Journal of Educational Sciences
Journal homepage: https://jes.ejournal.unri.ac.id/index.php/JES

P-ISSN 2581-1657
E-ISSN 2581-2203

Learning to Write Programs using Think-Pair-Share Programming Strategy: What are the Students' Perceptions and Experiences?

Adewale Owodunni Saka
Olabisi Onabanjo University, Ago-Iwoye, Ogun State, Nigeria

ARTICLE INFO

Article history:

Received: 29 June 2020
Revised: 09 Oct 2020
Accepted: 10 Oct 2020
Published online: 24 Oct 2020

Keywords:

Think-Pair-Share Programming
Plugged
Unplugged
Programs
Students' perception
Retention

A B S T R A C T

The need to learn programming to solve many complex problems facing humankind necessitated this study. The purpose of the study was to collect information about students' perceptions and experiences after exposure to the think-pair-share programming strategy. The sample consisted of 12 senior secondary school two students offering computer studies in Ijebu zone, Ogun State, Nigeria purposively selected from the two experimental groups. The data were collected through one-on-one in-depth interviews of the respondents using a Student Interview Guide (SIG). The data analysis was through thematic content analysis procedure. The study found that the respondents perceived the think-pair-share programming strategy helpful to learn programming concepts with or without computers. The study also found that the use of computer was more useful for the acquisition of programming skills than without the use of computers. Moreover, the study found that programming without computers was perceived to improve thinking. Therefore, the study argued that teachers should adopt the use of think-pair-share programming strategy for learning how to write programs notwithstanding the availability of computers due to its ability to aid knowledge retention.

1. Introduction

The growing importance of computers to solve many complex problems confronting humankind has necessitated the need for programming. Programming is the process of writing a set of instructions in a given programming language for the computer to solve problems and stacking the directions into the memory of the computer, running the program, and revising any resulting bugs (Halvorson, 2020). It is the only language that can explore the inner world of the computers, its central processing unit and without programming, human-computer interaction

* Corresponding author.
E-mail: wale.saka@oouagoiwoye.edu.ng
Doi: https://doi.org/10.31258/jes.4.4.p.705-717

may be difficult. Also, the fast-pace of computer dominance in virtually all the facets of human lives suggests that individuals need to learn programming to be able to work effectively in their workplaces.

The demand for individuals with programming skills is ever increasing and will continue for a very long time. There is a projection that many jobs in Science, Technology, Engineering and Mathematics (STEM) in 2020 will require possession of the knowledge and skills of programming (Massoud, Hallman, Plaisent, & Bernard, 2018). According to Patton (2020), the ever-increasing request for individuals with skills of programming makes them desirable for every individual to learn for productive participation in the wider society. However, studies have reported the difficulties of teaching and learning programming skills in all levels of education. Some of the challenges are students' lack of reasoning abilities (Akinola, 2016), newness to the syntax and semantics of the programming languages (Kwon & Scrhoroderus, 2017; Masura et al., 2012), lack of resilience to endure programming task (Sentance & Csizmadia, 2017) and complex nature of programming itself (Oyelere, Suhonen & Laure, 2017). Similarly, the lack of effective teaching strategies also has been identified as contributing to the challenges (Kastl & Romeike, 2018; Ahmed, Dykowski, Tooley, Helland & Barremkala, 2017; West African Examinations Council (WAEC) Chief Examiners' Reports, 2014-2018). WAEC is the regional examinations body for the English-speaking West African countries.

