Raspberry Pi and Rubber Ducks: Digital Pedagogy and Computer Anxiety in the LIS Classroom

Lindsay Kistler Mattock

The information professions demand digitally literate practitioners who can navigate the increasingly complex technological environments of libraries and archives. While basic literacy is presumed among graduate students, many students express fear and anxiety when approaching new technologies. These psychological barriers have not been addressed in the digital literacy discourse, yet they must be overcome if LIS educators are tasked with teaching future professionals to critically engage and evaluate the myriad technologies available for library systems and services. This article describes the introduction to computing course at University of Iowa that uses the Raspberry Pi (RPi) computer as a means for teaching critical digital literacy skills in the LIS curriculum. Designed to afford students opportunities to peer into the black box of the computer and build the confidence and knowledge to engage with unfamiliar technologies, the course adopts active learning pedagogy using the RPi. This article discusses the design and development of the course and presents the self-reported data from students regarding their perceived gain in experience and comfort with computing over the course of the term. While further research is required to fully understand the relationship between anxiety and computing, the findings suggest that adopting a critical digital pedagogy that focuses on the process over the end product provides space within the learning environment for students to take risks and develop heuristics for overcoming anxiety and building literacy skills.

Keywords: computing anxiety, critical making, digital literacy, digital pedagogy, LIS education

Many graduate students exhibit the knowledge and skills necessary to communicate via email, build documents with common word-processing software, navigate the web, and use mobile devices, but they often lack the knowledge of how these technologies work. While such students may be considered digitally literate, they understand computers as black boxes accepting input and producing output—the mechanics of the machine hidden behind the sleek packages of mobile devices, laptops, and desktops. This deeper engagement with technology can be anxiety producing for students who lack previous experience with programming or computer science. Yet library and information science demands students that can critically engage with technology, looking beyond the mass-produced packages to assess the utility and appropriateness of pre-packaged hardware, software, and the services offered by developers and vendors. Employers seek candidates who can manage databases, build websites, query large datasets, and critically evaluate the ever-changing catalog of new tools available for library systems and services. The American Library Association's Core Competences of Librarianship (2009) acknowledges the need for a robust technological skill set, stating that all persons graduating from accredited programs must be able to use, apply, and assess technologies utilized within the context of the library and information sciences. The question becomes, how should these digital literacy skills be taught in LIS classrooms?

Vol. 63, No. 2 DOI: 10.3138/jelis-2018-0062

This article presents one approach to teaching this critical understanding of computing technology adopted in the introductory computing course at the University of Iowa. This required course aims to familiarize students with the technologies they will encounter throughout the curriculum while modeling a heuristic approach to confidently engage new and unfamiliar technologies. The course includes a discussion of markup languages, programming, web design, and networking, while adopting an experiential pedagogy that aims to build a critical understanding of how each technology works. Engaging a critical making practice, students are asked first to create through practice-based exercises and then to reflect on their work. By emphasizing the process over the end product, this approach encourages students to think deeply and critically about their use of technology, while building their confidence to adopt and apply new technologies in LIS practice.

KEY POINTS:

- Despite demonstrating basic digital literacy, high levels of computing anxiety were found among first semester LIS students. While students were comfortable using email and other familiar software, computers remained "black boxes" to many students.
- Implementation of a novel classroom technology, the RPi computer, combined with critical making practices and metacognitive strategies for reflecting on learning, allowed students to overcome psychological barriers to engaging with new technologies.
- The pedagogical strategies employed in this course design afforded students opportunities to develop heuristics for engaging with new hardware and software. Students reported an increase in their perceived comfort and experience with both new and familiar technologies and higher confidence with troubleshooting and instructing others.

The myth of the digital native

Reflecting on digital humanities curricula, a cognate to library and information science similarly entangled with computing technology, Professor Alexander Reid (2012) observed that most graduate students are novices, having little exposure to digital tools and methodologies during their undergraduate education. This is true of the students completing the Master's of Library and Information Science; many students enter the program with undergraduate degrees from humanities disciplines, with few reporting exposure to basic computing concepts and technologies. While the students are comfortable navigating the learning management systems and the front end of many common software packages, they lack a basic understanding of how the technologies work behind the screen.

The myth of the digital native, a phrase coined by Marc Prensky (2001), however, has led to a more commonly held presumption that current generations of students are techologically adept, having been immersed in digital culture since their youth. It has been argued that this generation "possess sophisticated knowledge of and skills with information technologies" (Bennett, Maton, & Kervin, 2008, p. 778), as well as an expectation to use information technologies throughout their educational experiences, mirroring the embeddedness of technology in their daily lives (Frand, 2000). While the discourse surrounding the "information age" has perpetuated this mythology, empirical studies have demonstrated

that educational level, socio-economic status, access to technology, and locality are much more likely to affect a student's ability and comfort with digital tools (Akçayır, Dündar, & Akçayır, 2016; Kirschner & De Bruyckere, 2017). The ability to use a computer and expectations regarding the use of technology in educational settings cannot simply be generalized across generational divides. However, it is clear that the current generation of LIS professionals must be prepared to engage with digital technologies in their practice (Choi & Rasmussen, 2009; Gerolimos & Konsta, 2008; Kennan, Cole, Willard, Wilson, & Marion, 2006; Lynch & Smith, 2001).

The American Library Association (2009) includes "Technological Knowledge and Skills" among the Core Competences of Librarianship for students completing an ALA-accredited program. Graduates must be familiar with common technologies in the field, apply technologies according to norms and ethical standards, assess and evaluate the use of technology in praxis, and identify emerging trends. Rather than identifying a common set of technologies for LIS practice, the Core Competences emphasize the need for professionals to think with and use technology in context.

Several studies of LIS job postings have confirmed that while employers often list domain-specific knowledge of particular tools in advertisements, hiring agencies seek applicants with the generalized skills and competences acknowledged by the ALA. Choi and Rasmussen's (2009) study of academic library positions found qualifications matching the ALA's Core Competences among the top four desired skills for applicants. While many specific skills were included in the studied advertisements, over half of the postings included a knowledge of technological practices and standards, general literacy skills, and an understanding of the role of technology in LIS practice as general requirements.[1] Further studies have suggested that other personal and generic skills remain more desirable over specific digital skills. Arguing that digital skills have become artificially inflated in the current technological climate, these studies also identified the ability to work under pressure, be flexible and adaptable, and display critical-thinking and problem-solving skills as desirable qualities for applicants (Howard, 2010; Nonthacumjane, 2011).

