Education and Information Technologies (2024) 29:2679–2695
https://doi.org/10.1007/s10639-023-11955-7

The investigation of human attention networks on debugging performance

Arif Akçay[1] **· Arif Altun**[2]

Received: 12 March 2023 / Accepted: 4 June 2023 / Published online: 13 June 2023

Abstract

Debugging is an intellectually rich and a challenging process when learning a programming language. This process is important for increasing the quality of the program and making it functional. Debugging, by its nature, is thought to be a practice with a state of focus and concentration. This study explored whether the debugging performance could be predicted by attention networks of students studying at the IT department of vocational high schools. 108 vocational high school IT department students participated in the research. First, students' attention levels were determined by utilizing the Attention Network Test. Second, a Debugging Performance Test was administered to determine their debugging performance. The analysis result indicated that the general attention network statistically predicted the debugging performance but the effect size was found to be small. Furthermore, the singular and dual interactions of the alerting attention network, the orienting attention network, and the executive attention network did not predict debugging performances. Yet, the interactions of the three attention networks on predicting the debugging performances were found to be statistically significant. These findings were discussed together with the existing studies in the literature and suggestions to obtain more in-depth information regarding these results were provoked.

Keywords Bug · Debugging · Attention network · Programming

✉ Arif Akçay
aakcay@kastamonu.edu.tr

Arif Altun
altunar@hacettepe.edu.tr

[1] Faculty of Education, Department of Educational Sciences, Kastamonu University, Kastamonu 37200, Turkey

[2] Faculty of Education, Department of Computer Education and Instructional Technology, Hacettepe University, Ankara 06800, Turkey

1 Introduction

Programming is the practice of analyzing the situation, preparing an algorithm, designing the program, writing the code, testing it, and finally debugging it to solve a problem. The situations encountered during this implementation process that negatively affect the operation of the produced program are called bugs, and the process of detecting and eliminating these bugs is called debugging (Downey & Mayfield, 2016). A programmer is required to solve problems, understand the algorithm, know the programming language and debug it when needed (Downey & Mayfield, 2016). When encountered a problem, these bugs reduce the quality of the program and make it even dysfunctional (Peng et al., 2016). For this reason, finding and eliminating bugs in the development process will prevent negative situations that will occur in the future and cause more expenses (Chen & Lim, 2013).

Being a good debugger ensures that bugs are quickly found and fixed (Dooley, 2011). However, students have different strategies for debugging (Magana et al., 2019). Because, debugging requires a deeper skill than writing new code (Liu et al., 2017). It is stated that an expert debugger will also be an expert in programming, but the reverse is not always true (Ahmadzadeh et al., 2005; Fitzgerald et al., 2008; McCauley et al., 2008). In addition, the self-confidence of the programmer who has debugging skills will be high (Ahmadzadeh et al., 2005). Considering all these reasons, it is important for programmers to improve in debugging as well as writing the code.

Learning and teaching debugging is as difficult as it is important (McCauley et al., 2008; Stefik and Gellenbeck, 2011; Xinogalos, 2016). During debugging, abstract concepts, information, and complex cognitive skills are used beyond the common programming language requirements (Castelhano et al., 2019). While debugging, the programmer should know the difference between the planned software and the actual software, the application language, application area, error bugs, and debugging methods, and use them simultaneously (Ducasse & Emde, 1988). In this context, various researchers have studies comparing the debugging performance of expert and novice programmers (Ahmadzadeh et al., 2005; Bednarik, 2012; Bednarik & Tukiainen, 2007a, b; Chen & Lim, 2013; Lin et al., 2016; Romero et al., 2002; Sharif et al., 2012; Yen et al., 2012). In addition, mathematics, statistics, and software knowledge levels and attitudes (Ko, 2002; Ko & Uttl, 2003) are individual differences whose effects on debugging performance are examined. Also, the effect of demographic differences such as gender (Beckwith et al., 2006; Chandrika & Amudha, 2017; Hung & Wang, 2021; Ko, 2002; Ko & Uttl, 2003) and age (Chandrika & Amudha, 2017; Ko & Uttl, 2003) on debugging performance was examined. Verbal intelligence, spatial intelligence, mathematical intelligence (Ko, 2002; Ko & Uttl, 2003), cognitive styles (Hung & Wang, 2021), executive functions, and problem-solving skills (Kovari, 2020) are the other variables examined as cognitive individual differences. In addition to all these variables, the mood of the programmer and the effects of physical exercise were examined (Khan et al., 2011). Looking at the above-mentioned studies, the researchers examined the debugging performance of the students in terms of their knowledge levels in various fields, demographic characteristics, and cognitive and affective differences.

In a programming process, students or professional programmers may encounter bugs while writing codes. These bugs may result from a lack of experience or attention (Rahman et al., 2020). It is necessary to realize that there is a bug in the program, to search for the bug among the codes, and to find and remove it. Considering all these stages, programmers need to focus and pay attention (Fitzgerald et al., 2008). Programmers are expected to perform mental tasks such as focusing, paying attention to, and integrating the new ideas when they need to monitor code to find bugs (Lowe, 2019). Yet, no research is reported whether individual differences in attention levels would be another variable to be considered when teaching debugging as a skill. Furthermore, it is necessary to determine students' visual attention to develop debugging skills (Emhardt et al., 2020; Romero et al., 2004). Based on all these statements and suggestions, it is a valid quest to explore whether students' attention as an individual difference could affect their debugging performance.

2 Literature review

2.1 Bugs and debugging

It is difficult to write codes in a programming process and there is a high probability of bugs (Downey & Mayfield, 2016). In its first definition, programming is a goal-plan (goal/plan) process; bugs are defined as situations that prevent the implemented plans from reaching the goal (Johnson & Soloway, 1985; Soloway & Ehrlich, 1984; Spohrer et al., 1985). Bugs negatively affect the operability of the program and reduce the effectiveness of the program (Downey & Mayfield, 2016; Duraes et al., 2016). It is necessary to eliminate the bugs that occur in order to ensure that the program works effectively.

