Nonvisual Support for Understanding and Reasoning about Data Structures

Brianna L. Wimer*
bwimer@nd.edu
University of Notre Dame
Notre Dame, Indiana, USA

Ritesh Kanchi*
rkanchi@g.harvard.edu
Harvard University
Cambridge, Massachusetts, USA

Kaija Frierson
kaijaf@uark.edu
University of Arkansas
Fayetteville, Arkansas, USA

Venkatesh Potluri
potluriv@umich.edu
University of Michigan
Ann Arbor, Michigan, USA

Ronald Metoyer
rmetoyer@nd.edu
University of Notre Dame
Notre Dame, Indiana, USA

Jennifer Mankoff
jmankoff@cs.washington.edu
University of Washington
Seattle, Washington, USA

Miya Natsuhara
mnats@cs.washington.edu
University of Washington
Seattle, Washington, USA

Matt X. Wang
mxw@cs.washington.edu
University of Washington
Seattle, Washington, USA

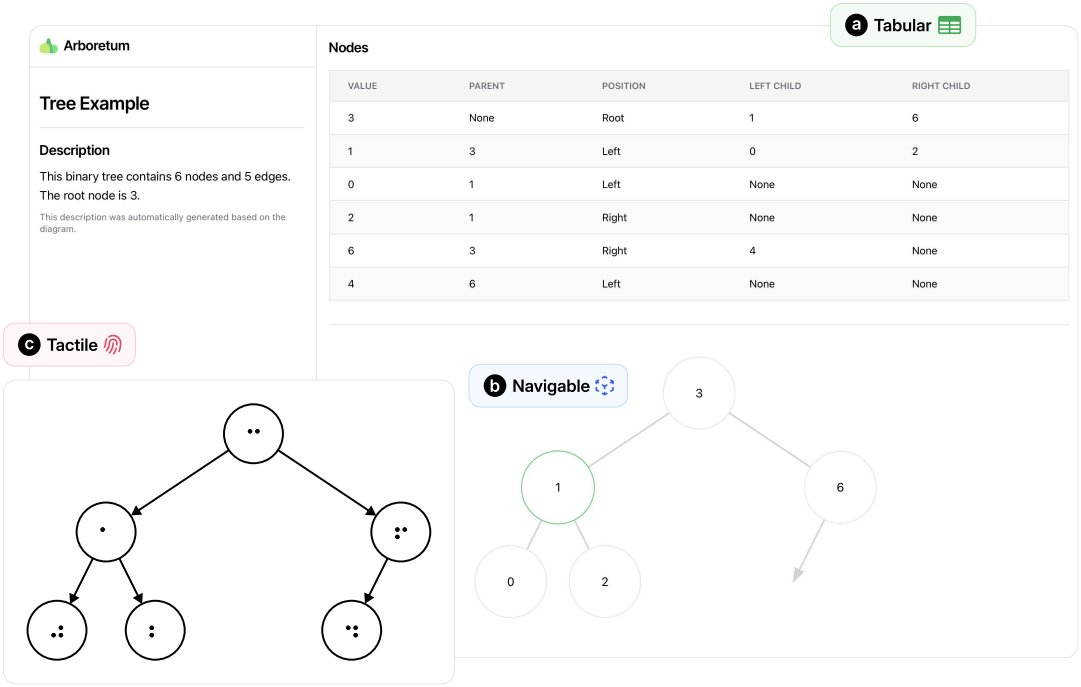


Figure 1: Arboretum, a web-based system that generates accessible diagrams of introductory data structures in (a) tabular, (b) navigable, and (c) tactile representations. This example highlights a binary tree in Arboretum across all three representations.

*The first two authors are equal contributors to this work.

CHI '26, April 13–17, 2026, Barcelona, Spain

ACM ISBN 979-8-4007-2278-3/2026/04
https://doi.org/10.1145/3772318.3791656

ABSTRACT

Blind and visually impaired (BVI) computer science students face systematic barriers when learning data structures: current accessibility approaches typically translate diagrams into alternative text, focusing on visual appearance rather than preserving the underlying structure essential for conceptual understanding. More accessible alternatives often do not scale in complexity, cost to produce, or both. Motivated by a recent shift to tools for creating

visual diagrams from code, we propose a solution that automatically creates accessible representations from structural information about diagrams. Based on a Wizard-of-Oz study, we derive design requirements for an automated system, Arboretum, that compiles text-based diagram specifications into three synchronized nonvisual formats—tabular, navigable, and tactile. Our evaluation with BVI users highlights the strength of tactile graphics for complex tasks such as binary search; the benefits of offering multiple, complementary nonvisual representations; and limitations of existing digital navigation patterns for structural reasoning. This work reframes access to data structures by preserving their structural properties. The solution is a practical system to advance accessible CS education.

CCS CONCEPTS

• **Human-centered computing** → **Accessibility**; • **Social and professional topics** → **Computing education**.

KEYWORDS

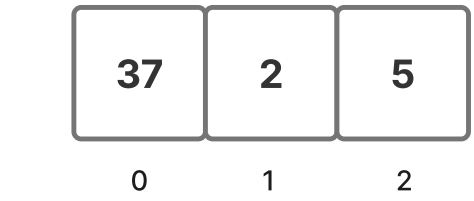
Accessibility, Diagramming, Computer Science Education

ACM Reference Format:
Brianna L. Wimer, Ritesh Kanchi, Kaija Frierson, Venkatesh Potluri, Ronald Metoyer, Jennifer Mankoff, Miya Natsuhara, and Matt X. Wang. 2026. Nonvisual Support for Understanding and Reasoning about Data Structures. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26), April 13–17, 2026, Barcelona, Spain.* ACM, New York, NY, USA, 21 pages. https://doi.org/10.1145/3772318.3791656

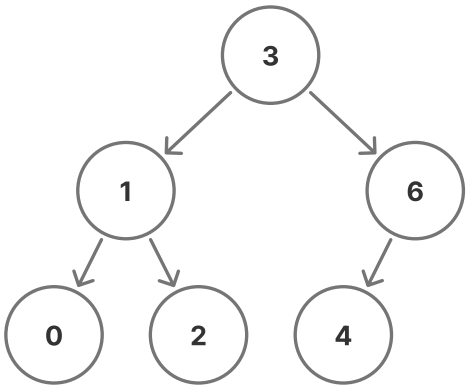
1 INTRODUCTION

The road into computer science (CS) is paved with visual representations [28]. They are central to reasoning about computing: programmers sketch designs to reason about their software and how their data is structured [14, 31, 83], and CS instructors rely on diagrams, whiteboard sketches, and animations to communicate the complexity of core concepts [27, 28]. In introductory CS education (*e.g.,* CS1, CS2), data structures such as arrays and binary trees are often introduced through such diagrams, aiding abstraction, supporting mental model development, and reducing cognitive load [31, 72, 73]. However, these representations remain inaccessible to blind and visually impaired (BVI) students [73], who already face barriers in CS education due to under-prepared educators and the limited availability of accessible instructional materials [41, 42].