The fact remains that LIS professionals cannot avoid working with digital tools and must develop some technological ability with databases, metadata, user interfaces, and content management systems if they are to be successful in the field. However, focusing solely on the myriad technologies listed in job advertisements does not build the competences that LIS professionals will need to remain nimble and flexible as technologies continue to adapt and change to the needs of users and collections. In response to these pressures, the introductory computing course at the University of Iowa attempts to balance the need to teach specific technologies that students will encounter in their coursework and professional practice, while embracing these broader goals.

Critical digital pedagogy and the Raspberry Pi: Designing the course

Clearly, a computing classroom needs computers. Like many LIS programs, the School has several computing labs available to students in both classroom spaces and common areas throughout the building. While these resources could be employed for instruction, the instructor assigned the Raspberry Pi computer as the course "textbook." The Raspberry Pi

(RPi) is a credit-card sized, single-board computer developed by the Raspberry Pi Foundation as a cost-effective tool for teaching digital literacy, particularly among young children. The introductory video "What Is Raspberry Pi" begins, "back in the 80's kids had to learn how to code computers to use them, and as a result these kids grew up with an inbuilt understanding of how computers work" (Raspberry Pi Foundation, n.d.), reflecting the black-boxing of computers that the tool is designed to combat. While most of the supplementary materials provided by the Foundation are aimed at the K–12 audience, the RPi has also been adopted by makerspaces, hobbyists, and in higher education settings, although more commonly within engineering and computer science curricula. The RPi is designed to be used like any standard desktop or laptop computer, but the flexibility of the machine allows it to be utilized for a variety of projects, from hosting databases to the Internet of Things (Bruce, Brock, & Riser, 2015). In addition, the low price point of about $35 makes the RPi more affordable than the average college textbook.[2]

While the RPi may be a popular new tool, the materiality of learning and critical digital pedagogy discourses caution against the blind adoption of novel technology in the classroom. Instructors must also consider "the *design of digital tools and their affordances, physical spaces, physical and virtual environments and the services and digital information within these environments*" (Kirschner & De Bruyckere, 2017, p. 140). Sociologist Estrid Sørensen (2009) observes that the computer is one of the many actors in the classroom; instructors adopting interventions must attend to the socio-material arrangements between the materials and technologies in the classroom (from computers to pencils) as well as the human actors (both instructors and students). As one of many elements in the larger learning environment, the pedagogy of the RPi must also align with the instructor's teaching philosophy, the course objectives, assignments, and overall course design.

This computing course is designed to work against the black-boxing of technology that has allowed users to understand the computer as a simple input and output device, leaving the inner workings of the machine a mystery to most. Unlike the devices that students encounter in their daily lives, the RPi is an open box. The processors, circuits, and ports are plainly visible on the credit card–sized board, inviting opportunities to discuss the hardware that drives the machine. This simple exposure of the inner workings of the computer allows students to reflect on the materiality of computing, as one student observed: "Without all the small bits of metal and whatnot that makes up the Pi, none of the functionalities that I worked with for this project would be possible." Similar observations from students after their first encounter with the RPi led to discussions of the physicality of "the cloud" as a server farm composed of clusters of computers rather than as an intangible construct for digital storage. The physicality of the RPi combined with the first lesson focused on binary encoding affords students opportunities to consider the materiality of the seemingly ephemeral bits and bytes stored within the computer and question the immateriality of the digital.

Unlike tablets, phones, and computers, the RPi comes out of the box as a blank machine without an operating system. For the second project, students install the Raspbian operating system, based on an open-source Linux distribution, on their newly purchased RPi's, building "from scratch" what often comes off the shelf as a pre-packaged and self-contained piece of technology. Not only does this configuration support the full customization

of the computer and the user experience, but this insistence on open-source technology also affords conversations regarding the politics of ownership, openness, access, and the digital divide. Students could easily install the same software and emulate a Linux OS on a virtual machine on their own laptops or lab computers, but the RPi offers a different relationship to the computer. The hardware and software are open for experimentation and customization; there is no need to "jailbreak" this machine or contact university IT administrators to make modifications to the system.

In addition to these technological affordances of the RPi, the low cost of the tool allows students to take full ownership of the computer without the fear of crashing or breaking expensive personal computers or university-owned machines. This ownership allows students to feel comfortable experimenting with the RPi without the fear of compromising the computers they regularly rely on to complete coursework for other classes.

While the RPi is clearly designed to afford this type of engagement, the coursework and learning environment must match the values of the machine. The affordances of the course management system, the physical layout of the classroom space and virtual spaces used by distance students, and the students' expectations regarding this course and graduate education were also considered. While designing the course, the instructor carefully considered the learning environment created for the students, acknowledging her role as a maker and learner in the classroom alongside the students.

The course description invites each student to "tinker, play, build, make, tweak, experiment, hack, and break things." Students are also encouraged to push their boundaries, ask questions, collaborate, and even fail. To create this collaborative and playful atmosphere, students are greeted with a small rubber duck during the first class meeting. While reviewing the syllabus and course expectations, students are introduced to the concept of rubber duck debugging—a think-aloud problem-solving methodology in which a programmer talks through the problems in the code with an inanimate object to reveal the errors in their logic. This "simple act of explaining, step-by-step, what the code is supposed to do often causes the problem to leap off the screen and announce itself" (Hunt & Thomas, 1999, p. 95). Students are encouraged to talk through errors and problems with their newly acquired rubber ducks or to use their classmates or instructor as a surrogate for the duck when they need additional support. The rubber ducks are an indispensable technology, reminding students that it is ok to "fail," as one student reflected at the end of the term: "[the rubber duck is] a token to show that it is normal and perfectly acceptable to get frustrated in the weeks to come." The ducks become constant companions for students, sitting next to the RPi's in the classroom, appearing at the edge of the table when help is required, and relieving tensions when problems arise. Referenced frequently in assignments, the ducks also add a humorous element as students celebrate successes and express frustration with imaginary squeaks, high-fives, and flights across the room, serving as reminders that troubleshooting is a natural part of the learning process.