Various classifications of bugs have been made to reduce the possibility of making them and eliminate them more effectively and efficiently (IEEE Computer Society, 2010; Robins et al., 2006). Although there are different classifications in the studies (Ahmadzadeh et al., 2005; Downey & Mayfield, 2016; Hristova et al., 2003; IEEE Computer Society, 2010; Johnson et al., 1983), it was seen that bug types were grouped under three headings when classifications, definitions, and examples were examined. These are compiler-time bugs, run-time bugs, and logical bugs. Compile-time bugs are syntactic (Syntax) errors that are encountered at compile time by making them in the syntax, which may vary specific to the programming language being written. The program goes through the compilation process when there is no syntactic error. Semantic errors that cause the program to generate an error while running and terminate are run-time bugs, the second of the bug types. The third type of bug, logical bugs, is the unexpected output of the compiled and running program.

Debugging is the process of examining the source of bugs encountered while testing written codes, finding and fixing them, and retesting to ensure that the bug has been eliminated (Dooley, 2011). As frustrating as it is to have bugs, debugging is an intellectually rich, challenging, and interesting part of programming. Debugging requires acting like a detective while searching for bugs using clues. It is also similar

to experimental science in terms of testing hypotheses to eliminate bugs when found (Downey & Mayfield, 2016).

Debugging is the simultaneous use of knowledge requiring different cognitive skills. The programmer encounters remarkable cautionary cues such as bug messages while debugging. He has to keep his attention on searching after he realizes there is a bug. It is necessary to direct his attention to the codes and detect the bugs. The programmer who detects the bugs must think, decide and fix them. In this respect, the attention capacity of programmers is an individual difference that can affect their debugging performance. In the following section, detailed information about the mechanism of attention networks in humans is described briefly.

2.2 Attention network

Attention is the clear and unequivocal possession of one of several objects or thoughts at the same time. At the core of attention lies focus, concentration, and awareness. Attention is giving up other things in order to deal with some things more effectively (James, 1950). It cannot be explained by human behavior alone. There is an anatomical structure underlying these behaviors. This structure generates neural signals to perform certain tasks that require attention (Fan et al., 2002, 2005). Three attention networks have been defined in the literature: (1) alerting, (2) orienting, and (3) executive attention networks (Fan et al., 2002, 2005; Posner & Fan, 2008). These attention networks are functionally separated, independent, and are not related to each other (Fan et al., 2002). In addition, this independence has been proven by neuroimaging studies (Fan et al., 2005). It has been found that different brain regions are activated during the performance of different attention tasks (Fig. 1).

Alerting Attention Network, provides the formation and maintenance of the arousal state of the individual's attention (Fan et al., 2002). It constitutes the state of being aroused, continuing, and ready to respond to stimuli in the environment (Raz, 2004). It is an internal change in which arousal and arousal occur (Fan et al., 2009). It can be thought that providing the necessary arousal for debugging when a bug is encountered and first noticed while programming may be related to the activity level of this attention network.

Orienting Attention Network, is about the shifting of attention from one place or object to another (Raz, 2004). This attention network is activated by a reflex for the appearance of a sudden stimulus or by a conscious visual search for the stimulus. While orienting is often associated with the eye, it is possible to process the stimulus implicitly without looking (Fan et al., 2009). During the debugging process, the programmer's search for the bugs, navigating through the codes, and detecting the codes with bugs may be related to the activity level of this attention network.

Executive Attention Network, related to the resolution of response complexity (Fan et al., 2002; Raz, 2004). This network is active when the individual needs to analyze the event and make decisions in conflicting situations. The executive attention network is needed in situations such as planning, making decisions, responding, and overcoming difficult or ordinary activities (Fan et al., 2009). It can be thought that decision-making situations while fixing the bugs in a debugging process may be related to the efficiency of the executive attention network.

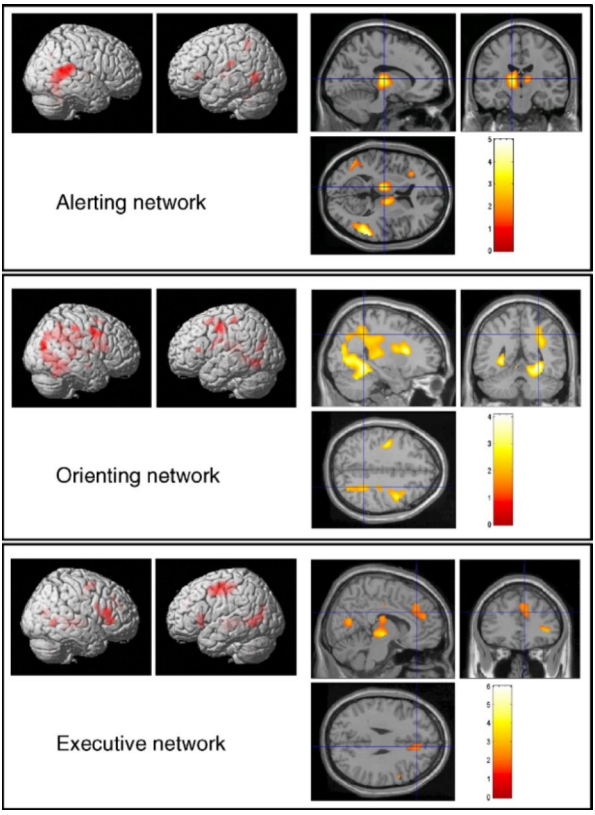

Fig. 1 fMRI results for the three attentional networks (Fan et al., 2005). There is the thalamic activation for alerting network. There is the parietal activation for the orienting network and ACC activation for the executive network

In Turkey, vocational high school Information Technologies Department students receive basic and field-specific programming language courses such as Programming Fundamentals, Object Oriented Programming, and Web Design and Programming (Milli Eğitim Bakanlığı [The Ministry of National Education], 2020a). The aim of these courses is to equip students with knowledge and skills such as having information about bugs, checking bugs, understanding bug messages, noticing and fixing bugs, and taking precautions against the occurrence of bugs (Milli Eğitim Bakanlığı [The Ministry of National Education], 2020b). While students gain this knowledge and skills, they employ their knowledge, skills, and characteristics in different disciplines. These individual differences affect their perceptions of bugs, their search behavior, and their detection and solution performance (Ko, 2002; Ko & Uttl, 2003; Lin et al., 2016; Sharif et al., 2012). In this study, it is aimed to examine the effects of attention network activity levels of vocational high school IT department students to predict their debugging performance. It is thought that this study will serve to explain the differences in students' debugging performance and determine the type of attention that is effective on them. In this way, it may be possible to make predictions about the behaviors that will affect the debugging performance of the students. In addition, determining the capacities of individuals can enable decision-making to increase debugging performance and the development of software to assist debugging (Khan et al., 2011). Because there is a need for activities and tools to support the

development of these challenging skills (Fields et al., 2021). In this way, permanent scientific solutions will be offered to reduce the number of bugs, which are one of the deep-rooted problems of the software development process (Duraes et al., 2016).

