IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, June 2026
Impact Factor: 8.2

Analysing Common Barriers in Programming: Translating Real-World Problems into Functional Code

Elmera J. Balingan[1], Marielle Ivy R. Salibay[2], Jess Rose E. Fernandez[3]
BSCS students, College of Computing and Information Sciences,
Surigao Del Norte State University, Surigao City, Philippines[1,2]
Faculty, College of Computing and Information Sciences,
Surigao Del Norte State University, Surigao City, Philippines[3]
elmera.balingan@ssct.edu.ph, marielleivy.salibay@ssct.edu.ph, jfernandez@ssct.edu.ph

Abstract: *This study investigated the common barriers faced by Bachelor of Science in Computer Science (BSCS) students in translating real-world problems into functional code. Despite high levels of reported technical proficiency, students continued to encounter significant cognitive and affective challenges, including cognitive overload, programming anxiety, and low self-efficacy. Using a quantitative descriptive-correlational design, data were collected from 76 participants through structured questionnaires and analysed with Jamovi statistical software. Results revealed that technical skills alone did not mitigate cognitive and psychological barriers, highlighting the multidimensional nature of programming difficulties. The findings emphasized the importance of integrated instructional approaches that combined algorithm design, computational thinking, and psychological support to strengthen students' problem-to-code translation skills. This research contributed to programming education by offering context-specific insights and recommendations for improving instructional strategies in BSCS programs.*

Keywords: Cognitive Load, Programming Anxiety, Computational Thinking, Algorithm Design, Self-Efficacy

I. INTRODUCTION

Overview of the Study

Programming had always been a core part of the Bachelor of Science in Computer Science (BSCS) curriculum, as programming is at the core of system design and software development, this has not changed for decades. Despite all the changes that had been made to the curriculum, many BSCS students were unable to successfully transform abstract concepts and real-world problems into executable code. You had to know not only the syntax, but you had to be trained in computational thinking, logical reasoning, and algorithm design (Shute et al., 2021). When we reviewed the literature again and conduct cognitive task analysis then previous evidences suggests that novice programmers found problem decomposition & abstraction a difficult skill to master which was crucial in realise the understanding of concepts into executable programs (Weintrop et al., 2021). Cognitive overload and low programming self-efficacy may also lead to poor coding performance (Chen & Kalyuga, 2022; Medeiros et al., 2021). The BSCS program is technical in nature; therefore, it was important to analyze the qualitative subset of impediments keeping individuals from successfully transcribing concrete problems into executable code.

DOI: 10.48175/IJARSCT-36512

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 4, June 2026 Impact Factor: 8.2

Review of Related Literature and Knowledge Gap

Recent study has identified several barriers affecting programming performance. Algorithm design and problem dissolution skills would be difficult for computer science students (Loksa et al., 2021). Weintrop et al. (2021) found that the computational thinking constructs abstraction and decomposition were significant predictors of positive programming outcomes. Also, Cognitive Load Theory described that programming tasks might insert substantial cognition demand on learners when they needed to hold the different levels of abstraction from logic and syntax (Chen & Kalyuga, 2022). Additionally, programming anxiety and self-efficacy had a significant negative impact on students' confidence and performance (Medeiros et al., 2021). Meanwhile, many of these studies study programming difficulties on a broad student smoker level and did not specifically analyze BSCS students that were involved at an intensive level in coding courses. A continuous lack of research explicitly addressing the multiple barriers these BSCS students face while translating real-world problems into functional code left a gap for in depth study.

Statement of the Problem

Despite the extensive studies on programming education, BSCS students still had a hard time converting real-world problems to executable code. Prior studies had analyzed cognitive load, computational thinking, and programming anxiety, but there was a lack of literature about their impact together on translating a problem to code among BSCS students. If the barriers preventing effective translation were not known, any instructional intervention would miss the actual source of the problem. In view of this, the study aimed at assessing the different barriers to programming, which prevented BSCS students from converting real-world problems into code.

Proposed Solution and Contribution of the Study

In order to fill this gap, the current study adopted the descriptive-correlational approach to explore the common barriers encountered by BSCS students when performing their programming tasks. The data for the current study were collected using programming tests and surveys assessing the level of proficiency in problem analysis and coding. Exploring the association between common barriers and students' inability to write effective codes was intended to provide the basis for practical recommendations on how to enhance instruction. The results of this study would be useful for programming instructors as they could implement certain practices aimed at developing problem decomposition, algorithmic thinking, and logical reasoning skills among BSCS students.