However, in addition to these tokens, the design of the course also assures students that there is room to "fail." The students need to gain mastery over the content while also building troubleshooting skills necessary to engage with unfamiliar technologies. The course is designed around seven different projects. Students are provided with a set of instructions

that guide them through the completion of the project at their own pace. The self-paced projects allow students to control their learning experience, while the accompanying assignments are graded on the students' ability to reflect on the learning process, rather than their ability to successfully complete the project (instructions are available at www.lindsaymattock.net).

The first project, titled "Binary, Bits, and Basics," explores the relationship between binary code and the representations of the code on the screen through a discussion of ASCII encoding and the interrogation of various file formats in a hex editor. This initial project allows students to familiarize themselves with the format of the class without adding in the additional unknown of the RPi and to discuss the technical details of how computers translate the 1's and 0's of binary code into meaningful information on the screen. The remaining lessons are completed using the RPi, beginning with the installation of the Raspbian OS, building a LAMP server, writing HTML and CSS with the Geany IDE, encoding information in XML, writing basic Python scripts, and installing WordPress. To monitor student progress, each week students complete a "check-in" assignment during the last 15–20 minutes of class. This open-book, open-neighbor quiz includes a few short-answer questions regarding the major concepts introduced that week, along with two open-ended questions regarding the "most interesting" and "muddiest" point from the class session. Students receive full credit as long as they attempt to answer the questions in their own words. Incorrect answers are not penalized; instead, the quizzes are designed to become a conversation between student and instructor, affording opportunities to clarify misunderstandings and confusions from the weekly lessons.

In addition to the Check-In, students submit a Lab Notebook and Reflection following the completion of each of the seven projects. The Lab Notebook details the process of how each project was completed, including documentation of any questions, problems, or "aha moments" along the way. Accompanying this process report, students are asked to write a short reflection describing how the project impacted their understanding of computing and its application in the LIS field. As with the Check-In, the grade is not reflective of whether the project was succesfully completed, but of the student's ability to describe what they learned, where errors occurred, and how they attempted to troubleshoot along the way. The assignments document the students' individualized experiences and the growth of understanding rather than testing their proficiency with particular technologies.

Mirroring the flexibility of the RPi, the course is designed to afford students the potential to build on their knowledge of specific technologies that they would continue to use in their coursework and professional practice while also encouraging experimentation and troubleshooting, the general skills that will ensure that students are prepared to engage with technologies outside of their comfort zones. However, this course structure also forced the instructor to take a critical look at her role in the classroom. While the instructor could anticipate many of the errors and missteps that students might take, the number of variables involved in configuring a LAMP server or writing a Python program are too great to account for. Instead, these become teachable moments where troubleshooting steps can be demonstrated. By walking through the process of revisiting the project instructions, support manuals, or technology forums such as Stack Overflow, students learn how to troubleshoot

errors themselves, becoming more reliant on their own knowledge as they complete the projects. These moments may reveal the limitations of the instructor's knowledge but allow students to see the process of learning in action.

Putting theory into practice: Testing the design

This course design was first implemented for the fall 2016 offering. Student activities and assignments served as the data points for studying the students' experiences and attitudes as they completed the course. When surveyed at the beginning of the term, over half of the 33 students enrolled in the course expressed a desire to build basic skills, while less than 10% of students named specific technologies that they would like to learn to make themselves more marketable as LIS professionals (see Figure 1). Students understood the demands for professionals in the LIS fields to be adaptable and flexible, as one student reflected: "while I'm looking forward to learning the basics in a lot of different areas, I'm most hoping to develop thinking patterns and skills that will assist me whenever I'm faced with new technology."

Many students also noted a general anxiety or fear of computers. While this apprehension could be associated with the normal anxiety experienced by students beginning a new degree program, students directly correlated their anxiety to a lack of knowledge about computers.

When surveyed about their thoughts or feelings at the beginning of the term, 81% of the students described their emotional state as "nervous," "scared," or "anxious" or noted a general apprehension about the course. Some commented on their lack of knowledge: "I am nervous because I don't know a lot about the basics of computers." Students noted a general difficulty with computing, suggesting that they were "not particularly good with technology," or that "the science of computing might be beyond me," or "it is hard for me to get comfortable with new technologies sometimes," while others simply stated that they were "luddites." Several cited previous negative experiences with computing courses in both high school and undergraduate programs. The anxiety was not related to particular tools or software; rather, students expressed a general dislike or discomfort with computing. Yet,

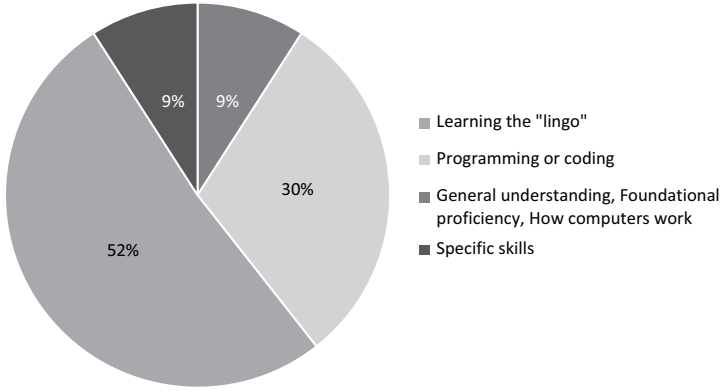

Figure 1: Responses to the question "What skills do you want to develop this term?"

