Teaching Programming to Students with Vision Impairment: Impact of Tactile Teaching Strategies on Student’s Achievements and Perceptions

Hind Alotaibi 1,2,* , Hend S. Al-Khalifa 3 and Duaa AlSaeed 3

1 College of Languages & Translation COLT, King Saud University, PO Box 145111, Riyadh 4545, Saudi Arabia
2 College of Humanities, Prince Sultan University, PO Box 66833, Riyadh 11685, Saudi Arabia
3 Information Technology Department, CCIS, King Saud University, PO Box 145111, Riyadh 4545, Saudi Arabia; hendk@ksu.edu.sa (H.S.A.-K.); dalsaeed@ksu.edu.sa (D.A.)

* Correspondence: Hialotaibi@ksu.edu.sa; Tel.: +966-55-320-8110

Received: 31 May 2020; Accepted: 26 June 2020; Published: 1 July 2020

Abstract: The United Nations (UN) 2030 agenda involved 17 Sustainable Development Goals (SDGs) to achieve a better and more sustainable world for all. The fourth Sustainable Development Goal called for “ensuring inclusive and equitable quality education and promoting lifelong learning opportunities for all”. Despite international efforts to achieve such a goal, many students with vision impairment (VI) who wish to pursue a degree in computer science face significant challenges and must overcome social and technical obstacles. One challenge is learning how to program as a key skill for pursuing a degree in the field of computer science. This paper explores practical issues in teaching students with VI the basics of programming and presents recommended practices based on a suggested workshop setup. The workshop ran for three weeks, for a total of 60 teaching hours, and involved designing and implementing complete curricula and multi-modal activities to simplify the acquisition of basic programming concepts. Workshop data was collected using several data collection methods—i.e., interviews, observation, questionnaires, performance records, and daily journals. The results indicated an improvement in participants’ programming skills, which was detected through their performance records and final project evaluations. The participants also showed a high interest in learning programming and positive attitudes towards the experience. However, the participants’ experience also involved some challenges such as understanding abstract concepts, code navigation, and some technical issues. The study is hoped to contribute to the literature on education inclusion and to bridge the digital divide in our society.

Keywords: vision impairment; programming; special needs; inclusion; accessibility; sustainability

1. Introduction

In 2015, United Nations Member States agreed on the 2030 Agenda for Sustainable Development [1]. The agenda included 17 global goals designed to be a “blueprint to achieve a better and more sustainable future for all”. Goal 4 in this agenda involves ensuring inclusive and equitable quality education and promoting lifelong learning opportunities for all. This goal explicitly states in one of its targets the importance of “eliminating gender disparities in education and ensure equal access to all levels of education and vocational training for the vulnerable, including persons with disabilities, indigenous peoples and children in vulnerable situations”. However, the recent report from the UN on the implementation of this agenda revealed that “persons with disabilities continue to face multiple disadvantages, denying them both life opportunities and fundamental human rights”. The report shows that refocused efforts are “needed to improve learning outcomes for the full life cycle, especially for women, girls and marginalized people in vulnerable settings” [2].
Along this line, the inclusion of individuals with VI in the digital age has been explored in several studies over the past decade. These studies tackled various issues on VI and technology from social, pedagogical, and technological perspectives. Some of these studies have looked at information seeking [3], accessibility [4,5], bias and social exclusion [6,7], cyber-racism, cyberbullying, the digital divide [8], navigation [9,10] misinformation, and usability [11], among other aspects of information and technology experiences. However, more work is still needed to address the issue of inclusion in the context of emerging technologies and social innovation.

Many students with VI who wish to pursue a degree in the field of computer science face significant challenges and must overcome social and technical obstacles. According to Corn and Lusk [12] (p. 13), “visual impairment or visual disability is a term that encompasses both those who are blind and those with low vision.” There are nearly 285 million people worldwide who have some degree of visual impairment; of these, 8.2% live in the Eastern Mediterranean region, according to the World Health Organization (WHO) [13].

While many studies have explored the inclusion of individuals with special needs, such a research area is often overlooked in the Middle East region [14,15]. According to WHO reports [13], this can be linked to various challenges associated with disability in the region, such as the lack of effective financing, the lack of clear and standardized definitions of disability and inclusion across countries, the limitation of rigorous and comparable data on disability, the insufficient involvement of people with disabilities in decision-making, and the lack of coordination among concerned entities.

Since the advent of the computer industry, people with VI have shown a keen interest in programming [16]. In fact, there is an international movement toward teaching programming to all, promoted through an education week that incorporates the activity, Hour of Code [17]. During this week, many sectors of the community including people with VI take part in programming [18]. However, many consider learning programming to be a challenging task in general, and for students with VI in particular, simply because of the way it is traditionally taught. Many programming courses rely heavily on abstraction and visualization such as the use of diagrams, flowcharts, tables, and images. Although such teaching strategies can help explain complex programming concepts, they cannot be used successfully among students with VI [19]. This study aims to demonstrate a framework for teaching the basics of programming to students with VI. To evaluate the effectiveness of such a framework, the following questions will be answered:

Q1: What strategies can be adopted to introduce the basics of programming to adults with VI?
Q2: How do the proposed strategies impact the participants’ programming achievements?
Q3: How do the proposed strategies impact the participants’ perceptions of learning programming?

To address these questions, we conducted a three-week-long workshop targeting adults with VI. The workshop focused on teaching the basics of programming using the Quorum Environment [20].

The rest of this paper is organized as follows: The following section presents the literature review and explores previous attempts in teaching programming to students with VI. Section 3 presents our study materials and methods, while Section 4 describes the results. Section 5 involves a discussion of the study findings and the study limitations. Finally, we present the conclusions and our plans for future work.

2. Background and Related Work

VI refers to any condition of vision loss that has an impact on everyday activities [21] and that cannot be corrected with correction aids such as eyeglasses or contact lenses. It is classified according to the International Classification of Diseases into low vision and blindness [22]. Low vision means a partial loss of vision, despite the correction, while blindness is the total loss of sight or the possession of only a few degrees of vision, despite the correction. According to Project IDEAL (Informing and Designing Education for All Learners), visual impairment is classified based on functional vision into low vision, functional blindness, and total blindness. Individuals with low vision use their vision
as their primary sense to read and write, with the help of magnifying devices. Functionally blind people can use limited vision for functional tasks but rely on tactile and auditory senses for learning. Totally blind people use touch and auditory senses for learning and functional tasks. The result of the process of formulating visual perceptions after receiving sensory information from the environment is known as a mental model [23]. Producing a mental model is referred to as “visualizing”, “seeing with the mind’s eye”, “hearing in the mind”, “imagining how something feels”, and so on [18]. To create a mental model, the individual imagines the object through simulations, copies, or reconstructions of past perceptual experiences or the anticipation of future ones. Visual mental models pertain foremost to the fields of philosophy, psychology, and cognitive science. The term is sometimes expanded to include other senses such as sound and smell [24].

Earlier cognitive studies concluded that people with VI have different types of mental models than sighted people [25,26]. People with VI predominantly imagine objects close to them; they are less likely to imagine objects farther away or larger objects. By contrast, sighted people tend to imagine large objects being at a greater distance from themselves [27]. However, another study concluded that the mental models of individuals with VI share distinct similarities with those of individuals who have full use of their sight; when receiving the same information as sighted people, participants with VI in the study demonstrated mental models similar to those of sighted participants [23].

For sighted people, sight is very important for learning and exploring the environment. Students without visual impairments learn through visual cues, while the other senses such as touch and hearing are often ineffective or insufficient for them. However, students with VI train themselves to depend on their other senses such as hearing, touch, taste, and smell to develop or adjust their mental models. Thus, students with VI are able to learn as well as their sighted counterparts, but they need assistive tools and different methods to learn things that depend on visual concepts. Individuals with VI must be taught skills to be able to acquire knowledge without sight, among which are orientation and mobility using assistive technology, independent living skills and residual vision, and the reading and writing of Braille [21].

An increasing number of individuals who are totally blind or who have low vision use computers and mobile devices in their daily lives to improve their professional qualifications and integrate more into society. Another study by Ashraf et al. [28] provided a systematic literature review of the application of Information Communication Technology (ICT) with people with VI. They highlight three pressing topics for people with VI: (a) assistive technology, (b) e-accessibility, and (c) virtual interfaces. They then emphasize ICT accessibility issues for people with VI, which include software environments such as websites and programming tools.

