2016

Integrating Creative Writing and Computational Thinking to Develop Interdisciplinary Connections

Candido Cabo
CUNY New York City College of Technology

Reneta Lansiquot
CUNY New York City College of Technology

Integrating Creative Writing and Computational Thinking to Develop Interdisciplinary Connections

Dr. Candido Cabo, New York City College of Technology

Candido Cabo is a Professor in the Department of Computer Systems Technology at New York City College of Technology, City University of New York (CUNY). He earned the degree of Ingeniero Superior de Telecomunicacion from the Universidad Politecnica de Madrid (Spain) in 1982, and a Ph.D. in Biomedical Engineering from Duke University (Durham, NC) in 1992. He was a post-doctoral fellow at Upstate Medical Center, State University of New York (Syracuse, NY), and a research scientist in the Department of Pharmacology at the College of Physicians and Surgeons of Columbia University (New York, NY). Since 2005, he has been a member of the doctoral faculty at the CUNY Graduate Center. His research interests include computer science and engineering education and the use of computational models to understand and solve problems in biology.

Dr. Reneta Davina Lansiquot, New York City College of Technology

Reneta D. Lansiquot is Associate Professor and Program Director of the Bachelor of Science in Professional and Technical Writing at New York City College of Technology of the City University of New York, where she earned an AAS in Computer Information Systems and a BTech in Computer Systems. She earned a MS in Integrated Digital Media at Polytechnic University and a PhD in Educational Communication and Technology at New York University. Her research focuses on interdisciplinary studies. Her first book is entitled Cases on Interdisciplinary Research Trends in Science, Technology, Engineering, and Mathematics: Studies on Urban Classrooms (Information Science Reference, 2013). Her two forthcoming books are entitled Interdisciplinary Pedagogy for STEM: A Collaborative Case Study (Palgrave Macmillan) and Technology, Theory, and Practice in Interdisciplinary STEM Programs: Connecting STEM and Non-STEM Approaches (Palgrave Macmillan).

Integrating Creative Writing and Computational Thinking to Develop Interdisciplinary Connections

Abstract

A typical college curriculum does not make it easy for students to establish connections between required general education courses and courses in their majors. Intentional linking of courses from different disciplines using interdisciplinary pedagogical strategies allows students to make those connections while developing the interdisciplinary skills which will benefit their college and post-college careers.

In addition to communication, critical thinking and reasoning, and collaborative skills, it has been recently argued that computational thinking (i.e., the application of computing concepts and methods to solve problems) should also be a part of a twenty-first century liberal education for a broad range of college students, including those not majoring in computing. Computational thinking concepts and skills can help students frame problems in a variety of fields and disciplines (not just STEM disciplines) using novel strategies, and, in so doing, to become better problem solvers in their professions.

At our institution, many students not majoring in computing (or a STEM discipline) take a first-year problem-solving with computer programming course (PS), which is designed for Computer Science majors, to satisfy the computer literacy/fluency requirement in their degree or to learn computational thinking concepts and skills. However, since PS is a gateway course for Computer Science majors, it is even more challenging for non-majors, resulting in high non-passing and withdrawal rates. To integrate computational thinking in required liberal arts courses, we created a general education interdisciplinary course, *Programming Narratives: Computer Animated Storytelling*, aimed at non-computer majors, which emphasizes creative writing and computational thinking. In this interdisciplinary course, students learn the structure of narrative, concepts of problem solving, and the logic of computer programming languages as they develop a narrative-driven video game prototype. This process helps students achieve the college-wide learning goal of making meaningful and multiple connections among the liberal arts majors, as well as between the liberal arts and the areas of study leading to a major or profession.

Our findings suggest that the learning objectives and the pedagogical approaches used in the course are adequate for a broad range of non-computer majors. Performance on writing and computing assessments as well as final grades (75% of students obtained a grade of C or better) indicated that a vast majority of students successfully achieved the learning objectives. These results were consistent with student perceptions as reflected in an end-of-course survey. There is also evidence that students satisfactorily integrated creative writing and computer programming to develop their video game prototypes, making in-depth interdisciplinary connections along the way. We believe that this intentional emphasis on connections between disciplines develops the interdisciplinary skills and perspectives which are important for graduation, and it lays the groundwork for interdisciplinary thinking in the workplace.

1. Interdisciplinary Learning in Undergraduate Education

Undergraduate degree programs consist of courses for a major and general education courses. Each degree program has learning outcomes mapped to the outcomes of its courses. Ideally, students establish synergistic connections among the different courses in the curriculum. However, abundant evidence suggests that transfer of skills between courses is relatively rare.[1-3] Students often do not make connections between general education courses and courses in the major, or among courses in the major.