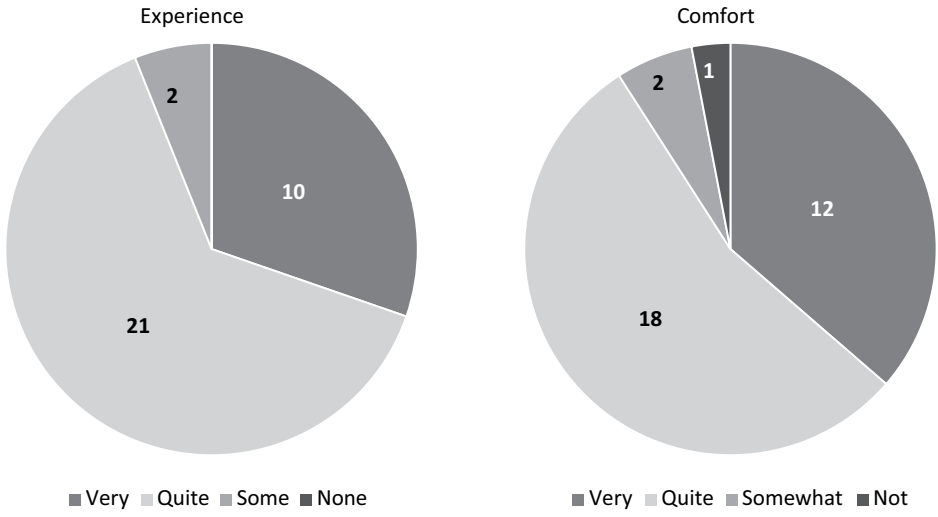

Figure 2: Student-reported experience and comfort with the Microsoft Office or similar software (*N* = 33)

when asked what technologies they engaged with at home or in their professional lives, students listed an assortment of computing technologies, including desktops, laptops, tablets, smart phones, game consoles, televisions, printers, and smart home devices.

While students were clearly consumers of technology, their unease indicated a lack of knowledge regarding how these technologies work. Anticipating a critical engagement requiring them to code, they expressed apprehension and fear. This discomfort was reflected in the responses to questions asking them to rate their experience and comfort with a list of common technologies, including those covered in the course: Microsoft OS, Mac OS, iOS, Android, Raspberry Pi, Microsoft Office (or similar software), Linux/Unix command line, computer programming (generally), web development, HTML, XML, Python, and WordPress. Figure 2 shows the distribution of the student responses regarding their comfort and experience with Microsoft Office or a similar software package.

Since this is a commonly used suite of programs, the question was designed to provide a gauge for comparing students' perceived comfort and experience levels with familiar technologies to the technologies that would be addressed in the course. The reported scores for the Office suite demonstrated that many students were at least somewhat experienced with the software and had some level of comfort using the tool, with 94% of students reporting that they were "quite" or "very" experienced with the software, and a similar number of students reporting high levels of comfort. Although experience and comfort scores did not correlate exactly, the distribution of reported scores across these two measures were similar for common operating systems and software. As the questions moved to less-familiar technologies, such as the Command Line Interface (CLI) and computer programming (see Figures 3 and 4), the reported scores demonstrated that students lacked experience working outside of the Graphical User Interface (GUI) of standard operating systems. Similar levels

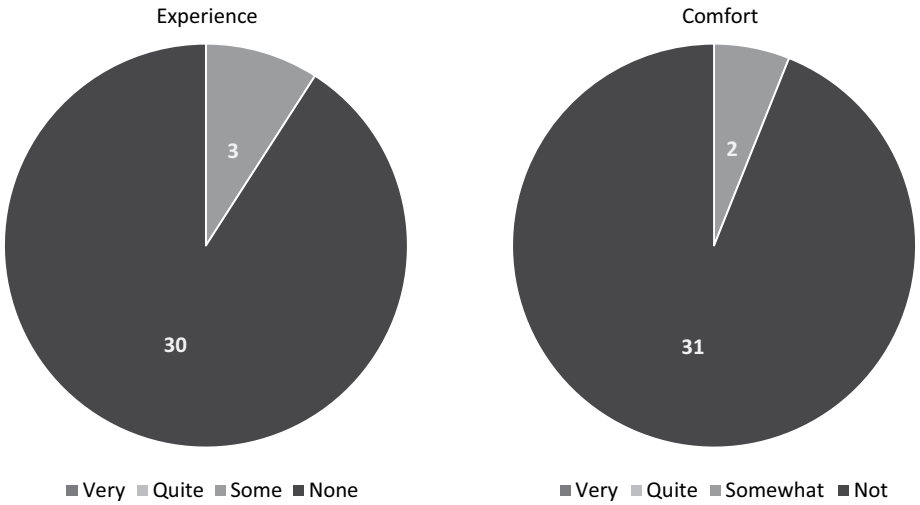

Figure 3: Student-reported experience and comfort with the command line interface: Beginning of term (*N* = 33)

of comfort and experience were reported for the specific technologies that would be used in the course: Raspberry Pi, HTML, XML, Python, and WordPress. At the beginning of the term, no students reported a comfort or experience score higher than "some," with the majority (on average 87%) reporting no comfort or experience with these technologies. HTML was the most familiar, with 11 of the 33 students indicating that they had some experience with HTML (only 9 reported some comfort). These statistics not only demonstrate the need for the introductory course but also suggest that students had little experience engaging with computers beyond the Graphical User Interface. Like many digital natives, students were taught to use computers to produce emails and documents, navigate the internet, and install software and applications, but they had not peered into the black box to discover how the machine works. Further, this lack of experience accompanied a lack of comfort with the technology that was producing a sense of fear for students entering the course.

Computing anxiety

These responses were somewhat surprising. The instructor presumed that students would be unfamiliar with many of the technologies that would be introduced in the course but had not anticipated the level of anxiety reported by students. Computing anxiety, however, is not uncommon. Studies regarding attitudes toward computing were prevalent in the mid-1980s as computers became common in the home, professional, and academic spaces. Defined as "emotional fear, apprehension and phobia felt by individuals towards interactions with computers or when they think about using computers" (Chua, Chen, & Wong, 1999, p. 610), computing anxiety cannot be correlated to any one root cause. Studies have explored a variety of variables, including gender, computer experience, personality type, age, learning style, profession, educational background, ownership, and training (Powell, 2013).