Teaching programming to students with VI has been the focus of many studies with various aims. Some of these studies looked at different programming environments such as text-based languages (TBL), Audio Programming Languages (APL), block-based languages (BBLs), and how to enhance the accessibility of these environments [29–33]. Several studies discussed the challenges VI programmers encounter and recommendations on how to overcome such challenges [34,35]. Another common theme in the literature involves evaluating tools and plug-ins to assist VI programmers [36,37].

One early study by Sánchez and Aguayo [38] looked at teaching Audio Programming Language (APL) to learners with VI to assist in developing their problem-solving and algorithmic thinking skills. In this study, students managed to interact with and program APL through audio-based commands, which helped them understand programming concepts and how to apply such concepts to code their own ideas. The study reported that students were satisfied with the experience of interacting with APL and were motivated to continue to use it. The researcher concluded that traditional instructional strategies such as theoretical explanations and traditional programming tasks fail to enhance their understanding of programming concepts. The researcher also argues that learners with VI depend heavily on concrete experience before building abstract thinking. This indicates that their mental model can be adjusted through the sensation of touch and hearing. In one of the studies by Kane and Bigham [31], a four-day computer science education workshop was conducted in which students with
VI learned how to program using Ruby. They reported that Ruby and its interactive interpreter offer a sufficient, if not perfect, environment for teaching students with VI to program. Although most participants completed the programming tasks, keeping track of their current program scope seemed challenging, and they occasionally entered code in invalid locations.

In Albusays and Ludi [35], the difficulties programmers with VI usually encounter were identified by surveying 69 programmers. They reported some difficulties with code navigation and inaccessible integrated development environments (IDEs) as well as with screen readers and working in teams. The results also showed that programmers with VI prefer using text editors to write code. However, further investigation is needed to obtain a better understanding of such challenges and how to tackle them.

To further investigate the issue of code navigation among programmers with VI, Albusays and Ludi [34] observed 28 developers with VI using their preferred coding tool while navigating complex codebases. The study reported several navigation challenges. Participants were not satisfied with the accessibility level of most IDEs and tended to compensate by using multiple input methods in order to navigate easily. However, such approaches usually affected their speed and accuracy, thus restraining their efficiency. On the other hand, Potluri et al. [36] proposed a solution to accessibility barriers to help developers with VI who use a graphical user interface (GUI) to improve their programming experience with a plug-in called CodeTalk.

In a study by Milne and Ladner et al. [29], the researchers examined nine programming environments—five web-based and four mobile device-based—and evaluated them using screen readers. They identified five accessibility barriers: accessing the output and programming elements, moving the blocks, and conveying programming structure and type information. To address these problems, they developed Blocks4All, a block-based programming environment that enables programmers with VI to code using tablet devices.

Considering all of the above, we can see that the focus has been on solving niche problems in the paradigm of programming such as developing plug-ins or IDEs to help individuals with VI program or teaching the use of specific programming languages. To the best of our knowledge, there are no studies that report the complete experience of setting up and implementing a full workshop for teaching the basics of programming to individuals with VI, which is the aim of our paper. In the next section, we describe the study materials and methods including participants, setup, course content and activities, and data collection methods.

3. Materials and Methods

The study involved three stages:

- Prior to the workshop: this stage covered all the processes involved in the preparation of this study, including analyzing needs, defining objectives, assigning team members, preparing course content, screening participants, and setting up the computer lab.
- During the workshop: a major part of the process was related to the actual implementation of this study, including the assessment of participants, classroom observation, out-of-class observation, and daily reporting.
- After the workshop: This stage mainly involved evaluating and assessing the results and outcomes of the implemented workshop and collecting and analyzing the data. This stage can be divided into two processes: (1) end-of-workshop evaluation and (2) analysis.

Figure 1 shows a detailed illustration of these processes with the related tools and data.
Our research team held a three-week programming workshop for participants with VI at Kafeef office center, an association of people with VI, located in Riyadh, Saudi Arabia [39]. The workshop was four hours a day, and the participants were engaged in learning how to write programs using the Quorum programming language. To understand how participants with VI experience a typical classroom setting and interactions, we held some preliminary discussions with computer science teachers who teach basic computer skills to students with VI (mainly Microsoft Office and Windows). We conducted the workshop after school hours (4:00 p.m.–8:00 p.m.) at Kafeef’s computer lab facility. The rationale for choosing to conduct the study at Kafeef and after school hours involves four reasons: First, Kafeef’s computer lab is equipped with all the special programs and devices required by students with VI—i.e., screen readers, headsets, and Braille Sense devices (see Section 3.2). Second, Kafeef provided participants with transportation, an important factor encouraging their enrolment. Third, the participants were familiar with Kafeef, since it had hosted several events and workshops in the past and they were accustomed to moving around easily and comfortably. Finally, most participants were only able to attend the workshop in the evening, due to their work or study commitments.

3.1. Participants

A registration form was distributed electronically through social media channels to recruit participants interested in learning programming. Due to cultural restrictions, only female participants could be accepted in the workshop. Seven female participants with VI with an average age of 27 years enrolled in the workshop. The registration form also involved a section asking the participants to self-evaluate their IT skills and English language ability. Based on their self-evaluations, the participants were selected according to the following criteria.

- Some experience in using computers, or, at least, in using Microsoft Office and Windows to ensure the satisfaction of the minimum accepted level of computer skills.
- Some experience in using an English keyboard, since Quorum is an English-based programming language.
- Ability to read and write using Braille, as the course material is provided in digital form and printed as hard copy in Braille.

Table 1 summarizes the demographic information and selection criteria of all the participants. After selecting the participants, they were informed that the workshop data would be used for
research purposes only and were assured that the privacy and personal identity information of all the participants would be protected.

Table 1. Demographic information and selection criteria of all the participants based on their self-evaluation.

| Participant | Info | Age | Blindness Type | Education | English Level | Braille Reading/Writing | Own a Braille Sense | Computer Skills | Programming Background |
|-------------|------|-----|----------------|-----------|---------------|------------------------|-------------------|-----------------|-----------------------|
| P1          | 20   | Totally blind | College | Excellent | Excellent | Yes | Excellent | Basic |
| P2          | 29   | Partially blind | BA | Good | None | No | Excellent | None |
| P3          | 19   | Partially blind | Diploma | Very good | Excellent | Yes | Excellent | Basic |
| P4          | 28   | Totally blind | College | Very good | Excellent | No | Excellent | None |
| P5          | 25   | Totally blind | College | Excellent | Excellent | Yes | Excellent | None |
| P6          | 36   | Partially blind | BA | Poor | Excellent | Yes | Good | None |
| P7          | 32   | Partially blind | BA | Good | Excellent | Yes | Excellent | None |

BA: Bachelor graduate.

3.2. Setup

Prior to the workshop, preliminary discussions were held with a computer science instructor who had been teaching basic computer skills to students with VI for more than 15 years. The instructor holds a bachelor’s degree in computer science and has worked as a computer instructor at Special Needs Center at King Saud University (KSU) for more than 10 years; she also provides training on basic computer skills outside KSU for VI students. She met with the research team several times prior to the workshop to discuss the course delivery requirements and help to prepare the lab. The following computer lab resources and learning materials were provided:

- A total of seven computers to create a 1:1 participant-to-computer ratio. All the computers were provided by Kafeef in their computer lab facility.
- Each computer was equipped with a headset to allow the participants to use screen readers without disturbing their colleagues.
- All the computers had internet access to help the participants access their emails and learning resources.
- Seven Braille Sense devices were provided; Braille Sense is a handy note taker that helps participants read documents during class. In our case, all the Braille Sense devices were owned by the participants. Extra devices were provided by Kafeef when needed.
- Each participant was provided with a digital copy of the course material, in addition to a Braille-printed hard copy.
- Quorum 6.0.9 was installed on participants’ PCs with a Sodbeans development environment, a programming language specially developed for programmers with VI. Quorum was selected since it is free and has built-in accessibility features—i.e., self-voicing support, magnification, and various sound settings (e.g., voices, beeping on errors, voice debugging).

