Decoding Success: Assessing the Online Pseudocode Interpreter through Users Insights

Tracy N. Tacuban*, Loreto G. Gabawa, Jr., Christian Lester D. Gimeno, May Florence J. Franco, and Lenny M. Amar

Computer Department, Faculty, Iloilo Science and Technology University, Iloilo City, Philippines
Email: tracy.tacuban@isatu.edu.ph (T.N.T.); loreto.gabawa.jr@gmail.com (L.G.G.); christianlester.gimeno@isatu.edu.ph (C.L.D.G.); mayfjf@gmail.com (M.F.J.F.); lenny.amar@isatu.edu.ph (L.M.A.)
*Corresponding author

Manuscript received November 20, 2023; revised January 2, 2024; accepted February 15, 2024; published May 24, 2024

***Abstract*—Pseudocode is vital in enhancing programming comprehension, making it an essential component of the Programming Logic Formulation (CS1) course. However, many CS1 students from various senior high school strands struggle with this concept, highlighting the need for additional learning resources. This research evaluated a pseudocode interpreter designed for first-year Information and Communications Technology (ICT) students at Iloilo Science and Technology University in Iloilo City, Philippines. Using random sampling, 147 first-year ICT students were selected to evaluate the software's user experience in usability, credibility, appearance, and loyalty using the Standardized User Experience Percentile Rank Questionnaire (SUPR-Q). Likewise, five ICT experts were selected using purposive sampling to evaluate the system's validity as instructional material based on its content and on the material's organization, form, and style using the university's Quality Form for Instructional Materials (IM). The user experience evaluation yielded positive results, with students unanimously agreeing on the software's ease of use, dependability, and user-friendliness. The expert evaluation confirmed the system's validity as instructional material, commending its content, organization, form, and style. These findings suggest that the pseudocode interpreter provides immediate benefits for students, offering a valuable learning tool. Moreover, the software has the potential to be adopted by other faculty members, promoting a more efficient and captivating learning experience. Future research could investigate how interpreters affect student learning and their usefulness in various educational settings.**

***Keywords*—instructional material, programming logic, educational innovation, pseudocode interpreter**

I. Introduction

The Programming Logic Formulation (CS1) course is a mandatory first-year subject for students across Information and Communications Technology (ICT) programs at Iloilo Science and Technology University, located at Burgos St., La Paz, Iloilo City. It is an essential subject wherein students learn programming fundamentals by designing and constructing a computer program's logical structure, including pseudocode. A pseudocode employs a natural syntax, enabling students to concentrate on program logic and structure without having to learn a particular programming language, thereby enhancing the learning process [1].

However, teaching pseudo-coding to CS1 students is a challenge for educators due to the diverse academic backgrounds of ICT students, who come from various tracks or specializations within the senior high school system. In Senior High School, the Information and Communications Technology (ICT) specialization from the Technical-Vocational-Livelihood (TVL) track is the only strand that provides specialized courses related to ICT. Besides TVL, undergraduate ICT students could also come from the Academic, Arts and Design, and Sports tracks, with each track further divided into strands where students have specializations distinct from ICT. Likewise, despite ICT being regarded as a tool to improve the teaching and learning process, its implementation has shortcomings, including the absence of information on how to use ICT, the lack of coordination of efforts from the public and private sectors, and insufficient teacher preparation [2]. ICT is also the top strand that is less offered due to limited funds for facilities and teaching personnel, and the situation is evident that even highly-developed regions like the National Capital Region (NCR), Ilocos Sur, Eastern Samar, and Davao del Sur still need to offer ICT strand fully [3].

Consequently, a significant proportion of students entering ICT-related programs in higher education institutions need a background in programming concepts, posing challenges in understanding abstract programming principles. Since the programming concepts are interdependent, teaching students basic computer skills before advancing into higher courses is required [4]. Students, therefore, must learn the basics, as it would be difficult for them to grasp the higher-level topics of the course when they fail to grasp the necessary programming concepts at the early stage of the course. This observation is supported by Hsiao *et al.* [5], where the researchers found that it is important to develop the students' programming logic ability before their actual programming courses. Computer programming courses are difficult for those who are new to these concepts and will soon lose interest in studying the course

To address this gap, the faculty members teaching CS1 introduce fundamental concepts such as variable naming, data types, operators, control structures, algorithms, pseudo coding, and desk checking using the concepts presented by Robertson [6]. An algorithm is a fundamental tool in mathematics and programming and an algorithmic approach is taught CS1 students to know how to create a series of step-by-step procedures or a set of well-defined instructions to solve a specific problem. The algorithm is written as pseudocode using natural language instead of actual programming code. These pseudocodes are then checked using desk check values and tables through manual walkthroughs to verify correctness.

According to Gimeno [7], pseudocoding is difficult for students because it is written on pen and paper, and they need help executing the code to validate its output. The

pseudocode should first be submitted to their professors for checking, and the feedback is usually delayed. Likewise, teaching pseudocode is difficult for educators as there is no standard syntax for teaching pseudocode, so book authors and educators have individual syntax, techniques, and rules for writing pseudocode [7].

A pseudocode student would need a platform to execute their pseudocode to see its output. Likewise, there needs to be learning material to serve as a reference for the pseudocode syntax to make the teaching and learning process efficient. Meanwhile, research on the development of pseudocode interpreters remains limited, and there needs to be more studies about the evaluation of pseudocode interpreters based on user experience and instructional material to ensure user satisfaction and validity.