OBJECTIVES OF THE STUDY

This research aims at determining the major constraints encountered in programming, particularly in transforming real-world problems into codes for BSCS students. The objectives of this research are:

1. Determine the main obstacles that BSCS students face when converting real-world issues into codes
2. Evaluate the performance of BSCS students in creating functional programs.
3. Identified the connection between the barriers to programming and the students' competence in writing functional codes.
4. Recommend possible solutions on how BSCS students could translate problems into functional codes.

II. REVIEW OF RELATED LITERATURE

Affective Barriers

The affective barriers to programming included the psychological aspects like stress, poor self-efficacy, the fear of failing, and lack of confidence that affected how students engaged in programming activities. According to Dastyni Loksa et al., programming self-efficacy had significant effects on the perseverance and behavior of students as they solved programming problems. In addition, poor self-efficacy caused students to lose interest easily if they encountered any challenges.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, June 2026
Impact Factor: 8.2

Likewise, Theodora Koulouri et al. discovered that programming anxiety resulted in decreased attention and poor performance in programming tasks among learners. Moreover, Jens Bennedsen and Michael E. Caspersen found out that the reason for a high level of failures in programming classes was due to motivational and emotional factors affecting persistence. This implies that the affective barriers had a direct impact on how students transitioned from problem-solving to writing codes.

Algorithm Formulation

Algorithm formulation was achieved by structuring the problem into a step-by-step process before coding. The difficulty faced by many students in programming is associated with algorithm formulation problems according to Rodrigo Medeiros et al. (2021). Most students found it difficult to structure logical processes hence failed in formulating good algorithms.

Further, according to Lye & Koh (2024), the ability to develop an efficient algorithm depended on the abilities of computational thinking such as disintegration and abstraction. Many students were not able to develop a proper algorithm even though they had knowledge of the concept of the program.

Cognitive Barriers

The cognitive barriers noted the limitations in processing information regarding skills related to cognition that made it difficult for students to be able to analyze and structure issues experienced during coding. Oi-man Kwok Chen and Slava Kalyuga (2022) used the Cognitive Load Theory framework when describing how coding had high levels of intrinsic and extraneous cognitive loads as it demanded that various forms of cognitive tasks such as logic and syntax analysis had to be performed, among others. In this context, those individuals who struggled with structuring issues experienced memory overload.

Also, according to Hellas, Arto et al. (2023), cognitive abilities significantly played a major role in determining coding success. For instance, cognitive problem had an impact on the coding steps of problem analysis and plan design.

Debugging and Refinement

Debugging and refinement involved the process of identification, analysis, and correction of mistakes. According to Ahadi et al. (2022), debugging involved trial and error because few students applied a systematic method. It is important to mention that it was essential to apply logical reasoning in debugging the code.

Additionally, Becker Loksa et al. (2021) found out that those students who had high programming self-efficacy continued debugging. This implies that debugging was a cognitive and affective activity because one required knowledge and self-confidence.

Problem-Solving Skills

Problem-solving skills were foundation in converting real-world experiences into virtual ones. According to Lye and Koh (2024), computational thinking improved considerably the skills of learners in decomposing problems to create appropriate solutions. Furthermore, Hellas et al. (2023) found out that analytical thinking was highly associated with program writing.

Individuals lacking the skills of solving problems faced challenges linking their understanding of concepts and programming codes. This shows that problem-solving ability was the basis of problem-solving activities.

Technical Skills

Technical skill was related to the student's proficiency with programming language syntax, structure, and programming technique. According to Medeiros et al. (2021), lack of competence in basic programming techniques such as loops, conditionals, and functions was the main impediment in the process of learning to program.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, June 2026
Impact Factor: 8.2

According to Bennedsen & Caspersen (2023), this was indeed the case and that indeed poor technical ability had led to failure in many programming courses. It is impossible to code correctly without adequate technical skills even for the best + algorithms.