To improve the learning of programming, researchers have suggested the use of pair programming (Lasisi, 2016; Yuan & Cao, 2019). Pair programming involves two learners working side-by-side with or without a computer on the same design, algorithm or code with each member playing a unique role at a given time (Sherriff, 2016). It can improve students' retention of knowledge of programming; increase confidence; promote knowledge sharing; and enhance creativity (Seo & Kim, 2016). Pair programming can also be an effective intervention for beginner programmers with little impact time. From the multidisciplinary approach standpoint, Massoud et al. (2018) compared the learning of programming to that of language learning with the suggestion that the application of language learning techniques to explore how to write computer programs can improve students' acquisition of the knowledge and skills of programming. The think-pair-share (TPS) strategy has been used effectively in the literature for language teaching (Aeni, 2020; Mundriyah & Parmawati, 2016; Sari, Komariah, & Isa, 2019). It consists of think, pair and share stages- thinking individually and pairing with partners. The TPS allows learners from diverse background to interact socially for learning purpose (Bamiro, 2015). It also supports learning by discovery and develops learners' higher cognitive and problem-solving skills. Think-pair-share strategy affords students the opportunities to reason and obtain seamless feedback on their actions.

Qualitative studies on the use of think-pair-share (TPS) strategy to learn how to write programs are scarce in the literature. Few qualitative studies on the application of the TPS strategy are in the other subject areas, especially language learning. Aeni (2020) used think-pair-share to teach speaking skills to twenty-five

students of the eighth grade of a junior high school in Bandung. The data collection was through observations and questionnaire. The findings revealed that students taught with TPS techniques experienced joy and excitement about the teaching activities. The study recommended TPS as the alternative teaching method to the English teachers. Sari et al. (2019) also conducted qualitative research using eleventh-grade students at SMAN 07 Banda Aceh to understand how think-pair-share could improve students' critical thinking in answering high-order questions in reading skills. The study collected data through observation, document collection and interview. The findings revealed that TPS made the learners active and helped them to use their previous knowledge to use during the interaction with resultant improvement in their critical thinking. Dwigustini and Widiya (2020) also investigated the efficacy of think-pair-share (TPS) to improve seventh-grade students' reading comprehension. The qualitative study collected data through observation, interviews and posttests. The results indicated that students' exposure to TPS improved their skills in comprehension aspect of English Language after exposure to the TPS. These results were contrary to the pretest results which indicated that students were unable to understand the meaning of the texts. Thus, the researchers described TPS as a suitable technique to improve reading comprehension.

Participation of the students in the class is core to the successful learning of language. In this regard, Alfino, Rochsantiningsih, and Sulistyawati (2019) examined the impact of think-pair-share to engage the students in classroom activities. The researchers collected data through observations, questionnaire, interviews, photographs and research diary. The findings showed that TPS strategy helped the learners to engage in robust class participation as noticed in the quality of answers to questions, group interaction, boldness to present their solutions in front of other colleagues and sharing of knowledge among the learners. However, TPS also had challenges such as students' familiarity with method; lack of class control and students' readiness to learn; and freedom of expression. The students were also unable to speak with confidence and had low mastery of grammar and vocabulary.

On pair programming, the qualitative aspect of the mixed-methods design study of Franklin (2015) used interview, questionnaire and observation to examine the benefits of implementation of pair programming on the students learning of how to program in the secondary schools. The study reported that students preferred to learn in pairs due to the nature of assistance enjoyed from group members during the teaching-learning process. However, the pairing of learners with similar ability could mitigate the success of pair programming. Also, the qualitative aspect of the mixed-methods study of Bailey and Mentz (2017) collected data from the students through open-ended questions added to the posttest and also interviewed the teachers using a semi-structured interview protocol. The findings revealed that pair programming enhanced the students' social skills and improved their programming skills. To understand the effectiveness of pair programming in the middle schools, Campe, Denner, Green, and Torres (2020) examined the impact of pair programming on students programming of a computer game. The research was conducted after the school hours with data collected through audio, video,

and screen captures from 66 students. The researchers found that the students did not spend the total time allocated for each lesson for the learning purpose. Instead, they used the majority of the time without interaction or doing unrelated things. The study also reported that students' inclination for social interaction and prior experience in programming significantly determined their extent of collaboration.

