In what ways can paired programming as a teaching and learning strategy facilitate High Quality Learning and Teaching?

Alex Johnson
University of Cumbria

Abstract

Key stage five A level students often find programming to be a challenging and inhibiting factor. This research focused on how paired programming as a teaching strategy could facilitate high quality learning while also increasing students' perceptions of their own programming ability. The data collected for this research was done through Interviews that were conducted prior to the pairings and then after the pairings, open questions were used in the pre-interview, these focused on the student's own feelings towards their ability and confidence towards programming. The pairings lasted for five 50-minute lessons. The post pairing interview used open questions that discussed key characteristics of good paired programming, students were asked to discuss what these characteristics meant to them, and if they felt they had experienced or exhibited these during the pairing, as well as the importance of them. The educational setting was an Inner London sixth form. The findings from this research show that there were real benefits for most participants of the pairings, with female participants identifying in more detail the characteristics from the post interview in a paired programming setting. All students of middle and low ability found the pairings to be beneficial, however some were reluctant to undertake paired programming on a more frequent basis.

Introduction

Computer programming has traditionally been perceived as an individual endeavour. Through my experience in teaching A level Computer Science I frequently see students become disengaged with programming as the complexity of learning increases. When students transition from year 12 to year 13 there is more taught programming content delivered, as well as the expectation that students spend more time programming independently outside of the classroom. This research will investigate whether using a paired programming approach when teaching a more complex aspect of programming will enable students of middle to low ability in programming make the expected progress and not become disengaged with their learning to the point where they do not program independently outside the taught classroom environment. Students will learn the Object-Oriented Programming Paradigm (OOP). OOP is a big transition at A level, when students first start their A level, they are introduced to programming through a procedural programming paradigm. When transitioning to OOP students struggle to change their programming approach and fail to develop their OOP skills. I have observed that students of middle to low ability in programming are likely to struggle more to transition, many students feel the concepts of OOP to be hard to implement, because of their basic understanding. I have observed however that some middle ability students, when working collaboratively, understand the more nuanced aspects of OOP. This has led me to undertake this practitioner research project. The literature studied indicates that paired programming can have transformative benefits to students who struggle with programming.

Literature Review

Paired Programming

Paired programming is common practice within software development. It is categorised as an 'Agile' software development paradigm. The main reason for it being classified as Agile is the adaptive nature of the process. It is human-centred, as the paired programmers work alongside each other in a focused and supportive role (Choi, 2007). This approach to software development, which is normally classified as Extreme Programming (XP) has many benefits within the software development sector, these benefits include, faster code production, better quality code, improved knowledge and enhanced learning (Hannay, 2009). The basic premise of paired programming is two programmers working together on one computer. Each programmer takes it in turn to write code, they are called the "driver" while the other programmer reviews the code being written, they are called the "navigator" the programmers frequently swap roles during the process. It is the role of the navigator to offer up suggestions/improvements to the code as it is being generated by the driver.

One of the fundamental components of successful paired programming is collaboration, rather than cooperation. It has been identified that cooperation does not always mean collaboration. Coman (2014) identified that the boundary between cooperation and collaboration is often blurred. They identify that collaboration can only occur if all participants agree upon and share a mutual goal. While participants might cooperate to complete a given task, it is no guarantee that they will collaborate as well.

The biggest factor for paired programming to be successful is the approach taken to pairing. Zhong et al (2016) identified skill, personality, gender and experience as all being important factors to consider. While an obvious approach would be to pair up programmers of different skill levels, Cao and Xu (2005) identified that pairing weaker programmers with stronger programmers led to a mis-match in their learning, the stronger programmer, while cooperating with the task, tended to avoid collaboration with the weaker programmer. DeClue (2003) found that students who worked with a stronger programmer felt inferior to the stronger programmer, and these stronger programmers felt that the weaker programmers slowed them down in producing their code. The literature read clearly indicates that pairing subjects based on similar ability levels produces a more successful and enjoyable experience (Hanks, 2006).

For pairings to be successful Chao and Atli (2006) identified five key attributes that each member of the pairing needs to have, of these five traits being open minded and creative were deemed the most important, while being attentive, responsible and having good logical thinking were also required. However, Carver et al (2007) and Grundy (2009) found that personality types or traits had no clear impact on the success/failure on paired programming pairs.