Recognizing this, the researchers have developed a pseudocode interpreter software to serve as an Integrated Development Environment (IDE) for pseudocoding. The researchers advocate for a more practical approach involving hands-on coding and problem-solving. This approach aligns with a study suggesting that to avoid the complexity of programming languages, students new to programming should be taught engagingly by improving pedagogy to introduce a more focused algorithm visualization [8]. The research aims to achieve two objectives. The first objective is to evaluate the user experience of students using the pseudocode interpreter in terms of the software's usability, credibility, appearance, and loyalty using the SUPR-Q questionnaire. It is essential to assess the user's experience from the perspective of the user to get a holistic view of the software, as they will constitute the end-users. The second objective is to assess the validity of the pseudocode interpreter as an instructional material. This assessment is crucial to ensuring the tool's measurement accuracy and that it improves learning outcomes, guides instructional designs, and aligns with instructional objectives. The study's outcomes will serve as a foundation for refining the software, ensuring it aligns with user expectations and needs. This study will help with continual improvement and develop a better instructional tool.

II. Related Literature

A. Pseudocode Interpreter

Teaching pseudocode in introductory programming subjects plays a vital role in learning [9–12]. According to these studies, the simplified language of pseudocode, which uses descriptive phrases, allows the students to focus on the logic and structure of the program without getting tied to the syntax of a programming language. Pseudocode is intended to enhance the algorithmic thinking and problem-solving skills of students and serve as their foundation for more advanced subjects

A primary drawback of pseudo-code is that it is not executable, and students would manually trace the execution of the pseudo-code line-by-line and validate its processing output using a desk check table. Several studies explored processing pseudocode to generate an equivalent programming language source code, like in C++ [13], in any language [14], or C/C++/Java [15]. The output source code of such systems can then be compiled, executed, or shared among developers. These studies did interpret pseudocodes but closely resembled pseudocode translators, and most of these studies employed machine learning and other advanced concepts in the translation process

Meanwhile, limited studies focused on directly executing pseudocode by constructing a pseudocode interpreter, even though pseudocode is not a programming language. Generally, an interpreter is a computer program that executes instructions in a high-level language without converting them into machine code [16]. However, a pseudo-code interpreter is a software tool that parses and executes pseudocodes [7, 17]. This type of software is particularly useful in educational settings, where students can implement the logic of algorithms and directly execute pseudocode without learning a specific programming language. It is also invaluable for educators, as they can use the tool as instructional material by simplifying the teaching process and standardizing the syntax.

PseudoJ is a pseudo-code interpreter designed to transform pseudocode to its equivalent Java code [17]. The authors identified two major components of the interpreter: (a) pseudocode collectors to import a pseudocode into the code editor for parsing, and (b) pseudocode parsers to process the code before being passed to the code editor. The parser used a trained data file or rule file to standardize the pseudocode and translate the result into its equivalent Java code.

The core of the program uses the Abstract Syntax Tree (AST) data structure to process the Java syntax. PseudoJ interprets pseudocode in the literal sense and does not execute the statements. Instead, the technology was implemented as a plugin for the Eclipse IDE to predict the Java syntax from a pseudocode input on the code editor. Like PseudoJ, this study utilized AST data structure but differs as an interpreter as this study processes pseudocode and executes each statement to produce an output.

Meanwhile, Gimeno [7] developed a pseudocode interpreter, or an integrated development environment consisting of a lexical analyzer, a parser, and an interpreter. The lexical analyzer processes the pseudocode into tokens, and through them, the parser creates commands, which it places into a data structure called an abstract syntax tree (AST). The interpreter then evaluates the AST to produce an output. The software allows the student to write, validate, and run their pseudocode to see its output.

This study is similar to Gimeno [7] in supporting the same set of operators, expressions, and statements. However, this study differs as it supports array definition and processing, function definition, recursive function, and the implementation of a desk check table. A desk check table, also known as a trace table, is a manual process used in debugging, primarily in the pseudocode design phase, and involves simulating the flow of control through the pseudocode and manually recording the changes to the variables on a table. It allows the student to check the logic of the pseudocode systematically. An automatically generated desk check table will eliminate the laborious manual process of tracing and updating the values of each variable used in an algorithm, thereby promoting efficiency and hastening the learning process.

Moreover, this study is remotely installed, delivered over the internet, and can be accessed by users using their desktop, laptop, or mobile devices. Being deployed on the web offers

several advantages, including access to the software anytime and anywhere, ease of use, security, and the possibility for students to collaborate and share their work with fellow students. In terms of validation, [7] used ISO 9126 to evaluate software quality in terms of functionality, reliability, usability, efficiency, maintainability, and portability. This study, however, used SUPR-Q to evaluate user experience in terms of usability, trust, loyalty, and appearance and validated the system as instructional material using a university's assessment form for instructional material.

B. Instructional Material

Instructional materials are resources that are used to improve students' knowledge and understanding of a subject. These can include textbooks, workbooks, software, online resources, videos, games, and other forms of media [18–20].

The primary purpose of instructional materials is to provide a rich learning experience for students [18]. They serve as a guide for instruction, allowing students to explore concepts more fully and apply their understanding in new and varied contexts [18]. Instructional materials also provide learners with real-life learning experiences [20].

Instructional materials are crucial in teaching and learning processes [18–20]. They help make learning more concrete and meaningful, and make abstract or complex concepts more understandable [18–20]. Effective instructional materials complement and enhance students' work both during and outside of class [20], especially if the material is accessible online.