While many factors hinder this transfer of learning, part of the problem may be the instructor's pedagogical approach.[2] Conversely, when instructors use reflective writing, contextualize learning experiences, and apply learning to real life,[3] they help students make connections among disciplines. However, regardless of the strategies used, curriculum and course design must emphasize the connections among courses to stimulate the transfer of learning.

The primary strategy used at our institution to connect courses is the first-year learning community (LC) model, in which a group of students enroll in two or more courses, generally in different disciplines, linked by a common theme in an academic semester. For over 10 years, the academic performance of our students participating in LCs reflects national trends.[4,5] To develop further an interdisciplinary culture, our institution now requires students to complete an interdisciplinary general education course. This emphasis on connections between courses and disciplines helps in developing interdisciplinary skills and perspectives important for degree completion while laying the groundwork for interdisciplinary thinking in the workplace upon graduation.[6]

2. Integrating Computational Thinking in a Liberal Education

Computing permeates all disciplines and areas of knowledge, and so it could be argued that computational thinking—including the application of computing concepts and skills to solve problems, not only in Computer Science but also in other disciplines—should be a part of a twenty-first century liberal education for a broad range of college students, including those not majoring in computing.[7,8] Computational thinking can help students to frame problems in STEM disciplines and a variety of other fields, and, in so doing, to become better problem solvers in their professions.

Currently, many students not majoring in Computer Science at our institution take the first-year PS course, designed for Computer Science majors, to satisfy the computer literacy requirement in their degree or to learn computational thinking concepts. However, since PS is a gateway course for Computer Science majors, it is even more challenging for non-majors. In a recent assessment of computer programming concepts and skills, 44 percent of Computer Science majors taking the PS course without being a part of a LC demonstrated an adequate understanding of computer programming concepts and could write viable programs. When computer majors take PS as part of a LC, the percentage of students with adequate performance increases to 56 percent.[9] In contrast, only 30 percent of non-majors taking PS courses for non-majors that are not part of a LC perform adequately in computer programming concepts and skills. These results of the assessments above indicate that teaching computational thinking concepts and skills to non-majors requires pedagogical strategies which are different than those that may work with Computer Science majors.

Two issues are important in designing either a computational thinking course, or a course that introduces computational thinking elements, for a broad range of college students, including non-majors: (1) what knowledge and skills students are expected to acquire in the course, and (2) what learning context and pedagogical approach to use in order to make computational thinking more accessible.[7] In our view, a computational thinking course should include a combination of procedural and object-oriented programming concepts, including the steps required in using computers to solve a problem and the use of flowcharting techniques and such programming structures as sequencing, repetition loops, and decision statements to solve an algorithm. It is also important that students are introduced to concepts of object-oriented programming, such as classes, objects, properties, and methods. The selection of the learning context and pedagogical approach used to teach those concepts results from our experience in linking writing and computer programming in the interdisciplinary LC.[10-13] Just as an interdisciplinary context linking writing and computer programming was beneficial for Computer Science majors, so can it also contribute to facilitate the learning of computational thinking concepts and skills for non-majors.

3. Interdisciplinary Creative Writing and Computational Thinking Course for Non-Majors

We have developed an interdisciplinary course, *Programming Narratives: Computer Animated Storytelling* (PN), designed to help non-computer systems major students develop computational thinking skills through computer programming combined with writing skills to satisfy the college requirement of an interdisciplinary liberal arts and science course.[14]

The PN course combines the perspectives and methodologies of two academic disciplines, English and Computer Science, in pursuit of a common goal. The common goal is to create a narrative-driven videogame prototype so students can identify with that which is immersive, engaging, and rewarding. To complete this video game prototype, students need the perspectives and methodologies in two distinct academic disciplines, English and Computer Science. Students rely on the perspectives and methodologies learned in the English component of the course to develop a story. The reading of various kinds of short narratives is very valuable in helping the students make the kinds of connections necessary to recognize synergies between writing stories and writing programs. As part of the course, students read, annotate, and discuss short narratives of various kinds (e.g., short stories, myth, plays, fantasy, horror, science fiction, historical fiction, and quest narratives as well as open-ended stories) and apply appropriate narrative structures to the construction of their video game prototype. As students study the structure of narratives and learn problem-solving strategies for writing, they are introduced to concepts of problem solving using constructs of logic inherent in computer programming languages. Students then implement the story as a computer program with the perspectives and methodologies learned in the computing component. Such gained knowledge facilitates creative writing and the application of solutions to computer programming problems. The distinct perspectives and methodologies in English and Computer Science are presented by two faculty members who are experts in their fields and co-teach the course.

