *Kaleidoscope: Educator Voices and Perspectives* 1, 4 (2017), 29–31.

[21] The Right Question Institute. [n.d.]. Experiencing the question formulation techniqueTM (QFTTM). https://condor.depaul.edu/tps/resources/level1/experienceqft.pdf

[22] Dan Rothstein and Luz Santana. 2011. *Make just one change: Teach students to ask their own questions.* Harvard Education Press.

[23] Luz Santana. 2015. Learning to Ask Questions: A Pathway to and through College for Students in Low–Income Communities. *About Campus* 20, 4 (2015), 26–29.

[24] Mindi Summers, Jordann Fernandez, Cody-Jordan Handy-Hart, Sarah Kulle, and Kyla Flanagan. 2024. Undergraduate Students Develop Questioning, Creativity, and Collaboration Skills by Using the Question Formulation Technique. *The Canadian Journal for the Scholarship of Teaching and Learning* 15, 2 (2024).

[25] LSA University of Michigan. [n.d.]. Question Formulation Technique - University of Michigan. https://sites.lsa.umich.edu/inclusive-teaching/wp-content/uploads/sites/853/2021/09/Question-Formulation-Technique-Draft.pdf

[26] Jackie Acree Walsh and Beth Dankert Sattes. 2016. *Quality questioning: Research-based practice to engage every learner.* Corwin Press.

[27] Susan Wiedenbeck, Deborah Labelle, and Vennila NR Kain. 2004. Factors affecting course outcomes in introductory programming.. In *PPIG*. 11.