The problem of the research is to determine whether the attention levels of the students studying in the Information Technologies departments of vocational high schools are predictive of their debugging performance.

RQ1: Does the general attention network of students predict their debugging performance?

RQ2: Do students' different attention networks (alerting attention network, orienting attention network, executive attention network) predict debugging performance?

3 Method

The study is designed as a correlational predictive design to examine whether students' attention networks predict their debugging performance. Figure 2 displays the research design process of the study.

3.1 Participants

A total of 108 high school students attending the information technologies department in three different vocational high schools located in the north of Turkey were invited to take part in the research. Data from 28 students were either incomplete or missing due to their absence; therefore, they were omitted. Finally, data from 80 students were included in the research. Of the participating students, 31 (38.8%) were male and 49 (61.3%) were female students. None of the participants received any programming training before high school. All of the students studied C# programming language training at the vocational high school with the same teaching curriculum.

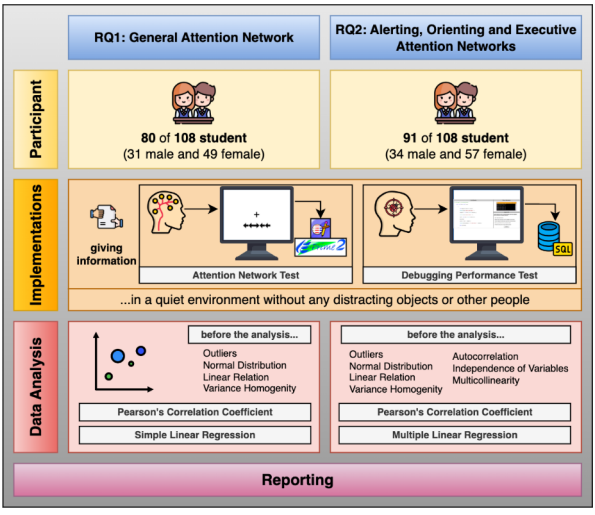

Fig. 2 Experimental design process

As a result of the analysis of the assumptions made to find the answer to the second research question, the data of 91 of 108 students are valid. Of these students, 34 (37.4%) are male and 57 (62.6%) are female students. None of the participants had any programming education before high school and all of them studied C# programming language education in a vocational high school with the same teaching curriculum. The project was approved by Hacettepe University Ethics Commission under-report no 35853172-300. All participating students and their families signed the informed consent forms.

3.2 Data collection tools

Attention Network Test The Attention Network Test was developed by Fan et al. (2002), and it measures the levels of three attention networks (alerting, orienting, and executive). The test also provides the general attention network score with the average of the three attention networks. Fan et al. (2002) provided information on the content of the test, its implementation, and the determination of attention network levels in their study. In the study, the reliability of the test was examined by test-retest analysis. As a result of the analysis, the correlation is 0.87 for the general attention network, 0.52 for the alerting attention network, 0.61 for the orienting attention network, and 0.77 for the executive attention network. Fan et al. (2005) showed that networks are independent of each other in their neuroimaging study. Only a low-level correlation was observed between the general attention network and the executive attention network (r = .44). The test was prepared at Hacettepe University Educational Ontology and Cognition Laboratory using E-Prime Professional 2.0 software.

Debugging performance test The Debugging Performance Test, developed by Akçay and Altun (2021), allows determining the debugging performance of the students studying in the IT department of vocational high schools in the C# programming language. There are 10 code scenarios in the test. These are compile-time bugs in 3 scenarios, runtime bugs in 3 scenarios, and logical bugs in 4 scenarios. The test is implied in a computer environment. In the test, there is a code scenario on the left side of the screen, and a question for the bugs in the lower right corner. In the upper right corner of the screen, a bug message for compile-time bugs and a screenshot for run-time bugs and logical bugs are displayed (Fig. 3). The highest 10 and the lowest 0 points are taken from the test.

Akçay and Altun (2021) conducted an item analysis to ensure the validity of the test. The difficulty of the items in the test ranges from 0.30 to 0.49. Item discrimination varies between 0.39 and 0.69. The reliability of the test was determined by the KR-20 Index and was found to be 0.58.

3.3 Implementations

The data collection process was carried out individually in the vocational high school where the participants studied, in a quiet environment without any distracting objects or other people. During the implementation process, we invited the students to the

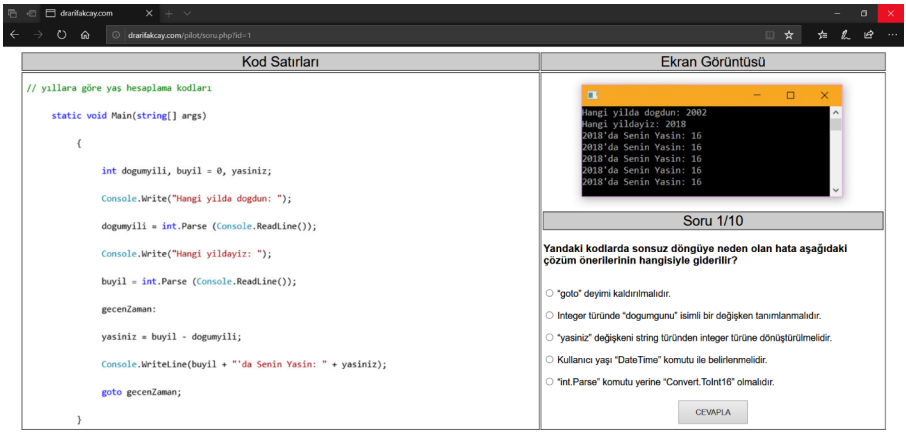

Fig. 3 A screenshot of Debugging performance test (Akçay & Altun, 2021)

experimental environment and gave information about the study. First, students completed the Attention Network Test to determine their attention network levels. Students used E-Prime Professional 2.0 software to complete the test. Students gave the desired responses in the test and their responses and response times were recorded by the software for analysis. The attention network test took approximately 18 min for each student.