To make the problems BVI learners face more concrete, consider a hypothetical student, Maya, exploring an array presented visually as a row of adjacent boxes, each containing a value and labeled with its index (Figure 2a). A hurried instructor, doing their best to follow Web Content Accessibility Guidelines (WCAG) for "complex images" (*e.g.,* charts, graphs, diagrams, illustrations, maps) [75], provides an alternative text description (*e.g.,* ALT text) that simply lists the values: "5, 2, 9, 7…". To identify the element at array index 6, Maya must count sequentially through each value, unlike her sighted peers who can can immediately see which value sits in the box labeled with index "6." This strategy quickly becomes cognitively intensive for BVI students as arrays grow in size, and instructors often struggle to maintain consistent nonvisual descriptions as diagrams become larger or get updated shortly before class.



(a) Array.



(b) Binary Tree.

Figure 2: Representations of standard array and binary tree diagrams as seen in a slide deck.

A similar issue arises with node-link diagrams for binary trees: sighted learners rely on spatial cues—circles connected by arrows with parents above children and left/right positioning (Figure 2b)—while Maya receives ALT text that states: "The binary tree has a root with the value 4. The 4's children have values of 2 and 8. The 2's child has a value of 1. The 8's children have values of 6 and 10." Even this level of detail varies depending on the instructor's expertise or time to prepare. For Maya, reasoning about operations like binary search or verifying the binary search tree (BST) property results in an over-reliance on working memory or repeated passes over the description to decipher semantic roles and relationships. ALT text often omits the *computational properties* of data structures: the rules, relationships, and reasoning procedures that define how a structure is organized and how it can be traversed or manipulated. Although Maya's experience reflects the minimum legal standard for classroom accessibility [74], it falls short of providing equitable access: diagrams are treated as images to be described rather than as structured representations that can be encoded. Larkin and Simon [43] note that diagrams support reasoning by organizing information spatially, yet this structure is lost when diagrams are exported as images and later paired with text. Supporting equitable access therefore requires representing diagrams in a form where structural relationships are captured directly rather than left implicit.

Prior research has solved this problem by developing alternatives to text. Tactile representations can preserve layout and relationships, while supporting conceptual understanding of CS concepts

for BVI learners [2, 17, 54]. However, producing tactile materials requires time and expertise and is manually done separately for each new diagram, making them challenging to create consistently for instruction [17, 35, 57, 63]. If structural information about diagrams, instead of images of diagrams, were available, it would be feasible to automate this process, as well as automating ALT text generation and other accessibility solutions. An automated solution could vastly improve the speed and scale at which diagrams could be made accessible. Structural information is often available in some form during the diagram creation process. Some instructors use tools to create diagrams, ranging from general-purpose presentation software (e.g., PowerPoint, Google Slides) to specialized diagramming environments (e.g., draw.io, Lucidchart). However, each tool introduces its own stylistic conventions and offers limited accessibility features. Others use text-based diagram specifications (*e.g.,* Mermaid[1], Graphviz DOT[2] PlantUML[3], d2[4]). Our key insight is that **these tools naturally capture structural information including nodes, relationships, and reasoning patterns in a single source of truth from which accessible multimodal representations can be automatically generated.**

Building on this insight, we present the iterative design of an automated solution for making diagrams accessible. Our work focuses on arrays and binary trees, foundational data structures in introductory CS: arrays demonstrate sequential ordering and index-based access, while binary trees demonstrate different traversal strategies [45]. We have three primary contributions. First, we **derive five design requirements from a small Wizard-of-Oz study of three nonvisual prototypes of general-purpose flowcharts**: textual with Q&A support, navigable, and tactile formats. Our goal was to understand accessibility barriers across representation modalities. Participants had difficulty forming complete mental models from any single prototype, and different prototypes had complementary strengths and weaknesses. The resulting requirements shaped the design and implementation of Arboretum, **a web-based system that compiles diagram specifications into tabular, navigable, and tactile representations that make a data structure's computational properties explicit** (Figure 1). Arboretum automates this process through access of structured data rather than images. To validate Arboretum, we asked 8 BVI learners to use Arboretum's representations. Participants successfully worked through five task types: (1) locating specific elements, (2) identifying parent–child relationships, (3) identifying leaf nodes, (4) verifying BST properties, and (5) performing binary search. We show that the **nonvisual representations generated by Arboretum support comprehension and reasoning over arrays and binary trees.** Based on these contributions, we highlight opportunities for expanding the concepts to a wider array of educational settings.

2 RELATED WORK

Our research intersects two strands of prior work: accessible CS education for BVI learners, which highlights the barriers students face and the systems developed to support them; and accessible diagrams in STEM, which examines tactile, auditory, and multimodal alternatives to visual representations.

2.1 Accessible CS Education and Systems for BVI Learners

BVI learners encounter a range of visually-oriented barriers in CS education—spanning curriculum, tools, and classroom practices [3]. Current computing education environments often lack strategies for making learning both inclusive and accessible, though educators and researchers have explored approaches to address these gaps.

At the curriculum level, efforts include the creation of screen reader-friendly instructional content and physical components that embody abstract computing concepts [1, 36, 48, 70]. Visual representations—commonly used to teach data structures, algorithms, and other programming concepts—continue to present significant barriers for BVI learners when not paired with accessible alternatives [3, 37, 46, 50]. To counter this, schools for the blind and visually impaired have long used tactile manipulatives–boxes to represent variables, dice for numerical values, and switches for Boolean values [66]. Additional work has focused on curriculum redesign efforts: for example, Stefik et al. [68] adapted the Code.org AP Computer Science Principles course by offering accessible and unplugged activity alternatives and ensured that content adhered to WCAG 2.1 AA [75] and ARIA [76] guidelines.

Some systems have focused on making computing concepts accessible through alternative representations. For example, the *GSK* system [5, 6] enables BVI and sighted learners to create, edit, and share graphs using familiar interaction mechanisms (*e.g.,* mouse, keyboard, screen reader). To support both groups, GSK provides synchronized grid and connection views of a graph, ensuring proper representation and interaction.