The role of gender in paired programming pairs has largely been ignored, Hanks et al (2011) identify that the role of gender, specifically women is often only mentioned as an afterthought. Maguire (2014) found that pairing women together led to greater confidence and ability in programming skill. There was little mention of pairing women with men in any literature read.

Experience and understanding of the roles within paired-programming is directly linked to the success of pairings. Without any guidance or instruction of the roles students need to undertake, they are likely to fail in the production of the required work. Somervell (2006) and Williams et al (2008) identify that training students on how to pair program before implementing it within the classroom is more beneficial to both student and educator, than just allowing the students to undertake the task with no training.

Quality of Teaching

It is generally agreed that high quality teaching leads to high quality learning (Gore, 2017). When trying to identify what high quality teaching is Hattie (2008) identifies that quality teachers are visible within their classrooms to their students. Hattie (2008) is explicit that visible teachers are not didactic in their approach, rather they become immersive in the learning themselves, they activate the learning for the students, then direct the flow of the learning. While these are clear attributes that a teacher could have, they are by no means a clear indicator of what high quality teaching is.

Ramsden (1992) identifies that high quality teaching is grounded within its context, and that the teaching is continuously improvable. Coe et al (2014) identify six common components of great teaching, based on multiple frameworks.

1. Subject knowledge is essential for all teachers, however, to be an effective teacher, a practitioner needs deep subject knowledge, as well as a keen understanding of how the students will perceive the material, along with how to address the common misconceptions that arise from teaching the content. As OOP is perceived by the students as being a substantive step up in ability, a transitive approach is required to bridge the change from a procedural programming paradigm to the OOP paradigm. Misconceptions are common place when teaching Computer Science, but in my experience both within the classroom and in discussion with other A level teachers, OOP has many more misconceptions, with many based on how hard it is to learn.

2. Quality of instruction within a classroom using previous learning, questioning and scaffolding will allow learners to develop and build their confidence towards the content being taught. When transitioning from procedural programming to OOP many students are reluctant to write their own code independently, it is beneficial as a teacher to model how the main components of OOP work, this allows students to practice writing code and embed their skills.

3. The climate in which students learn needs to be both demanding of the student to push themselves while also respecting a student's self-worth. It must also celebrate the success of all students, whatever the ability, and also promote resilience when something has not been successful. Programming ability can be very divisive within a classroom environment, many lower ability students become detached from the learning when observing fellow peers make quick progress. When teaching OOP, differentiating for all abilities can be very demanding on a teacher's time, and with low resilience towards failure in programming, students become detached from their learning.

4. Classroom management is more than just behaviour, it incorporates timings of lesson activities, use of resources as well as expectations on how students approach and undertake tasks. Embedding a collaborative approach to problem solving in the classroom can benefit students when learning new programming components. In the dynamic model of teaching (Creemers & Kyriakides, 2006), establishing student to student interactions towards learning and developing a student's ability to solve problems can have a positive effect on student outcomes. It is common to pair students together when programming, however the pairings do not always work, this leads to ineffective learning from one or both students within the pairing.

5. Confidence and belief in a teacher's approach to delivering content is a key indicator of how successful the teaching and learning will be (Askew et al, 1997). However, it is worth identifying that no matter how the confident a teacher is towards the content being taught and the learning that needs/is taking place, a student's own self-efficacy will play a fundamental role within their learning (Gorard, See & Davies, 2012). Careful consideration needs to be taken when pairing students within a classroom, as a low ability programmer paired with a high ability programmer, may in fact experience more negative feelings about their ability than positive.

6. A teacher's professionalism is on display in every lesson for all students to see. Adapting approaches to learning, engaging students' parents on the learning their child is undertaking, all help frame how your practice and teaching is perceived. Delivery of lessons can start to form a similar pattern, while this is beneficial regarding classroom management (see point 4) it can also detract from the overall learning environment. Continued Professional Development (CPD) along with being a reflective practitioner allows a teacher to adapt and develop teaching. I hope by utilising a paired programming approach to the teaching of OOP I can learn and develop my own practice, which can be shared within my department, and with other teaching professionals.

Quality of Learning

Quality learning is directly linked to quality teaching. However, there is no prescriptive set of processes that a teacher can follow. The multitude of factors that are ever present in a classroom makes teaching more of an art than a science (Coe et al, 2014). Because learning is symbiotic with teaching, using a framework like Creemers and Kyriakides' (2006) Dynamic Model of Teaching has been shown to deliver a positive impact on a student learning and outcomes.