After determining the levels of the attention networks of the students, a debugging performance test was implied to determine their debugging performance. The data collection process of the test took place in the computer laboratories of the students' high schools. Before the test was implied, the students were informed about the test. Students answered the test and their answers were recorded in the database.

3.4 Data analysis

Simple Linear Regression was run to examine whether the general attention network would predict the debugging performances. To predict the model, first, the relationship between the variables were explored with Pearson's Correlation Coefficient. Before running the analysis, outliers, normal distribution, linear relation, and variance homogeneity assumptions were checked. Outliers were checked with Q-Q graph and z score, normal distribution with Kolmogorov-Smirnov, linear correlation, and variance homogeneity with the scatterplot.

Second, Multiple Linear Regression was run to examine whether the alerting, orienting, and executive attention networks could predict the debugging performances. In addition to the assumptions mentioned above, the Durbin-Watson d statistic was used for the autocorrelation assumption and the Pearson's Correlation Coefficient value between the independent variables and VIF (Variance Inflation Factors) for the multicollinearity assumption.

4 Findings

4.1 General attention network and debugging performance

In the study, first, whether the general attention network would predict the debugging performance was examined. Descriptive statistics for students' debugging performance and general attention network levels, and the correlation between them were presented in Table 1.

As shown in Table 1, it was observed that students debugged an average of about 4 questions ($\bar{X}$ =3.59; Sd = 1.25; Min=1; Max=6). There is a significant and negative relationship between students' general attention network and debugging performance (r=-.44; p<.01). Simple Linear Regression Model analysis was conducted to examine whether this relationship is a predictor. The results are presented in Table 2.

The regression model, in which students' general attention network predicts their debugging performance, was found to be significant (F = 19.13; p < .05). According to this result, it can be said that the general attention network could predict the debugging performance (t=-4.37; p < .05). With each grade increase in the overall attention network, their debugging performance will decrease by 0.01.

4.2 Predicting the debugging performance: alerting, orienting, and executive attention networks

In the study, whether each attention network scores (alerting, orienting, and executive attention) would predict students' debugging performance was investigated. The descriptive statistics of students' attention networks and debugging performances and the correlations between them are given in Table 3.

As seen in Table 3, it was observed that students answered approximately four of the 10 debugging test questions correctly ($\bar{X}$ =3.86; Sd = 1.55; Min = 1; Max = 7). It was determined that students' debugging performance had a significant and low-level relationship with the orienting attention network (r = .32; p < .01). However, alerting attention network activity level (r=-.07; p > .05) and executive attention network activity level (r=-.20; p > .05) did not correlate with debugging performance.

Table 1 Correlation between Students' General Attention Network and Debugging Performance and Descriptive Statistics

	1	$\bar{X}$	Sd	Min	Max
1. Debugging Performance	1.00	3.59	1.25	1	6
2. General Attention Network	−0.44*	101.21	47.52	31.87	266.43

*p<.01

Table 2 Simple Linear Regression Analysis Results for Predicting Debugging Performance

	b_j	$S(b_j)$	β	t	p
Constant	4.76	0.29	-	16.00	0.00
General Attention Network	−0.01	0.00	−0.44	-4.37	0.00

n = 80; R = .44; R^2 = 0.20; F = 19.13; p < .05

Table 3 Correlation Coefficient between Students' Different Attention Networks and Debugging Performance and Descriptive Statistics

	1	2	3	4	$\bar{X}$	Sd	Min	Max
1. Debugging Performance	1.00				3.86	1.55	1	7
2. Alerting Attention Network	−0.07	1.00	-	-	43.57	30.50	-20.83	135.52
3. Orienting Attention Network	0.32**	0.24*	1.00	-	44.49	32.72	-57.27	146.01
4. Executive Attention Network	−0.20	−0.09	−0.31**	1.00	212.25	133.99	63.34	664.14

*p<.05; ** p<.01

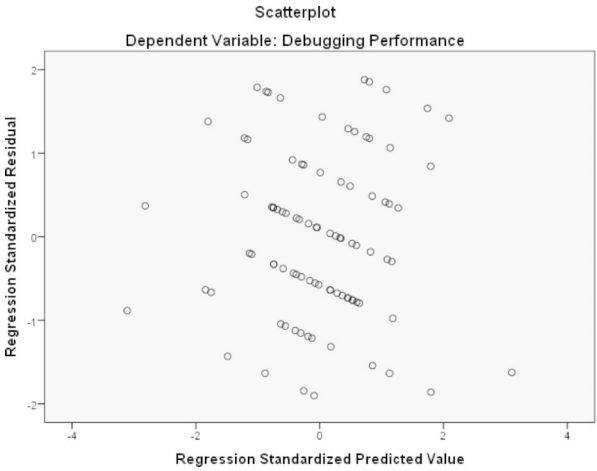

Fig. 4 Scatterplot of Multiple Linear Regression Model Standardized Residuals and Standardized Predicted Values

Before starting the multiple linear regression analysis, the assumptions were tested. Outliers were examined with Q-Q Plot and z score which showed that 91 out of 108 participants' scores were valid. Kolmogorov-Smirnov Test was used to examine the normal distribution and it was seen that the data were normally distributed (p>.05). The linearity and homogeneity of the data were examined by scatterplot (Fig. 4).

In Fig. 4, the data were determined to be linear and homogeneous. Durbin-Watson d statistics was used to examine the independence of standardized predicted residuals from each other. As a result of the statistics, it was determined that the d value was 1.96 and there was no autocorrelation when checked with the critical values table (p<.01). Pearson's Correlation Coefficient (Table 4) and VIF values (Alerting Attention Network: 1.06; Orienting Attention Network: 1.00; Executive Attention Network: 1.10) were used to examine multicollinearity among different attention networks, which are the independent variables of the regression analysis. There is no multicollinearity among attention networks. After examining all these assumptions, multiple regression analysis was performed and the results are given below (Table 4).