In this workshop, the researchers assumed the role of mentor if needed, but mainly played the role of observer, taking notes and recording classroom interactions and dynamics. Figure 2 is a photo of one of the computers provided for each student. During the workshop, the instructor adopted inquiry-based and hands-on teaching approaches. Such instructional approaches focus on the role of the student in the learning process. The students were encouraged to be active participants, asking
questions and sharing ideas and solutions. The role of the co-instructor was to help in resolving technical issues, guide participants when needed, and help the instructor during the course.

![Figure 2. Examples of in-class multi-modality activities; (a), (b) and (c) pictures of a problem-solving and algorithms activity (Prepare a dish); (d) a picture of a condition statement activity (Condition cups); (e) a picture of a problem-solving and algorithms activity (Giving directions); (f) a picture of a problem-solving and algorithms activity (Sorting numbers).]

3.3. Course Content

The first session of the first day of week 1 was orientation, and the participants were asked to fill out the pre-questionnaire. For the remainder of the week, the instructor introduced the participants to the basics of programming, including problem-solving, algorithms, and the use of the Quorum IDE. In week 2, the first two days were dedicated to the completion of part 1 of the concept of types and variables, while the rest of the week covered the control statements—IF—and conditions. In week 3, the instructor covered part 2 of control structures, including looping and the different forms of the Repeat statement, while the rest of the week covered intensive practical exercises and the final project. A detailed description of the workshop’s content is shown in Table 2.

| Week 1 | Workshop Weeks | Week 2 | Week 3 |
|--------|----------------|--------|--------|
| 1. Introduction to programming: | 1. Types and variables II | 1. Control structures II |
| 1.1 What is programming? | 1.1 Mathematical operators | 1.1 Introduction to repeat statement types |
| 1.2 Problem solving and algorithms | 1.2 Logical operators | 1.2 Repeat times |
| 1.3 Getting started with IDE | 1.3 Output and user input | 1.3 Repeat while |
| 1.4 Quiz | 1.4 Quiz | 1.4 Repeat until |
| 2. Types and variables I | 2. Control structures I | 2. Programming exercises |
| 2.1 Introduction to types and variables | 2.1 Conditional IF statement | 2. Final project |
| 2.2 Typecasting | 3.1 Conditionals | |
| | 3.2 Conditionals | |

3.4. Classroom Activities

During the three weeks sessions, a variety of instructional strategies were used to deliver the content. The inquiry-based and hands-on teaching approaches employed aimed at encouraging the students to be active participants, asking questions and sharing their ideas and suggestions. To simplify the programming constructs, the workshop provided several multi-modal activities specifically designed to accommodate the mental model of students with VI to offer learners actual, concrete,
and hands-on experience to simplify and facilitate their learning process. As discussed previously, traditional teaching approaches, such as theoretical explanations and traditional programming tasks, are inadequate to promote VI learners’ understanding of programming concepts, as they depend greatly on concrete experience before building abstract thinking. This indicates that their mental model can be adjusted through various senses—i.e., touch, hearing, smell, and even taste—which were implemented in the classroom activities (see Figure 2).

Table 3 lists examples of some of the activities used during the workshop. One of the examples involves explaining logical operators using the sense of taste. The instructor provided each student with two small cups—one with sugar, representing true, and the other with salt, representing false. Students had to taste each cup and mix the flavors to identify and “sense” the differences when using the Boolean operators, And, Or, and Not.

| Topic/Concept Learned | Sample Activity | Short Description | Teaching Strategy |
|-----------------------|----------------|-------------------|-------------------|
| What is programming?  | Video + group discussion | Students discussed their views on programming | Brainstorming, activating schema |
| Problem-solving and algorithms | Preparing a dish | Students were provided with ingredients and asked to prepare a dish. Figure 3a–c | Problem-based learning, learning through play, collaborative learning |
| Giving directions | One student gave directions while the others traced on a map. The winner was the one who arrived first. Figure 3e. | Problem-based learning, learning through play |
| Sorting numbers | Students were required to sort numbers on a magnetic board. Figure 3f. | Problem-based learning, learning through play |
| Calculate the ideal weight | Students were asked to write an algorithm to calculate the ideal weight. | Problem-based learning |
| Getting started with IDE | Printed tactile IDE screenshot | A simplified tactile version of the IDE main interface was designed and printed. Participants used it to learn the layout, items, and sections of the IDE editor screen. Figure 4 | Conceptual mind maps |
| Introduction to types and variables | Tracing Braille programming code to discover errors in variables’ names | Students were given coded programs printed in Braille with which to trace and detect errors. | Problem-based learning |
| Using different variables | Students were required individually and in pairs to write simple code using different variables and types. | Problem-based learning |
| Typecasting | Tracing Braille programming code | Students were given coded programs printed in Braille with which to trace and detect errors. | Problem-based learning |
| Mathematical operators | Mathematical exercises to practice order of operations | Students are required individually and in pairs to code some mathematical exercises to practice order of operations. | Problem-based learning, collaborative learning |
| Logical operators | “Sugar and salt” | The instructor provided each student with two cups, one with sugar, representing true, and the other with salt, representing false. Students had to taste each box and mix the flavors to recognize and “sense” the differences when using the Boolean operators, And, Or, and Not. | Learning through play, collaborative learning |
Table 3. Cont.

| Topic/Concept Learned                  | Sample Activity                  | Short Description                                                                 | Teaching Strategy                      |
|----------------------------------------|----------------------------------|-----------------------------------------------------------------------------------|----------------------------------------|
| Output and user input                  | Input-Processing-Output chart    | Students used a printed tactile aid to learn the concepts.                         | Conceptual mind maps                   |
| Control structures (IF statement)      | Acting out a scene               | Two students acted out a scene, while the others decided where the input, processing, and output was. | Role-playing, collaborative learning    |
|                                        | Role-play game                   | Students gathered, facing the instructor, as she instructed them to do something if a condition applied—e.g., if you have long hair, clap. If you are wearing black, knock on the table, etc. | Learning through play                  |
| Condition cups                         | Students were provided with two cups, representing True and False and several cards with Braille numbers. In pairs, they had to classify each card according to a certain condition being either true or false. Figure 3d. | Learning through play, collaborative learning                                    |
| Control structures (loops)             | Coding                           | Write a program to calculate a student’s grade, based on her score.                | Problem-solving                        |
|                                        | Role-play game                   | All students line up except one. The first student gives her a card. The rest of the students were tied with ribbon. The goal was to learn by checking how many cards she has: If less than 10, she gets through the ribbon; if it is more than 10, she will stay out of the line and will go directly to being the last girl standing outside of the ribbon. | Role-playing, learning through imagination |
| Project fair                           | Students are required to come up with an idea for a certain program and must include all the programming concepts presented during the workshop—i.e., variables, conditions loops, etc. After implementing their projects, students presented their work to an evaluation committee at the project fair. The students’ projects were evaluated based on an evaluation rubric. | Problem-based learning, collaborative learning, learning through brainstorming. |

The instructor used practical programming sessions to check the participants’ understanding of the learned concepts and determine how they employed them in their coding. Following these discussions, it was easier for the instructor to identify and address the difficulties faced by the participants. During the programming exercises, the instructor starts by explaining and/or reviewing a certain construct—e.g., mathematical operators or the control structure, and then she presents a programming problem asking students to suggest possible solutions. The participants’ proposed solutions are discussed individually or in groups. Finally, the students begin to code (individually or in pairs), run their codes, and check for errors, while the instructor walks around to provide assistance when needed.

In another activity, a simplified tactile version of the Integrated Development Environment (IDE) main interface was designed and printed. The participants used it to learn the layout, items, and sections of the IDE editor screen (see Figure 3).
After the course content had been covered, the participants spent the final week of the workshop planning, developing, and testing their projects. The project topics were proposed by the participants themselves based on their own interests. The instructor encouraged team members to collaborate and discuss their projects’ coding and findings. More details on the final project are presented next.

3.5. Beyond the Workshop: Project Fair

The workshop encouraged problem-based collaborative learning, whereby participants worked in pairs and proposed a programming project, wrote an algorithm for the solution, and implemented and tested their programming code. The students were given the chance to freely choose their partners. P1 worked with P2, P4 worked with P5, and P6 worked with P7. Since we had 7 students, one participant (P3) preferred to work individually instead of working in a group of 3.

On the final day, the workshop concluded with a project fair, where each group presented and discussed their project ideas and demonstrated their final work. The projects were evaluated using the evaluation rubric shown in Appendix B. The researchers asked each team questions after the presentation to better understand the rationale behind each project and to evaluate the teamwork and attained programming skills. The best project award was announced at the end of the fair.