Methodology

Hegarty (1998) defined good research as research that is relevant to the needs of teachers' intelligence, as this will lead to the best possible outcome for children. This research will be focused on a small-scale practitioner led investigation. As this research is being undertaken to help develop the practice of the researcher it is appropriate to use a paradigm which will generate the most meaningful data.

Positivist approaches to research generate Quantitative data. This can be through the form of controlled experiments, testing & statistical records, the aim of this approach is to be as objective as possible, this detached approach allows the researcher to generalise their findings (Cole McGrath, 2010). In contrast to this, an Interpretivist approach mainly produces Qualitative data. The data generated using a qualitative approach can help show a picture of a social reality, some common data collection approaches used are interviews, observations & questionnaires. While there is a clear correlation to the paradigms and their data gathering approaches, they are not exclusive to these methodologies, and it is common for researchers to adopt a flexible approach to data gathering. (Middlewood, Coleman & Lumby, 1998)

Table 1.1 shows some key features of positive and interpretive paradigms.

Positivist	**Interpretive**
Objective and external view of world	Social and subjective view of the world
Observer is independent of research	Observer is part of the research
Focus on facts only	Focus is on meaning
Using numerical based concepts to measure	Multiple methods to generate different views of situation
Large samples	Small samples
Reduce findings to simplest terms	Looking at overall situation

As this research is focused on developing my teaching proficiency an interpretive approach was most appropriate.

Using interviews as a data collection method for qualitative research allows the researcher to judge the responses given by the participants (Walliman, 2001). If questions are not understood the researcher can support and develop the answer further. This technique is called probing and it allows the interviewer to extract more data from a response, it is essential however that the interviewer does not influence the direction of the answer through probing (Dawson, 2013).

Interviews were used as the primary source of qualitative data. The interview questions were all open-ended questions to allow for each answer to be specific to the participant. Six participants were chosen for interview. The questions in the pre-pairing interview were focused on:

- How the participant felt towards programming in general
- Their own ability in programming
- Their perception of OOP

After the pairings had completed a set of post-pairing questions were asked during a second interview. These questions included the original three questions from the pre-pairing interview, along with additional questions about the characteristics that are required for successful paired-programming, the questions asked:

- What does openness mean to you, have you experienced it during your pairing, did you exhibit this behaviour?
- What role did your skill/experience of programming play in how successful your pairing was when completing the tasks?
- Did you use logical thinking a lot during the planning and development stages? How much creative thinking was used for your scenario?
- How do you feel the gender of your pairing had in regards your learning?
- What are feelings towards paired-programming?

The use of the interview questions enabled two sets of data to be produced. To add authenticity to this data, the student's assessment and programming practical's data was also used for triangulation. Triangulation of data is important as it allows the researcher to compare different accounts of the same situation (Altricter, 2008). Triangulation is a one-phase design where two types of data are collected in the same time period, in this instance, before the pairing then after the pairing. The responses from both data collection methods are given the same weighting (Punch , 2005). It should be noted that triangulation can lead to bias in qualitative research, Patton (1980) states that triangulation does not bring consistency to data and should not be completely relied

upon. However, for this research, the triangulation of the qualitative data was deemed the most appropriate technique.

Ethical Considerations

Ethics is the study of what is good or right. One branch of ethics is applied ethics, which focuses on complex issues. Research ethics exists within the larger facet of applied ethics. Punch (2005) identified five main issues surrounding social research as harm, consent, deception, privacy and confidentiality.

Harm: the study was undertaken in an environment the students were familiar with, at no point were they exposed to any additional risk from the research being undertaken.

Consent: Consent was granted from the head of school where the research was undertaken, from the University of Cumbria's ethic panel, along with the voluntary consent from participants within the class. All participating students were given a document which explained what the research was about, and identified their rights to either participate or not, and how they can leave the research at any stage.

Deception: All information given within the consent form was adhered to. No additional data was recorded and profiled during the data gathering stage.

Privacy: All data gathered was kept secured and used for its intended purpose. No reference to either participants or the educational setting have made, to ensure complete anonymity.

Confidentiality: All participant responses were recorded in privacy, away from other participants.

Findings

Pre-pairing Interview

All six participants of the interviews were of a middle to low ability regarding their programming and academic performance.