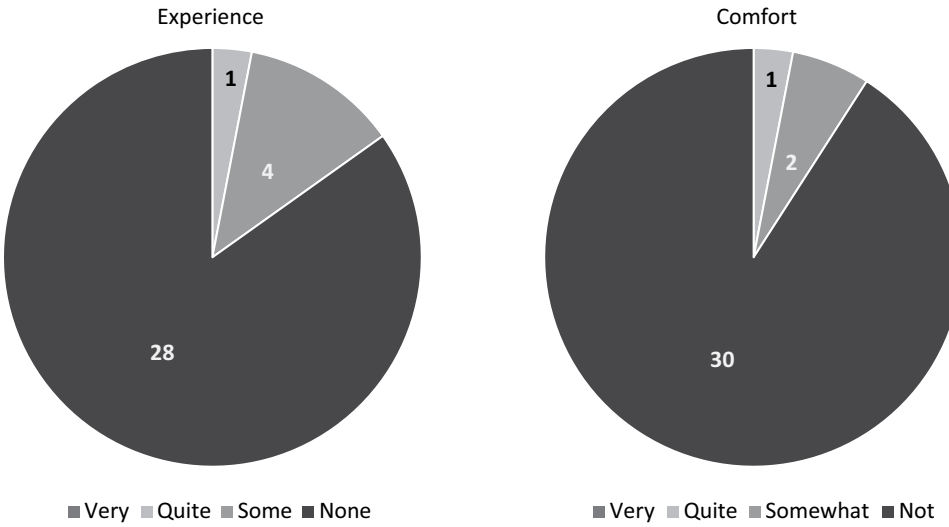

Figure 4: Student-reported experience and comfort with computer programming: Beginning of term (*N* = 33)

Overall, the most significant correlations have been found between attitudes toward computing—those with more positive attitudes toward computers are less likely to report anxiety (Kay, 2008; Popovich, Gullekson, Morris, & Morse, 2008)—and competency—those with lower competency levels report higher levels of anxiety (Orr, Allen, & Poindexter, 2001).

However, most of these studies focus on the same digital competencies expected of digitally literate individuals, that is, the ability to use software to generate documents, spreadsheets, databases, and presentations. The populations of concern in these studies were adopting the use of computers in their workplaces, academic spaces, or for personal use and were interacting with the machine at the GUI interface; they were not required to code or to work at the command line interface but simply to accept the computer as a new tool. Negative emotions were more commonly associated with the subject's perceived ability to use the software, and anxiety was directly correlated with the individual's knowledge and competency of each subject. The students' reported comfort and experience scores reflect the findings from the literature; students were more comfortable with familiar technologies. While the computing anxiety and digital literacy discourse provide little guidance about how to overcome these psychological barriers in the classroom, the design of this course afforded students space to overcome their fears, build domain-specific skills, and develop positive attitudes toward computing and the confidence to engage with new software and hardware in their professional posts.

Overcoming anxiety

The course structure reflects a need for students to build a heuristic for critically engaging new technology. The experiential approach emphasizes the learning process, rather than the nuances of particular technologies, while the course projects afford an opportunity

to explore beyond the polished interfaces of contemporary computers. Each project was designed to look behind the interface and explore the computer as a constructed tool. Students first install the LAMP software that will allow their RPis to host webpages as servers. This work requires students to configure a firewall, install MySQL and PHP, and configure the server settings using the Linux command line. The project opens discussions about cloud computing, the Internet of Things, and the history of computing, as CLI is compared to the more familiar GUI interface. Other projects focused on HTML, XML, and Python ask students to code using the Geany IDE, building websites, encoding text, and writing Python scripts "from scratch." The hands-on experience with coding affords opportunities for students to understand how websites are constructed and designed and how data are manipulated and stored in digital form. While these lessons provide only a brief introduction to the basic concepts for each technology, the projects offer a new perspective on the digital environments that we consume. The final project requires students to install the self-hosted version of WordPress on their RPi servers. Students are encouraged to explore the software and to look at the CSS used to style the WordPress themes, the PHP files that generate the dynamic content, and the databases that store the contents of the site. This exercise demonstrates how the elements of the LAMP server, HTML and CSS, and programming languages work together in a self-hosted content management platform. While many students in the fall 2016 offering of the course were not yet comfortable enough to manipulate the CSS and PHP files, they reported that they were at least familiar with the contents and could draw on their knowledge from the previous projects that introduced these technologies.

Students were given step-by-step instructions for each exercise and encouraged to use the resources available to them, including the instructor, teaching assistant, and their classmates, yet the majority of students approached each new project with apprehension. However, as they became more familiar with the structure of the course, the expectations for the assignments, and the RPi, they reported more frustration than anxiety. The students' emotions were associated with their unfamiliarity with the computing environments and a general lack of confidence in their abilities.

The work in the Shell or the Command Line Interface (CLI) was the most unfamiliar and challenging territory for many students. At the CLI, users must rely on text-based commands and navigate the interface using the keyboard rather than the icon-based interface and point-and-click of the mouse in the GUI. Frustrated by the unforgiving syntax of the shell commands, students suggested that computing was a natural ability rather than a learned skill, as one student reflected: "I am often reminded that I have no natural gift when it comes to working with computers." However, by the final week of the project, students described an increased comfort with the CLI and growing confidence that allowed them to share their experiences and assist one another. When students worked through their anxieties and fears, they also demonstrated an ability to grapple with the bigger picture and begin to peer into the black box:

> I also think this project has given me a better understanding of what actually goes into the process of connecting to the web and how websites are available for other computers to

> access. It's one thing to read about the process in [the textbook], even with [the author's] very helpful diagrams and visual representations, but it is another thing entirely to initiate the process on one's own and follow each individual step.

This confidence and understanding extended beyond the classroom as well. One student reported that after the first project they were more confident with troubleshooting other problems, such as helping a relative with their home internet connection. Where this student would have called on the expertise of a spouse before, they reported that they were confident enough to take on the challenge on their own after the successful completion of the project in the classroom.

While confidence increased, students continued to report frustrations and anxiety throughout the term. The anxiety related to computing seemed to be different from the fears related to other challenges in the students' lives:

> This was such a tough few weeks and it really taught me, most importantly, about how I handle stress in technological situations. I've noticed that the anxiety produced by working on the terminal is different that anxiety in other aspects of my life.

Students also found correlations to other situations: "I noticed the same terminal-related command-line-infused anxiety when playing board games or video games, which makes me think that this type of activity (command line, programming, etc.) should be treated as a puzzle." The key to overcoming this anxiety was related not only to the opportunities to engage with and build comfort with a new technology but also to the ability to reframe these fears into a positive experience and reflect on the learning process.