In Table 4, the regression model indicated a significant model (F=4.63; p<.05). Alerting attention network (t=-1.48; p>.05), orienting attention network (t=-0.70; p>.05) and executive attention network (t=-1.46; p>.05) regression coefficient was found to be insignificant on the prediction of debugging performance. In addi-

Table 4 Multiple Regression Analysis Results for Predicting Debugging Performance

	b_j	$S(b_j)$	β	t	p
Constant	4.63	0.93	-	4.99	0.00*
AAN	−0.03	0.02	−0.56	-1.48	0.14
OAN	0.02	0.02	−0.26	−0.70	0.48
EAN	−0.00	0.00	−0.30	-1.46	0.15
AAN X OAN	0.00	0.00	1.04	1.92	0.06
AAN X EAN	0.00	0.00	0.38	1.20	0.24
OAN X EAN	0.00	0.00	0.58	1.88	0.06
AAN X OAN X EAN	−0.00	0.00	0.94	-2.23	0.02*
n=91; R=.43; R^2=0.19; F=2.73; p<.05					

*p<.05; AAN: Alerting Attention Network; OAN: Orienting Attention Network; EAN: Executive Attention Network

tion, alerting attention network and orienting attention network interaction (t=1.92; p>.05), alerting attention network and executive attention network interaction (t=1.20; p>.05), and orienting attention network and executive attention network interaction (t=. 1.88; p>.05) was found to be insignificant as a predictor of debugging performance. However, the regression coefficient for the interaction of all three attention networks is significant to predict debugging performance (t=-2.23; p<.05). In the model, the interaction variable explains 19% of the debugging performance (R^2=0.19).

5 Discussion

This study investigated whether students' different attention networks predicted their debugging performance, which requires a state of focus and concentration. First, the effect of the general attention network on predicting the debugging performances was examined. The results indicated that there was a negative relationship between the general attention network of the students and their debugging performance. In addition, when the relationship between attention networks and debugging performance is examined, it is seen that only one of the three attention networks (orienting attention network) has a positive and low correlation with the debugging performance. The reason for this unexpected result may be the use of attention networks that are not related to debugging performance in calculating the general attention network level. It was found that the general attention network was statistically predictive of debugging performance. Yet, due to its low percentage in prediction, it would not be practical to utilize it alone. Some other related variables could be integrated to improve the model in future research.

This study also examined whether different attention networks would predict the debugging performance. The findings indicated that the alerting attention network, the orienting attention network, and the executive attention network did not predict the debugging performance. Furthermore, the interaction of attention networks with each other as a predictor of debugging performance was examined. Accordingly, it was determined that attention networks did not predict the debugging performance across dual interactions.

Further, the tripartite interactions of the alerting attention network, the orienting attention network and the executive attention network were found to be statistically significant in predicting the debugging performance. But, this predictor value yielded

a low coefficient. Therefore, it can be concluded that although it is statistically significant, it may not be practical to utilize it alone. These variables could be examined together with other related variables in future research.

In the literature, researchers have generally examined learners' attention limited to visual attention through eye tracking and self-report interviews during debugging (Bednarik, 2012; Bednarik & Tukiainen, 2007a, b; Kovari et al., 2020; Romero et al., 2002; Sharif et al., 2012). In their research of examining the debugging behaviors of the students, Xinogalos (2016) stated that the students exhibited irrational behaviors in debugging the program after compilation; also, they did not read the bug message. Accordingly, even if the students were given the necessary alerting, not examining the bug messages sufficiently might affect the debugging performance of the students. In addition, the use of a controlled environment in the experiment may not have achieved the required arousal. Failure to provide the necessary arousal while debugging may negatively affect the debugging performance of the student (Khan et al., 2011).

Attention was described by Mangaroska et al. (2022) as a "state of arousal when humans selectively concentrate on a discrete aspect of information. "The results of this study indicated that there was a correlation between the attention levels of Computer Science majors and their debugging performance. It is well-reported that students need a variety of skills and knowledge (i.e., Xie et al., 2019) in order to debug. It was also questioned that novice programmers might lack the necessary debugging skills and knowledge (Gao & Hew, 2023). One strategy novice programmers employ might be their choice of a trial-and-error approach rather than planning their techniques in advance like experts do (Michaeli & Romeike, 2019). Compared to the think-through strategy, the trial-error approach leads to lower debugging performances (Kovari et al., 2020). It is plausible to presume that the participants in this study are less skilled and perform worse than Computer Science major students in light of each of these characteristics. Additionally, Mangaroska et al. (2022) assessed students' debugging performance using five tasks created in Java without syntax bugs. When assessing the debugging skills of K–12 students, compile-time bugs must be taken into account. Because compile-time bugs are a common and serious problem for K–12 students while programming (Altadmri & Brown, 2015; Michaeli & Romeike, 2019). The results of this study bring a novel approach to explore the possible effects of students' attention performances in understanding the debugging process in K-12. Further research is needed to include an attention network to better conceptualize the debugging process to reach a broader model to represent the debugging process.

Students may also think that a boring practice such as debugging (Downey & Mayfield, 2016) will not benefit them; therefore, they may perform less attentively (Isen, 2008), assuming there will be no problem even if they don't answer correctly in the debug test. On the contrary, due to the student's anxiety during the debugging process (Isen, 2008), excessive arousal skills-based practices may cause the interruption of automatic behaviors and worsening of behaviors (Ko & Myers, 2005). Therefore, this situation may be reflected in debugging behavior. Further, the alerting attention network is affected by the instantaneous mood (Finucane et al., 2010), while the orienting and executive attention network is affected by the insomnia (Martella et

al., 2011). Future research would include those parameters when exploring the attention network on debugging.

6 Conclusion

This study investigated how human attention networks (alerting, orienting, executive, and general) would predict students' debugging performance, which is an important component of programming. The findings demonstrated a negative correlation between debugging performance and the general attention network. It is concluded that the debugging performance had a correlation with the orienting attention network. But, performance in debugging had not been found to be associated with either the executive attention network or the alerting attention network.

This study also showed that debugging performance was predicted by the general attention network. The predictive level was low; thus, it may not be practical to utilize it alone. Furthermore, the findings also indicated that different attention networks and dyadic interactions did not predict debugging performance. But, when the triple interaction of these attention networks had been taken into account, it could predict the debugging performance.

Some limitations should be raised. First, students' level of programming language knowledge level for this study was considerably limited. Debugging instruction was also limited in their curricula. Therefore, when presented with such exercises to debug for research purposes, they could have just disregarded error messages. Secondly, how much importance the students placed on the debugging activity and how they were feeling at the time of the activity may have had an impact on how well they performed the debugging. Finally, more research is needed to explore the correlational effects of attention networks along with other variables across students with different background knowledge and disposition levels.