	Question	Themes			
Participant ID	How do you feel about programming?	Frustration	Debugging	Motivation	N/A
1	I enjoy programming in the classroom, especially when the teacher is in the room to offer support. When I program by myself and it doesn't work, I feel frustrated and normally give up.	1		1	
2	I think programming is enjoyable, but only when it works. It is annoying when I spend lots of time fixing the code to make it work.		1		
3	Programming at A level is a lot harder than at GCSE. I struggle to program outside of lessons, because it is hard.			1	
4	I like programming, it is harder at A level. I find it challenging outside of lessons when there is not teacher support.				1
5	I don't like programming, I enjoyed it at GCSE, but it is now harder, and my programs never work, even if I spend lots of time trying to make them work.	1	1		
6	I enjoy programming in lessons. There is more help available. When I program at home I generally just give up if a program does not work.		1	1	

	Question	Themes		
Participant ID	How would you describe your programming ability?	Good	Poor	N/A
1	I think I have a good level of skill in programming. I generally understand what is required.	1		
2	My programming is a lot better than when I did my GCSE's. I understand more about how a program actually works.	1		
3	I think my programming ability is not very good. I have not made much progress since GCSE.		1	
4	My programming is ok, I can generally understand what is required to make a program work.	1		
5	I would say my programming is not good at all.		1	
6	In lessons I think my programming skill is enough, but at home its bad.		1	

	Question	Themes	
Participant ID	What is your perception of Object Oriented Programming?	Hard	N/A
1	I have not heard about it before.		1
2	My cousin does Computer Science at University, he said it was really hard.	1	
3	Only that it is a lot harder than the programming that we do now.	1	
4	I know its popular for making games using C++, and it's difficult to learn.	1	
5	I'm not sure what that is.		1
6	It uses objects based on real world objects, and its tough to write programs for.	1	

Key Findings from pre-pairing interview

From the responses to the questions, debugging a program is a cause of frustration within the participants, their motivation to complete a program is heavily influenced by being present within a classroom environment. There is an equal split across participants about their programming ability, it should be noted here, that all male participants responded positively about their programming ability. There is a generally agreed consensus across participants who know about OOP, that it is hard, although their responses are based on other people's experiences or mis-information.

Overview of pairing lessons

Lesson 1: Students were put into their pairings as identified from their current academic data. Most students were happy with their pairings. A few students asked to swap pairings based on their social interactions with a student in a different pairing, this was declined. The lesson focused on the basic premise of OOP. Several examples were given, students designed real world models in their pairings.

Lesson 2: The students were shown the basic syntax structure and methods used when programming in OOP. The pairings then designed pseudocode programs of their objects; these were then marked by another pairing.

Lesson 3: In their pairings, students created a programmed version of their pseudocode model. Although initially hit and miss, all students managed to undertake the role of driver and navigator.

Lesson 4: The topic Inheritance and Polymorphism was explained using models, pairings then implemented these components within their current program. Driver and navigator roles again were used.

Lesson 5: As in previous lesson, implementation of Polymorphism was time intensive. Pairings shared programs they had created with another pairing.

Post-pairing Interview

The post-pairing interview was broken down into two parts.

For part one, all six participants were asked the same three questions from the pre-pairing interview.

For part two Each participant was then asked four questions based on the identified characteristics that might be exhibited in successful paired-programming, and if they felt they themselves had exhibited this characteristic during the pairing, and if it was important to the success of the pairing.

	Question	Themes			
Participant ID	How do you feel about programming?	Enjoyed pair-programming	Debugging	Support	N/A
1	As before, I enjoy programming in the classroom, especially when there is other students and a teacher available to help. I find it hard to program at home.			1	
2	I still think programming is enjoyable, but only when it works, however working with another student has made it easier to fix a program that does not work.	1	1		
3	I still feel programming is hard, I have enjoyed programming in a pair, as I felt I understood some of the structures I was struggling with.	1			
4	I enjoyed paired-programming and I still like programming on my own, I did find it frustrating when being the person who was coding	1			
5	I still find programming a struggle, it was nice to work in a pair and have someone help me develop a program, but I still don't like programming	1		1	
6	Programming in lessons is still my preferred option, being in a pair was interesting, but I prefer to program by myself.				1