Beyond CS concepts, other systems have aimed to make programming environments accessible. For instance, Smith et al. [61] introduced *JavaSpeak*, a specialized Java programming environment with aural feedback that "speaks" a program's semantics and structure analogous to how syntax highlighting and indentation visually communicate structure to a sighted learner. Similarly, *Sodbeans* offers a computer programming environment with custom screen reader, *Hop* programming language, and accompanying talking debugger [65, 66]. Other accessible programming languages include Blocks4All [64] and Quorum [67]. At a broader level of abstraction, Schanzer et al. [58] developed a language-independent toolkit for creating accessible programming environments, providing spoken descriptions and accessible navigation through a unified block editor for BVI programmers. Plugins to adapt existing IDEs have also been developed; *StructJumper,* an Eclipse plugin that exposes the structure of a Java class via a hierarchical tree for easy code navigation [4] and *CodeTalk*, a Visual Studio plugin which addresses accessibility issues in code comprehension, editing, debugging, and collaboration for the BVI developer experience [56].

Collectively, these curricular efforts and systems demonstrate that accessible design and alternative representations are critical for BVI learners in CS. They also highlight that accessibility must be built into both instructional materials and educational tools. Our system, Arboretum, builds on this foundation by focusing on data

[1] https://mermaid.js.org
[2] https://graphviz.org/doc/info/lang.html
[3] https://plantuml.com/
[4] https://d2lang.com/

structure diagrams, providing a low-barrier interface for educators and students to create and explore accessible representations.

2.2 Nonvisual Presentations of Data from Diagrams

Beyond CS education settings, researchers have explored various nonvisual approaches, from tactile graphics to multimodal audio-tactile systems, to preserve the essential properties of diagrams across computing contexts including flowcharts [2, 18, 37], node–link diagrams [25, 38, 86], UML diagrams [12, 20, 34, 40, 47, 51, 52, 80], and graphs [5, 6, 13, 15, 16, 69].

Tactile representations preserve spatial layouts and relationships through physical formats such as raised-line graphics on swell paper thermoformed materials, as well as digital refreshable tactile displays [29, 54, 57]. For instance, hyperbraille displays have enabled users to directly follow interconnections in mind maps via raised nodes and lines [54]. While tactile representations excel at conveying spatial structure and supporting direct manipulation, they face significant scalability challenges: physical production requires specialized equipment and expertise [57].

Researchers have attempted to preserve a visual representation's structure through auditory approaches that map to screen reader navigation patterns [7, 23, 32, 55, 85], but these sacrifice the spatial grounding that makes tactile representations effective. Multimodal representations emerged to combine the spatial grounding of tactile exploration with detailed information delivered auditorily, as users often value this parallel presentation of information across modalities [21, 37, 80]. Early work by Blenkhorn and Evans [9] created audio-tactile matrices for data flow diagrams, demonstrating how combined modalities could support both spatial layout and content detail. Building on this foundation, researchers have developed various interaction approaches: Kennel et al. [38] enabled users to trigger sounds through touch, Alzalabny et al. [2] augmented two-dimensional tactile interfaces with audio feedback for technical diagrams, and Kawulok et al. [37] implemented gesture-driven audio descriptions. More recent tablet-based systems allow users to navigate through touch while receiving auditory cues to identify objects and interconnections [80, 86]. These systems all enable ease for constructing diagrams, taking advantage of multimodality for nonvisual presentations of data.

Evaluating nonvisual representations has become increasingly important as approaches have diversified. Since nonvisual representations vary widely and offer different benefits, prior work has examined how to compare their efficiency to access information [78, 79, 84], user preferences [30], representation usability [30, 82], workload [84], and user performance across tasks [24, 30, 82, 84]. While prior research focuses on general-purpose diagrams, there is little systematic understanding of how nonvisual representations support learning and conceptualizing domain-specific diagrams in computing remains limited. Our work addresses this by focusing specifically on data structure diagrams and operationalizing access through complementary representations that preserve computational meaning.

3 WIZARD-OF-OZ STUDY FOR REPRESENTATION DESIGN

The goal of our first study was to understand the tradeoffs between different representations of a generic class of frequently used diagrams: node-link diagrams (specifically, flowcharts). These diagrams use the same node-link structure as data structure diagrams, which appear widely in CS education [28]. Using a Wizard-of-Oz approach allowed us to explore tradeoffs between representations before fully implementing them [19].

Representations. We created three nonvisual prototypes: 1) an ALT-text description with a high-level overview and a linearized list of nodes and edges, supplemented by an LLM-based question-answering component; 2) a keyboard-navigable digital graph whose custom navigation followed the diagram's directional flow (necessary because flowcharts with cycles do not map cleanly onto existing screen reader patterns); and 3) a tactile diagram printed on swell paper with a digital legend. We implemented a back-end system supporting our Wizard-of-Oz study of the Q&A interface and the digital graph. When participants wanted to ask questions, or explore the digital graph, they verbalized their intent and researchers executed the commands needed to generate spoken output. Additionally, since four of our participants were unfamiliar with reading braille, researchers provided verbal label information on request for the tactile diagram.

Method and Analysis. We described the study's purpose and procedures, and introduced flowchart-related concepts and terminology to participants. We then asked them to explore the three prototypes, starting with the ALT text representation. After exploring each prototype, participants answered usability questions and provided feedback on their experience. Sessions were about 90-minutes long and took place in-person at the **(anonymized location and organization for review)**. All sessions were recorded with consent, and participants received a $60 Amazon gift card as compensation. Recordings and transcripts were analyzed using an affinity diagramming process [59]. We focused on the usability of each prototype, their effectiveness in assisting participants in answering flowchart-related questions, and identifying areas for enhancement based on participant feedback.

Participants. Eight participants, several who had prior experience with flowcharts, were recruited through a national organization supporting BVI professionals. Ages ranged from 36 to 67 (mean = 51 years, SD = 12). We refer to these participants as FP1-FP8 throughout our findings.

3.1 Findings and Resulting Design Requirements

This first study revealed five recurring accessibility needs for diagram representations:

DR1. Adopt standardized screen reader navigation models. We implemented a Wizard-of-Oz navigation pattern that allowed participants to move directionally along the diagram's logical flow (*e.g.,* advancing through outgoing edges, returning via incoming edges, and following labeled branches) rather than relying on a

parent–child hierarchy. Although this pattern matched the structure of the flowcharts, participants had difficulty using it; when sibling nodes reflected different decision paths or cycles brought them back to earlier nodes, the unfamiliar commands became hard to interpret. In contrast, participants readily used ARIA-style commands when they applied, drawing on existing expertise. As FP2 noted: "Using the keystrokes is easy…knowing that parent was left, child was up and down…that makes it easier." Such reactions guided us toward standardizing navigation to match familiar screen reader interaction models, reducing both the learning burden and interaction friction.