Meanwhile, the challenges in learning programming persist despite efforts by researchers to solve the challenges. Several jobs that require programming permeate the workplaces without applicants with the knowledge and skills needed to get employed and succeeded (Thayer, 2020). The observations from the WAEC Chief Examiners reports (2014-2018) have followed a similar trend. For example, West African Examinations Council (WAEC) Chief Examiners' Reports (2014-2018) expressed that majority of students had little understanding of Beginners' All-purpose Symbolic Instruction Code (BASIC), which is the approved programming language for senior secondary education in Nigeria. Moreover, the reports indicate that students could not distinguish between core concepts such as algorithm and flowcharts, avoided questions relating to programming concepts and those that attempted often missed the answers. These observations imply that the individuals churned out by the schools do not have the requisite knowledge and skills to write programs.

Shreds of evidence from the literature have shown that think-pair-share programming and pair programming strategies have been used to learn English language and programming respectively. However, no study has fused the two strategies into a new strategy to learn to program. This fusion is also a response to the observation in the literature that the techniques of language learning can be extended to teach computer programming. Hence, this study fused the two strategies into a single strategy called think-pair-share programming strategy. The think-pair-share programming strategy is different from the conventional think-pair-share proposed by Lyman (1981) due to its integration of pair programming strategy into the 'pair' stage. The strategy was implemented in two variations-with computers (plugged) and without computers (unplugged). It also puts the mechanism for conflict resolution in place.

To this end, the following research question guided the study:

i. What are the students' perceptions of the TPSPS for learning programming concepts?
ii. What are the students' experience about programming after exposure to TPSPS with or without computers?

2. Methodology

This qualitative research was extracted from the mixed-methods study of Saka (2020) which examined the instructional benefits of think-pair-share programming strategy (TPSPS) to learn the programming aspect of Computer Studies in senior secondary schools in Nigeria. The TPSPS was implemented in plugged (i.e. with computers) and unplugged (i.e. without computers) modes. Mixed-methods

involve the collection and integration of the data from qualitative and quantitative research in a study with clearcut designs (Creswell & Creswell, 2018). The study adopted an explanatory sequential mixed-methods design. In this design, the quantitative study was conducted and the data analysed before the conduct of qualitative research. The findings from the qualitative aspect were employed to understand and explain the results from the quantitative data analysis more succinctly. Therefore, this study is a report of the qualitative aspect of the mixed-methods research.

The study was conducted in three purposefully selected senior secondary schools in Ijebu zone of Ogun State, Nigeria. The schools were selected based on the criteria that they were offering computer studies up to senior secondary school two (SS2 is the second year in the senior secondary school level in Nigeria), availability of qualified teachers to handle the subject and availability of computer laboratories in the schools. Two of the schools served as the experimental groups for plugged (with computers) and unplugged (without computers) programming using think pair share programming strategy. The remaining school served as the control group where the students were exposed to the traditional method of teaching without computers and unplugged activities. Each student in the experimental groups was allocated 10 minutes to think individually about the solutions to a given programming task. Then, students of mixed abilities were paired using their results in pretest without their knowledge and then allowed them to learn in the group using established rules and guidelines of pair programming. Each group then presented their solution to the general class after which the individual tasks were assigned to the students to measure their level of understanding of the programming concepts. The research assistants/researcher took students without notable improvement for remedial action. The study lasted for six weeks.

One hundred and eighty-six (186) SS 2 students offering computer studies in the selected schools participated in the study such that the control group had 59 students while the groups with computers(plugged) and without computers (unplugged) had 64 and 63 students respectively. From the sample, six students were purposefully selected from each of the experimental groups (i.e. six students each from plugged and unplugged groups) for in-depth interviews after exposure to the interventions on the criteria that they participated in the study and were willing to participate in the interview. Initially, the students were told that the interview was an extension of the just concluded teaching-learning process and for interaction with them to collect information about their perception and experiences about the teaching strategy. The students were also informed that they could opt-out of the interview at will. A semi-structured interview guide of six questions was used to conduct the one-on-one in-depth interviews. During the interview, students were asked questions bothering on their experience and perception of think-pair-share programming strategy. Where necessary, the respondents were made to provide clarity on their responses with the use of probing questions. The researcher hand-recorded the responses of the interviewees while the research assistants recorded the audios and videos of the proceedings.