III. CONCEPTUAL FRAMEWORK

Key Variables

This study was conducted to examine the impact of cognitive, technical, and psychological factors on the ability of the BSCS students to translate problem statements to code (Shute et al., 2021; Weintrop et al., 2021). The cognitive, technical, and psychological factors analyzed in the study included computational thinking, algorithm development, cognitive load, and programming self-efficacy (Weintrop et al., 2021; Loksa et al., 2021; Chen & Kalyuga, 2022; Medeiros et al., 2021). Generation of correct, credible, and executable code was considered as the measure of programming skills because the inability to perform these skills could affect coding, whereas good skills and confidence in performing them would aid the programming process.

DIAGRAM:

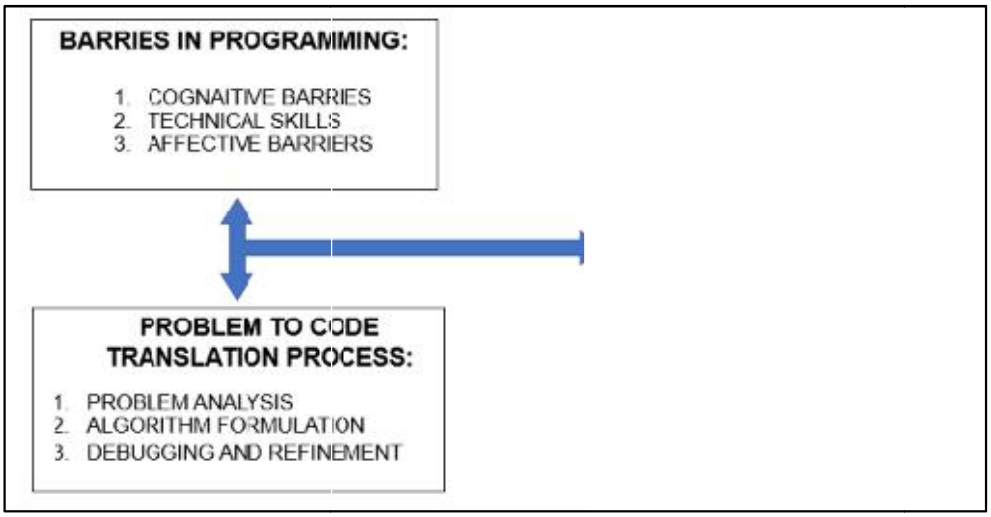

Fig. 1: ***Conceptual Framework***

This research study was founded on the theory of Cognitive Load, which stated that problem-solving capabilities were affected by the level of cognitive load. These factors included programming barriers, which would affect the translation of real-world problems into code. Figure 1 shows how these factors will be able to affect programming performance, which includes accuracy, efficiency, and number of errors made.

METHODS

This section outlined the research design, data collection techniques, and analysis methods used in the study. It was presented in sufficient detail to allow replication of the research process.

2.1 Research Design

This study involved a descriptive correlation approach that entailed an examination of the association between Cognitive Barriers, Technical Skills, and Affective Barriers (psychological factors) concerning the identified topic of the study. This research method was appropriate because the data collected through the survey were numerical and

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, June 2026
Impact Factor: 8.2

therefore subjected to statistical analysis. The descriptive method helped measure each variable, whereas the correlation method helped identify the relationship between variables, if any.

2.2 Participants and Sampling Method

This study included 76 respondents who were recruited by adopting the Random Sampling Technique. Selection of these respondents was done based on their availability and willingness to take part in an online survey. The population targeted included all those people who satisfied the study criteria and were internet users. This particular sampling method seemed appropriate since data collection would be effective under the given circumstances.

2.3 Data Collection Methods

The data were gathered by means of Google Forms survey questionnaires which were sent out to the respondents using online channels. The questions were formulated based on the three constructs which were Cognitive Barriers, Technical Skills, and Affective Barriers (psychological barriers).

In gathering data, the ethical aspect was strictly observed. Prior to answering the questionnaire, the purpose of the study was made clear to the respondents as well as the voluntary nature of their participation and their right to withdrawal at any point without repercussions. Their privacy was respected, with no need to provide personal information except if needed.

2.4 Data Analysis Techniques

Data gathered from Google forms was downloaded and saved in CSV file format. The data obtained through Google forms was carefully analyzed, sorted out, cleaned up, and checked to eliminate any incomplete entries, duplication, and inconsistencies. The clean database was thereafter uploaded into Jamovi statistical software for further analysis.