DR2. Provide structural information in tables. Participants consistently suggested presenting nodes and edges in a tabular structure instead of a list inside prose because it aligns with existing screen reader expertise in table navigation: "Using arrow keys with screen readers and spreadsheets…it's much easier for me. I can quickly go back and forth" (FP8). Tables offer a stable, predictable baseline format for making data accessible [77, 87]—a format that separates structural facts from explanation, supports random access, and reduces the variability inherent in authored descriptions.

DR3. Explicitly encode structural relationships rather than requiring inference. Participants had difficulty determining how elements were connected when relationships were described in the ALT text format; they lacked clear cues about which nodes were connected or how they were positioned relative to one another. Because the ALT text listed nodes and edges linearly, participants could not determine which elements were related without reconstructing structure through inference. One participant stated: 'Trying to understand which nodes are attached to the edges and the different labels was somewhat difficult." (FP3), while another reported: "I wasn't really getting the layout or structure" (FP8).

DR4. Separate structural facts from narrative explanation in nonvisual representations. Participants struggled with the ALT text prototype because moving back and forth between a high level overview and a lists of nodes and edges made it difficult to track what information was explanatory and what was structural. As one participant stated, "You get this big data dump…trying to break it down is hard" (FP7). Others reported difficulty understanding the ordering of details, with FP1 noting, "I wasn't getting a cohesive picture…the order of things wasn't clear." Representations should therefore separate conceptual framing from structural detail, providing a clear overview alongside a distinct, precisely structured enumeration of nodes, edges, and roles.

DR5. Enable integrated access across complementary modalities. No single representation contained enough information for participants to understand the full structure; when restricted to one representation, they often felt uncertain or confused. Instead, participants preferred when they combined representations, using each to compensate for the limitations of another. Participants explicitly described integration strategies: "I'd use a tactile printout to verify the visual map in my head" (FP8) and "I would check [the Q&A answers] by verifying with the raw data table" (FP1). Here, "integrated access" refers to supporting smooth movement between representations and enabling users to cross-check and coordinate information across modalities.

Together, these requirements expose the structural information that is normally implicit in visual diagrams and motivate synchronizing multiple modalities and representations.

4 THE ARBORETUM SYSTEM

Arboretum is a web-based system that generates accessible data structure—currently array and binary tree—diagrams for introductory CS environments, ensuring that structural information is preserved[5]. Arboretum operationalizes the design requirements elicited in our Wizard-of-Oz study by compiling diagrams written in diagram specification languages into three synchronized representations: tabular, navigable, and tactile. By "synchronized," we mean that all representations are generated from the same underlying structural model, ensuring they remain consistent and convey the same structural information. Rather than implementing many-to-many translators, we compile the text-based diagram specifications into an intermediate representation (IR) to provide a single unified format and infer additional accessibility properties of the data structure.

Arboretum was developed with both CS educators and students in mind: **educators** who need to produce accessible data structure diagrams for instruction, and **students**, who engage with multimodal outputs to explore and learn data structures accessibly. There are many text-based diagram specification languages which describe nodes, edges, and layout rules in a simple textual syntax that a renderer converts into a visual diagram. Typically, these languages offer limited accessibility support, often only allowing authors to add manual ALT text. By directly supporting familiar diagram specification languages, Arboretum makes accessibility effectively zero-cost for pre-existing diagrams, requiring minimal additional work from diagram creators.

4.1 Implementation

Arboretum is a client-side web application developed with TypeScript, React, and Next.js[6][7]. The system follows a three-stage pipeline: input specification, translation, and output generation (Figure 3). Arboretum introduces an IR that encodes the accessibility-relevant semantics (*e.g.,* traversal order, role labeling, explicit structural relationships) needed for accessible data structure representations. Input diagram specifications are compiled into this IR, and a separate compiler pass generates the accessible outputs from it.

Input. Some instructors visually author diagrams in direct manipulation tools (*e.g.,* PowerPoint, Lucidchart, Canva), while others use text-based diagram specification languages to use declarative syntax that supports abstraction and reuse in ways that direct manipulation tools do not [53]. Arboretum accepts diagrams from some of the more popular of these text-based languages, specifically Mermaid (Figure 4a) and Graphviz DOT (Figure 4b). Arboretum also requires that authors must specify the input's data structure type to ensure proper compilation into the IR (Figure 3a). .

Translation. In the translation stage, Arboretum compiles inputs into the IR by first parsing the source specification into an

[5]A live demo can be accessed at g.riteshkanchi.com/chi26-arboretum/demo
[6]https://nextjs.org
[7]The source code is available at github.com/ritesh-kanchi/Arboretum

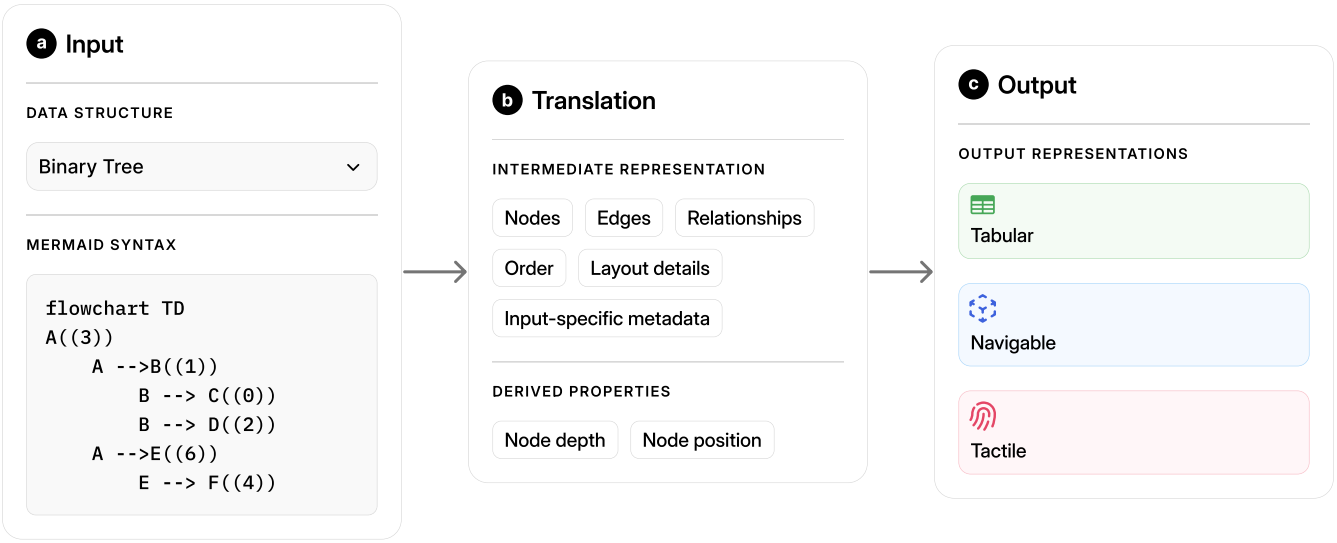


Figure 3: A system overview of Arboretum: an (a) input stage with a selected data structure and a provided Mermaid diagram specification, a (b) translation stage compiles the input into the IR, extracting and deriving additional properties, and an (c) output stage that generates accessible output representations.

abstract syntax tree that captures its elements and their connections (Figure 3b). Because diagram specification languages can differ in syntax and expressive power, each compiler pass is tailored to the source language (*e.g.,* Mermaid or Graphviz DOT) and the data structure type the author specifies. Figure 4 shows how Mermaid and Graphviz specifications of the same binary tree are compiled to the same IR. Figure 5 illustrates how the IR supports arrays and naturally generalizes to linked lists and two-dimensional arrays.