	Question	Themes			
Participant ID	How would you describe your programming ability?	Good	Poor	Improved from pairing	N/A
1	It is still good. In fact, it's better as I now understand OOP	1			
2	I think my programming is just as good as before we started OOP	1			
3	I still think my programming ability is not good, but during the paired task, I felt I produced a good program.		1	1	
4	I think my programming has improved, as I get what OOP is about, and I can write an OOP program.	1			
5	I would say my programming has improved from working in a pair.	1		1	
6	I think my ability is still the same.		1		

	Question	Themes			
Participant ID	What is your perception of Object Oriented Programming?	Syntax	Hard	Practice	N/A
1	Its fairly straight-forward, you just need to plan how you are going to write your program.				1
2	I found it challenging to begin with, but during the practical's in lesson, I found myself getting quicker at writing the code.			1	
3	IT is very different from how we have programmed, I got lots of syntax errors in programs at the beginning.	1			
4	Its is very tricky when we first learnt it, but I enjoyed the practical's, and the pairings made fixing the errors easier.	1	1		
5	You need to practice a lot, because I got lots of syntax errors when we first started.	1		1	
6	It is very hard to write programs using OOP, I am not keen to use this approach to my programs.		1		

Post-pairing interview part 2.

The table below shows the question and responses from all six participants.

Key: Yes  No 

Question		1	2	3	4	5	6
When you think about openness towards working with your paired partner, do you feel you were open to them, and was this important towards how successful your pairing was in completing your programs?	Displayed						
	Important						
Looking at your skill as a programmer, do you feel this was an important factor in how well your programs worked, and did you display your skill during the pairing?	Displayed						
	Important						
From my research, creative and logical thinking are identified as key indicators of good paired programming, during your time in your pair, do you believe you exhibited either of these, and if you did, do you feel they are important characteristics?	Displayed						
	Important						
How important do you feel gender is regarding how successful your pairing was?	Important						

Discussion

Paired-programming

One of the main themes that emerged from the post-pairing interviews was how students felt that paired-programming had led to better programs being produced, for some participants that had less confidence, their responses centred around having a peer support them when being the driver removed some of the frustration surrounding debugging, with one student stating *"Having another student explain the code as I wrote it, allowed me to see what the code was doing".* One student felt having an additional layer of support during the pairing made them *"feel more relaxed"* about their

program not working. Another student mentioned *"collaborating on a program"* while learning about OOP was enjoyable. Coman (2014) also identified that collaboration needs to take place for pairings to be successful. Several students mentioned having "confidence in their solutions" that they had produced in their pairs, this mirrors what DeClue (2003) and VanDeGrift (2003) found.

Not all students found the pairings to be so enjoyable. One student stated that *"I did find it frustrating when being the navigator"* while students were paired of a similar ability, some students found the basic concepts of OOP straightforward and would have naturally progressed at a quicker pace if they had been working independently. Another student also mentioned that *"I found the paired programming ok, but do prefer to program by myself in the lesson"* when probed further the respondent added *"I usually ask for help if I'm stuck, but I enjoy solving the problem more than getting help"*. This is like Chaparro et al (2005) who found that debugging was the least enjoyable aspect of paired-programming.

Understanding of OOP

The main theme from respondents when asked about OOP after the pairing was on the syntax of the program being the biggest inhibiting factor to a successful program. One student stated that the syntax is *"a lot harder to understand at the beginning*" another student mentioned *"getting the main objects written correctly is really annoying"* while the syntax is more structured when using OOP it was also observed that sharing the role of navigator when debugging syntax sped up the process and allowed me to support more students within the lesson.

Another theme that arose when discussing the understanding of OOP was on how practice of writing programs in OOP led to a better understanding. Some respondents were very honest about their approach to programming outside of lesson, one student felt that programming at home was *"pointless, as when the program doesn't work, I cannot get any support",* another student identified that they felt *"more confident"* from working in a pair and had *"practiced"* the content they had learnt from that lesson.

However, two students found writing programs using an OOP approach to be *"hard"* and while the paired programming approach clearly supported one students understanding, this did not transfer into making it more accessible. One student said *"When I was navigating in my pair, I did not actually do much debugging as I did not understand what they were coding".* Hanks (2006) found that while students may produce better programs in a paired situation, when interviewed some students admitted not fully understanding what the code was doing.

Openness to paired-programming

Rather surprisingly was the mixed response towards how being open to your partner during the paired-programming is important. While all participants said they had displayed openness, which they identified as *"discussing", supporting", being patient",* only half the respondents felt it was important regarding how successful the pairing would be.