Descriptive statistics which included frequency counts, mean, median, mode, standard deviation, minimum score, and maximum score were calculated to understand the nature of the response received from participants and also estimate the general magnitude of variables involved. The use of descriptive statistics made the understanding of the distribution of scores easier.

In addition, the use of graphical presentation in the form of boxplots and density plots made the interpretation of data even easier. Graphs provided more detailed information about variability, spread, and presence of any outliers within the dataset.

Inferential analysis was done by using correlational statistical test to identify the existence of relationship between cognitive barriers, technical skills, and affective barriers. The correlation test results obtained using Jamovi software were interpreted accordingly.

Lastly, the research results were interpreted in light of the objectives of the study. The statistical results were used to draw conclusions regarding the factors affecting the respondents and to provide evidence-based recommendations.

TABLE 1. Scale of Interpretation for Likert Responses

Scale	Range	Verbal Description	Interpretation
5	4.50 – 5.00	Strongly Agree	Very High
4	3.50 – 4.00	Agree	High / Moderate
3	2.50 – 3.49	Neutral	Moderate
2	1.50 – 2.49	Disagree	Low

DOI: 10.48175/IJARSCT-36512

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, June 2026
Impact Factor: 8.2

1	1.00 – 1.49	Strongly Disagree	Very Low

Table 1 presents the scale of interpretation used for the Likert responses in the study. It serves as the basis for interpreting the respondents' answers according to their corresponding numerical values, verbal descriptions, and interpretation levels. The table helps determine the extent of agreement or disagreement of the respondents regarding the statements provided in the questionnaire.

IV. RESULT AND DISCUSSION

4.1 Presentation of Results

Three main variables were considered in the statistical analysis, namely; Cognitive Barriers, Technical Skills, and Affective Barriers (Psychological Factors). This analysis is detailed in Table 1. The data set is composed of N = 76 data points with no missing observations.

Table 2: Descriptive Statistic of Research Variables

	COGNITIVE BARRIERS	TECHNICAL SKILLS	AFFECTIVE BARRIERS(PSYCHOLOGICAL FACTORS)
N	76	76	76
Missing	0	0	0
Mean	3.87	3.92	3.88
Std. error mean	0.0694	0.0641	0.0665
Median	3.89	4.00	3.94
Mode	4.00	4.00	4.00
Standard deviation	0.605	0.559	0.580
Variance	0.366	0.312	0.336
Range	3.22	3.56	3.44
Minimum	1.78	1.44	1.56
Maximum	5.00	5.00	5.00
Skewness	-0.331	-0.953	-0.836
Std. error skewness	0.276	0.276	0.276

Plot

This plot was made up of three graphs plotting the Kernel Density Estimate of the data for three factors namely, Cognitive Barriers, Technical Skills, and Affective Barriers (Psychological Factors). Density on the vertical axis denoted the frequency of occurrence of each value while the horizontal axis showed the scale of rating between 2 and 5. From the graph, it can be noted that all three curves were smooth and not symmetrical with peaks to the right hand side and longer tails extending to the left hand side. The skewed nature of the curves as seen in the graph was similar to the calculated negative skewness shown in the statistical table above.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, June 2026
Impact Factor: 8.2

COGNITIVE BARRIES

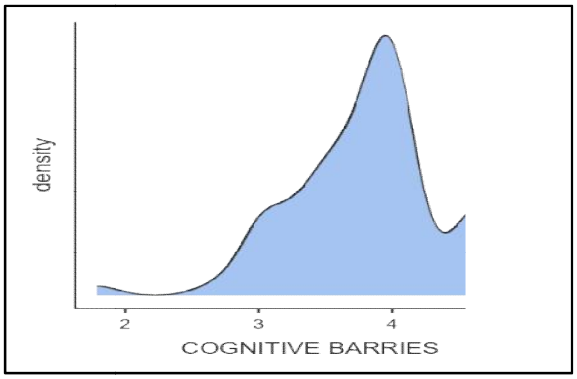

Fig.2: ***density plot of Cognitive Barriers***

The following graph displayed how responses were distributed in relation to Cognitive Barriers. It was evident from the above graph that the peak occurred at 4. This meant that majority of the respondents had assigned cognitive barriers at this level. Since the skewness value was -0.331, the graph demonstrated slight negative asymmetry where few people responded with a score between 2-3 and many responded with a score of 4-5. The minimum value on the data values was 1.78 whereas the maximum was 5.00.