6.1 Limitations and recommendations

This study carries a few limitations that might have influenced the research results. First of these limitations is related to the course content. It is assumed that all students followed the same curriculum, attended all courses, and studied all units of the course. The reason for this assumption is the lack of a valid and reliable measurement tool to measure programming knowledge in C# for Turkish students. Thus, it may be suggested to researchers to develop a measurement tool that can be used in such studies and that will enable students to control their prior knowledge.

Another limitation of the study is that the use of a valid and reliable debugging performance test may have provided students with a very controlled environment during their code writing process. Accordingly, errors and debugging process may contain individual-specific situations. From this point of view, it can be suggested to examine the debugging performances in a real-time programming process with qualitative and quantitative approaches.

Students may not read error messages and perform sensible debugging behaviors while debugging. In this context, a course in which students review the codes and teach debugging strategies can be added to the curriculum (Ciolkowski et al., 2002).

In this study, arousal, anxiety, stress, sleep, or mood, which can affect the attention of students and can change instantly, were not controlled. In order to obtain deeper information about these results, it is recommended that these variables be controlled and re-examined.

Funding This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Data availability The datasets generated and analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Informed consent Informed consent was obtained from all individual participants included in the study. The study procedures were conducted by attending the Declaration of Helsinki.

CRediT author statement Conceptualization: [Arif AKÇAY]; Data Curation: [Arif AKÇAY]; Formal Analysis: [Arif AKÇAY]; Investigation: [Arif AKÇAY]; Methodology: [Arif AKÇAY]; Project Administration: [Arif ALTUN]; Resources: [Arif ALTUN]; Software: [Arif AKÇAY]; Supervision: [Arif ALTUN]; Validation: [Arif AKÇAY, Arif ALTUN]; Visualization: [Arif AKÇAY]; Writing - Original Draft: [Arif AKÇAY]; Writing - Review & Editing: [Arif ALTUN]

Conflict of interest Authors declare they have no competing interests.

Research involving human participants - ethical approval All procedures performed in study involving human participants were in accordance with the ethical standards of the institutional and/or national research committee and with the 1964 Helsinki declaration and its later amendments or comparable ethical standards. The study was approved by Hacettepe University Ethics Commission under report n° 35853172-300. Informed consent was obtained from all individual participants and their families included in the study.

References

Ahmadzadeh, M., Elliman, D., & Higgins, C. (2005). An analysis of patterns of debugging among novice computer science students. *ACM SIGCSE Bulletin*, *37*(3), 84–88. https://doi.org/10.1145/1151954.1067472.

Akçay, A., & Altun, A. (2021). Hata ayıklama performansı testi geliştirme: Geçerlik ve güvenirlik çalışması [Test development for debugging performance: Validity and reliability study]. *Erzincan University Journal of Education Faculty*, *23*(3), 667–685. https://doi.org/10.17556/erziefd.815922.

Altadmri, A., & Brown, N. C. C. (2015). 37 million compilations: Investigating novice programming mistakes in large-scale student data. In A. Decker, K. Eiselt, J. Tims, & C. Alphonce (Eds.), *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (pp. 522–527). The Association for Computing Machinery. https://doi.org/10.1145/2676723.2677258.

Beckwith, L., Kissinger, C., Burnett, M., Wiedenbeck, S., Lawrance, J., Blackwell, A., & Cook, C. (2006). Tinkering and gender in end-user programmers' debugging. In R. Grinter, T. Rodden, P. Aoki, E. Cutrell, R. Jeffies, & G. Olson (Eds.), *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 231–240). https://doi.org/10.1145/1124772.1124808.

Bednarik, R. (2012). Expertise-dependent visual attention strategies develop over time during debugging with multiple code representations. *International Journal of Human-Computer Studies*, *70*(2), 143–155. https://doi.org/10.1016/j.ijhcs.2011.09.003.

Bednarik, R., & Tukiainen, M. (2007a). Validating the restricted focus viewer: A study using eye-movement tracking. *Behavior Research Methods*, *39*(2), 274–282. https://doi.org/10.3758/BF03193158.

Bednarik, R., & Tukiainen, M. (2007b). Analysing and interpreting quantitative eye-tracking data in studies of programming: Phases of debugging with multiple representations. *19th Annual Workshop of the Psychology of Programming Interest Group*. https://ppig.org/files/2007-PPIG-19th-Bednarik.pdf.

Castelhano, J., Duarte, I. C., Ferreira, C., Duraes, J., Madeira, H., & Castelo-Branco, M. (2019). The role of the insula in intuitive expert bug detection in computer code: An fMRI study. *Brain Imaging and Behavior*, *13*, 623–637. https://doi.org/10.1007/s11682-018-9885-1.

Chandrika, K. R., & Amudha, J. (2017). An eye tracking study to understand the visual perception behavior while source code comprehension. *International Journal of Control Theory and Applications*, *10*(19), 169–175. https://serialsjournals.com/abstract/11377_20-chandrika_k.r.pdf.

Chen, M., & Lim, V. (2013). Eye gaze and mouse cursor relationship in a debugging task. In Stephanidis C. (Ed.), *HCI International 2013 - Posters' Extended Abstracts* (pp. 468–472). Springer. https://doi.org/10.1007/978-3-642-39473-7_93.

Ciolkowski, M., Laitenberger, O., Rombach, D. H., Shull, F., & Perry, D. (2002). Software inspections, reviews and walkthroughs. In W. Tracz (Ed.), *Proceedings of the 24th International Conference on Software Engineering* (pp. 641–642). IEEE. https://doi.org/10.1145/581339.581422.

Dooley, J. (2011). *Software development and professional practice*. Springer. https://doi.org/10.1007/978-1-4302-3802-7.

Downey, A. B., & Mayfield, C. (2016). *Think Java: How to think like a computer scientist*. Green Tea Press. https://open.umn.edu/opentextbooks/textbooks/think-java-how-to-think-like-a-computer-scientist.

Ducasse, M., & Emde, A. M. (1988). A review of automated debugging systems: Knowledge, strategies and techniques. In T. N., Nam (Ed.), *Proceedings of 11th International Conference on Software Engineering* (pp. 162–171). IEEE Computer Society Press. https://doi.org/10.1109/ICSE.1988.93698.