Computing anxiety and digital literacy

When surveyed at the end of the term, students reported a higher level of comfort with technology. This result correlates with the findings from the studies of computing attitudes and anxiety that have found that with instruction, experience, and exposure, anxiety and negative feelings toward computing decrease. A comparison of Figure 6 to the beginning of the semester scores in Figure 5 clearly demonstrates the increase in students' comfort over the course of the term. Whereas the majority of students reported that they were not comfortable with the listed technologies, they now responded with a wider range of scores. As a survey course, students did not have the time to build expertise with each of the technologies introduced, but they felt more confident to continue to work with these tools: "After the past several weeks of HTML, XML, and Python I think my biggest takeaway is a higher level of comfort. My skill level is still pretty basic, but I know where to go for help and I know the kinds of things to search for." But, what is most interesting is that technologies not specifically addressed in the course also showed an increase in comfort and experience scores. Students used word-processing software to generate their assignments, but the software included in common office suites were not directly addressed in the class projects. Yet, as Figure 7 demonstrates, the number of students reporting that they were "very comfortable" with the software increased from 36% to 60% of students by the end of the term, with an increase in the students' reported comfort with other technologies as well.

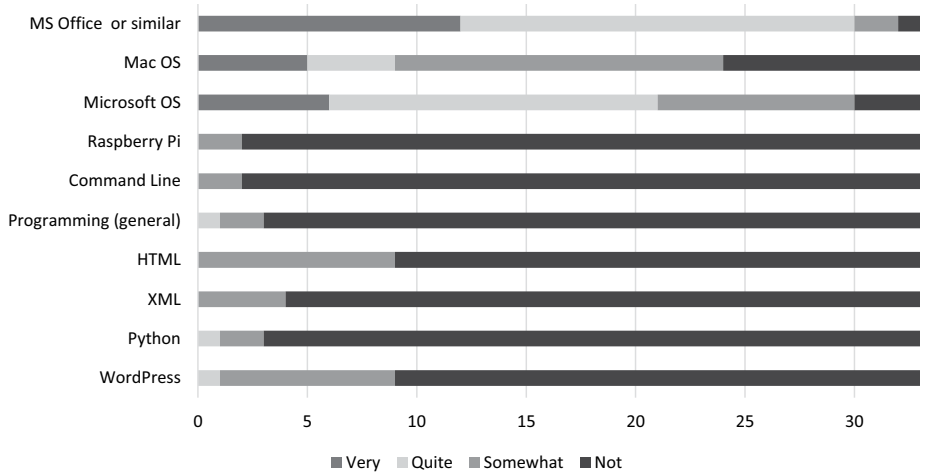

Figure 5: Student-reported comfort at beginning of term: All technologies (*N* = 33)

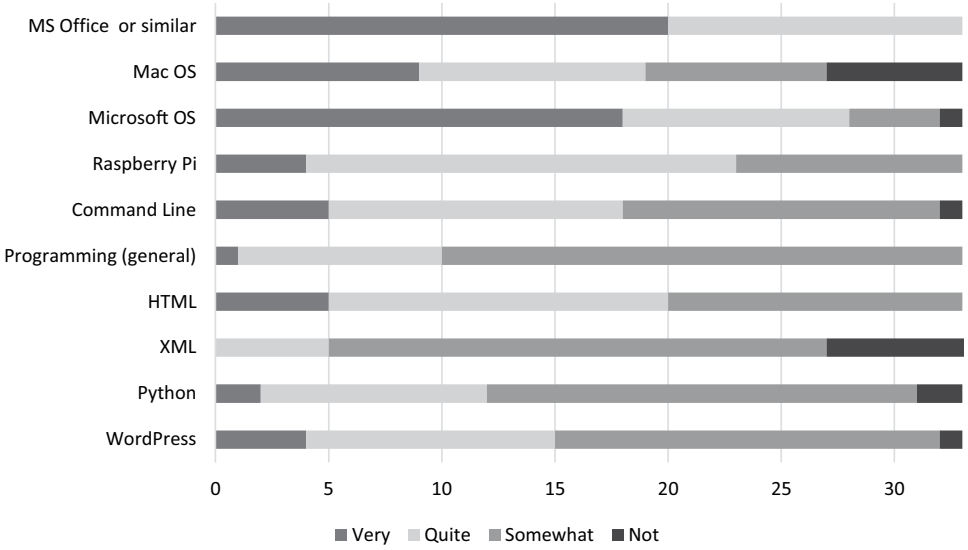

Figure 6: Student-reported comfort at end of term: All technologies (*N* = 33)

Comparisons of the students' reported experience and comfort scores also reflect the findings from studies of computing anxiety. While experience and comfort scores do not directly match, Figures 8 and 9 demonstrate a close correlation between the students' perceived level of experience working with a technology and their perceived comfort, suggesting that as students continue to work with these technologies, their fears and anxieties

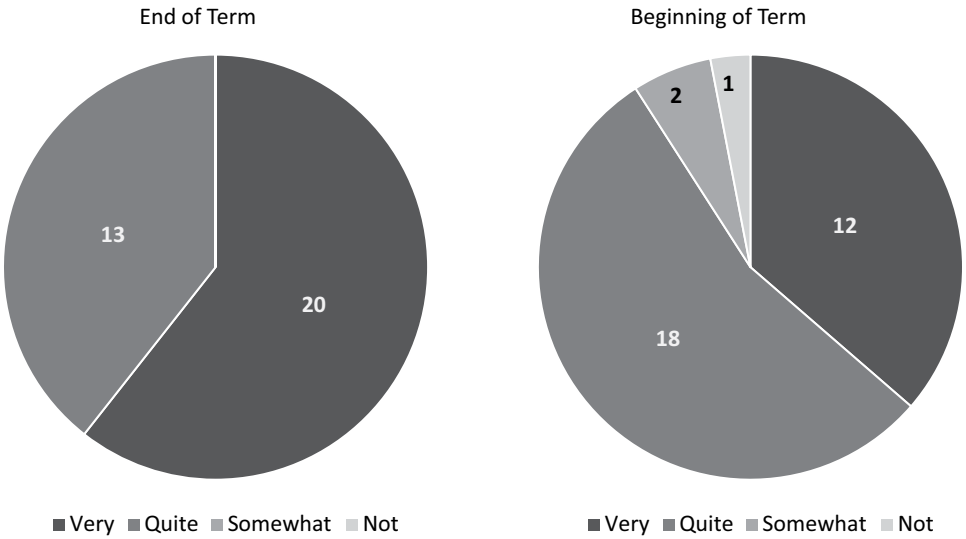

Figure 7: Student-reported comfort with Microsoft Office or similar: Beginning and end of term comparison (*N* = 33)

will continue to dissipate. The student assignments also reflected this increase in comfort, with students reporting that they were more confident in their skills and more willing to experiment without the guidance from the instructor at the end of the term:

> Can I just take a moment to say that again, I . . . the girl who fears computers secretly hate her . . . was helping my entire row [of classmates] during this last exercise. Not once in my wildest dreams did I expect that to happen. I mean, I wasn't able to solve all their problems, but I was actually helpful in some cases.

Additionally, a number of students reported that they were now confident enough to adopt the RPi in their professional practice or to continue using the RPi for personal projects.

The RPi afforded an engagement with the technology that was perceived as different from students' experiences working with traditional computers:

> I think I enjoy working with the Pi in this class because it feels as though we are doing real, tangible things—projects that have practical implications—rather than simply writing programs that draw pictures or solve easy math problems like I did in the computer science class I took in college. People may say that the Pi is a toy, but that's the best way to learn how to do something—to play around with it.

But, as this student's response suggests, it is not the RPi alone that afforded the positive experiences in the classroom. The RPi is one of many technologies that can be employed to teach basic computing skills to a range of students at different age and skill levels. The course design also had to reflect the values represented in the design of the RPi. The careful design of the course incorporated active learning strategies that allowed students to move through the course materials and projects at their own pace and opportunities to reflect on

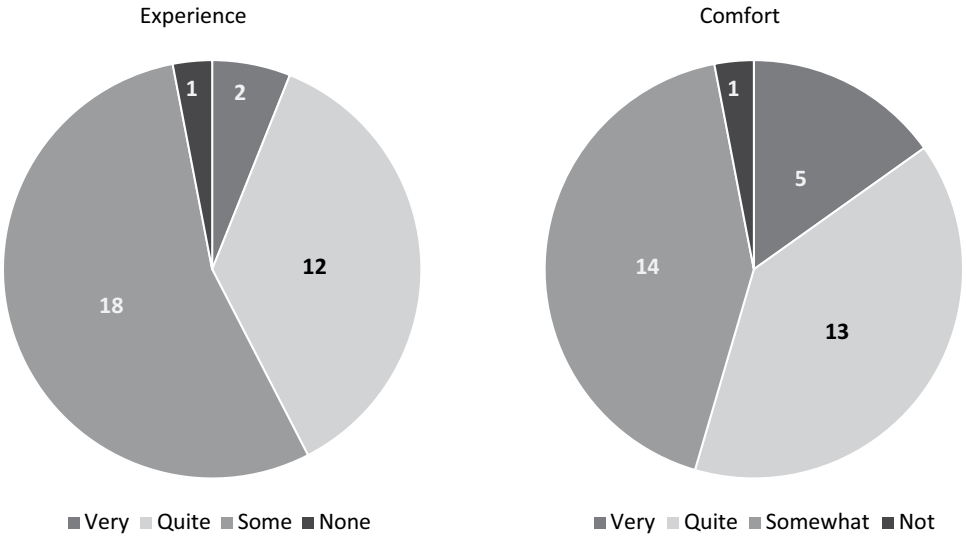

Figure 8: Student-reported experience and comfort with the command line interface: End of term (*N* = 33)

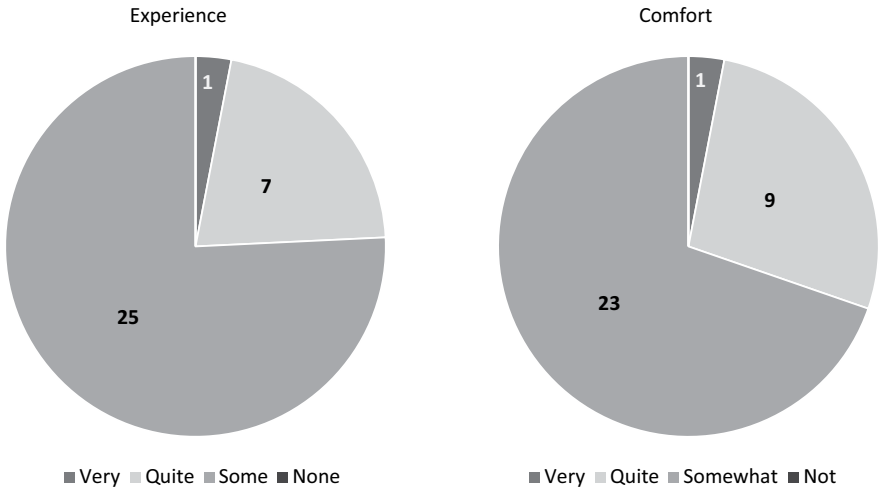

Figure 9: Student-reported experience and comfort with computer programming: End of term (*N* = 33)

their learning. Assignments provided room for experimentation, affording opportunities for students to "fail" and attempt to troubleshoot solutions without risking their grade. The instructor also acknowledged her role as a facilitator and learner in the classroom, modeling strategies for engaging with technology and reflecting on the psychological barriers to the learning process.

The notebook assignments and weekly "Check-In" quizzes were critical feedback mechanisms for tracking progress for both students and the instructor. One student reflected:

> Reading my notebook and doing the last check-in has really made me reflect on this last exercise and the process around it. I can actually see where I started and the progress I have made. . . . I'm practicing, and my progression is showing in the notebook.

The critical making praxis afforded by the projects asked students to critically engage the technologies that they consume daily: "I found these labs interesting because while I use websites daily, many with questions and forms, I had never really considered the code that went into the creation, layout, and function of the webpages I visited."