Despite their importance, the use of instructional materials is challenging. These can include issues such as the quality and relevance of the materials, the availability and accessibility of resources, and the need for training and support in their use [21, 22]. The paramount issue with instructional materials is that they are evaluated for validity before utilization.

III. Materials and Methods

The study used the ADDIE instructional design, which includes the analysis, design, development, implementation, and evaluation phase. As the instructional designers move from one phase to another, the instructional material is assessed and evaluated against instructional requirements to determine whether it conforms to the needs of the learners. The ADDIE training model helps instructional designers provide more effectively designed programs and learning objectives and materials that are more clearly defined and structured [23]. The model was selected for its comprehensiveness and has been proven to be effective in enhancing the quality and clarity of instructional materials and programs.

The study also integrated Design-Based Research (DBR) since it included developing and evaluating an online pseudocode interpreter. The design-based model bridges the gap between theory and practice, aiming for a deeper comprehension of the learning processes and their associated factors. Typically, a DBR approach necessitates utilizing both quantitative and qualitative methods, combining theory-driven designs with empirical research, all within a continuous cycle of "design, enactment, analysis, and redesign", which can be meticulously documented. DBR delves into the intricacies of education to gain a deeper comprehension of the processes related to learning, including various factors and their effects. Researchers accomplish this through various methods, a systematic approach, and examining instructional strategies and tools.

The data generated through DBR offers valuable insights into "how, when, and why educational interventions come into practice," ultimately leading to improved outcomes and the facilitation of evidence-based innovation implementation. This process thrives on close collaboration among researchers, designers, and participants [24, 25]. The developed software was evaluated to determine whether it is functional and easy to use.

A. Research Participants

The research participants are first-year students enrolled in the Programming Logic Formulation (CS1) class randomly selected during the first semester of the academic year 2023–2024. The participants were invited to use the software for the entire midterm period and requested to evaluate the system using a Google form. The number of responses was monitored, and after having reached 147 respondents, the researchers tabulated and interpreted the result.

The researchers also invited five experts composed of ICT experts and Faculty members to evaluate the online pseudocode interpreter. Among the evaluators, three came from the academe, while two were software developers from the ICT industry. The respondents were chosen using purposive sampling as they have specific characteristics relevant to the study. The ICT experts demonstrated proficiency in the design of information systems, while the faculty members possessed backgrounds in ICT and were deemed end-users of the system as instructional material.

B. Data Gathering Instruments

The researchers utilized the Standardized User Experience Percentile Rank Questionnaire (SUPR-Q) to evaluate the pseudocode interpreter from the perspective of the students. The SUPR-Q is a valid and reliable instrument to measure the quality of the user experience in terms of usability, trust and credibility, appearance, and loyalty [26, 27]. User experience goes beyond usability and considers the user's emotional response, enjoyment, and the overall experience of interacting with the technology, aligned with the user's sense of beauty [28].

Comprising eight standardized questions, the SUPR-Q's first seven questions were scored on a scale of 1–5, with five indicating "Strongly Agree" and one indicating "Strongly Disagree." In addition, the eighth question included a Net Promoter Score (NPS) measurement, which assessed users' likelihood of recommending the software to others on a scale of 0–10.

The following formula is used to calculate the average SUPR-Q score:

$$\text{SUPR-Q score} = \frac{Q1+Q2+Q3+Q4+Q5+Q6+Q7+ (1/2 \times Q8)}{8}$$

To calculate the SUPRQ score, the researchers averaged the scores for the seven categories. For the NPS or the loyalty category, the score for question eight is multiplied by $\frac{1}{2}$ to

convert it to the same scoring method used for the first seven questions.

The interpretation of SUPR-Q scores followed the scale range outlined by Brigula *et al.* [29] as shown in Table 1.

Table 1. The mean range and verbal interpretation of SUPR-Q scores

Weight/ Scale	Mean Range	Verbal Interpretation
5	4.51–5.0	Strongly Agree
4	3.51–4.50	Agree
3	2.51–3.50	Moderately Agree
2	1.51–2.50	Slightly Agree
1	1.00–1.50	Disagree

In addition to SUPR-Q, the students were asked how the online pseudocode interpreter helped them with their lessons. The students' qualitative responses added an essential layer of knowledge to the quantitative data.

Concurrently, the researchers tested the validity of the software as instructional material using the university's quality form for assessing instructional materials. The assessment form is the standard instrument used university-wide to validate instructional materials. The questionnaire was divided into two parts, the first focusing on the material's content and the second measuring the IM's organization, form, and style. The instrument allowed evaluators composed of ICT experts and faculty members to evaluate the system using a rating scale. The scale ranges from 1, indicating strong disagreement, to 4, indicating strong agreement for accepting it as instructional material. The interpretation scale for the university quality form was derived from Nee and Yunus [30] as shown in Table 2.

Table 2. Likert Four-Point Scale Range Interpretation

Point	Scale Range	Interpretation
4	4.00–3.00	Strongly Agree
3	2.99–2.00	Agree
2	1.99–1.00	Disagree
1	0.00–0.99	Strongly Disagree

C. Research Procedure

The researchers followed the ADDIE Model in conducting this research. The model consists of five phases, namely, the analysis, design, development, implementation, and evaluation phase.

Analysis. The researchers identified the study's objectives, scope, and constraints and determined the necessary functions of the software based on the course syllabus and various references. To do this, the researchers conducted interviews and observations with students and faculty members who are the end-users of the system and reviewed related literature and previous studies to identify the strengths and opportunities supporting the software solution. The researchers likewise interviewed several ICT experts who could provide further input into the system's design.