TECHNICAL SKILLS

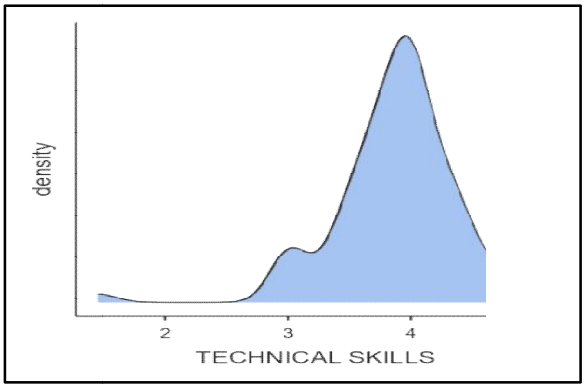

Fig.3: ***density plot of Technical Skills***

The plot presented was the plot of Technical Skills. This graph plotted a narrow and tall curve with its maximum value at 4, the most frequent response given by the respondents. This is because of the presence of the highly negatively skewed plot of -0.953, which showed a high concentration towards the higher values. There were almost no responses of the lower values (2–3). The range of the data was from 1.44 to 5.00.

DOI: 10.48175/IJARSCT-36512

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, June 2026
Impact Factor: 8.2

AFFECTIVE BARRIERS (PSYCHOLOGICAL FACTORS)

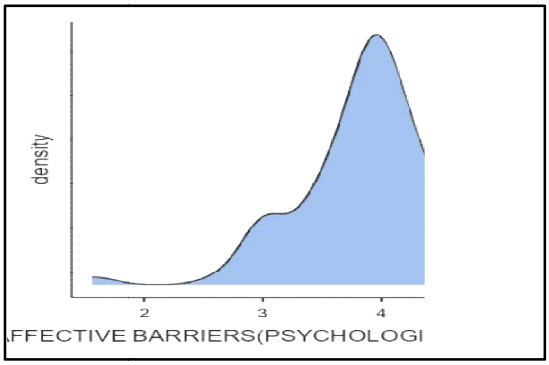

Fig.4: ***density plot of Affective Barriers (Psychological Factors)***

This density plot presented values for Affective Barriers or Psychological Factors. The plot's maximum was observed at 4, thus this response was the mode; however, as a whole, this density plot looked broader and less concentrated than the density plots of the other two variables. This was corroborated by the calculated skewness, which was equal to -0.836. Negative skewness implied that high values (4-5) were received more often than low values (2-3). Values were in the range from 1.56 to 5.00, and the values varied significantly within the range of 2 to 5.

Key findings from the distributions include:

- The average scores for all three scales mean, median, and mode were relatively high throughout the study, ranging from 3.87 to 4.00.
- Skewness was negative in all variables, meaning that most of the scores had been concentrated at the upper end of the scale.
- Skills were the most negatively skewed variable (−0.953) and also had the least amount of variance (0.312).

4.2 Interpretation of Findings

The findings revealed that the respondents generally felt that they had very high technical skills (Mean=3.92) but at the same time experienced relatively high Cognitive and Affective Barriers. The higher median compared to the mean value in all the three cases further confirmed the left-skewness in the distribution, where a few low-value outliers (as shown in the box-plots) dragged down the overall mean value.

The high score values obtained for Affective Barriers (Mean=3.88) implied that psychological variables were significantly important for this case. This finding was consistent with Self-Efficacy Theory where the respondents may have had technical skills, but they had to face substantial psychological problems. The small standard deviation obtained in all cases (0.559 to 0.605) implied that there was a high degree of agreement on the issue among the 76 respondents.