Output. Arboretum uses the IR to generates three accessible representations (Figure 3c)— tabular, navigable, and tactile—and each output is automatically generated from the IR, so any update to the input is automatically reflected across all representations. Section 4.3 provides a detailed description of each output representation.

4.2 Interface

Arboretum was developed following WCAG 2.1 AA guidelines [75] to ensure accessibility, and tested with the JAWS[8], NVDA[9], and VoiceOver (macOS) [10] screen readers to confirm that interactions follow common patterns for screen reader users.

Editor mode. In editor mode, educators can author diagrams using Mermaid or Graphviz DOT syntax while simultaneously previewing the generated outputs. The live preview ensures that edits are immediately synchronized across all output modalities, allowing authors to verify both correctness and accessibility. Authors can also provide titles and descriptions for diagrams, giving students additional context. If an author-provided description is absent, Arboretum generates concise descriptions, satisfying **DR4** by providing a brief, high-level explanation separate from the data structure representation itself. For example, for the binary tree in Figure 8, if no author description is provided, Arboretum displays: *"This binary tree contains 6 nodes and 5 edges. The root node is 3."*

Preview mode. In preview mode, students can view the accessible representations of a diagram without interacting with the underlying diagram specification language. The interface highlights the generated representations through a representation-selection dropdown while minimizing distractions from the Editor counterpart, providing a focused space for students to engage with data structures representations.

Across both modes, Arboretum supports local favoriting to revisit examples as well as sharing and embedding diagrams via hyperlinks, QR codes, or embedded iFrames. Authors and users can also easily switch between the editor and preview modes to edit or preview the Arboretum diagram respectively.

4.3 Output Representations

Arboretum's pipeline generates three synchronized representations that follow the design requirements identified in Section 3.1, and are illustrated for arrays (Figure 7) and binary trees (Figure 8). Figure 2 shows the original visualizations of the array and binary tree provided to sighted students before using Arboretum to create alternative, accessible representations.

Tabular. Tabular representations present a consistent, unambiguous encoding of data structures in a table format. This modality leverages familiar table navigation patterns and HTML semantics to make structural relationships explicit while supporting efficient nonvisual access and exploration. Given participants' strong suggestion for tabular formats (**DR2**), this view also supports **DR3**

[8]https://www.freedomscientific.com/products/software/jaws/
[9]https://www.nvaccess.org/download/
[10]https://support.apple.com/guide/voiceover/welcome/mac

[11]A screen reader-friendly version of this array diagram is available at: g.riteshkanchi.com/chi26-arboretum/array-outputs

```
flowchart TD
A((1))
    A -->B((2))
        B --> C((3))
        B --> D((4))
    A -->E((5))
        E --> F((6))
```

(a) Mermaid syntax.

```
digraph bt {
    A[label="1"];
    B[label="2"];
    C[label="3"];
    D[label="4"];
    E[label="5"];
    F[label="6"];

    A->B;
        B->C;
        B->D;
    A->E;
        E->F;
}
```

(b) Graphviz DOT syntax.

```
ArborBinaryTree:
  meta:
    type = "binary_tree"
    ...    # optional metadata

  nodes = [
    Node(id = "A", value = "1", ...)
    Node(id = "B", value = "2", ...)
    ...    # additional nodes
  ]

  edges = [
    Edge(parent = "A", child = "B", ...)
    Edge(parent = "B", child = "C", ...)
    ...    # additional edges
  ]
```

(c) Intermediate representation.

Figure 4: Examples of Mermaid (a) and Graphviz DOT (b) input syntax for the same binary tree compiled into Arboretum's shared IR (c), which unifies distinct diagram formats into a shared, semantically consistent model.

```
ArborArray:
  meta:
    type = "array"
    ...    # optional metadata

  elements = [
    Element(id = "A", value = "1", ...)
    Element(id = "B", value = "2", ...)
    ...    # additional elements
  ]
```

(a) Array.

```
ArborLinkedList:
  meta:
    type = "linked_list"
    ...    # optional metadata

  nodes = [
    Node(id = "A", value = "1", ...)
    Node(id = "B", value = "2", ...)
    ...    # additional nodes
  ]
```

(b) Linked list.

```
Arbor2DArray:
    meta:
        type = "2d_array"
        ...     # optional metadata

    rows = [
        Row(children = [
          Element("A", "1"), Element("B", "2"), ...
        ]),
        Row(children = [
          Element("D", "3"), Element("E", "4"), ...
        ])
        ...     # additional rows of elements
    ]
```

(c) Two-dimensional array.

Figure 5: IRs generated by Arboretum for arrays (a) with examples of generalization to linked lists (b) and two-dimensional arrays (c).

by explicitly encoding roles and relationships—such as parent and child—rather than requiring users to infer them from narrative descriptions. The table encodes the semantics of each data structure: for arrays and lists, rows capture explicit indices and values (Figure 7a); for binary trees, rows capture node attributes such as value, parent/child relationships, and positional role (root, left, right) (Figure 8a).