An instructional analysis was conducted, including the learning objectives to contribute, the existing pseudocode interpreter features, the features to be enhanced or added, and the need to validate the system. The overall process enabled the researchers to identify the development of a pseudocode interpreter as a web application and establish the system's functional, software, hardware, and network requirements.

Design. The logical model of the system was designed, including the structure and flow of the content. It involved designing the underlying architecture and logic that govern how the system processes and interprets pseudocode, as well as how it interacts with users. The Graphical User Interface (GUI) was appropriately designed, using wireframes and mock-ups and observing design principles to ensure user-friendly design and acceptance.

GUI refers to an environment that allows the user to communicate with the software or the computer [31]. A well-designed interface will make it easy for the user to understand the functionality and interact with the system, which, according to Miranda [31], will result in better user acceptance.

The researchers used the Unified Modelling Language (UML), Use Case, Activity, and Deployment diagrams to visually represent the system flow, efficiently conveying its functionalities to its target users. As a general-purpose modeling language, UML allows software system specification, design, visualization, and documentation [32]. The diagrams enabled the researchers to effectively communicate the functionalities and features of the system and gather feedback from stakeholders for design refinement.

Overall, the design phase ensured that the content was user-friendly and aligned with learning objectives and outcomes.

Development. The researchers developed the software by adopting the design as formulated. The software design as shown in Fig. 1 can accept pseudocodes using simple declarations, mathematical operations and control structures such as sequence, selection, and iteration. It can also accept recursive functions and arrays.

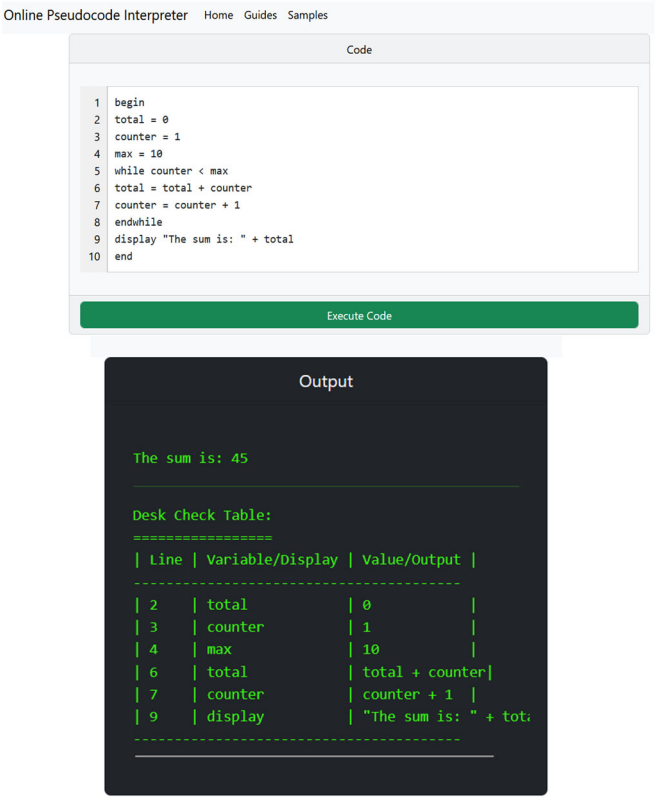

Fig. 1. The graphical user interface of the software.

Likewise, the SUPR-Q questionnaire was created using a Google Form to evaluate the design and extended with an open-ended question to obtain the opinion of the students

about the pseudocode interpreter.

Implementation. The software was deployed online, and students enrolled in the CS1 course were invited to use it whenever they needed to work on their pseudocode through their subject professors. The students were encouraged to use the software as it would be a subject for study and they could act as respondents. They were also informed that their participation in the study was voluntary and that their responses would be treated with utmost respect and confidentiality. Since it is accessed online, students can use the system at home through their personal computers, laptops, or mobile devices.

Evaluation. The researchers assessed the system based on the student's user experience using the SUPR-Q questionnaire. The instrument has convergent validity with other questionnaires that evaluate the same construct, high internal consistency reliability (Cronbach-α = 0.86), and content validity, covering constructs from expert judgment [27]. The students who used the system during the implementation phase were randomly selected to answer the SUPR-Q survey a week before the midterm examination to ensure that the users had used the software for the midterm period. The researchers monitored the number of submitted responses to reach 147, which represents the sample size, before terminating the data collection process. The result of the evaluation was tabulated and analyzed using the Statistical Program for Social Science (SPSS).

The researchers also assessed the system's validity and reliability as instructional material using the university's Evaluation for Instructional Materials Form. The instrument is the official assessment form for IM university-wide, and the evaluation process was performed solely by the Office of Instruction. An instructional material must pass the evaluation process before its utilization.

A request was made with the office of the Director for Instruction and the office of the Vice President for Academic Affairs to evaluate the software as an Instructional Material for CS1. The Office of the Instructions prepared the letters for the five evaluators. After their approval, the Evaluation for Instructional Materials form and the approved letter were manually given to evaluators who were faculty members and emailed to ICT experts working in private companies. After all the evaluation forms are returned to the researchers, they are forwarded to the Office of Instruction for tabulation. The result of the evaluation was signed and the online pseudocode interpreter was approved for utilization by the Vice President for Academic Affairs.