Implementing a narrative into a machine-executable computer program is a complex task. Students are challenged to map the structure of their narrative, including character and setting development, into constructs of logic inherent in computer programming languages. We expect these challenges to give students more insight into both their creative writing processes as well as their computer programming writing processes. Throughout the semester, students are challenged to understand, think critically to solve writing and computing problems, analyze narrative structure, compare and contrast stories, and apply various narrative structures to their project. Students work collaboratively on this group project to create a video game prototype and an accompanying game design document. The game design document describes the project and discusses elements of analysis and design. Moreover, students prepare and revise an annotated bibliography to facilitate their ability to make connections across academic disciplines. This strategy requires students to write one paragraph to summarize and assess narrative structure of classic short fiction and reflect on assigned course readings as they relate to interactive storytelling.

The computational thinking component of the course introduces procedural and object-oriented programming concepts. Students are introduced to the use of flowcharting techniques and programming structures, such as sequencing, repetition loops, and decision statements, to solve a problem with an algorithm. It is also important that students are introduced to concepts of object-oriented programming, such as classes, objects, properties and methods. In the current implementation of the course, students use *Alice* (www.alice.org) to write the computer program to implement their narratives. *Alice* is helpful because it allows students to implement animations and video games using procedural and object-oriented programming constructs.[15-18]

4. Student Performance

In spring 2015 and fall 2015, we offered four sections of the PN course with a total of 91 students. Twelve students (13%) withdrew from the course and were not considered in the analysis below. Of the 79 students who completed the course, 59 (75%) obtained a grade of C or better, which we consider adequate performance.

4.1. Creative writing

We assessed student understanding of narrative and game structures as well as their ability to write effectively (n=79). An assessment performance of 70% (equivalent to a C) or above was considered adequate. Students were prompted to write an essay to analyze the narrative structure of *Oedipus the King* using Aristotle's concept of "Unity of Action" and apply the Hero's Journey plot structure.[19] They were also tasked with explaining how these literary concepts relate to the structure of video games (i.e., how they could be translated into an action-adventure or role-playing game). About 71% of students demonstrated an adequate understanding of narrative and game structures and were able to write effectively.

4.2. Computational thinking

We assessed student understanding of procedural and object-oriented programming concepts (n = 79). An assessment performance of 70% (equivalent to a C) or more was considered adequate. Students' understanding of object-orient programming concepts was assessed by asking students to create a concept map, relating concepts like classes, object, methods, instances, conditions, and so forth. Students were asked to create a node for each concept, link related nodes/concepts, and label the nature of the relationship. Students' understanding of procedural programming was assessed by giving students a piece of code created with *Alice*, and asking them to create a flowchart (including sequencing, selection (if/else). and repetition constructs) that represents the computer code. Overall, about 57% of students demonstrated an adequate understanding of computer concepts.

5. Student survey

We analyzed student perceptions using an end-of-course survey. The surveys were completed (at least in part) by 66 of the 79 students who completed the course. About 80% of students agreed or strongly agreed with each of the following statements: "I understand the various narrative structures"; "I understand the structure of video game stories"; "I understand the steps required to solve a problem with a computer"; "I understand concepts of object-oriented programming"; and "I can program using sequencing, selection and repetition structures" (Figure 1).