The recordings were done with the permission of the students after assurance that the researcher would mask their responses.

The data collected through the interviews were prepared and coded into six themes manually. The thematic content analysis helps researchers identify students' pattern of responses from the data collected and demonstrates the relevance of the data pattern to the theory (VanDevanter et al., 2012). The respondents were allowed to go through the transcript and coded documents to ascertain the accuracy of the information. Also, the documents were given to a graduate student who had conducted a thesis using mixed methods for critique.

3. Results and Discussion

Six main themes were defined from the students' responses: benefits of TPSPS; challenges of TPSPS; plugged mode benefits; plugged mode challenges; unplugged mode benefits; and unplugged mode challenges.

The summary of the themes and subthemes according to students' responses are as shown in Table 1:

Table 1. Themes and subthemes from the responses of the students

	Themes	Subthemes
1.	Benefits of TPSPS	Shared knowledge
		Knowledge retention
		Team spirit building
		Better understanding
		Enhanced confidence
		Motivation to program
		Enhanced contribution to group activities
		Clarity of tasks
2.	Challenges of TPSPS	Uncooperative attitudes and overreliance of group members
		Inability to learn at their own pace
3.	Plugged mode benefits	Practical learning
		Quality learning
		Knowledge retention
4.	Plugged mode challenges	Inadequate computers
		Irregular power supply
5.	Unplugged mode benefits	Understanding of concepts
		Understanding of program compilation by computer
		Opportunity to practice
		Improved thinking
6.	Unplugged mode challenges	Absence of computers for direct interaction
		Time-consuming nature of the activity

The detailed description of the views of respondents is as follow:
i. Benefits and challenges of think-pair-share programming strategy (TPSPS): The responses of the students from both plugged and unplugged groups as to the benefits of the strategy for learning programming were identical. Some respondents expressed that the TPSPS supported sharing of knowledge: *"nobody*

knows all, we can always learn from each other…it enables me to share my ideas and was able to solve some of the programming tasks with the help of my friends…no matter how difficult a programming task is, there is always a member of the group that would have clues to the solution and this helped us to solve the problems…those of us with little knowledge of programming learned from others". For some students, it helps in knowledge retention: *"learning with the friends helps me to remember what I learned because I always remember the discussion, we had…I always remember what I learn with friends and besides I don't want to disappoint when they ask for my contribution to group work"*. Some students believed that TPSPS builds team spirit among the learners: *"it teaches me how to relate with others in the class. Initially, I don't like sharing what I know with others but now I know it is good to learn with others…I am a shy person who likes individual work before but now I contribute when we are discussing in the group".*

All the interviewees stated that group learning in the form of think-pair-share programming strategy led to their better understanding of the topics learned: *"group learning improves my understanding of programming concepts…what I cannot learn on my own, I learn with the help of my partners".* The excitement of being able to solve programming tasks after group learning motivated some of the students to solve more problems and see the results: *"Anytime we get the answer to the programming problems, I am always happy and want to try more programs…I develop the confidence that I can learn programs…I have the feeling that programming may not be as difficult as I think." One participant emphatically stated that group learning builds her confidence: "group learning helps me to learn better and boost my confidence!"*

On the challenges of TPSPS, the responses of students from both groups were similar in some aspects and varied in others. For example, respondents from both groups pointed to the uncooperative attitudes and overreliance of some partners on the high-ability students in the groups: *"I prefer individual programming because some partners would not want to contribute to the solution. I will only like it if there is a way teacher can make every student contribute equally…some students do not take their studies seriously, they were in the group without saying anything"; and that group learning did not allow them to learn at their own pace: "I don't want any students to give me too much burden because I have my way of learning…I can't allow any students to disturb m learning in the name of group learning"*. Some respondents from the plugged group believed that group learning increased their anxiety especially when they could not contribute to the group discussion and it is time-consuming: *"Group learning exposes some of us who are weak and creates fear that other group members may see us as unserious…you spend a lot of time explaining to other members and you can't proceed until they all understand… it also takes time to put the ideas of all members together"*. Some students in the unplugged group observed that members did not always agree with contributions of others and this made them passive in the group: *"…they will be saying that you don't contribute whereas your answers will always be rejected without explanation which makes me to always keep quiet".*