Conclusion

The Raspberry Pi is designed as a tool for teaching twenty-first-century skills to students who have not peered into the black box of their personal computers and digital devices. The price point of the RPi makes it affordable for adoption in classrooms or for personal use, and it costs considerably less than the textbooks required for other courses in the LIS curriculum. However, access to technology is not enough. As the reflections from the students have demonstrated, these spaces must also be designed to encourage experimentation and provide support to novice users that may not have the knowledge or confidence to openly and critically engage with these tools. An open and critical interrogation of the design of the classroom that considers the materiality of the entire learning environment is required if such endeavors are to be successful. This course design aimed to build critical literacy skills for LIS professionals in training but revealed the psychological barriers that have impeded learning in computing classrooms. As a pilot study, this experience has suggested more questions than answers. More research is required to better identify the sources of computing anxiety that students bring into the classroom; however, the interventions employed by this course design have demonstrated some success in overcoming these emotional and psychological barriers to learning in the computing classroom.

Lindsay Kistler Mattock, *School of Library and Information Science, University of Iowa*, is an associate professor at the University of Iowa School of Library and Information Science. Her work centers on community archives. Her digital project *Mapping the Independent Media Community* seeks to understand how the historical conditions surrounding independent media production have influenced contemporary archival praxis. Email: lindsay-mattock@uiowa.edu

Notes

1. Other specific competences with technologies are named: mark-up languages, programing/scripting languages, web servers, and computer operating systems, among others.
2. The National Association of College Stores reported in 2018 that the average textbook price in 2015–16 was $80 for a new book and $51 for used (https://www.nacs.org/research/HigherEdRetailMarketFactsFigures.aspx; accessed August 30, 2018).

References

Akçayır, M., Dündar, H., Akçayır, G. (2016). What makes you a digital native? Is it enough to be born after 1980? *Computers in Human Behavior*, *60*, 435–440. https://doi.org/10.1016/j.chb.2016.02.089

American Library Association (ALA). (2009). Core competencies of librarianship. Retrieved from http://www.ala.org/educationcareers/sites/ala.org.educationcareers/files/content/careers/corecomp/corecompetences/finalcorecompstat09.pdf

Bennett, S., Maton, K., & Kervin, L. (2008). The "digital natives" debate: A critical review of the evidence. *British Journal of Educational Technology*, *39*(5), 775–786. https://doi.org/10.1111/j.1467-8535.2007.00793.x
Bruce, R. F., Brock, J. D., Reiser, S. L. (2015). Make space for the Pi. Proceedings of the IEEE SoutheastCon 2015. https://doi.org/10.1109/SECON.2015.7132994
Choi, Y. & Rasmussen, E. (2009). What qualifications and skills are important for digital librarian positions in academic libraries? A job advertisement analysis. *Journal of Academic Librarianship*, *35*(5), 457–467. https://doi.org/10.1016/j.acalib.2009.06.003
Chua, S. L., Chen, D.-T., & Wong, A. F. L. (1999). Computer anxiety and its correlates: A meta-analysis. *Computers in Human Behavior*, *15*(5), 609–623. https://doi.org/10.1016/S0747-5632(99)00039-4
Frand, J. L. (2000). The information-age mindset: Changes in students and implications for higher education. *EDYCAUSE Review*, *35*, 15–24.
Gerolimos, M. & Konsta R. (2008). Librarians' skills and qualifications in a modern informational environment. *Library Management*, *29*(8/9), 691–699. https://doi.org/10.1108/01435120810917305
Howard, K. (2010). Programming not required: Skills & knowledge for the digital library environment. *Australian Academic & Research Libraries*, *41*(4), 260–275. https://doi.org/10.1080/00048623.2010.10721480
Hunt, A., & Thomas, D. (1999). *The pragmatic programmer: From journeyman to master*. Reading, MA: Addison-Wesley Professional.
Kay, R. H. (2008). Exploring the relationship between emotions and the acquisition of computer knowledge. *Computers and Education*, *50*, 1269–1283. https://doi.org/10.1016/j.compedu.2006.12.002
Kennan, M. A., Cole, F., Willard, P., Wilson, C. & Marion, L. (2006). Changing workplace demands: What job ads tell us. *Aslib Proceedings*, *58*(3), 179–196. https://doi.org/10.1108/00012530610677228
Kirschner, P. A., & De Bruyckere, P. (2017). The myths of the digital native and the multitasker. *Teaching and Teacher Education*, *67*, 135–142. https://doi.org/10.1016/j.tate.2017.06.001
Lynch, B. P., & Smith, K. R. (2001). The changing nature of work in academic libraries. *College and Research Libraries*, *62*(5), 407–420. https://doi.org/10.5860/crl.62.5.407
Nonthacumjane, P. (2011). Key skills and competencies of a new generation of LIS professionals. *IFLA Journal*, *37*(4), 280–288. https://doi.org/10.1177/0340035211430475
Orr, C., Allen, D., Poindexter, S. (2001). The effect of individual differences on computer attitudes: An empirical study. *Journal of Organizational and End User Computing*, *13*(2), 26–39. https://doi.org/10.4018/joeuc.2001040103
Popovich, P. M., Gullekson, N., Morris, S., & Morse, B. (2008). Comparing attitudes towards computer usage by undergraduates from 1986 to 2005. *Computers in Human Behavior*, *24*(3), 986–992. https://doi.org/10.1016/j.chb.2007.03.002
Powell, A. L. (2013). Computer anxiety: Comparison of research from the 1990s and 2000s. *Computers in Human Behavior*, *29*(6), 2337–2381. https://doi.org/10.1016/j.chb.2013.05.012
Prensky, M. (2001). Digital natives, digital immigrants: Part 1. *On the Horizon*, *9*(5), 1–6. https://doi.org/10.1108/10748120110424816
Raspberry Pi Foundation. What is Raspberry Pi. Retrieved from https://www.raspberrypi.org/help/videos/#what-is-a-raspberry-pi
Reid, A. (2012). Graduate education and the ethics of the digital humanities. In M. K. Gold (Ed.), *Debates in the digital humanities* (pp. 350–367). Minneapolis, MN: University of Minnesota Press.
Sørensen E. (2009). *The materiality of learning*. New York, NY: Cambridge University Press.