3.6. Data Collection Methods

To answer the research questions, several data collection methods were used. A pre-questionnaire was administered at the beginning to collect data on the participants’ backgrounds and interest in learning programming and to evaluate their understanding of some programming concepts such as variables, mathematical operations, Conditional Statements, and Loops. In addition, the pre-questionnaire involved three questions to test the participants’ logical thinking abilities. A translated copy of the questionnaire and model answer to the logical questions are shown in Appendix A. A post-questionnaire was administered at the end of the workshop to detect any changes in the participants’ interest in learning programming and their logical thinking after attending the workshop. The instructor was asked to record her reflections and observations on the participants’ progress, main challenges, and actions taken to resolve any difficulties or obstacles faced in or out of class. Classroom observation notes were taken during each session. One researcher attended each session as an observer. The observer focused on instructor-participant and participant-participant interaction and classroom dynamics, mainly covering three broad domains of instructor-participant interaction that support participants’ learning and development: emotional support, classroom organization, and instructional support. In addition, the participants were encouraged to share their views, challenges, and achievements through a WhatsApp group created especially for the workshop, or through their Twitter accounts, by posting under the workshop’s hashtag. Their posts were recorded daily. Finally,
an end-of-workshop questionnaire was administered to assess the quality of the workshop and examine the participants’ views on their overall experience.

4. Data Analysis and Results

The pre- and post-questionnaires which involved data on the participants’ interest in learning programming and their understanding of some programming concepts were analyzed as presented in Table 4. The table shows the changes in the students’ understanding of basic programming concepts.

Table 4. Differences in the participants’ understanding of programming concepts in the pre- and post-questionnaires.

| Variable | Pre- | Post- | Pre- | Post- | Pre- | Post- | Pre- | Post- | Pre- | Post- |
|----------|------|-------|------|-------|------|-------|------|-------|------|-------|
| Q1       | 3    | 7     | 2    | 0     | 2    | 0     | 4    | 0     | 4    | 0     |
| Q2       | 1    | 7     | 2    | 7     | 1    | 6     | 2    | 6     |      |       |
| Q3       | 4    | 0     | 4    | 0     | 4    | 0     | 4    | 0     | 1    | 0     |
| Q4       | 1    | 6     | 1    | 1     | 1    | 1     | 1    | 1     |      |       |
| Q5       | 2    | 6     | 2    | 6     | 2    | 6     | 2    | 6     |      |       |
| Q6       | 42%  | 100%  | 42%  | 100%  | 14%  | 100%  | 28.5%| 100%  | 14%  | 100%  |
| % of positive responses | 42% | 100% | 42% | 100% | 14% | 100% | 28.5%| 100% | 14% | 100% |

TSE: To some extent.

The students’ responded positively when asked about their understanding of the programming concepts in the post-questionnaire compared to their initial responses in the pre-questionnaire (see Figure 4).

Figure 4. Change in the participants’ understanding of the programming concepts in the pre- and post-questionnaires. The maximum value on the vertical axis of Figure 4 is adjusted to 100%.

In addition, the pre- and post-questionnaires evaluated the participants’ perceptions of learning programming before and after attending the workshop. The data were analyzed and presented in Table 5.
Table 5. Change in participants’ perceptions of learning programming in the pre- and post-questionnaires.

| Learning programming is important for my self-development. | Pre-Questionnaires | Post-Questionnaires |
|----------------------------------------------------------|--------------------|---------------------|
| I would like to learn more about programming.             | Yes (100%)        | Yes (100%)          |
| Learning programming will enhance my confidence.          | 100%               | 100%                |
| Learning programming will be fun.                         | 86%                | 14%                 |
| Learning programming will promote my career.              | 100%               | 0%                  |
| I have the required capabilities to learn to program.     | 100%               | 0%                  |
| Learning programming will be difficult.                   | 0%                 | 100%                |

TSE: To some extent.

The pre- and post-questionnaires involved two logical thinking questions (see Appendix A). The participants’ scores on the logical thinking questions are presented in Table 6. The table shows the means and standard deviation (SD).

Table 6. Participants’ scores on logical questions.

| Participant | Q1-A Pre | Q1-A Post | Q1-B Pre | Q1-B Post | Q2 Pre | Q2 Post |
|-------------|----------|-----------|----------|-----------|--------|---------|
| P1          | 0.5      | 0.5       | 0        | 0         | 1      |         |
| P2          | 0        | 0         | 0        | 0         | 0.25   |         |
| P3          | 1        | 1         | 1        | 1         | 0.5    | 0.5     |
| P4          | 0.5      | 0.57      | 1        | 1         | 0.5    | 0.25    |
| P5          | 0        | 0.57      | 1        | 1         | 0      | 1       |
| P6          | 0        | 0         | 0        | 0         | 0.25   | 0.5     |
| P7          | 0        | 0.25      | 0        | 1         | 0      | 0.25    |

Mean 0.29 0.41 0.43 0.57 0.18 0.54
SD 0.39 0.36 0.53 0.53 0.24 0.34

The analysis of students’ mean scores and standard deviation shown in Figure 5 indicates that students performed better in the post-questionnaire, which can be attributed to the fact that students in the post-questionnaire gained more understanding of the programming concepts necessary to solve the logical questions.

![Figure 5](image_url)

**Figure 5.** (a) Students’ mean scores on the logical questions in the pre- and post-questionnaires. (b) Standard deviation of students’ scores on the logical questions in the pre- and post-questionnaires.

As stated earlier, evaluating the participants’ performance was also done through recording and analyzing their coding and identifying their errors, as shown in Table 7.
Table 7. Participants’ coding error types.

| Common Coding Error Types                          | P1  | P2  | P3  | P4  | P5  | P6  | P7  |
|---------------------------------------------------|-----|-----|-----|-----|-----|-----|-----|
| 1. Typing a text variable without “”              | −   | •   | −   | −   | −   | −   | −   |
| 2. Defining variables and never using them        | −   | −   | •   | −   | −   | •   | −   |
| 3. Typing an integer variable with “”             | −   | •   | −   | −   | −   | •   | −   |
| 4. Typing the wrong brackets—e.g., [ ] instead of ( ) | −   | −   | −   | •   | •   | −   | −   |
| 5. Opening quotations without closing them—e.g.,   | “How old are you?” | −   | •   | •   | −   | •   | −   |
| 6. Opening brackets without closing them          | −   | −   | •   | •   | −   | −   | •   |
| 7. Forgetting to print the output                 | −   | •   | −   | −   | •   | •   | •   |
| 8. Not knowing which variable to output           | −   | −   | −   | •   | •   | •   | •   |
| 9. Forgetting to type the input command and using text | −   | −   | •   | •   | −   | −   | −   |
| 10. Typing the variable between quotation marks—e.g., text x = “test” output “x” | •   | •   | −   | −   | −   | •   | •   |
| 11. Typing all commands on one line               | −   | −   | •   | −   | −   | •   | −   |
| 12. Spelling errors when typing variables         | −   | •   | •   | •   | −   | •   | −   |
| 13. Typing symbols incorrectly—e.g., >= instead of <= | •   | •   | •   | •   | •   | •   | •   |
| 14. Not knowing where to end the condition        | −   | −   | •   | •   | •   | •   | •   |
| 15. Not knowing where to end the loop             | −   | −   | •   | •   | •   | •   | •   |
| 16. Using the same variable name more than once   | −   | −   | −   | •   | •   | •   | •   |

The number of error types (out of 16) 2 6 5 8 4 11 7
Percentage of error types 12.5% 37.5% 31.25% 50% 25% 68.75% 43.75%

• means that the participant had a coding error(s) of this type.

In addition, the end-of-workshop questionnaire data were analyzed to evaluate the participants’ overall satisfaction with the workshop components—i.e., the workshop’s objectives, coursework, environment, delivery, evaluation, and outcomes, as shown in Table 8.