Some respondents espoused the advantages of the thinking time embedded in the think-pair-share programming: *"it enables me to have at least an understanding of the programming tasks before joining my partners to solve the problems…the initial understanding always helps me to say something in the group"* (Plugged group). Individual thinking led to personal skill development, clarity of the given programming tasks and trust in relating with the others: *"it helps me to understand the work before going into the group… I develop the habit of solving the problems before meeting my group members and this makes it clearer after group work"* (Unplugged group)

ii. Benefits and challenges of learning programming with computers (plugged): On the benefits, respondents stated that the use of computers made the learning practical due to opportunities for direct interaction with the machines: *"Using the computer to learn computer programming gives me a real idea because it changes the learning of programming from being paperwork…It enables me to see the outputs of my programs immediately and can play with different values to obtain different outputs"*. Students also stated that it leads to a better understanding of the programming concepts compared to when they did not have opportunities to use computers: *"It leads to a better understanding of programming concepts… it improves my thinking because I can always think of other solutions if I am wrong". A student expressed that plugged programming helps to retain the learned programming concepts: "it helps me to retain the knowledge of what I learn"*.

On the challenges, students expounded the problems militating against their use of computers for effective learning of programming. All the students stated the issue of inadequate computers on a one-to-one basis to enable them to individually practice the learned programming concepts: *"As you can see, the number of computers in this lab is less than the number of students to enable our practice…I am unable to practice on the computers because they are not enough"*. Besides, students observed irregular power supply to the school: *"there is no regular power supply to school which makes the lab to rely on a generator for power supply…before this time, sir, we don't always have practical except during the one-week practical period towards the end of a term."*

iii. Benefits and challenges of learning programming without computers (unplugged): Students' in the unplugged group perceived benefits of unplugged activities to learn programming concepts in diverse ways. Some observed that the unplugged programming improved their thinking and led to better understanding: *"it improves my understanding of computer programming…for example, serving as a human representation of computers to run programs alone, can improve human thinking"*. Some students expressed that unplugged programming helped them to understand how computers would interpret programs before running them to solve particular problems. The activities changed their perception that programs can only be executed on the computers: *"it makes me understand what is happening when a computer is running programs…it simplifies programming and makes me understand that programs can be run without computer".*

Some respondents commented that unplugged programming afforded them opportunities for regular practice because the materials needed for the activities were readily available. *"We usually practice programming using unplugged activities on our own because the materials are easy to get… Most times, we use papers instead of the cardboard that teacher normally used as learning materials."*

In terms of challenges, students mentioned that unplugged programming did not give them the opportunities to interact with the physical computers: *"although using unplugged activities to learn programming helped in learning programming, but making use of computers would help us to understand programming better…unplugged activities are fun but the activities leading to solutions most times are too many…it also takes much time and steps to arrive at the answer".*

Discussion

Overall, the respondents appeared to find the think-pair-share programming strategy (TPSPS) useful for the learning of programming. This was corroborated with the feedback from the interview report which indicated that majority of the students did not have access to computers outside the school premises and the few with access did not use it for learning how to program. This means that whatever experiences the students provide are based on their exposure to TPSPS. The reported instructional benefits are better understanding, knowledge sharing and knowledge retention, motivation to solve more programming tasks and building of team spirits. Scotts and Palincsar (2013) argued that when learners are grouped to learn, they acquire socially shared experiences and gained improved learning as well as better problem-solving methods. This finding aligns with previous studies on the effectiveness of the think-pair-share strategy (Alfino et al., 2019; Dwigustini & Widiya, 2020) and pair programming (Bailey & Mentz, 2017; Campe et al., 2020). However, the experiences of some students such as unequal contributions of group members, uncooperative attitudes and over-reliance on high-ability members of the groups are indications that the TPSPS needs mechanisms for engendering quality and equal contributions of members. For example, two respondents reported that group learning prevented them from learning at their own pace because they had to teach the low-ability learners. This finding is not unexpected because the unique feature of group learning is in supporting members to learn what they would have been unable to learn with their efforts. The finding is congruent with that of Alfino et al. (2019) on the challenges of TPS.