4.3 Discussion of Unexpected Results

That extreme outliers existed on the low end of the scale was a fascinating finding. As a matter of fact, Technical Skills and Cognitive Barriers had minimum scores of 1.44 and 1.78, respectively, but the respondents rated themselves close to 4.00.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, June 2026
Impact Factor: 8.2

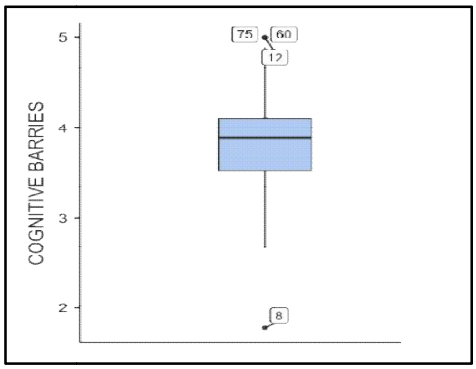

Fig.5: ***Distribution of Cognitive Barriers Scores***

This plot portrayed the distribution of scores on Cognitive Barriers. Box ranged from 3.5 to 4.2, while median was around 4.0, indicating that respondents felt that the cognitive barriers did exist and were important. Whiskers extended up to 2.8 at their lowest and 5.0 at their highest point, thus covering the normal range of data distribution. Minor outliers were found on the upper side of the diagram within range 4.8–5.0. Most interesting is the fact that the lowest score received by one of the respondents – "8" – was 1.8, which was actually the lowest score possible among all the variables.

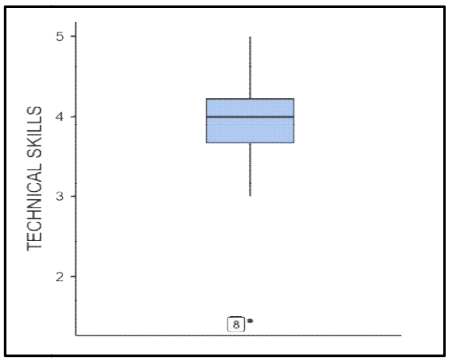

Fig.6: ***Distribution of Technical Skills Scores***

This box plot represent the scores for Technical Skills. It showed the tightest and most compact distribution of all three variables; the box ranged from about 3.8 to 4.2, with the median right at 4.0, which reflected strong consensus — nearly all respondents rated their technical skills similarly and highly. The whiskers were short, stretching only from 3.2 to 4.8, meaning very few people gave low or very high scores. However, respondent "8" once again appeared as an extreme outlier at the absolute lowest value of 1.44. This score was far outside the normal range and confirmed that this one individual assessed their technical abilities in a completely different way compared to the whole group.

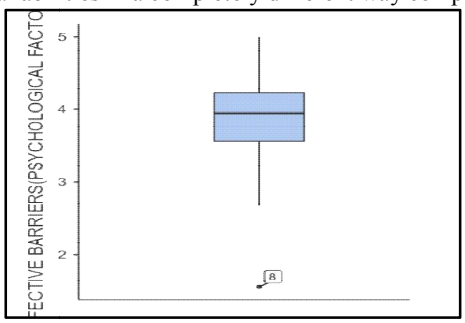

Fig.7: ***Distribution of Affective Barriers (Psychological Factors) Score***

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 4, June 2026 Impact Factor: 8.2

The box plot technique has been used for taking into account all psychological factors. It can be seen from the graph presented below that the box is drawn based on the values lying between 3.7 and 4.3. In this case, the median is 4.0.The lines (whiskers) reached from 3.0 to 5.0, representing the entire range of answers. Nonetheless, there was one particular outlier marked "respondent 8," which was significantly lower, at approximately 1.7. Thus, it can be seen that whereas many participants scored psychological barriers rather high, there was one person who did it extremely low, probably because of individual reasons.

The box plots revealed that respondent number 8 was an outlier in all categories. It could be said that for a small group of people, their experience could have been drastically different than that experienced by the larger group of the population. This could have happened because of unique conditions prevailing amongst individuals like no exposure or high levels of technophobia.

In the present study, distributions of variables like Cognitive Barriers, Technical Skill, and Affective Barriers among a population of 76 respondents have been investigated. The major result found through statistical investigation of these variables was that while technical skills were very high in the sample size, there were similar highs in the case of cognitive and psychological factors as well. The negative skewness of the data across variables revealed that these problems were common for almost everyone.