Navigable. While tables expose explicit values and relationships, they do not reflect how learners typically traverse or interact with a data structure. The navigable representation provides an interactive, nonvisual analogue to common navigation patterns (*e.g.,* iterating through an array, traversing a tree). This representation maintains the visual and spatial layout familiar to sighted instructors but is implemented with screen reader-friendly HTML. To satisfy **DR1**, we grounded the navigation model in established WAI-ARIA interactions—such as list and tree navigation [76]. Arrays are rendered as horizontally adjacent boxes but are represented as list items containing each element's index and value (Figure 7b). Binary trees are rendered visually with circular nodes and edges but are represented as nested lists following ARIA tree patterns, enabling users to expand and collapse child nodes for efficient traversal. Following the WAI-ARIA conventions preserves visual clarity for

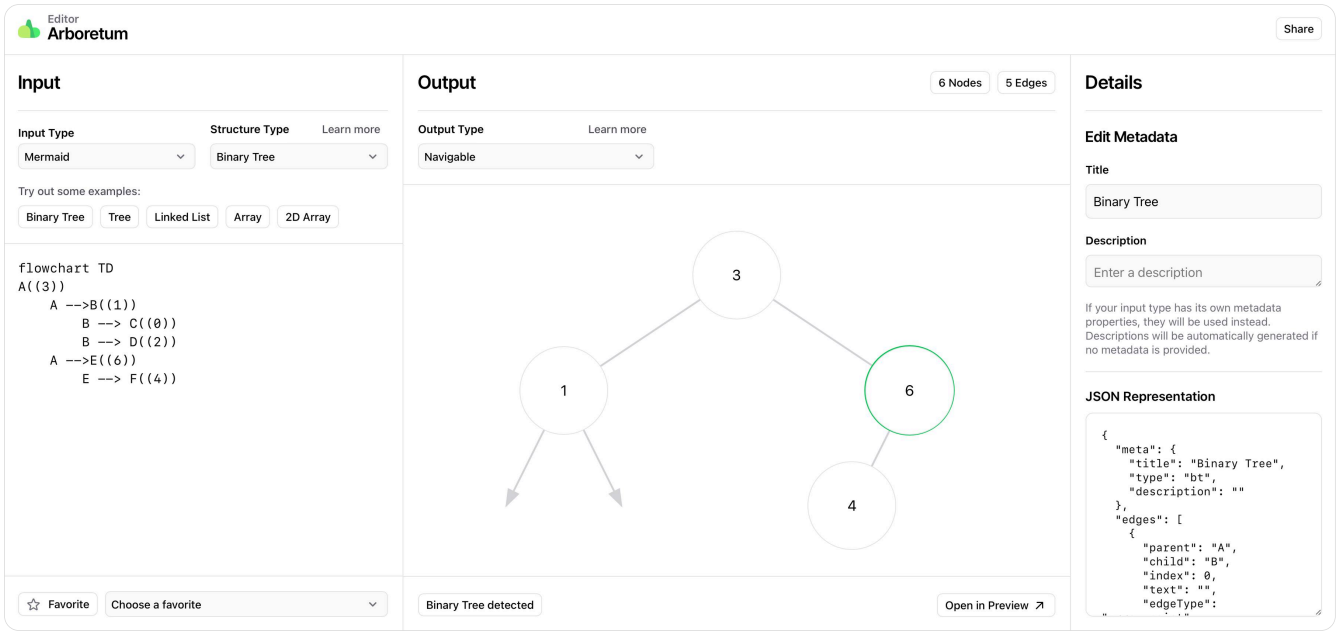


Figure 6: Arboretum's Editor mode, which allows educators to write diagram specifications and preview synchronized accessible outputs.[9]

sighted learners while offering BVI learners a consistent and accessible interaction model. Table 1 summarizes the commands used in the binary tree navigable representation.

Tactile. Whereas the tabular emphasizes explicit relationships and navigable supports interactive exploration, the tactile representation provides an explicit spatial dimension. Tactile representations allow learners to physically perceive structure and layout, offering affordances particularly useful for reasoning about hierarchy and positional relationships. In Arboretum, the tactile generator produces scalable SVG diagrams that can be embossed using common tactile-production equipment found in most university assistive technology offices, such as embossers or Swell Form (fuser) machines. Arrays are represented as adjacent boxes with Braille-labeled values and indices, supporting both sequential and positional reasoning (Figure 7c). Tactile representations generated by Arboretum contain Braille-labeled nodes and arrowheads that mark directed edges for parent–child node connections in a branching layout to preserve hierarchy and directionality (Figure 8c). Educators can also combine multiple tactile-ready SVGs for a single tactile sheet—for example, showing an original tree and a version with a node removed—by arranging Arboretum 's SVG outputs in a slide deck or layout tool.

Together, these three representations satisfy **DR5** by enabling integrated, complementary access: learners can fluidly switch between representations to cross-check relationships, confirm structural details, and form stable mental models.

Table 1: Supported interactions for Arboretum's Binary Tree Navigable Representation.

Command	Action
→→ (Right arrow twice)	Expands the current node and moves to its first child (left child if both exist).
←← (Left arrow twice)	Collapses the current node and its siblings, then moves to the parent node (no action if no parent).
↑ (Up arrow)	Move to the prior child; if none, moves to the previous node at the same level.
↓ (Down arrow)	Move to the next child; if none, moves to the next node at the same level.

4.4 Limitations of Arboretum

While Arboretum illustrates a low-barrier to adoption, educator-centered approach to accessible data structure representations, several limitations remain. First, only supporting Mermaid and GraphViz DOT may present a learning curve for educators unfamiliar with these languages, and array creation is limited to Mermaid as GraphViz DOT lacks a commonly adopted convention for block-style diagrams. Second, screen reader support was evaluated only with JAWS, NVDA, and VoiceOver (macOS); future work should test Narrator[13], mobile screen readers such as VoiceOver (iOS, iPadOS) and TalkBack[14], and additional assistive technologies to ensure broader compatibility. Third, node and element labels are length-limited—approximately ten characters visually and three characters in Braille for tactile representations—restricting use of larger values.

[12]A screen reader-friendly version of this binary tree diagram is available at: g.riteshkanchi.com/chi26-arboretum/tree-outputs

[13]https://www.microsoft.com/en-us/windows/tips/narrator

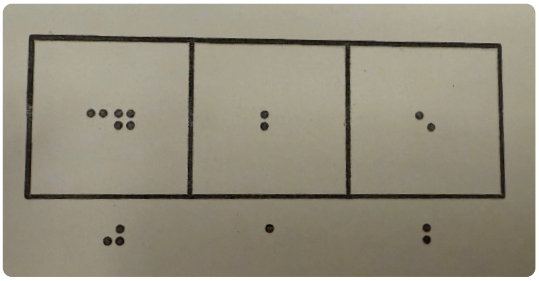
[14]https://support.google.com/accessibility/android/answer/6283677

INDEX	VALUE
0	37
1	2
2	5

(a) Tabular.

Index 0, 37	Index 1, 2	Index 2, 5

(b) Navigable.



(c) Tactile.

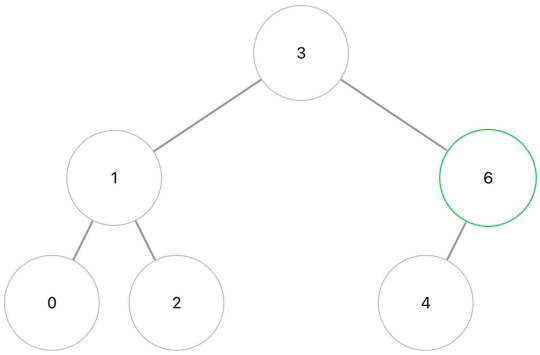
Figure 7: Representations of an array: (a) a tabular representation of elements' indexes and values, (b) a navigable representation that is screen reader navigable, and (c) a tactile representation with braille [11].