Table 8. End of workshop questionnaire.

| Questions                                                                 | Strongly Agree | Agree | Agree to Some Extent | Disagree | Strongly Disagree |
|---------------------------------------------------------------------------|----------------|-------|----------------------|----------|-------------------|
| Satisfaction:                                                             |                |       |                      |          |                   |
| I’m happy and satisfied with this workshop because I acquired knowledge that contributes to my overall development. | 86%            | 14%   | 0                    | 0        | 0                 |
| I’m happy and satisfied with this workshop because I acquired skills that contribute to my overall development. | 100%           | 0     | 0                    | 0        | 0                 |
| The workshop contributed to my overall development.                      | 100%           | 0     | 0                    | 0        | 0                 |
| About the workshop:                                                      |                |       |                      |          |                   |
| I will encourage my friends to join the workshop in the future.          | 86.00%         | 14%   | 0                    | 0        | 0                 |
| I would like to join an advanced programming workshop if provided.       | 86.00%         | 0     | 14%                  | 0        | 0                 |
| Workshop Outcomes:                                                       |                |       |                      |          |                   |
| At the end of the workshop, I found that the objectives and planned outcomes were achieved. | 57%            | 43%   | 0                    | 0        | 0                 |
| After this workshop, I believe that I have gained the basic knowledge I wanted from it. | 100%           | 0     | 0                    | 0        | 0                 |
| I can apply the knowledge gained from joining the workshop.              | 86%            | 14%   | 0                    | 0        | 0                 |
| I can formulate solutions to programming problems.                        | 43%            | 43%   | 14%                  | 0        | 0                 |
Table 8. Cont.

| Questions                                                                 | Strongly Agree | Agree | Agree to Some Extent | Disagree | Strongly Disagree |
|---------------------------------------------------------------------------|----------------|-------|----------------------|----------|-------------------|
| Learning programming contributed to the development of my analytical skills. | 86%            | 14%   | 0                    | 0        | 0                 |
| The workshop contributed to the development of my logical thinking skills.  | 71%            | 29%   | 0                    | 0        | 0                 |
| The workshop contributed to the development of my communication skills.     | 57%            | 29%   | 14%                  | 0        | 0                 |
| The workshop contributed to the development of my teamwork skills.         | 57%            | 29%   | 0                    | 14%      | 0                 |
| The workshop contributed to strengthening my confidence in my abilities.    | 71%            | 29%   | 0                    | 0        | 0                 |
| I feel that learning and practicing programming is possible for the blind.  | 100%           | 0     | 0                    | 0        | 0                 |

The above data were compared and contrasted with the data collected from participants’ responses to the interviews and classroom observations. A thematic analysis was used to identify repeated ideas and patterns of meaning, which then were categorized into major themes and sub-themes related to the students’ performance, perceptions, and challenges. Several challenges were detected through classroom observation notes and students’ interviews. These challenges were categorized into several subthemes: understanding of abstract concepts, code navigation, technical issues, and educational background. A detailed discussion of these findings is presented in the following section.

5. Discussion

From the lens of these research questions, several themes were identified from the data analysis: teaching strategies, participants’ achievements, participants’ perceptions, and challenges.

5.1. Teaching Strategies

The workshop incorporated a variety of instructional strategies to introduce the basics of programming to students with VI and simplify programming constructs. Several multimodal activities specifically designed to accommodate the mental models of students with VI were provided to offer learners concrete, hands-on experience to simplify and facilitate their learning process (Table 3). As previously stated, conventional teaching and learning strategies, such as theoretical lecturing and traditional programming tasks, are insufficient for enhancing VI learners’ understanding of programming concepts. Learners with VI depend heavily on tactual experiences before they can build abstract thinking skills. Moreover, their mental models can be modified through various senses such as touch, smell, hearing, and even taste. Such teaching strategies proved to be effective, in our study, since they had a positive impact on students’ achievements and perceptions, as discussed next.

5.2. Participants’ Achievements

The data analysis revealed some interesting aspects related to student achievements. First, the pre- and post-questionnaires showed an improvement in their understanding of programming concepts (see Table 4 and Figure 4). The pre- and post-questionnaires aimed at evaluating the participants’ knowledge of basic programming concepts, such as their understanding of variables, the ability to write condition statements, or looping. The chart in Figure 4 shows that almost all the participants had a limited understanding of such concepts before the workshop. However, all the participants responded positively when asked about their knowledge of these concepts after attending the workshop. The participants were also asked explicitly whether they could write a program to solve a specific problem and execute it. In the pre-questionnaire, four of the seven participants reported that they did not have these experiences before the workshop, while in the post-questionnaire, all the participants...
reported that they had the ability to write a program and execute it, indicating an improvement in their skills as novice programmers.

Analyzing the students' scores on the logical thinking questions showed an overall improvement in their performance, as shown in Figure 5. Looking closer at the students’ scores shown in Table 5 reveals that P7 scored better on all three questions. The scores of the other participants varied, as some of them scored higher in some questions or retained their original score (P1, P2, P4, P5, P6), while no change was detected in P3’s scores. On the other hand, two students scored lower in one question—i.e., P5 in Q1-B and P4 in Q2—which can be attributed to the short duration of the workshop (3 weeks) and/or to the participants’ educational background, which will be discussed later under the challenges section.

To evaluate the validity of these findings, the data from the pre-and post-questionnaires were compared against the data gathered from classroom observations and participants’ performance reports. For each task, the participants’ performance reports were used to record their coding errors and their task completion rates. The data showed that five out of seven participants managed to successfully complete all the required programming tasks, either autonomously or with minor help from their instructor. P1, P3, and P5 were among the top participants in almost all sessions; they managed to complete all the tasks independently in a relatively short time, with minimal coding errors (see Table 7). P2 and P4 took a relatively longer time to complete the tasks, asking for assistance from the instructor and the assistant occasionally. However, the performance of P6 and P7 was relatively lower than their peers. Although they managed to complete all the tasks, they seemed to take a longer time to finish, compared to the rest of the class, and referred more often to their instructor or the assistant to ask for help to solve a coding error or to explain the task further. Looking at the participants’ background information in Table 1, variations in their abilities can be linked to their ages and educational background. The participants who performed very well (P1, P3, and P5) were the youngest (age 20, 19, and 25 years) and were still pursuing their studies, which may have had a positive impact on their performance. The participants’ English language abilities seemed to influence their performance to some extent. While basic English language skills were not a prerequisite for acceptance in the workshop, it was preferred that the participants had some English language knowledge to be able to understand and type the Quorum commands. During the workshop, a few participants struggled to cope with typing and memorizing commands in English. In the pre-questionnaire, the participants were asked to rate their English language ability. While the top three participants rated their English as Excellent and Very good, P6 and P7 rated their ability as Poor and Good, respectively.

A careful look at the analysis of coding errors (Table 7) confirms the above-mentioned findings. P1 made the fewest mistakes in the group, and her mistakes involved opening parentheses and not closing them or typing a variable between quotation marks. P6 had the highest number of coding mistakes, which mainly involved syntax errors. P4 has the second most coding mistakes, which also linked to navigation and syntax. It was interesting to see how the analysis revealed that programming errors in this group of novice programmers with VI did not differ much from errors typically made by sighted programmers and revolved mainly around incompatible types and syntax errors [40]. Some researchers suggest that providing participants with a list of common mistakes and detailed examples of how to avoid them is an effective strategy for raising their awareness of such errors and helping to minimize them.

The participants’ performance was also evaluated based on their final projects. The students managed to develop impressive projects reflecting their unique personalities and interests, despite the time limit. The projects’ evaluation showed that all the students were able to find an idea appropriate to what they had learned in the workshop. They were able to implement and test their programs successfully and managed to provide suggestions for future improvement.

In summary, the workshop had an overall positive impact on the participants’ programming skills that could be detected through their performance records, pre-/post-questionnaires, and final projects. Such improvement was also reported by the participants themselves through their positive responses
to the end-of-workshop questionnaire, during interviews, and in classroom observation, as discussed in the following section.

5.3. Participants’ Perceptions

From the very beginning, the participants were highly interested in learning programming. Their responses to the pre-questionnaire showed they all agreed on the importance of learning programming and that it could promote their self-confidence and careers (Table 5). Their positive views were also expressed in the post-questionnaire, as all the participants were eager to learn more about programming. During the three-week workshop, the participants sustained their interest despite the challenges and frustrations. Their interest in learning programming was evident in their attendance rates and participation level. Over the three weeks, only two absences were recorded, which indicates an 87% attendance rate. All the required tasks (in-class and homework) were completed and submitted. During the sessions, all the students were observed to be interested and engaged in the lessons and activities. They appeared comfortable seeking support from their instructor; her assistant; and, on many occasions, their peers.