The findings also indicated that learning through the strategy with computers led to an improvement in learning of programming. The use of computer provided respondents opportunities for direct interaction with the computer and enhanced their learning achievement and retention of acquired programming knowledge. It also made learning more practically inclined. This finding concurs with that of Aggarwal, Gardner-McCune and Touretzky (2017) that using computers to learn programming is better than using other means. However, these benefits are not

without challenges, the majority of the participants reported that the computers could not go round the students on a one-to-one basis for individual practice of the learned programming concepts. Also, there was an epileptic power supply to the school that even made the usage of the available computers difficult.

Further, analyses suggest that learning through the strategy without computers also have some benefits. Majority of the students observed that using unplugged activities improved their thinking and enhanced learning achievement and retention of programming concepts. The human-representation of computers exposed them to the fundamentals of programming and enabled them to understand how computers interpret programs (codes) before execution. This has invariably changed the perception that programming can only be learned using computers in a well-equipped laboratory. Besides, learning without computer afforded the students the opportunities for practice outside the classroom because the materials used for the demonstrations are cheap and do not involve the use of electricity. However, the downside to this finding is that the students in the unplugged group still indicated a preference for the use of computers for learning of programming. They lamented that the steps leading to solutions of programming tasks using unplugged activities are also too long and time-consuming.

4. Conclusion

Learning programming through the think-pair-share programming strategy can improve achievement in programming and also enhance the retention of learned programming concepts. However, the effectiveness of the strategy can be improved if the concerns of the students such as uncooperative attitudes of students, over-reliance on the more intelligent learners and unequal participation in group activities are addressed by the teachers/facilitators. Also, efforts towards doubling the number of computer and regular power supply to schools are valued but it is recommended that teachers should introduce students to the use of unplugged activities to learn programming concepts because it aids retention of knowledge acquired during learning and also assists the transfer of knowledge gained for use on the computers. Moreover, learners can improve their understanding of programming through the fun-filled activities embedded in the unplugged mode of learning programming. Hence, it is imperative to develop the capacity of teachers on the use of unplugged activities to learn programming especially in schools where there are no computers for teaching and learning.

Acknowledgement

I wish to acknowledge the support of Professor S. Y. Erinosho and Dr A. S. Ifamuyiwa as my doctoral thesis supervisors, and their editorial contributions to this paper. The Principals, Staff and students of the three secondary schools used for this study- Molusi College Ijebu-Igbo and Ago-Iwoye Secondary Schools,

Ago-Iwoye in Ijebu-North as well as Ogbobo Baptist Grammar School in Ijebu-North East Local Government Areas of Ogun State are also appreciated.

Conflict of Interest: There is no conflict of interest as regards this article.

References

Aeni, Y. K. (2020). The use of Think Pair Share technique in teaching speaking. *PROJECT (Professional Journal of English Education)*, *3*(5), 570–576. doi: 10.22460/project.v3i5.p570-576

Aggarwal, A., Gardner-McCune, C. & Touretzky, D. S. (2017). *Evaluating the effectiveness of using manipulatives to foster computational thinking in Elementary School*. Paper presented at Special Interest Group on Computer Science Education (SIGCSE) '17 March 8-11, Seattle, WA, USA. ACM. doi: https://dx.doi.org/10.1145/301/7680.3017791.