Programming skill

Most participants felt that their skill/ability level played an important role in the success of their pairing. By pairing programmers of a similar ability, I was interested in seeing if confidence levels would increase as there would be dominant programmer. McDowell et al (2003) showed that pairing students of a similar ability led to better programs and better programming confidence. It is clear this is also the case in this study. Based on previous experience of teaching OOP to middle ability students I felt the programs that were produced were generally more robust.

However, it should be noted that the all-female pairing, while agreeing that skill/ability is important, both felt they did not display their skill. Their responses during the interview mentioned relying on the other person in the pairing. It was observed that they spent a lot of time actively collaborating on their program and did not fully assume the rigid Driver/Navigator roles. I felt this was a great example of high-quality learning, both students had merged their roles through a simple mirroring of each other's strengths and weaknesses.

Creative and logical thinking

All participants felt this was important to success in a paired-programming scenario. The openness of the main task afforded all pairs the opportunity to spend time being both creative with their own example, as well as utilising logical thinking in the creation of the program structure. Two students mentioned how much they *"enjoyed"* planning their program.

Gender

Interestingly, the only participants who felt gender was important to the success of paired-programming was the all-female pairing. It had been identified in the literature studied, that women feel less confident than men in programming courses. In the current setting this is also true, a common response to any programming related task is *"I can't do it", "I don't understand it", "programming Is not my skill".* According to Hanks (2006) female participants of paired-programming reported a greater level of enjoyment towards all aspects of programming. I felt that the pairing of the all-female pair, because of their similar ability levels allowed them to develop their confidence in a situation that may not have been as easily afforded in a male-female pairing.

Drawbacks

While there is a positive result regarding the impact paired-programming has on learning OOP, there are however some drawbacks. The relatively small size of the participants involved in the study, do not best represent all year 12 A level Computer Science students. There is also a possibility of greater bias in some of the participants responses, due the mixed nature of students who attend the sixth-form. Another drawback is the exclusion of High ability students from the interviews, while they participated in the pairings, they did not respond in the interviews. The inclusion of these students may have produced some very different responses in the data. Some of the literature identified the positives of pairing a stronger programmer with a weaker one, this might also have had additional benefits for participants.

Conclusion

The paired-programming lessons were well received by all year 12 computer science students. There was a general agreement that it was more enjoyable writing programs when learning new aspects of programming using this approach. I noted a better approach to completing programming tasks while students were in their pairings. The understanding of the roles and then implementing them was troublesome, and a few students did not feel confident enough to debug other students work to begin with. Using this approach in future teaching is certainly one I will implement; however, I will spend more time developing the students understanding of their roles within the pairings. The findings indicate there was a higher quality of program completed from the pairs.

In the literature it was identified that good teaching is directly linked to good learning, and this investigation sought to see if paired-programming could facilitate both. All participants responded positively regarding their understanding of OOP and the programs they produced. Within my own department, I feel it would be beneficial for both GCSE and A level students to regularly undertake

paired-programming tasks, to help develop a students', understanding, collaboration and communication skills within computer science.

References

Akinola, O. and Ayinla, B. (2014) 'An Empirical Study of the Optimum Team Size Requirement in a Collaborative Computer Programming/Learning Environment', *Journal of Software Engineering and Applications,* 7(12), pp. 1008.

Altintas, T. (2016) 'A peer-assisted learning experience in computer programming language learning and developing computer programming skills', *Innovations in Education & Teaching International,* 53(3), pp. 329-338. doi: 10.1080/14703297.2014.993418.

Askew, M., Rhodes, V., L Brown, M., Wiliam, D. and C Johnson, D. (1997) *Effective teachers of numeracy: final report.*

Bowman-Perrott, L. (2013) 'Academic Benefits of Peer Tutoring: A Meta-Analytic Review of Single-Case Research', *School Psychology Review,* 42(1), pp. 39-56.

Burns, R.W. (2015) 'High-Quality Teaching Requires Collaboration: How Partnerships Can Create a True Continuum of Professional Learning for Educators', *Educational Forum,* 79(1), pp. 53-68. doi: 10.1080/00131725.2014.971990.

Carver, J.C., Henderson, L., He, L., Hodges, J. and Reese, D. (2007) *Increased Retention of Early Computer Science and Software Engineering Students Using Pair Programming.* Washington, DC, USA: IEEE Computer Society, pp. 115.