IV. Result and Discussion

The pseudocode interpreter is an application developed for Programming Logic Formulation students that serves as a learning medium to write, debug, and execute pseudocode in a specifically designed development environment instead of writing on a piece of paper [7]. The application provides a user guide for the supported pseudocode statements and syntax, as well as a set of samples to practice with pseudocode online. The application, accessed through the internet using either a desktop, laptop, or mobile device, provides users with the flexibility to learn pseudocode anytime, anywhere, and at their own pace [33–35].

The software had undergone evaluation by students to measure user experience and by ICT experts to measure the validity and reliability of the software as instructional material.

Table 3 presents the evaluation results of CS1 students using the SUPR-Q criteria where the system's overall SUPR-Q Score is 3.89, with SD = 1.10. The score is equivalent to Strongly Agree which indicates that the software is usable and credible, and the overall presentation of the software content is simple and easy to follow. The result implies a better user experience for the students which translates to user satisfaction [36–38] and the intent of users to continue usage of the system [36]

Regarding the software's usability evaluation, it has a mean of 4.03, with SD = 0.99. Usability is a combination of effectiveness, efficiency, and satisfaction with the software [27]. The evaluation implies that the respondents agree to the software's design and development features, which allow users to interact effectively and efficiently with the system, with a positive and satisfactory user experience. The result likewise signifies that the system focuses on effectiveness and efficiency with appropriate context, including the task, the user's experience, and the environment [39].

Table 3. Standardized User Experience Percentile Rank Questionnaire (SUPR-Q) result

SUPR-Q Criteria	Question	Mean	SD
Usability	The website is easy to use.	4.01	1.02
	It is easy to navigate within the website.	4.04	0.95
	Average	***4.03***	***0.99***
Credibility	The information on the website is credible.	3.97	0.94
	The information on the website is trustworthy.	3.54	0.99
	Average	***3.76***	***0.97***
Appearance	I find the website to be attractive.	3.94	1.04
	The website has a clean and simple presentation.	3.65	0.93
	Average	***3.80***	***0.99***
Loyalty	I will likely return to the website in the future.	4.22	0.94
	How likely are you to recommend the website to a friend or colleague?	7.56	2.02
	Average	***4.00***	***1.48***
	SUPR-Q Score	3.89	1.10

Credibility is the extent to which one perceives the website information to be believable [40] and is synonymous with trustworthiness. The respondents evaluated this aspect of the system with a mean of 3.76 and SD of 0.97. This positive evaluation implies that the respondents agree that the software is trustworthy and reliable to provide accurate information [27]. Positive credibility leads to increased trust and continued use of the software.

Regarding software appearance, the evaluation result has a mean of 3.80 and SD of 0.99. Appearance refers to the visual appeal of software or a website [27]. This outcome signifies that the respondents agree that the software or website is attractive and has a clean and straightforward presentation.

Loyalty is the likelihood of returning to the website and recommending the website or software to others [27]. The evaluation of software's loyalty has a mean value of 4.00 and

SD of 1.48. This result implies that the respondents agree that they will still use the software in the future and even recommend it to other users. The result signifies a satisfactory user experience and trust in the software or website to the point that they will recommend it to others.

The survey of student's opinions on the pseudocode interpreter yielded an overwhelmingly positive response, with 144 respondents expressing satisfaction, two providing irrelevant feedback, and one deferring. Students praised the system's user-friendliness, accessibility, and adaptability to mobile devices. The real-time feedback mechanism was consistently praised for its role in efficiently identifying and correcting errors, contributing to a more effective learning process. Another significant response was the appreciation for the system's interactive nature, which allowed students to visualize the execution of their pseudocode and served as a valuable self-assessment tool. While the majority of respondents were content, a subset provided constructive feedback, suggesting the addition of more features for enhanced functionality. This recommendation highlights the potential for continuous improvement to better meet the diverse needs of users. The positive feedback underscores the effectiveness of the pseudocode interpreter in supporting pseudocode learning, while the identified recommendations provide valuable directions for further development, emphasizing the need for ongoing efforts to refine and expand the system based on user feedback

The software was also evaluated as an Instructional Material by ICT experts using the university's quality form for instructional materials. Validating instructional materials will help ensure that they are effective, appealing, efficient, and suitable for students' learning needs. The result of the evaluation is shown in Table 4.

Table 4. Evaluation of the software as an instructional material

Criteria	Assessment	Mean	SD
Content	The content is based on the course syllabus with substantial coverage and concepts.	4.00	0.00
	It encourage higher order thinking skills, critical thinking and problem solving.	4.00	0.00
	The lesson lead to the attainment of the course outcome.	3.60	0.55
	Real life applications are given.	3.80	0.45
	Activities are developmental and with varied degree of complexity.	3.40	0.89
	The content is free of gender, cultural, religious and racial biases.	4.00	0.00
	Information is accurate and directions are clearly explained	3.40	0.55
	Average	***3.74***	***0.35***
Organization, Form and Style	There is logical presentation of topic	4.00	0.00
	Layout is consistent and arrange logically	3.80	0.45
	Font size and font style is appropriate and format and lay-out is visually appealing	3.40	0.89
	Average	***3.73***	***0.45***

The content of an instructional material refers to the specific information, knowledge, or educational substance presented or conveyed within that material. It encompasses the subject matter, concepts, facts, explanations, and any other educational content relevant to the learning objectives and goals of the instructional material. The content is what learners are expected to engage with, understand, and potentially master as part of their educational experience. The evaluators rated the content of the software with a mean of 3.74 and SD of 0.35. The result shows that the evaluators strongly agree that the software content provides accurate information, data, or results, and it functions as intended without errors or inaccuracies. The evaluation implies that the instructional material aligns with standards and learning objectives and can foster critical thinking and problem-solving skills [41, 42].