The use of a variety of modalities, including auditory, sensory, and mobile modalities, effectively expanded students’ involvement and promoted their participation and interaction during class time. They were observed on many occasions to be sharing and discussing their ideas with the instructor and with their peers.

It was interesting to note how their interaction extended far beyond the classroom. Before the workshop, the researchers created a WhatsApp group aimed at facilitating out-of-class communication and information sharing among the participants. The students actively exchanged their posts either through text or audio. Their posts involved discussing their views and ideas, asking for support from their peers and the instructor, or simply expressing their feelings. They were also encouraged to share their opinions through their Twitter accounts. The students’ posts on both platforms truly reflected their positive perceptions regarding the workshop and how it contributed to the development of their programming skills and boosted their confidence.

The end-of-workshop questionnaire showed high satisfaction rates in general regarding the workshop components—i.e., its objectives, coursework, environment, delivery, evaluation, and outcomes (See Table 8). All the participants strongly agreed that the workshop contributed to their overall development. A total of 86% of them strongly agreed that they were happy and satisfied with the workshop and that they could apply the knowledge they had gained. They also reported an interest in joining an advanced programming workshop in the future.

The end-of-workshop questionnaire also included an open-ended question to allow the participants to comment on the most important aspects of the workshop. They reported:

“The delivery of information in a concrete and tactile way and linking it to real life; I loved the activities!”

“Learning algorithms and how computers think in a logical and organized way”

“Learning through different strategies”

“The enriching activities provided to us, the kind spirits, the group of beautiful trainees, the novelty of the topic, and my anxiety about it—and then overcoming all of that; that’s an advantage! Even my concerns and clumsiness ended in victory; I see that as an advantage!”

“The advantages are enormous, but the most important ones are: I was amazed by the new information, the way we learned, the great variety of activities, and by how much they accommodated our needs. Through this workshop, you managed to reactivate my brain mathematically and logically—I owe you for that!”

The students’ positive perceptions were consistent with other studies, such as Kane and Bigham [31], Seraj et al. [33], and Sánchez and Aguayo [38], which reported that VI students show high levels of motivation and enjoyment in learning programming. However, despite the participants’ positive attitudes toward this experience, there were some challenges involved as discussed next.
5.4. Challenges

Even with the positive impact of this workshop on the participants’ performance and their perceptions, a data analysis revealed that there were some challenges involved. These challenges were categorized into the following subthemes: the understanding of abstract concepts, code navigation, technical issues, and educational background.

5.4.1. Understanding of Abstract Concepts

One of the main challenges faced by the participants was their struggle to understand abstract concepts owing to the lack of visual input. In some instances, they seemed overwhelmed when they were first introduced to programming concepts such as types and variables, control structures, conditions, and loops. They expressed frustration explicitly during class time or when discussing homework: “Oh, it’s difficult!”; “It’s impossible to understand!” However, multi-modal activities were specifically designed for this workshop to take into account learners’ mental models and provide them with tactual, concrete, and hands-on experience to simplify and facilitate the learning of such constructs. Thus, there seemed to be a gradual decline in their struggle as they worked on such tasks.

It was interesting to observe how simple hands-on or role-play activities reduced students’ anxiety and provided them with a better understanding of what had seemed to be an abstract and complex concept while creating a fun and engaging atmosphere. Such an instructional approach allowed the students to quickly enhance their understanding of many of the abstract concepts related to programming and to focus on coding and testing their programs.

Another effective pedagogical approach that seemed to minimize this problem was collaborative work, sometimes referred to as pair programming [41,42]. Although some tasks involved students working individually, the participants were also encouraged to work in pairs, particularly after introducing a challenging concept. They appeared to be more enthusiastic when working together and were observed discussing and explaining to one another; the pattern of interaction involved one student holding the Braille activity sheet and reading the instructions aloud, while the other worked on the PC with the headset on. Their interaction pattern also involved the discussion and negotiation of possible solutions. Asking for the instructor’s help appeared to lessen when the participants worked in pairs, as they seemed to rely on each other to overcome any challenges. Working in pairs appeared to increase the task completion rate and promote their success as a group. The participants seemed to produce higher-quality programs with a minimum of coding errors, as reported by their instructor.

5.4.2. Code Navigation

One of the main challenges faced by the students was navigation. Due to the lack of structure conveyed by screen readers, as they read code sequentially, students faced some coding difficulty, which was confirmed through observation and a coding error analysis (Table 7). It seems that the participants occasionally had difficulty keeping track of their current position and entered code in invalid locations. This is a common challenge among programmers with VI reported in the literature [16,19,31,43]. Some students expressed frustration when faced with such a problem and were observed deleting the entire code on many occasions and starting over again as a coping strategy, which was slowing them down. Some researchers [31] recommend encouraging students to add comments to facilitate navigation, since they can indicate the start and end of a coding block. Some researchers [16,44] have pointed out that providing code samples in Braille would enhance learners’ understanding of the overall structure of the code.

5.4.3. Technical Issues

Although the programming environment selected for this workshop was fully accessible and easy to navigate using a screen reader, some participants encountered few challenges. When installed on older versions of Windows, some screen readers might crash or fail to pronounce Arabic script.
This clearly caused some frustration and task delays detected through classroom observation and reported by the students. As a coping strategy, the students facing this problem avoided using Arabic script in their programming code. To address the problem, all the devices should have the appropriate operating system to minimize any incompatibility issues with screen readers.

Another technical issue could be linked to language incompatibility, which was reported in other studies [41,42]. Many coding exercises in this workshop involved Arabic script—e.g., prompting a user to input value or displaying output messages. When switching between languages while coding, students could lose track of which language was set, opening parentheses in English and closing them in Arabic, for example. They could also mistake text direction or complete an English command in Arabic characters, which inevitably caused run-time errors. When they encountered such problems, students usually asked for the instructor’s help to switch the language back to English. This issue can be easily addressed by allowing screen readers to prompt the user when switching between languages while typing.

5.4.4. Educational Background

As previously discussed, a few students progressed relatively slower than the rest of the group and, in the error analysis, most of their errors could be linked to their low English language ability (Table 1). Many researchers have argued that, since programming languages are English-based, students’ lack of language skills may hinder their learning [42–44]. According to Noshin and Ahmed [45] (2018, p. 192), almost all popular programming languages are English language-based. This is deemed an obstacle for non-native English speakers. On the one hand, there is difficulty understanding different programming languages’ syntax; on the other, there is the added pressure of understanding English vocabulary.

Another difficulty faced by many participants was their high school preparation. During interviews, the participants’ reported that their general education took place at Al-Noor Institute in Riyadh, a special school for educating people with poor vision, blindness, or visual impairments, supported by the Ministry of Education, from the elementary level through middle school and ending in high school. The elementary and middle school curricula focus on teaching humanities courses such as social sciences, Arabic, and basic math and science [46]. However, the high school curriculum is limited to humanities courses, with no exposure to science and math. This could explain why the students struggled with basic mathematical concepts such as how to calculate an average, convert a unit, or calculate an area. There is a strong need to integrate VI students into regular schools in order to achieve inclusive and equitable opportunities for all. According to UNESCO [46], inclusive education means “putting the right to education into action by reaching out to all learners, respecting their diverse needs, abilities, characteristics and eliminating all forms of discrimination in the learning environment”. Consequently, this will promote the achievement of the Sustainable Development Goals (SDGs).

After discussing the study results, it is crucial to acknowledge some limitations that should be addressed in future work. Due to the limited sample size, only a qualitative analysis was conducted. A mixed methods research design with a larger sample size could be adopted to enhance the results of this study. Furthermore, only female participants were involved in this study owing to cultural restrictions. The gender factor may have an impact on the study results, and it is recommended to include male participants in future work. Another limitation is the odd number of participants, which sometimes hindered working in pairs, especially during the final project. Finally, the workshop’s short duration (only three weeks) may have influenced the students’ training, since they could not practice at home and were not able to extend their practice in class.