Ahmed, M., Dykowski, S., Tooley, T., Helland, T. & Barremkala, M. (2017). Influence of learning paradigms on the retention of anatomical knowledge in medical students. *Federation of American Societies for Experimental Biology (FASEB) Journal,* 31:1_supplement, 732.9-732.9

Akinola, S. O. (2016). Computer Programming Skills and gender difference: An empirical Study. *American Journal of Scientific and Industrial Research*, 7(1), 1-9

Alfino, B. Y., Rochsantiningsih, D., & Sulistyawati, H. (2019). Improving students' class participation by optimizing the use of think-pair-share technique. *English Education Journal*, *7*(2), 193–201.

Bailey, R., & Mentz, E. (2017). The value of pair programming in the IT classroom1 2. *The Independent Journal of Teaching and Learning*, *12*(1), 90–103.

Bamiro, A. O. (2015). Effects of guided discovery and think-pair-share strategy on secondary school students' achievement in chemistry. *SAGE Open*, 1-7. doi:10.1177/2158244014564754

Campe, S., Denner, J., Green, E., & Torres, D. (2020). Pair programming in middle school: Variations in interactions and behaviors. *Computer Science Education*, *30*(1), 22–46. doi: 10.1080/08993408.2019.1648119

Creswell, J. W., & Creswell, J. D. (2018). *Research Design: Qualitative, quantitative and mixed methods approaches* (5th ed.). Los Angeles: SAGE Publications Inc. Retrieved from https://us.sagepub.com/en-us/nam/research-design/book255675

Dwigustini, R., & Widiya, J. (2020). Think Pair Share Technique to Promote Students' Reading Comprehension. *Jurnal Ilmu Pendidikan (JIP) STKIP Kusuma Negara*, *12*(1), 25–34. doi: 10.37640/jip.v12i1.270

Franklin, J. P. (2015). *Perception of young people of pair programming when learning text languages* (Master Thesis, King's College). King's College, London. Retrieved from https://www.axsied.com/wp-content/uploads/2016/06/Pair-Programming-Dissertation-James-Franklin.pdf

Halvorson, M. J. (2020). Code Nation: Personal Computing and the Learn to Program Movement in America. S.l.: ACM Books.

Kastl, P. & Romeike, R. (2018). Agile Project to foster Cooperative learning in heterogeneous classes. Paper presented at the 2018 IEEE Global Engineering Education Conference (EDUCON). 17-20 April. Tenerife, span. https://doi.org/10.1109/EDUCON.2018.8363364

Kwon, S. & Schroderus, K. (2017). *Coding in Schools- Comparing integration of programming into BASIC Education curricula of Finland South Korea*. Retrieved from www.mediakasvatus.fi/materialali/coding-in-schools/

Lasisi, A. (2016) *Creativity in Software Engineering: A Systematic Literature Review*. Master Thesis, University of Oulu. Retrieved from Jultika.oulu.fi/Record/nbnfioulu-201605221871

Lyman, F. (1981). The Responsive Classroom Discussion. (Anderson, A.S., Ed), Mainstreaming Digest, 109-113. College Park, MD: University of Maryland College of Education.

Massoud, L., Hallman, S., Plaisent, M., & Bernard, P. (2018). Applying and Improving Multidisciplinary Teaching Techniques to the Programming Classroom Environment. *DACEE-18,BEHIS-18 May 11-12, 2018 Zagreb (Croatia)*. Presented at the May 11-12, 2018 Zagreb (Croatia). doi: 10.17758/HEAIG3.H0518408

Masura, R., Shahrina, S., Rodzich, L., Noor, F. M. Y., Noor, F. A. Z. & Rohizah A. (2011). Major Problems in Basic Programming that Influence Student Performance. *Procedia-Social and Behavioural Sciences,* 59(2012). 287-296. doi:10.1016/j.sbspro.2012.09.277.

Mundriyah, M., & Parmawati, A. (2016). Using think-pair-share (TPS) to improve students' writing creativity (A Classroom Action Research in the Second Semester Students of STKIP Siliwangi Bandung). *Jurnal Ilmiah P2M STKIP Siliwangi*, *3*(2), 84–91. doi: 10.22460/p2m.v3i2p84-91.630

Oyelere, S. S., Suhonen, J. & Laure, T. (2017). Integrating Parson's programming puzzles into a game based mobile learning application. In the proceedings of the 17th Koli calling International Conference on Computing Education Research. November, 16-18 (pp 158-162). ACM New York, NY, USA. https://doi.org/10.1145/3141880.3141882.