Duraes, J., Madeira, H., Castelhano, J., Duarte, C., & Branco, M. C. (2016). WAP: Understanding the brain at software debugging. In Y. Labiche (Ed.), *2016 IEEE 27th International Symposium on Software Reliability Engineering* (pp. 87–92). IEEE. https://doi.org/10.1109/ISSRE.2016.53.

Emhardt, S. N., Kok, E. M., Jarodzka, H., Brand-Gruwel, S., Drumm, C., & van Gog, T. (2020). How experts adapt their gaze behavior when modeling a task to novices. *Cognitive Science*, *44*(9), 1–26. https://doi.org/10.1111/cogs.12893.

Fan, J., McCandliss, B. D., Sommer, T., Raz, A., & Posner, M. I. (2002). Testing the efficiency and independence of attentional networks. *Journal of Cognitive Neuroscience*, *14*(3), 340–347. https://doi.org/10.1162/089892902317361886.

Fan, J., McCandliss, B. D., Fossella, J., Flombaum, J. I., & Posner, M. I. (2005). The activation of attentional networks. *Neuroimage*, *26*(2), 471–479. https://doi.org/10.1016/j.neuroimage.2005.02.004.

Fan, J., Gu, X., Guise, K. G., Liu, X., Fossella, J., Wang, H., & Posner, M. I. (2009). Testing the behavioral interaction and integration of attentional networks. *Brain and Cognition*, *70*(2), 209–220. https://doi.org/10.1016/j.bandc.2009.02.002.

Fields, D. A., Kafai, Y. B., Morales-Navarro, L., & Walker, J. T. (2021). Debugging by design: A constructionist approach to high school students' crafting and coding of electronic textiles as failure artefacts. *British Journal of Educational Technology*, *52*(3), 1078–1092. https://doi.org/10.1111/bjet.13079.

Finucane, A. M., Whiteman, M. C., & Power, M. J. (2010). The effect of happiness and sadness on alerting, orienting, and executive attention. *Journal of Attention Disorders*, *13*(6), 629–639. https://doi.org/10.1177/1087054709334514.

Fitzgerald, S., Lewandowski, G., McCauley, R., Murphy, L., Simon, B., Thomas, L., & Zander, C. (2008). Debugging: Finding, fixing and flailing, a multi-institutional study of novice debuggers. *Computer Science Education*, *18*(2), 93–116. https://doi.org/10.1080/08993400802114508.

Gao, X., & Hew, K. F. (2023). A flipped systematic debugging approach to enhance elementary students' program debugging performance and optimize cognitive load. *Journal of Educational Computing Research*, *0*(0), 1–32. https://doi.org/10.1177/07356331221133560.

Hristova, M., Misra, A., Rutter, M., & Mercuri, R. (2003). Identifying and correcting Java programming errors for introductory computer science students. In S. Grissom (Ed.), *Proceedings of the 34th SIGCSE technical symposium on Computer science education* (pp. 153–156). Association for Computing Machinery. https://doi.org/10.1145/611892.611956.

Hung, J. C., & Wang, C. C. (2021). The influence of cognitive styles and gender on visual behavior during program debugging: A virtual reality eye tracker study. *Human-Centric Computing and Information Sciences*, *11*(22), 1–20. https://doi.org/10.22967/HCIS.2021.11.022.

IEEE Computer Society (2010). *IEEE standard classification for software anomalies*. https://doi.org/10.1109/IEEESTD.2010.5399061.

Isen, A. M. (2008). Some ways in which positive affect influences decision making and problem solving. In M. Lewis, J. M. Haviland-Jones, & L. F. Barrett (Eds.), *Handbook of Emotions* (pp. 548–573). The Guilford Press. https://psycnet.apa.org/record/2008-07784-034.

James, W. (1950). *The principles of psychology: Volume one*. Dover Publications.

Johnson, W. L., & Soloway, E. (1985). PROUST: Knowledge-based program understanding. *IEEE Transactions on Software Engineering*, *SE-11*(3), 267–275. https://doi.org/10.1109/TSE.1985.232210.

Johnson, W. L., Soloway, E., Cutler, B., & Draper, S. (1983). *Bug catalogue: I*. https://cpsc.yale.edu/research/technical-reports/1983-technical-reports.

Khan, I. A., Brinkman, W. P., & Hierons, R. M. (2011). Do moods affect programmers' debug performance? *Cognition Technology and Work*, *13*(4), 245–258. https://doi.org/10.1007/s10111-010-0164-1.

Ko, A. J. (2002). *Individual differences in programming, testing, and debugging strategies in a statistical end-user programming environment* [Bachelor Thesis]. Oregon State University.

Ko, A. J., & Myers, B. A. (2005). A framework and methodology for studying the causes of software errors in programming systems. *Journal of Visual Languages and Computing*, *16*, 41–84. https://doi.org/10.1016/j.jvlc.2004.08.003.

Ko, A. J., & Uttl, B. (2003). Individual differences in program comprehension strategies in unfamiliar programming systems. In S. Kawada (Ed.), *Proceedings of the 11th IEEE International Workshop on Program Comprehension* (pp. 175–184). IEEE Computer Society. https://doi.org/10.1109/WPC.2003.1199201.

Kovari, A. (2020). Study of algorithmic problem-solving and executive function. *Acta Polytechnica Hungarica*, *17*(9), 241–256. https://doi.org/10.12700/APH.17.9.2020.9.13.

Kovari, A., Katona, J., & Costescu, C. (2020). Evaluation of eye-movement metrics ina software debbuging task using GP3 eye tracker. *Acta Polytechnica Hungarica*, *17*(2), 57–76. https://doi.org/10.12700/APH.17.2.2020.2.4.

Lin, Y. T., Wu, C. C., Hou, T. Y., Lin, Y. C., Yang, F. Y., & Chang, C. H. (2016). Tracking students' cognitive processes during program debugging-An eye-movement approach. *IEEE Transactions on Education*, *59*(3), 175–186. https://doi.org/10.1109/TE.2015.2487341.

Liu, Z., Zhi, R., Hicks, A., & Barnes, T. (2017). Understanding problem solving behavior of 6–8 graders in a debugging game. *Computer Science Education*, *27*(1), 1–29. https://doi.org/10.1080/08993408.2017.1308651.

Lowe, T. (2019). Debugging: The key to unlocking the mind of a novice programmer? In P. K. Imbrie (Ed.), *2019 IEEE Frontiers in Education Conference* (pp. 1–9). IEEE. https://doi.org/10.1109/FIE43999.2019.9028699.