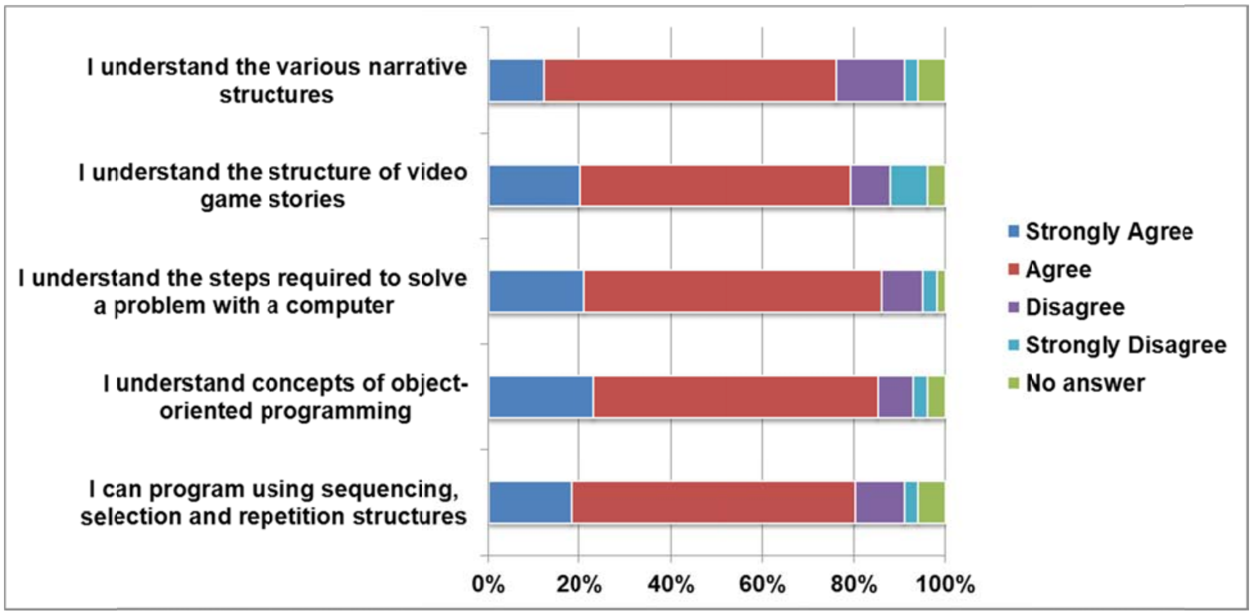

Figure 1. Student survey responses.

Students satisfactorily integrated creative writing and computer programming to develop their video game prototypes, making in-depth interdisciplinary connections along the way. Specifically, 62% of students responded affirmatively to the question, "Were you able to make interdisciplinary connections between writing stories and writing code?" 20% responded negatively, and 18% of students did not answer the question (Figure 2).

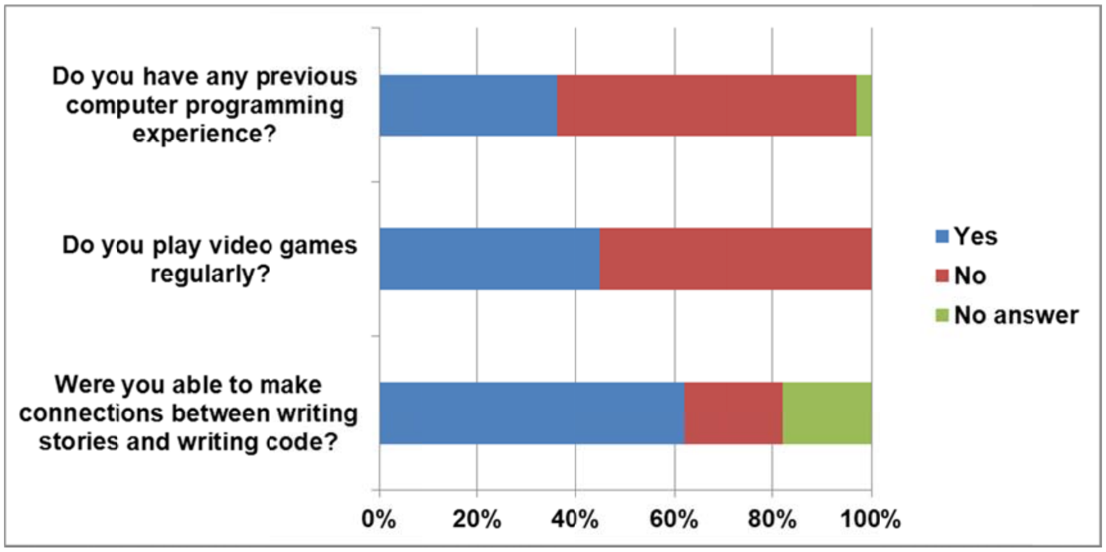

Figure 2. Student survey responses (continuation).

Further analysis of survey responses revealed that students who reported playing video games regularly in their personal life were more readily able to make these connections (Table 1). Of the subgroup (45% of all students) who played video games regularly, 70% indicated that they were able to find interdisciplinary connections, in contrast to only 56% of those who did not play video games. Of the 20% who did not find interdisciplinary connections, 77% were not gamers. The relationship between playing video games and interdisciplinary awareness is intriguing and merits further study.

Table 1. Student gamers making interdisciplinary connections.