Tactile output also carries inherent constraints independent of Arboretum. Physical production technologies (*e.g.*,, swell paper, embossers) impose constraints on sheet size, resolution, and production cost [17, 57]. In our case, a standard Swell Form machine ($1,475) [15] and a box of 100 sheets of swell paper ($138) [16] kept per-page costs low enough for routine instructional use. In addition, tactile media does not scale well to very large structures such as multi-level search trees or arrays with hundreds of elements, where multiple sheets or sequential prints would be required. For these larger or more complex structures, Arboretum's digital modalities (*e.g.*,tabular and navigable) are better suited for nonvisual exploration; however, large tables can themselves become cumbersome to navigate with a screen reader [77].

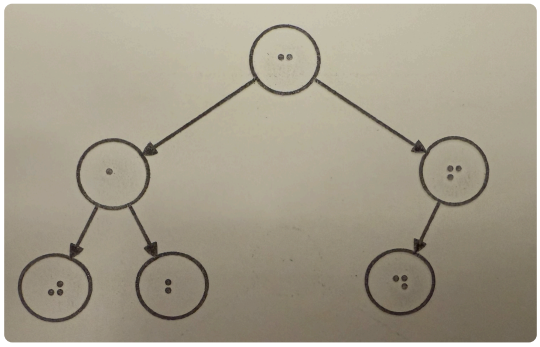
[15]https://tinyurl.com/swell-paper-machine
[16]https://tinyurl.com/swell-paper-cost

VALUE	PARENT	POSITION	LEFT CHILD	RIGHT CHILD
3	None	Root	1	6
1	3	Left	0	2
0	1	Left	None	None
2	1	Right	None	None
6	3	Right	4	None
4	6	Left	None	None

(a) Tabular.



(b) Navigable.



(c) Tactile.

Figure 8: Representations of a binary tree: (a) a tabular representation of the parent and children nodes, (b) a navigable representation that is screen reader navigable, and (c) a tactile representation with braille[12].

5 EVALUATING ACCESSIBLE ARRAYS AND BINARY TREES WITH BVI USERS

We conducted a within-subjects study to examine how participants perceive multimodal access to data structures through three nonvisual representations and how these representations affect comprehension, navigation, and confidence. Our user study was guided by four specific research questions:

- **RQ-Comprehension:** How do Arboretum's representations (tabular, navigable, tactile) support *comprehension* of arrays and binary trees?
- **RQ-Navigation** How do participants *navigate* and interact with the different representations?
- **RQ-Confidence:** How does multimodal access (having multiple representations available) support participants' *confidence* in understanding across data structures?
- **RQ-Application:** How do participants *apply* binary search using Arboretum's representations, and how efficiently do they do so?

RQ-Comprehension and **RQ-Application** examine whether participants can extract computational information (element location, BST properties, algorithmic steps) from each representation. **RQ-Navigation** explores the interaction mechanisms that make these properties accessible, while **RQ-Confidence** measures how multimodal access affects participants' certainty and reasoning strategies across different data structures.

We designed each study session to build conceptually from simpler to more complex tasks. Participants first explored arrays, a linear data structure that establishes basic notions of indexing and bi-directional navigation. Next, they explored binary trees, which introduce hierarchical relationships and more complex navigation challenges. Finally, participants applied the binary search algorithm on binary trees, requiring them to integrate structural understanding with algorithmic reasoning. This progression allowed us to examine how Arboretum's representations supported learning from foundational comprehension through to applied problem-solving.

5.1 Participants

We recruited 8 (5 female, 3 male) BVI participants through accessibility-focused mailing lists and referrals from organizations serving BVI individuals. Our inclusion criteria included participants that: (1) were at least 18 years old, (2) self-identified as blind or visually impaired, and (3) were comfortable navigating digital content with commonly-used screen readers (*e.g.,* VoiceOver, JAWS, NVDA). One additional participant completed the study, but was excluded from analysis because she was not familiar with standard screen reader interactions during the session, despite screening as comfortable with a commonly used screen reader. As a result, her data was not comparable to that of other participants. Although prior experience with CS was not required, half of respondents (n=4) described having some prior exposure that came through formal education, work, or informal learning. Participants ranged in age from 37–65 (M = 50.8, SD = 12.4), with vision loss occurring at different stages of life. Each participant received a $40 Amazon gift card as compensation for their time. Table 2 presents participants' demographics.

5.2 Task Design and Structure

The study design included 6 tasks (T1-T6, shown in Table 3). The tasks were chosen to align with our research questions, and originally included two array tasks (T1 and T2) and four Binary Tree tasks (T3-T6). Each task was designed to require interacting with the data structure to answer one or more questions. Participants were free to use whichever representation(s) they found most most helpful, switch between them as needed, and were encouraged to think aloud while working. For the study, we simplified Arboretum's preview mode by removing sharing, local favoriting, and editor mode, allowing participants to focus solely on interacting with the output representations.

The number of questions varied across tasks. Questions probed concepts such as array indexing (T1), parent–child relationships, leaf identification, and verifying the BST property (T3–T5). Finally, a binary search task (T6) included questions that probed participants' algorithmic reasoning.

After each task question, we asked participants to think aloud while answering a more detailed follow up question. For example, to answer T4, participants counted the number of leaf nodes in the tree. The follow up question was 'what are their values?' Our intent was to understand the role of the representations in their learning.

5.3 Procedure

Sessions were conducted either remotely or in person and lasted approximately 60 minutes. Participants received a consent form and a short demographic survey in advance. At the start of each session, facilitators confirmed participants' background and CS experience, addressed any questions, and ensured both consent and the demographic survey were completed. To enable consistent screen recording of computer interactions, all participants joined a Zoom call, including those attending in person. Once these steps were completed, participants shared their screens so facilitators could observe their interactions.

We employed a within-subjects design in which all participants completed activities for both arrays and binary trees. All participants completed the same tasks in the same conceptual order. Because our goal was to examine how participants reasoned with full access to all representations—rather than compare performance across modalities—we did not create separate task variants. Instead, participants chose whichever representation(s) they preferred, enabling us to observe authentic strategy selection.

The three representations—tabular, navigable, tactile—were presented in a counterbalanced order. Counterbalancing applied only while participants were learning each representation, ensuring that no modality benefited from being taught first while participants were still forming initial mental models of the data structures. After this phase, participants were free to use any representation(s) for the tasks.