6. Conclusions

This paper reports the planning and conducting of a three-week workshop to teach the basics of programming to adults with VI. The aim was to "ensure inclusive and equitable quality education and promoting lifelong learning opportunities for all”, signifying the fourth Sustainable Development Goal
(SDG4). The study explored strategies that can be adapted to introduce the basics of programming to adults with VI and how these strategies impact participants’ programming achievements and perceptions. Data were collected using several methods: questionnaires, interviews, classroom observations, performance records, and daily reports. The workshop incorporated multimodal teaching strategies to simplify the understanding of main programming concepts taught in an introductory programming course. The strategies involved problem-based, role-play, pair work, and project-based activities. Such teaching strategies proved to have a positive impact on the participants’ achievements and attitudes. The improvement in the participants’ programming skills was detected based on their performance records and final project evaluations. Such improvement was also reported by the participants themselves in their positive responses to the post-questionnaire, during interviews, and in classroom observation. As for their attitudes, the participants demonstrated high levels of interest in learning programming and positive attitudes towards the experience. However, their experience involved some challenges, such as understanding abstract concepts, code navigation, and some technical issues.

Future work involves conducting a large-scale study and including both genders to examine how our findings apply to a different group. Our future work also involves evaluating and comparing other programming environments and their accessibility and suitability to VI. Furthermore, we plan to conduct intermediate and advanced programming workshops to encourage novice programmers with (VI) and to help bridge the digital divide in our society, making sure that “no one is left behind”.

Author Contributions: Conceptualization, H.S.A.-K. and H.A.; methodology, H.S.A.-K., H.A., and D.A.; project administration, H.S.A.-K. and D.A.; resources, H.S.A.-K., H.A. and D.A.; visualization, H.S.A.-K. and D.A.; writing—original draft, H.S.A.-K., H.A. and D.A.; writing—review and editing, H.S.A.-K., H.A. and D.A.; funding acquisition, H.A. All authors have read and agreed to the published version of the manuscript.

Funding: The authors wish to extend their appreciation to the Research Center for Humanity-Deanship of Scientific Research at King Saud University for funding this Research [Group No. HRGP-1-19-03].

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A Study Questionnaire

Introduction to programming workshop
* Required

1. Name: ____________________________________________

2. Age: ____________________________________________

3. Why did you join this workshop? *

   ____________________________________________
   ____________________________________________
   ____________________________________________

Do you agree with the following statements?

4. I realise the importance of programming. *
   Mark only one oval.
   [ ] Yes
   [ ] No
   [ ] To some extent
5. I understand variables in programming. *
   Mark only one oval.
   ☐ Yes
   ☐ No
   ☐ To some extent

6. I understand mathematical operations in programming. *
   Mark only one oval.
   ☐ Yes
   ☐ No
   ☐ To some extent

7. I understand logical operators in programming. *
   Mark only one oval.
   ☐ Yes
   ☐ No
   ☐ To some extent

8. I can use conditional statements in my programming code. *
   Mark only one oval.
   ☐ Yes
   ☐ No
   ☐ To some extent

9. I can use loops in my programming code *
   Mark only one oval.
   ☐ Yes
   ☐ No
   ☐ To some extent

10. I can write and implement a program that solves a specific problem *
    Mark only one oval.
    ☐ Yes
    ☐ No
    ☐ To some extent

Do you agree with the following statements?

11. Learning programming is important for my self-development. *
    Mark only one oval.
    ☐ Yes
    ☐ No
    ☐ To some extent

12. I would like to learn more about programming *
    Mark only one oval.
    ☐ Yes
    ☐ No
    ☐ To some extent
13. Learning programming will enhance my confidence. *
Mark only one oval.
☐ Yes
☐ No
☐ To some extent

14. Learning programming will be fun. *
Mark only one oval.
☐ Yes
☐ No
☐ To some extent

15. Learning programming will promote my career. *
Mark only one oval.
☐ Yes
☐ No
☐ To some extent

16. I have the required capabilities to learn programming. *
Mark only one oval.
☐ Yes
☐ No
☐ To some extent

17. Learning programming will be difficult. *
Mark only one oval.
☐ Yes
☐ No
☐ To some extent

---

**Logical thinking question:**
Your colleague asked you to calculate her average grade for her 4 courses. Below are the suggested steps to implement the required task. Arrange them according to the sequence you find correct (the steps with no particular order):
1. Repeat the previous two steps
2. Ask for a course grade
3. Survey the total grade
4. Add the total grade to the total score
5. Add the score to the total score

**Logical thinking question:**
You have 20 envelopes; the envelopes are numbered from 1 to 20. Each envelope contains 100 Saudi Riyals, except for one empty envelope. You do not know which one. You have a small piggy bank and you’re required to empty the envelopes one by one according to the envelope number and put the amount that you find inside each envelope in the piggy bank until you reach the empty envelope. You then have to calculate the amount that you collected in the piggy bank.

18. Write the steps in the correct logical sequence. Remember to press the enter key after each number. *

19. Continuing with the same question, if we assume that the empty envelope is the twelfth envelope, how much will be the amount you have collected? *
Appendix B  Project Evaluation Rubric

| Elements                                      | Excellent | Very Good | Good | Satisfactory |
|-----------------------------------------------|-----------|-----------|------|-------------|
| Program Specifications                        | 4         | 3         | 2    | 1           |
| The program works correctly on all inputs     |           |           |      |             |
| The program does not contain code errors      |           |           |      |             |
| The code is generally clear and understandable|           |           |      |             |
| Readability (easy to read)                    |           |           |      |             |
| The code is well organized                    |           |           |      |             |
| Meaningful variable names                     |           |           |      |             |
| The code is well documented with comments     |           |           |      |             |
| Code efficiency                               |           |           |      |             |
| Adequate scope                                |           |           |      |             |
| Well written algorithm                        |           |           |      |             |
| Follows logical thinking based on the context of the problem | | | | |
| The code is not complicated                   |           |           |      |             |
| The code is free of repeated code             |           |           |      |             |
| Loops used when needed                        |           |           |      |             |
| Conditional sentences used when needed        |           |           |      |             |
| Presentation: excellence in project idea presentation; commitment to presentation and demo time | | | | |

References

1. United Nations. Sustainable Development: Knowledge Platform: SDGs & Topics. 2015. Available online: https://sustainabledevelopment.un.org/ (accessed on 20 May 2020).
2. Convention on the Rights of Persons with Disabilities (CRPD). Available online: https://www.un.org/development/desa/disabilities/convention-on-the-rights-of-persons-with-disabilities.html (accessed on 20 May 2020).
3. Xie, J.; Babu, R.; Castillo, M.D.; Lee, T.H.; Youi, S. Developing Digital Library Design Guidelines to Support Blind Users. In Proceedings of the ACM SIGACCESS Conference on Computers & Accessibility, Galway, Ireland, 22–24 October 2018; pp. 401–403.
4. Giraud, S.; Thérouanne, P.; Steiner, D.D. Web Accessibility: Filtering Redundant and Irrelevant Information Improves Website Usability for Blind Users. *Int. J. Hum. Comput. Stud.* 2018; 111, 23–35.
5. Reynal, M.; Imbert, J.-P.; Arico, P.; Toupillier, J.; Borghini, G.; Hurter, C. Audio Focus: Interactive Spatial Sound Coupled with Haptics to Improve Sound Source Location in Poor Visibility. *Int. J. Hum. Comput. Stud.* 2019, 129, 116–128. [CrossRef]
6. Du, J.T.; Haines, J. Working with Indigenous Communities: Reflections on Ethical Information Research with Ngarrindjeri People in South Australia. *Proc. Asoc. Inf. Sci. Technol.* 2018, 55, 794–796. [CrossRef]
7. Meza-de-Luna, M.E.; Terven, J.R.; Raducanu, B.; Salas, J.A. Social-Aware Assistant to Support Individuals with Visual Impairments during Social Interaction: A Systematic Requirements Analysis. *Int. J. Hum. Comput. Stud.* 2019, 122, 50–60. [CrossRef]
8. Du, J.T. Research on Indigenous People and the Role of Information and Communications Technology in Development: A Review of the Literature. *J. Aust. Libr. Inf. Assoc.* 2017, 66, 344–363. [CrossRef]
9. Guerrón, J.; Sato, D.; Ahmetovic, D.; Ohn-Bar, E.; Kitani, K.M.; Asakawa, C. Virtual Navigation for Blind People: Transferring Route Knowledge to the Real-World. *Int. J. Hum. Comput. Stud.* 2020, 135, 102369. [CrossRef]
10. Guerrón, N.E.; Cobo, A.; Olmedo, J.J.S.; Martin, C. Sensitive Interfaces for Blind People in Virtual Visits inside Unknown Spaces. *Int. J. Hum. Comput. Stud.* 2020, 133, 13–25.
11. Tekli, J.; Issa, Y.B.; Chbeir, R. Evaluating Touch-Screen Vibration Modality for Blind Users to Access Simple Shapes and Graphics. *Int. J. Hum. Comput. Stud.* 2018, 110, 115–133. [CrossRef]
12. Corn, A.L.; Erin, J.N. *Foundations of Low Vision: Clinical and Functional Perspectives*; American Foundation for the Blind: New York, NY, USA, 2010.
13. World Health Organization (WHO). Available online: https://www.who.int/disabilities/world_report/2011/report/en/ (accessed on 10 September 2010).