The evaluators strongly agree that the software was designed to include the topics in the course syllabus, and the result of their evaluation has a mean value of 4.00 and SD of 0.00. The result implies that the instructional material aligns with standards and learning objectives [41, 42] and can provide accurate and relevant information that supports students' understanding of the subject matter, which is essential for developing conceptual knowledge and skills [43]. The evaluators strongly agree that the software promotes higher-order thinking skills, critical thinking, and problem-solving. This evaluation has a mean value of 4.00 and an SD of 0.00. This result suggests that the approach, strategy, and learning materials promote critical thinking [44] and can significantly enhance the student's learning outcomes and achievements [45].

The evaluators also strongly agree that the software leads to the attainment of the course outcome with a mean value of 3.60 and SD of 0.55. This evaluation signifies that the software contains tangible or specific materials that are essential to the realization of the instructional objectives [46]. The result also shows that the evaluators strongly agree that the lessons include real-life applications. This feature was rated with a mean of 3.80 and SD of 0.45. This positive evaluation implies that the software promotes a higher level of engagement and motivation [47].

The evaluators also strongly agree that the activities using the software are developmental and with varied degrees of complexity. This evaluation was given a mean of 3.40 and SD of 0.89.

The evaluator also strongly agrees that the content is free of gender, cultural, religious, and racial biases. This feature was rated with a mean of 4.00 and an SD of 0.00. Moreover, the evaluators strongly agree that the software information is accurate and directions are clearly explained, with a mean value of 3.40 and SD of 0.55.

The selected evaluators also evaluated the software's Organization, Form, and Style. The "Organization, Form, and Style" of instructional material refers to how the educational content is structured and presented and how it is designed to facilitate learning. The overall mean score of the software's Organization, Form, and Style is 3.73, and SD is 0.45. This result means that the evaluators strongly agree that the system is easy to read and follows a consistent coding style, making it understandable to its users.

The evaluators strongly agree that the software logically presents the topic. This feature was rated with a mean score of 4.00 and an SD of 0.00. The evaluators also strongly agree that the layout is consistent and arranged logically. This feature was rated with a mean score of 3.80 and an SD of 0.45. It signifies that the software material can optimize student

learning through systematic organization where the topic sensibly follows the previous one, even allowing independent learning at home [47].

Furthermore, the evaluators strongly agree that the software's font size and font style are appropriate, and its format and layout are visually appealing. This feature was rated with a mean of 3.40 and an SD of 0.89. The result indicates that the chosen font style, format, and layout are visually appealing and conducive to readability, fostering a user-friendly environment. The result implies that the visual presentation of the software is not only aesthetically pleasing but also contributes to a coherent and comprehensible user interface.

Regarding the reliability or internal consistency of the measurement taken, the McDonald's ω scale was 0.628. This result indicates that the instrument was a good measure of construct validity. Finally, the evaluation form was signed and approved by the Vice President for Academic Affairs, signifying the qualification of the software for use as instructional material.

V. Conclusion

The pseudocode interpreter is a web application developed as a medium for CS1 students to learn pseudocode and for educators to use as instructional material. The result of the student's evaluation of the software using the SUPR-Q affirmed their satisfaction with the features as well as their tendency to use the software continually. The pseudocode interpreter is indispensable in supporting independent learners who want to learn pseudocoding, as they can access the application anytime and anywhere, using any of their IoT devices, whether desktop, laptop, or mobile devices. The features and content of the software will allow the students to practice with pseudocoding, the intent of which is to build a solid foundation in problem-solving and algorithmic thinking to hasten the transition to more advanced subjects.

Meanwhile, as affirmed by experts, the pseudocode interpreter has valid content, organization, form, and style and can be utilized by educators as instructional material for CS1 students. The immediate execution of pseudocode and the automatically generated desk check table will enable educators to demonstrate every aspect of pseudocode, from the variable declaration to iteration, processing flow, error tracing, and output generation. The software can aid ICT educators in teaching pseudocode to efficiently impart knowledge, enhance instructional strategies and techniques, and improve the overall teaching process.

The positive evaluation of the system and the opinions of the students led the researchers to recommend further study on the effectiveness of the software to improve the performance of CS1 students using it. A comparative analysis could be performed on the grades of previous and current students using the system to evaluate the system's effectiveness.

Likewise, the researchers recommend extending the system to support other CS1 concepts, like flowcharting. Such a system may allow users to write their pseudocode and view its equivalent flowchart or allow users to construct their flowchart and view its equivalent pseudocode as well as switch between flowchart and pseudocode mode. A collaborative feature may also be introduced, allowing students to work on group projects involving pseudocode or flowcharts, which will encourage teamwork and peer learning.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Dr. Tacuban assisted in designing and analyzing the software requirements, facilitated the software implementation, tabulated the results, and wrote the paper or the manuscript. Mr. Gabawa and Mr. Gimeno analyzed user requirements, designed the model, and developed and tested the software. Dr. Franco assisted in tabulating results, checking the manuscript, and implementing the software, and Dr. Amar helped with software implementation. All authors approve the final version of the manuscript.

References

[1] J. Yu and M. Bozic, "Investigating the role of pseudocode in learning programming language: A language transfer and typological similarity perspective," *Cambridge Open Engage*, 2023. doi: 10.33774/coe-2023-jdvw9

[2] M. M. Rodrigo, "Information and communication technology use in Philippine public and private schools," *Loyola Schools Review: School of Science and Engineering*, vol. 1, pp. 122–139, 2001.