Choi, K.S., Deek, F.P. and Im, I. (2008) *Exploring the underlying aspects of pair programming: The impact of personality*.

Coe, R., Aloisi, C., Higgins, S. and Major, L.E. (2014) *What makes great teaching? Review of the underpinning research.* Available at: http://www.suttontrust.com/wp-content/uploads/2014/10/What-makes-great-teaching-FINAL-4.11.14.pdf(Accessed: .

Cohen, L. (2011) *Research methods in education.* 7th ed.. edn. London; New York: London; New York : Routledge.

Coleman, S.A. (2011) 'Embedding Inquiry based learning into Programming via Paired Assessment', *ITALICS: Innovations in Teaching & Learning in Information & Computer Sciences,* 10(1), pp. 72-78. doi: 10.11120/ital.2011.10010072.

Coles, A. (2010) *Your education research project handbook.* Harlow : Pearson Education.

Coman, I.D., Robillard, P.N., Sillitti, A. and Succi, G. (2014) *Cooperation, collaboration and pair-programming: Field studies on backup behavior*.

Creemers, B.P.M. and Kyriakides, L. (2006) 'Critical analysis of the current approaches to modelling educational effectiveness: The importance of establishing a dynamic model', *School Effectiveness and School Improvement,* 17(3), pp. 347-366. doi: 10.1080/09243450600697242.

da Silva Estácio, Bernardo José and Prikladnicki, R. (2015) *Distributed Pair Programming: A Systematic Literature Review*.

David, J. (2008) 'Project-Based Learning', *Educational Leadership,* 65(5), pp. 80.

Dawson, C. (2009) *Introduction to research methods : a practical guide for anyone undertaking a research project.*

DeClue, T.H. (2003) 'Pair Programming and Pair Trading: Effects on Learning and Motivation in a CS2 Course', *J.Comput.Sci.Coll.,* 18(5), pp. 49-56.

Denscombe, M. (2012) *Research proposals a practical guide.* Berkshire, Eng.; New York: Berkshire, Eng; New York : McGraw Hill Open University Press.

Echeverría, L., Cobos, R., Machuca, L. and Claros, I. (2017) 'Using collaborative learning scenarios to teach programming to non-CS majors', *Computer Applications in Engineering Education,* 25(5), pp. 719-731. doi: 10.1002/cae.21832.

Edgar Acosta Chaparro, Aybala Yuksel, Pablo Romero and Sallyann Bryant (2005) *Factors Affecting the Perceived Effectiveness of Pair Programming in Higher Education*.

Elton, L. (2008) 'Teachers investigate their work – By Altrichter, Herbert', *British Journal of Educational Technology,* 39(4), pp. 749. doi: 10.1111/j.1467-8535.2008.00870_1.x.

F Punch, K. (2005) 'Introduction to Social Research – Quantitative & Qualitative Approaches', *http://lst-iiep.iiep-unesco.org/cgi-bin/wwwi32.exe/[in=epidoc1.in]/?t2000=023993/(100),* .

Gary, K. (2015) 'Project-Based Learning', *Computer,* 48(9), pp. 98-100. doi: 10.1109/MC.2015.268.

Gorard, S., See, B. and Davies, P. (2012) 'The Impact of Attitudes and Aspirations on Educational Attainment and Participation', .

Hanks, B. (2006) *Student Attitudes Toward Pair Programming.* , Bologna, ItalyNew York, NY, USA: ACM, pp. 113.

Hanks, B., Fitzgerald, S., Mccauley, R., Murphy, L. and Zander, C. (2011) 'Pair programming in education: a literature review', *Computer Science Education,* 21(2), pp. 135-173. doi: 10.1080/08993408.2011.579808.

Hannay, J.E., Dybå, T., Arisholm, E. and Sjøberg, D.I.K. (2009) *The effectiveness of pair programming: A meta-analysis*.

Hattie, J. (2008) *Visible learning: A synthesis of over 800 meta-analyses relating to achievement.* Routledge.

J. Chao and G. Atli (2006) *Critical personality traits in successful pair programming.*

Lan Cao and Peng Xu (2005) *Activity Patterns of Pair Programming.* IEEE, .

Lemons, R. and Helsing, D. (2008) 'HIGH QUALITY TEACHING AND LEARNING: DO WE KNOW IT WHEN WE SEE IT (AND WHEN WE DON'T)?', *Education Canada,* 48(5), pp. 14-18.