14. Al-Ratta, N.M.; Al-Khalifa, H.S. Teaching Programming for Blinds: A Review. In Proceedings of the Fourth International Conference on Information and Communication Technology and Accessibility (ICTA), Hammamet, Tunisia, 24–26 October 2013; pp. 1–5.

15. Achcar, G. On the “Arab Inequality Puzzle”: The Case of Egypt. In Development and Change; International Institute of Social Studies: Den Haag, The Netherlands, 2020.

16. Konecki, M.; Ivković, N.; Kaniški, M. Making Programming Education More Accessible for Visually Impaired; IEEE: Opatija, Croatia, 2016; pp. 887–890.

17. Hour of Code. Available online: https://hourofcode.com/us/supporting-special-needs-students/ (accessed on 11 October 2018).

18. Wilson, C. Hour of Code—A Record Year for Computer Science. ACM Inroads 2015, 6, 22. [CrossRef]

19. Bigham, J.P.; Aller, M.B.; Brudvik, J.T.; Leung, J.O.; Yazzolino, L.A.; Ladner, R.E. Inspiring Blind High School Students to Pursue Computer Science with Instant Messaging Chatbots. In Proceedings of the 39th SIGCSE technical symposium on Computer science education (SIGCSE ’08); Association for Computing Machinery: New York, NY, USA, 2008; pp. 449–453.

20. Stefkis, A.; Ladner, R. The Quorum Programming Language. In Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education (SIGCSE ’17); Association for Computing Machinery: New York, NY, USA, 2017; p. 641.

21. Project IDEAL (Informing and Designing Education for All Learners). Available online: http://www.projectidealonline.org/v/visual-impairments (accessed on 11 October 2018).

22. Mariotti, A.; Pascolini, D. Global Estimates of Visual Impairment. Br. J. Ophthalmol. 2012, 96, 614–618.

23. Kurze, M.T. Draw: A Computer-Based Tactile Drawing Tool for Blind People. In Proceedings of the second annual ACM conference on Assistive technologies (Assets ’96); Association for Computing Machinery: New York, NY, USA, 1996; pp. 131–138.

24. Mental Imagery. Available online: https://plato.stanford.edu/cgi-bin/encyclopedia/archinfo.cgi?entry=mental-imagery (accessed on 13 April 2019).

25. Millar, S. Theory, Experiment, and Practical Application in Research on Visual Impairment. Eur. J. Psychol. Educ. 1997, 12, 415–430. [CrossRef]

26. Johnson-Laird, P.N. Mental Models: Towards a Cognitive Science of Language, Inference, and Consciousness; Harvard University Press: Cambridge, MA, USA, 1983.

27. Arditi, A.; Holtzman, J.D.; Kosslyn, S.M. Mental Imagery and Sensory Experience in Congenital Blindness. Neuropsychologia 1988, 26, 1–12. [CrossRef]

28. Ashraf, M.M.; Hasan, N.; Lewis, L.; Hasan, M.R.; Ray, P. A Systematic Literature Review of the Application of Information Communication Technology for Visually Impaired People. Int. J. Disabil. Manag. 2017, 11, e6. [CrossRef]

29. Milne, L.R.; Ladner, R.E. Blocks4All: Overcoming Accessibility Barriers to Blocks Programming for Children with Visual Impairments. In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI ’18); Association for Computing Machinery: New York, NY, USA, 2018; pp. 1–10.

30. Ludi, S.; Ellis, L.; Jordan, S. An Accessible Robotics Programming Environment for Visually Impaired Users. In Proceedings of the 16th international ACM SIGACCESS conference on Computers & accessibility (ASSETS ’14); Association for Computing Machinery: New York, NY, USA, 2014; pp. 237–238.

31. Kane, S.K.; Bigham, J.P. Tracking@ Stemxcomet: Teaching Programming to Blind Students via 3D Printing, Crisis Management, and Twitter. In Proceedings of the 45th ACM technical symposium on Computer science education (SIGCSE ’14); Association for Computing Machinery: New York, NY, USA, 2014; pp. 247–252.

32. Wang, Z.; Wagner, A. Evaluating a Tactile Approach to Programming Scratch. In Proceedings of the 2019 ACM Southeast Conference, Kennesaw, GA, USA, 18–20 April 2019; pp. 226–229.

33. Seraj, M.; Katterfeldt, E.; Bub, K.; Autexier, S.; Drechsler, R. Scratch and Google Blockly: How Girls’ Programming Skills and Attitudes are Influenced. In Proceedings of the 19th Koli Calling International Conference on Computing Education Research; Association for Computing Machinery: New York, NY, USA, 2019; pp. 1–10.

34. Albusays, K.; Ludi, S.; Huenerfauth, M. Interviews and Observation of Blind Software Developers at Work to Understand Code Navigation Challenges. In Proceedings of the ACM SIGACCESS Conference on Computers & Accessibility, Baltimore, MD, USA, 29 October–1 November 2017; p. 91.
35. Albusays, K.; Ludi, S. Eliciting Programming Challenges Faced by Developers with Visual Impairments: Exploratory Study. In Proceedings 9th International Workshop on Cooperative and Human Aspects of Software Engineering, (CHASE), Austin, Texas, USA, 14–22 May 2016; Association for Computing Machinery, Inc.: New York, NY, USA, 2016; pp. 82–85.

36. Potluri, V.; Vaithilingam, P.; Iyengar, S.; Vidya, Y.; Swaminathan, M.; Srinivasa, G. CodeTalk: Improving Programming Environment Accessibility for Visually Impaired Developers. In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI ’18); Association for Computing Machinery: New York, NY, USA, 2018; pp. 1–11.

37. Schanzer, E.; Bahram, S.; Krishnamurthi, S. Accessible AST-Based Programming for Visually-Impaired Programmers. In Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE ’19); Association for Computing Machinery: New York, NY, USA, 2019; pp. 773–779.

38. Sánchez, J.; Aguayo, F. Blind Learners Programming through Audio. Presented at the CHI’05 Extended Abstracts on Human Factors in Computing Systems, Portland, OR, USA, 2–7 April 2005; pp. 1769–1772.

39. National Association of the Blind. Kafeef. Available online: http://www.kafeef.org (accessed on 14 October 2018).

40. Mohammed, M.; Lawan, A.; Galadanci, B.S.; Mijinyawa, M.K. Detection and Categorization of Errors by Novice Programmers in a First-Year Java Programming Class: A Comparative Analysis. In Proceedings of the Third International Conference on Digital Enterprise and Information Systems, Shenzhen, China, 16–18 April 2015; p. 104.

41. Faja, S. Evaluating the effectiveness of pair programming as a teaching tool in programming courses’. Inf. Syst. Educ. J. 2014, 12, 36.

42. Guo, P.J. Non-native English speakers learning computer programming: Barriers, desires, and design opportunities. Presented at the 2018 CHI Conference on Human Factors in Computing Systems, Montreal, QC, Canada, 21–26 April 2018; pp. 1–14.

43. Mrwan, B.I. The Correlation between Arabic Students’ English Proficiency and their Computer Programming Ability at the University Level. Int. J. Manag. Public Sect. Inf. Commun. Technol. (IJIMPICT) 2019, 70, 207.

44. Veerasamy, A.K.; Shillabeer, A. Teaching English Based Programming Courses to English Language Learners/Non-Native Speakers of English. Int. Proc. Econ. Dev. Res. 2014, 70, 311.

45. Noshin, J.A.; Ahmed, S.I. Teaching Programming to Non-Programmers at Undergraduate Level. Int. J. Eng. Manag. Res. (IJEMR) 2018, 8, 191–194. [CrossRef]

46. Aldabas, R.A. Special Education in Saudi Arabia: History and Areas for Reform. Creat. Educ. 2015, 6, 1158. [CrossRef]

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).