[3] Asian Development Bank (ADB) and Department of Education of the Government of the Philippines (DepEd), "Youth education investment and labor market outcomes in the Philippines survey," July 2019. doi: 10.22617/TCS190267-2

[4] M. Ramat, S. Shahran, R. Latih, N. Yatim, N. Zainal, and R. Rahman, "Major problems in basic programming that influence student performance," *Procedia—Social and Behavioral Sciences*, vol. 59, pp. 287–296, 2012. doi: 10.1016/j.sbspro.2012.09.277.

[5] C. Hsiao, Y. H. Chuang, T. L. Chen, C. Y. Chang, and C. C. Chen, "Students' performances in computer programming of higher education for sustainable development: The effects of a peer-evaluation system," *Frontiers in Psychology*, vol. 13, 911417, 2022.

[6] L. Robertson, "Simple program design, a step-by-step approach fifth edition," *Thomson Course Technology*, 2007.

[7] C. L. Gimeno, "Pseudocode interpreter (pseudocode integrated development environment with lexical analyzer and syntax analyzer using recursive descent parsing algorithm)," *Asia Pacific Journal of Multidisciplinary Research*, vol. 5, no. 4, pp. 31–38, 2017.

[8] J. C. Giordano and M. Carlisle, "Toward a more effective visualization tool to teach novice programmers," in *Proc. the 7th Conference on Information Technology Education*, October 2006, pp. 115–122. doi: 10.1145/1168812.1168841

[9] E. Bachu and M. Bernard, "Visualizing problem solving in a strategy game for teaching programming," in *Proc. the International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS)*, 2014, p. 1.

[10] C. Rezende and E. L. Bispo, "Comparison between pseudocode usage and visual programming with scratch in programming teaching," in *Proc. 2018 XIII Latin American Conference on Learning Technologies (LACLO)*, Sao Paulo, Brazil, 2018, pp. 492–498. doi: 10.1109/LACLO.2018.00087

[11] A. L. Olsen, "Using pseudocode to teach problem solving," *Journal of Computing Sciences in Colleges*, vol. 21, no. 2, pp. 231–236, 2005.

[12] J. Yu and M. Bozic, "Investigating the role of pseudocode in learning programming language: A language transfer and typological similarity perspective," *Cambridge Open Engage*, 2023. doi: 10.33774/coe-2023-jdvw9

[13] N. U. Koyluoglu, K. Ertas, and A. Brotman, "Pseudocode to code translation using transformers," in *Proc. Natural Lang. Process. with Deep Learn.*, Stanford, CA, USA, Tech. Rep. CS224N, 2021.

[14] P. Mandal and P. Pedamkar. (2023). System flow diagram. [Online] Available: https://www.educba.com/system-flow-diagram/

[15] T. Dirgahayu, S. N. Huda, Z. Zukhri, and C. I. Ratnasari, "Automatic translation from pseudocode to source code: A conceptual-metamodel approach," in *Proc. 2017 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom)*, Phuket, Thailand, 2017, pp. 122–128. doi: 10.1109/CYBERNETICSCOM.2017.8311696.