Maguire, P. (2014) 'Enhancing Collaborative Learning Using Pair Programming: Who Benefits?', *AISHE-J: The All Ireland Journal of Teaching & Learning in Higher Education,* 6(2), pp. 1411-1436.

Martens, E. and Prosser, M. (1998) 'What constitutes high quality teaching and learning and how to assure it', *Quality Assurance in Education,* 6(1), pp. 28-36. doi: 10.1108/09684889810200368.

May, G.L. (2005) *Collaborative Learning Techniques: A Handbook for College Faculty*, Durham: Wiley Subscription Services, Inc.

McAteer, M. (2013) *Action research in education.* Los Angeles, California : SAGE Publications.

McDowell, C., Werner, L., Bullock, H.E. and Fernald, J. (2003) *The Impact of Pair Programming on Student Performance, Perception and Persistence.* , Portland, OregonWashington, DC, USA: IEEE Computer Society, pp. 602.

Middlewood, D., Coleman, M. and Lumby, J. (2019) *Practitioner Research in Education: Making a Difference*, London:.

Patton, M.Q. (1980) *Qualitative Evaluation Methods.* Beverly Hills, Calif. : Sage.

Plonka, L., Sharp, H., van der Linden, J. and Dittrich, Y. (2015) *Knowledge transfer in pair programming: An in-depth analysis*.

Ramsden, P. (1992) *Learning To Teach in Higher Education.*

Shaw, A. (2018a) *A critical reflection on how immersive learning can be used to facilitate high quality teaching and learning in educational settings.* University of Cumbria.

Shaw, A. (2018b) *A critical reflection on how immersive learning can be used to facilitate high quality teaching and learning in educational settings* University of Cumbria.

Somervell, J. (2006) *Pair Programming: Not for Everyone?* . 01. pp. 303.

Sonya A Coleman and Eric Nichols (2011) 'Embedding Inquiry based learning into Programming via Paired Assessment', *Innovations in Teaching and Learning in Information and Computer Sciences,* 10(1), pp. 72-77. doi: 10.11120/ital.2011.10010072.

Umar, I.N. and Hui, T.H. (2012a) *Learning Style, Metaphor and Pair Programming: Do they Influence Performance?*.

Umar, I.N. and Hui, T.H. (2012b) 'Learning Style, Metaphor and Pair Programming: Do they Influence Performance?', *Procedia - Social and Behavioral Sciences,* 46, pp. 5603-5609. doi: 10.1016/j.sbspro.2012.06.482.

VanDeGrift, T. (2004) 'Coupling Pair Programming and Writing: Learning About Students' Perceptions and Processes', *SIGCSE Bull.,* 36(1), pp. 2-6. doi: 10.1145/1028174.971306.

Vega, C. (2013) 'A scalable and incremental project-based learning approach for CS1/CS2 courses', *Education & Information Technologies,* 18(2), pp. 309-330. doi: 10.1007/s10639-012-9242-8.

Vega, C., Jiménez, C. and Villalobos, J. (2013) 'A scalable and incremental project-based learning approach for CS1/CS2 courses', *Education and Information Technologies,* 18(2), pp. 309-329. doi: 10.1007/s10639-012-9242-8.

Walliman, N. (2011) *Research Methods: The Basics.*

Walton, Richard E., DMD, MS (2013) 'Outcomes', *Journal of Endodontics,* 39(3), pp. S66. doi: 10.1016/j.joen.2012.11.032.

Wang, X. (2017) 'A problem posing-based practicing strategy for facilitating students' computer programming skills in the team-based learning mode', *Educational Technology Research & Development,* 65(6), pp. 1655-1672. doi: 10.1007/s11423-017-9551-0.

Wang, X. and Hwang, G. (2017) 'A problem posing-based practicing strategy for facilitating students' computer programming skills in the team-based learning mode', *Educational Technology Research and Development,* 65(6), pp. 1655-1671. doi: 10.1007/s11423-017-9551-0.

Williams, L., McCrickard, D.S., Layman, L. and Hussein, K. (Aug 2008) *Eleven Guidelines for Implementing Pair Programming in the Classroom.* IEEE, pp. 445.

Zhong, B., Wang, Q. and Chen, J. (2016) 'The impact of social factors on pair programming in a primary school', *Computers in Human Behavior,* 64, pp. 423-431. doi: 10.1016/j.chb.2016.07.017.