V. CONCLUSION AND RECOMMENDATIONS

1. Conclusion

This research highlighted the most commonly faced obstacles by BSCS students in converting practical situations to effective programming codes. From the statistical analysis of the gathered data, the researcher drew the following conclusions. First, with regard to the presence of barriers despite technical competence, it is noted that while BSCS students claimed their technical competencies ($\bar{x} = 3.92$) to be high, their experience of cognitive barriers ($\bar{x} = 3.87$) and affective barriers ($\bar{x} = 3.88$) was similarly high. It can thus be said that mere possession of technical skills such as mastery of syntax and coding techniques would not be enough to overcome the mental and psychological struggles involved in problem-code conversion.

As for the influence of cognitive and affective factors, the negative skewness found in the variables showed that most respondents struggled continuously with Cognitive Load Theory and self-efficacy. Indeed, it was found that BSCS students faced many affective barriers, implying that programming anxiety and fear of failure persist among students regardless of their immersion into code-based subjects.

Concerning the connection between the variables of skill and barrier, the results revealed that there was an important deficiency – advanced technical skill did not necessarily mean reduced perceived cognitive overload and psychological pressure. In summary, the researcher argued that the conversion of actual problems into programming language involved more than technical expertise; emotional resilience and cognitive skills also played an equally significant role.

Lastly, concerning the detection of outliers, it was observed that the existence of extreme data points – particularly respondent "8" – suggested that there were a small number of students who encountered far greater challenges than what was considered average. Clearly, there were certain individual factors, such as different exposure to technology and instances of technophobia, that made a huge difference.

2. Recommendations

Moreover, to lessen cognitive barriers, the proponent suggested the implementation of framed tasks that gradually increase in difficulty. From the results and conclusions obtained through the conduct of the study, the proponent made the following recommendations in order to enhance the programming performance and problem-solving abilities of the BSCS students. It was suggested that the programming teachers should go beyond just teaching about programming syntax and adopt more holistic strategies for teaching problem solving and algorithms. This would help learners cope with the high cognitive demands of programming by helping them handle their cognitive loads better.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 4, June 2026 Impact Factor: 8.2

Furthermore, it was recommended that interventions be carried out in order to address any concerns regarding programming anxiety among learners. In contrast, there are actually many strategies that could be as effective in helping develop a good attitude towards programming through peer assessment and formative assessment prior to undertaking any program itself. With the aid of tools such as pseudocode and flowchart, students could successfully transition from conceptualization to actual implementation without burdening themselves with cognitive overload and mental strain.

In addition to this, teachers were advised to employ diagnostic testing at the onset of the course to single out the "outlier" students, who may have no prior exposure to programming and/or experience heightened levels of technophobia. With this approach, these students could be provided with the appropriate assistance needed to avoid early disengagement from the course and possible high dropout rates.

Finally, there was the suggestion for future research to look into the long-term impacts of these barriers among students enrolled in different years within the BSCS course. Moreover, qualitative research could reveal valuable information concerning the particular "real-world" scenarios and situations that are difficult for students to apply into functional programming tasks.

REFERENCES

1. Ahadi, A., Lister, R., Haapala, H., & Vihavainen, A. (2022). *Exploring the effectiveness of explicit debugging instruction in introductory programming.*
   Link: https://scholar.google.com/scholar?q=Exploring+the+effectiveness+of+explicit+debugging+instruction+in+introductory+programming
2. Bennedsen, J., & Caspersen, M. E. (2023). *Failure rates in introductory programming revisited: A meta-analysis of recent studies.*
   Link: https://scholar.google.com/scholar?q=Failure+rates+in+introductory+programming+revisited+A+meta-analysis+of+recent+studies
3. Chen, O., & Kalyuga, S. (2022). *Cognitive load theory and programming education: Managing intrinsic and extraneous load in novice learners.*
   Link: https://scholar.google.com/scholar?q=Cognitive+load+theory+and+programming+education+Managing+intrinsic+and+extraneous+load+in+novice+learners
4. Hellas, A., Ihantola, P., Petersen, A., Ajanovski, V. V., Gutica, M., & Loksa, D. (2023). *Predicting programming performance: A multi-institutional study of cognitive and non-cognitive factors.*
   Link: https://scholar.google.com/scholar?q=Predicting+programming+performance+A+multi-institutional+study+of+cognitive+and+non-cognitive+factors
5. Koulouri, T., Lauria, S., & Macredie, R. (2024). *Programming anxiety and student performance in computing courses: An empirical analysis.*
   Link: https://scholar.google.com/scholar?q=Programming+anxiety+and+student+performance+in+computing+courses+An+empirical+analysis
6. Loksa, D., Ko, A. J., Jernigan, W., Oleson, A., & Mendez, C. J. (2021). *Programming self-efficacy and its relationship to programming performance in introductory computing courses.*
   Link: https://scholar.google.com/scholar?q=Programming+self-efficacy+and+its+relationship+to+programming+performance+in+introductory+computing+courses
7. Lye, S. Y., & Koh, J. H. L. (2024). *Computational thinking, abstraction, and decomposition skills in computer science undergraduates.*
   Link:

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 4, June 2026
Impact Factor: 8.2

https://scholar.google.com/scholar?q=Computational+thinking+abstraction+and+decomposition+skills+in+computer+science+undergraduates

8. Medeiros, R. P., Ramalho, G. L., & Falcão, T. P. (2021). *A systematic literature review on teaching and learning introductory programming in higher education.* Link: https://scholar.google.com/scholar?q=A+systematic+literature+review+on+teaching+and+learning+introductory+programming+in+higher+education
9. Loksa, D., Ko, A. J., Jernigan, W., Oleson, A., Mendez, C., & Burnett, M. (2021). *Programming self-efficacy and its relationship to learning behaviors.* Link: https://scholar.google.com/scholar?q=Programming+self-efficacy+and+its+relationship+to+learning+behaviors
10. Koulouri, T., Lauria, S., & Macredie, R. D. (2024). *Programming anxiety and its effects on learning and performance.* Link: https://scholar.google.com/scholar?q=Programming+anxiety+and+its+effects+on+learning+and+performance
11. Bennedsen, J., & Caspersen, M. E. (2023). *Failure rates and motivational factors in introductory programming.* Link: https://scholar.google.com/scholar?q=Failure+rates+and+motivational+factors+in+introductory+programming
12. Medeiros, R., Ramalho, G., & Falcão, T. (2021). *Difficulties in algorithm design among novice programmers.* Link: https://scholar.google.com/scholar?q=Difficulties+in+algorithm+design+among+novice+programmers
13. Lye, S. Y., & Koh, J. H. L. (2024). *Computational thinking and algorithm development in education.* Link: https://scholar.google.com/scholar?q=Computational+thinking+and+algorithm+development+in+education
14. Chen, O.-M. K., & Kalyuga, S. (2022). *Cognitive load theory and programming instruction.* Link: https://scholar.google.com/scholar?q=Cognitive+load+theory+and+programming+instruction
15. Hellas, A., Ihantola, P., Petersen, A., Ajanovski, V., Gutica, M., & Liao, S. N. (2023). *Predictors of programming performance in higher education.* Link: https://scholar.google.com/scholar?q=Predictors+of+programming+performance+in+higher+education
16. Ahadi, A., Lister, R., Haapala, H., & Vihavainen, A. (2022). *Novice programmers' debugging strategies and difficulties.* Link: https://scholar.google.com/scholar?q=Novice+programmers+debugging+strategies+and+difficulties
17. Chen, O.-M. K., & Kalyuga, S. (2022). *Cognitive load theory and programming instruction.* Link: https://scholar.google.com/scholar?q=Cognitive+load+theory+and+programming+instruction
18. Loksa, D., Ko, A. J., Jernigan, W., Oleson, A., Mendez, C., & Burnett, M. (2021). *Programming self-efficacy and its relationship to learning behaviors.* Link: https://scholar.google.com/scholar?q=Programming+self-efficacy+and+its+relationship+to+learning+behaviors
19. Medeiros, R., Ramalho, G., & Falcão, T. (2021). *Difficulties in algorithm design among novice programmers.* Link: https://scholar.google.com/scholar?q=Difficulties+in+algorithm+design+among+novice+programmers
20. Shute, V. J., Sun, C., & Asbell-Clarke, J. (2021). *Demystifying computational thinking.* Link: https://scholar.google.com/scholar?q=Demystifying+computational+thinking
21. Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2021). *Defining computational thinking for mathematics and science classrooms.* Link:https://scholar.google.com/scholar?q=Defining+computational+thinking+for+mathematics+and+science+classrooms