[16] V. Parekh and D. Nilesh, "Pseudocode to source code translation," Intl. *J. Emerging Technologies and Innovative Research (JETIR)*, vol. 3, no. 11, pp. 45–52, 2016.
[17] J. Carette, O. Kiselyov, and C. Shan, "Finally tagless, partially evaluated: Tagless staged interpreters for simpler typed languages," *Journal of Functional Programming*, vol. 19, no. 5, pp. 509–543, 2009. doi: 10.1017/S0956796809007205
[18] T. Amarasingha, H. Pitipana, S. Lanka, and R. Ranaweera, "PseudoJ: A pseudo-code interpreter for transforming Pseudo-code into JAVA," in *Proc. 1st International Conference on Business Innovation*, 2018, p. 7.
[19] Y. An, "A history of instructional media, instructional design, and theories," *International Journal of Technology in Education (IJTE)*, vol. 4, no. 1, pp. 1–21, 2021. doi: 10.46328/ijte.35
[20] A. Paolini, "Enhancing teaching effectiveness and student learning outcomes," *Journal of Effective Teaching*, vol. 15, no. 1, pp. 20–33, 2015.
[21] I. N. Olokooba, "Effective utilization of instructional materials for social studies in upper basic schools in Kwara state," *Anatolian Journal of Education*, vol. 6, no. 1, pp. 167–174, 2021.
[22] H. Aydin, B. Ozfidan, and D. Carothers, "Meeting the challenges of curriculum and instruction in school settings in the United States," *Journal of Social Studies Education Research*, vol. 8, no. 3, pp. 76–92, 2017.
[23] I. Bouchrika. (2023). The ADDIE model explained: Evolution, steps, and applications. [Online]. Available: https://research.com/education/the-addie-model
[24] L. Tinoca, J. Piedade, S. Santos, A. Pedro, and S. Gomes, "Design-based research in the educational field: A systematic literature review," *Educ. Sci.*, vol. 12, no. 6, 410, 2022. doi: 10.3390/educsci12060410
[25] The Design-Based Research Collective, "Design-based research: An emerging paradigm for educational inquiry," *Educ. Res.*, vol. 32, pp. 5–8, 2003.
[26] M. İ. Berkman and Ş. Şahin, "Adapting SUPR-Q into Turkish for assessing user experience in web and mobile services," *Turkish Online Journal of Design Art and Communication*, vol. 11, no. 4, pp. 1328–1347, 2021.
[27] J. Sauro, "SUPR-Q: A comprehensive measure of the quality of the website user experience," *JUX—the Journal of User Experience*, vol. 10, no. 2, pp. 68–86, 2015.
[28] M. Hassenzahl and N. Tractinsky, "User experience—A research agenda," *Behaviour & Information Technology*, vol. 25, no. 2, pp. 91–97, 2006. doi: 10.1080/01449290500330331
[29] R. P. Bringula, M. Y. C. Batalla, S. D. Moraga, L. D. R. Ochengco, K. N. Ohagan, and R. R. Lansigan, "School choice of computing students: A comparative perspective from two universities," *Creative Education*, vol. 3, no. 6, 1070, 2012.
[30] C. C. Nee and M. M. Yunus, "RollRoll dice: An effective method to improve writing skills among year 3 pupils in constructing SVOA sentences," *Universal Journal of Educational Research*, vol. 8, no. 6, pp. 2368–2382, 2020.
[31] M. Miranda, "The importance of graphic users interface, analysis of graphical user interface design in the context of human-computer interaction," in *Proc. Edulearn11*, 2011, pp. 7137–7144.
[32] M. Seidl, M. Scholz, C. Huemer, and G. Kappel, *UML @ Classroom an Introduction to Object-Oriented Modeling*, Springer International Publishing AG, 2012, ch. 1, pp. 11–21.
[33] X. Du, J. Yang, B. Shelton, and J.-L. Hung, "Is learning anytime, anywhere a good strategy for success? Identifying successful spatial-temporal patterns of on-the-job and full-time students," *Information Discovery and Delivery*, vol. 47, no. 4, pp. 173–181, 2019. doi: 10.1108/idd-09-2019-0060
[34] M. S. Andrade and B. Rivers, "Developing a framework for sustainable growth of flexible learning opportunities," *Higher Education Pedagogies*, vol. 4, no. 1, pp. 1–16, 2019. doi: 10.1080/23752696.2018.1564879
[35] H. Abuhassna, W. M. Al-Rahmi, N. Yahya, M. A. Zakaria, A. B. Kosnin, and M. Darwish, "Development of a new model on utilizing online learning platforms to improve students' academic achievements and satisfaction," *International Journal of Educational Technology in Higher Education*, vol. 17, no. 1, 2020. doi: 10.1186/s41239-020-00216-z
[36] L. Deng, D. Turner, R. Gehling *et al.*, "User experience, satisfaction, and continual usage intention of IT," *Eur. J. Inf. Syst.*, vol. 19, pp. 60–75, 2010. doi: 10.1057/ejis.2009.50
[37] Y. A. Shidqi, E. R. Mahendrawathi, and B. W. Otok, "The effect of user's experience, characteristics, and satisfaction toward the adoption of ERP," *SISFO*, vol. 9, no. 3, pp. 1–10, 2020.
[38] M. Bano, D. Zowghi, and F. Rimini, "User satisfaction and system success: An empirical exploration of user involvement in software development," *Empirical Software Engineering*, vol. 22, pp. 2339–2372, 2017.
[39] J. Brooke, "SUS: A retrospective," *Journal of Usability Studies*, vol. 8, no. 2, pp. 29–40, 2013.
[40] B. J. Fogg, T. Kameda, J. Boyd *et al.* (2002). Stanford-Makovsky web credibility study 2002: Investigating what makes web sites credible today. [Online]. pp. 512–519. Available: http://captology.stanford.edu/pdf/Stanford-MakovskyWebCredStudy2002-prelim.pdf
[41] M. J. F. Tan-Espinar and R. S. Ballado, "Content validity and acceptability of a developed worktext in Basic Mathematics 2," *Asia Pacific Journal of Multidisciplinary Research*, vol. 5, no. 1, pp. 72–84, 2017.
[42] M. C. R. Selga, "Instructional materials development: A worktext in science, technology and society," *LCCB Development Education Journal of Multidisciplinary Research*, vol. 2, no. 1, 2013.
[43] M. A. Romarate, A. B. Aquino, E. J. Punongbayan, G. R. Quizon, L. A. Balilla, and N. P. Ramos, "Development of outcomes-based instructional materials in professional teacher education courses for a flexible set-up," *Journal of Education and e-Learning Research*, vol. 10, no. 1, pp. 61–67, 2023.
[44] P. Andreucci-Annunziata, A. Riedemann, S. Cortés, A. Mellado, M. T. del Río, and A. Vega-Muñoz, "Conceptualizations and instructional strategies on critical thinking in higher education: A systematic review of systematic reviews," *Frontiers in Education*, March 2023, 1141686.
[45] D. Liu and H. Zhang, "Improving students' higher order thinking skills and achievement using Wechat based flipped classroom in higher education," *Educ. Inf. Technol.*, vol. 27, pp. 7281–7302, 2022. doi: 10.1007/s10639-022-10922-y
[46] H. Ojating and J. H. Ojating, "Incorporating tangible instructional materials in teaching and learning: Implications for educational assessment and evaluation," *International Journal of Quantitative and Qualitative Research Methods*, vol. 10, no. 1, pp. 1–6, 2022.
[47] J. Mazgon and D. Stefanc, "Importance of the various characteristics of educational materials: different opinions, different perspectives," *Turkish Online Journal of Educational Technology-TOJET*, vol. 11, no. 3, pp. 174–188, 2012.