Magana, A. J., Fennell, H. W., Vieira, C., & Falk, M. L. (2019). Characterizing the interplay of cognitive and metacognitive knowledge in computational modeling and simulation practices. *Journal of Engineering Education*, *108*(2), 276–303. https://doi.org/10.1002/jee.20264.

Mangaroska, K., Sharma, K., Gašević, D., & Giannakos, M. (2022). Exploring students' cognitive and affective states during problem solving through multimodal data: Lessons learned from a programming activity. *Journal of Computer Assisted Learning*, *38*(1), 40–59. https://doi.org/10.1111/jcal.12590.

Martella, D., Casagrande, M., & Lupiáñez, J. (2011). Alerting, orienting and executive control: The effects of sleep deprivation on attentional networks. *Experimental Brain Research*, *210*, 81–89. https://doi.org/10.1007/s00221-011-2605-3.

McCauley, R., Fitzgerald, S., Lewandowski, G., Murphy, L., Simon, B., Thomas, L., & Zander, C. (2008). Debugging: A review of the literature from an educational perspective. *Computer Science Education*, *18*(2), 67–92. https://doi.org/10.1080/08993400802114581.

Michaeli, T., & Romeike, R. (2019). Current status and perspectives of debugging in the K12 classroom: A qualitative study. In A. K. Ashmawy & S. Schreiter (Eds.), *Proceedings of the 10th IEEE Global Engineering Education Conference* (pp. 1030–1038). IEEE. https://doi.org/10.1109/EDUCON.2019.8725282.

Milli Eğitim Bakanlığı [The Ministry of National Education] (2020b). *Programlama temelleri ders bilgi formu [Programming fundamentals course information sheet]*. http://kitap.eba.gov.tr/panel/dosyalar/upload/0/0/P_0_24_07_2020_1_20_31_342.pdf.

Milli Eğitim Bakanlığı [The Ministry of National Education]. (2020a). *Bilişim Teknolojileri alani çerçeve öğretim programi [A framework curriculum of Information Technology Department]*. Programi&tur=mtal&sinif=9. http://meslek.eba.gov.tr/?p=Ogretim.

Peng, F., Li, C., Song, X., Hu, W., & Feng, G. (2016). An eye tracking research on debugging strategies towards different types of bugs. In S. I. Ahamed, C. K. Chang, W. Chu, I. Crnkovic, P.-A. Hsiung, G. Huang, & J. Yang (Eds.), *2016 IEEE 40th Annual Computer Software and Applications Conference* (pp. 130–134). IEEE. https://doi.org/10.1109/COMPSAC.2016.57.

Posner, M. I., & Fan, J. (2008). Attention as an organ system. In R. James, & Pomerantz (Eds.), *Topics in Integrative Neuroscience: From cells to Cognition* (pp. 31–61). Cambridge University Press. https://doi.org/10.1017/CBO9780511541681.005.

Rahman, M. M., Watanobe, Y., & Nakamura, K. (2020). Source code assessment and classification based on estimated error probability using attentive LSTM language model and its application in programming education. *Applied Sciences*, *10*, 1–21. https://doi.org/10.3390/app10082973.

Raz, A. (2004). Anatomy of attentional networks. *The Anatomical Record (Part B: New Anat)*, *281B*, 21–36. https://doi.org/10.1002/ar.b.20035.

Robins, A., Haden, P., & Garner, S. (2006). Problem distributions in a CS1 course. In D. Tolhurst & S. Mann (Eds.), *Proceedings of the 8th Australasian Conference on Computing Education* (Vol. 52, pp. 165–173). Australian Computer Society, Inc. https://doi.org/10.5555/1151869.1151891.

Romero, P., Cox, R., du Boulay, B., & Lutz, R. (2002). Visual attention and representation Switching during Java program debugging: A study using the restricted focus viewer. In M. Hegarty, B. Meyer, & N. H. Narayanan (Eds.), *Diagrammatic Representation and Inference* (Vol. 2317, pp. 221–235). Springer. https://doi.org/10.1007/3-540-46037-3_23.

Romero, P., Boulay, B., Cox, R., Lutz, R., & Bryant, S. (2004). Dynamic rich-data capture and analysis of debugging processes. *16th Annual Workshop of Psychology of Programming Interest Group*. https://ppig.org/files/2004-PPIG-16th-romero.pdf.

Sharif, B., Falcone, M., & Maletic, J. I. (2012). An eye-tracking study on the role of scan time in finding source code defects. In S. N. Spencer (Ed.), *Proceedings of the Symposium on Eye Tracking Research and Applications*, (pp. 381–384). Association for Computing Machinery. https://doi.org/10.1145/2168556.2168642.

Soloway, E., & Ehrlich, K. (1984). Empirical studies of programming knowledge. *IEEE Transactions on Software Engineering*, *SE-10*(5), 595–609. https://doi.org/10.1109/TSE.1984.5010283.

Spohrer, J. C., Soloway, E., & Pope, E. (1985). A goal/plan analysis of buggy pascal programs. *Human-Computer Interaction*, *1*(2), 163–207. https://doi.org/10.1207/s15327051hci0102_4.

Stefik, A., & Gellenbeck, E. (2011). Empirical studies on programming language stimuli. *Software Quality Journal*, *19*, 65–99. https://doi.org/10.1007/s11219-010-9106-7.

Xie, B., Loksa, D., Nelson, G. L., Davidson, M. J., Dong, D., Kwik, H., Tan, A. H., Hwa, L., Li, M., & Ko, A. J. (2019). A theory of instruction for introductory programming skills. *Computer Science Education*, *29*(2–3), 205–253. https://doi.org/10.1080/08993408.2019.1565235.

Xinogalos, S. (2016). Designing and deploying programming courses: Strategies, tools, difficulties and pedagogy. *Education and Information Technologies*, *21*, 559–588. https://doi.org/10.1007/s10639-014-9341-9.

Yen, C. Z., Wu, P. H., & Lin, C. F. (2012). Analysis of experts' and novices' thinking process in program debugging. In K. C. Li, F. L. Wang, K. S. Yuen, S. K. S. Cheung, & R. Kwan (Eds.), *Engaging Learners Through Emerging Technologies* (Vol. 302, pp. 122–134). Springer. https://doi.org/10.1007/978-3-642-31398-1_12.