		Interdisciplinary Awareness and Connections			
		Yes	**No**	**No Answer**	**Total**
Play Video Games	**Yes**	21	3	6	30
	No	20	10	6	36
	Total	41	13	12	66

6. Conclusions

We believe that interdisciplinary competence should be an integral part of undergraduate education. Interdisciplinary courses help students make connections between courses in general education and their majors, and among courses in their majors. Our experience with designing, developing, and teaching an interdisciplinary course linking creative writing and computational thinking has shown us that such courses help students make these connections. In the course, students develop original stories which they later implement as a video game prototype using computer programming. This interdisciplinary approach seems to be effective in teaching computational thinking concepts and skills to non-computer majors. Moreover, students are able to make interdisciplinary connections between creative writing and computational thinking. There is an intriguing relationship between video game playing and interdisciplinary awareness.

References

1. Barnett, S. & Ceci, S (2002). When and where do we apply what we learn? A taxonomy for far transfer. *Psychological Bulletin*, *128*(4), 612-637.
2. Benander, R., & Lightner, R. (2005). Promoting transfer of learning: Connecting general education courses. *The Journal of General Education*, *54* (3), 199-208.
3. DeCorte, E. (2003). Transfer as the productive use of acquired knowledge, skills, and motivations. *Current Directions in Psychological Science*, *12*(4), 142-146.
4. Assessment and Institutional Research. (2014). *CUNY Student Experience Survey*. New York City College of Technology, CUNY.
5. Kuh, G. D. (2008). High-impact educational practices: What they are, who has access to them, and why they matter. Washington, D.C.: AAC&U.
6. Lansiquot, R. D., & Cabo, C. (2015). Strategies to integrate writing in problem-solving courses: Promoting learning transfer in an interdisciplinary context. In *Proceedings of the 122nd American Society for Engineering Education Annual Conference*. Washington, DC: ASEE.
7. Guzdial, M. (2008). Paving the way for computational thinking. *Communications of the ACM*, *51*(8), 25-27.
8. Wing, J. M. (2006). Computational thinking. *Commun*ications *of the ACM*, *49*(3), 33-35.
9. Cabo, C., & Lansiquot, R. D. (2014). Synergies between writing stories and writing programs in problem-solving courses. In *Proceedings of the 2014 IEEE Frontiers in Education Conference* (pp. 888-896). New York: IEEE.
10. Cabo, C., & Lansiquot, R. D. (2013). Development of interdisciplinary problem-solving strategies through games and computer simulations. In R. D. Lansiquot (Ed.) *Cases on interdisciplinary research trends in science, technology, engineering, and mathematics: Studies on urban classrooms* (pp. 268-294). New York: Information Science Reference.
11. Lansiquot, R., & Cabo, C. (2014). Strengthening the narrative of computing with learning communities. In *Proceedings of World Conference on Educational Media and Technology 2014*. Chesapeake, VA: AACE.
12. Lansiquot, R., & Cabo, C. (2010). The narrative of computing. In *Proceedings of World Conference on Educational Media and Technology 2010* (pp. 3655-3660). Chesapeake, VA: AACE.
13. Lansiquot, R. D., Satyanarayana, A., & Cabo, C. (2014). Using interdisciplinary game-based learning to develop problem solving and writing skills. In *Proceedings of the 121st ASEE Annual Conference*. Washington, DC: ASEE.
14. Lansiquot, R. D., & Cabo, C. (2016). Making connections: Writing stories and writing code. In R. D. Lansiquot (Ed.), *Interdisciplinary pedagogy for STEM: A collaborative case study*. New York: Palgrave.
15. Cooper, S., Dann, W., & Pausch, R. (2000). Alice: A 3-D tool for introductory programming concepts. *Journal of Computing Sciences in Colleges*, *15*(5), 107-116.
16. Cooper, S., Dann, W., & Pausch, R. (2003). Using animated 3D graphics to prepare novices for CS1. *Computer Science Education*, *13*(1), 3-30.
17. Lansiquot, R. D., & Cabo, C. (2011). Alice's adventures in programming narratives. In C. Wankel & R. Hinrichs (Eds.), *Transforming virtual learning: Cutting-edge technologies in higher education* (Vol. 4, pp. 311-331). Bingley, UK: Emerald.
18. Mullins, P., Whitfield, D., & Conlon, M. (2009). Using Alice 2.0 as a first language. *Journal of Computing Sciences in Colleges*, *24*(3), 136-143.
19. Campbell, J. (1949). *The hero with a thousand faces*. Navato, CA: New World Library.