Patton, B. (2020). An analysis of factors that may Influence student satisfaction in computer programming courses at an online Midwestern University. Retrieved from https://etd.auburn.edu/handle/10415/7428

Saka, A. O. (2020). Effects of think-pair-share programming strategy on senior secondary school students' achievement and retention in programming aspect of Computer Studies. *Undergoing Assessment* at Olabisi Onabanjo University, Ago-Iwoye, Ogun State, Nigeria.

Sari, D. F., Komariah, E., & Isa, R. A. (2019). The use of think pair share to improve students critical thinking in reading skill. *International Conference on Early Childhood Education*, *0*(0), 344–350.

Scotts, S. & Palincsar, A. (2013). Sociocultural theory. Retrieved from http://dr-hatfield.com/theorists/resources/sociocultural_theory.pdf

Sentance, S. & Csizmadia, A. (2017): Computing in the curriculum: Challenges and strategies from teachers' perspective. *Education and Information Technologies*, 22(2), 469-495.

Seo, Y., & Kim, J. (2016). Analyzing the Effects of Coding Education through pair Programming for the computational thinking and creativity of Elementary School students. *Indian Journal of Science and Technology*, 9 (46), 1-5. doi: 10 17485/ljst/2016/19: 46/107837.

Sherriff, M. (2016). *Pair. Programming in the Classroom*. Retrieved 6 May, 2018 from https://pdfs.semanticscholar.org/.../e196739701ddaaab93755609cc7218ad4095.pdf

Thayer, K. M. (2020). *Practical Knowledge Barriers in Professional Programming* (Thesis). Retrieved from https://digital.lib.washington.edu:443/researchworks/handle/1773/45471

VanDevanter, N., Combellick, J., Hutchinson, M. K., Phelan, J., Malamud, D. & Shelley, D. (2012). A qualitative study of patients attitudes toward HIV in the Dental setting. Nursing Research and Practice, Article ID803169, 6 pages. doi:10.1155/2012/803169

West Africa Examinations Council Chief Examiners' Report. (2014). General Comment, Weakness/Remedies and Candidate's Strength. Retrieved from https://waeconline.org.ng/e-learning/Computer/Comp223mq1.htm

West Africa Examinations Council Chief Examiners' Report. (2015). General Comment, Weakness/Remedies and Candidate's Strength. Retrieved from https://waeconline.org.ng/e-learning/Computer/Comp224mq1.htm

West Africa Examinations Council Chief Examiners' Report. (2016). General Comment, Weakness/Remedies and Candidate's Strength. Retrieved from https://waeconline.org.ng/e-learning/Computer/Comp225mq1.htm

West Africa Examinations Council Chief Examiners' Report. (2017). General Comment, Weakness/Remedies and Strength. Retrieved from https://waeconline.org.ng/e-learning/Computer/Comp226mq1.htm

West Africa Examinations Council Chief Examiners' Report. (2018). General Comment, Weakness/Remedies and Candidate's Strength. Retrieved from https://waeconline.org.ng/e-learning/Computer/Comp227mq1.htm

Yuan, H. & Cao, Y. (2019). Hybrid Pair Programming-A Promising Alternative to Students Pair Programming. Paper presented at the SIGCSE '19 Proceedings of the 50th ACM Technical Symposium on Computer Science Education (Pp. 104-1052). Minneapolis, MN, USA- February 27- March 02, 2019. ACM New York, NY, USA. https://doi.org/10.1145/3287324.3287

How to cite this article:

Saka, A. O. (2020). Learning to write programs using Think-Pair-Share Programming Strategy: What are the Students' perceptions and experiences?. *Journal of Educational Sciences, 4(4)*, 705-717.