The activity sequence was fixed (arrays → binary trees → binary search), reflecting a common progression from simple linear (*i.e.* arrays) to hierarchical structures (*i.e.* binary trees) to algorithmic operations over those structures. Each task was administered once per participant to prioritize rich think-aloud reasoning rather than repeated-measures performance comparisons.

For each activity, we began with a brief introduction of the data structure and its application, followed by the representations. Facilitators guided participants through practice questions using a simple example to illustrate how each representation could be applied to answer structural questions. For remote participants, tactile materials were shipped in advance, and we standardized orientation by cutting the top-left corner of each page and embossing tactile page numbers in the top-right corner.

Table 2: Evaluation Participant demographics.

ID	Gender	Age	Vision Level	Education	CS Experience	Env.
P1	Male	43	Totally blind (no light perception)	Some college	Took a CS class in school; CS-related job; studied CS independently	Remote
P2	Male	41	Totally blind (some light perception)	Bachelors	None	Remote
P3	Female	62	Low vision (some usable vision)	Masters	Studied CS independently	Remote
P4	Female	65	Totally blind (some light perception)	Masters	None	In-Person
P5	Female	37	Totally blind (no light perception)	Masters	None	In-Person
P6	Female	64	Low vision (some usable vision)	Masters	CS-related job	Remote
P7	Male	57	Totally blind (no light perception)	Bachelors	Took a CS class in school; CS-related job; studied CS independently	Remote
P8	Female	37	Totally blind (some light perception)	Masters	None	Remote

Table 3: Task list for the Arboretum user study. All specific tasks can be found in Supplemental Materials. # is # of Questions

Task	Category	Structure	Example Question(s)	#
T1	Element Location	Array	Is the element 54 in the array? If so, at what index?	3
T2	Order/Sorting	Array	Is the array sorted in increasing order?	1
T3	Parent/Child Identification	Binary Tree	What is the value of the root's right child?	1
T4	Leaf Node Identification	Binary Tree	How many leaf nodes are in the tree?	1
T5	BST Property Check	Binary Tree	Do all nodes satisfy the BST property?	1
T6	Binary Search	Binary Tree	Does the tree contain the numbers 0, 5, 6, or 9?	3

This introduction allowed participants to see how different representations afforded different strategies. For instance, a leaf node could be identified in the tactile graphic by locating nodes without outgoing edges, in the table by finding rows where left/right child columns were empty, or in the navigable representation by checking which nodes could not be expanded or collapsed. Once participants were comfortable using the representations, they were provided with the official task (T1-T6) for that data structure.

For arrays, the learning example included three elements, while T1 and T2 included eight elements. For binary trees, the practice example was a binary tree of height 3 with 6 nodes, while T3-T5 had a binary tree of height 3 with 7 nodes (a full binary tree). Prior to T6, participants were taught binary search step-by-step using the practice example in their preferred representation, giving them an opportunity to learn the procedure before applying it independently. The practice tree satisfied the BST property and differed in both structure and values from the T6 tree; T6 had a binary tree of height 4 with 9 nodes that did not satisfy the BST property.

Participants were free to use any representation(s) they preferred to complete the tasks listed in Table 3. At the end of each activity, participants completed a short 5-point Likert-scale survey rating their comprehension of the data structure, clarity of the representation, ease of navigation, confidence in their answers, and how well each representation conveyed the underlying structure. After all activities were complete, we conducted a short semi-structured interview to elicit reflections on the usability of each representation, the experience of having multiple representations available, strategies for switching among them, and the transfer of understanding across data structures (*e.g.,* moving from arrays to binary trees).

5.4 Data Collection and Analysis

We recorded the time needed to answer each question, and the correctness of the answer. In the case of T2, our process uncovered three warning signals. First, participant answers took less time than any other question (2.1s on average vs 6.9-62.5s on average for the other questions). Next, participants did not describe data structure use in the follow up question. Finally, there was no correlation with **RQ-Comprehension**, **RQ-Navigation**, **RQ-Confidence**, or **RQ-Application**. Our interpretation is that participants may have answered T2 from memory alone. For this reason we excluded T2 from our quantitative analysis. For the remaining tasks, we report descriptive statistics because of our small sample size (n=8). This approach aligns with recommendations for small-sample accessibility research [44]. Our primary insights derive from an inductive thematic analysis of participants' interactions and feedback [11], with quantitative measures providing complementary context about performance patterns.

For our descriptive statistics, we used Accuracy of Extracted Information (AEI) as our primary performance measure, following prior work in accessible data representations [33, 60]. AEI is coded as a binary outcome for each task question, with a value of 1 ("accurate") if the participant answered correctly and 0 ("inaccurate") otherwise. Overall accuracy for each task was calculated as the proportion of correct responses out of the total number of questions, expressed as a percentage. In addition to AEI, we collected time spent for each task, time spent using each data structure, and participants' Likert-scale ratings after each activity. For tasks with multiple questions, (*e.g.,* T1, T6), we calculated the mean accuracy, mean time spent, and standard deviation of time spent across questions within each task.

For qualitative analyses, we analyzed participants' anonymized transcripts and screen recordings. Two authors independently conducted open coding, generating 444 initial codes that captured fine-grained aspects of participants' feedback. Through iterative discussion and constant comparison, these were consolidated into a codebook of 78 focused codes. The authors then collaboratively organized codes into 17 higher-order categories and 5 overarching themes: representation affordances & preferences, multimodal integration & strategic use, learning processes & mental model development,conceptual understanding & algorithm application, and accessible education implications.

6 RESULTS

Our results begin with an overview of task performance and subjective ratings to establish the quantitative landscape, then explore five key themes that emerged from our thematic analysis.

6.1 Overview of Performance and Preferences

Participants demonstrated strong performance on most tasks, with accuracy patterns supporting the effectiveness of our representations. As shown in Table 4, participants achieved perfect accuracy on some tasks (parent/child identification, binary search), while encountering the most difficulty in checking the BST property across all nodes in a tree.

Our array task revealed strong understanding in both performance and timing (T1, 91.67% accuracy, M=22.8s, SD=6.9s). Participants were effective in finding the location of an element based on the value or index, as well as determining adjacent element values. However, some participants struggled in conditional aggregation (i.e., counting the number of elements greater than 10) across elements as they needed to maintain additional information (a counter) while iterating through the array.

Binary tree tasks revealed greater complexity in both performance and timing. While participants successfully identified parent-child relationships (T3, 100% accuracy), there was decreased performance for leaf node identification (T4, 87.5% accuracy) and BST property verification (T5, 62.5% accuracy). These complex tasks also required substantially more time (T4: M=39.8s, SD=39.5; T5: M=62.5s, SD=52.1). BST verification saw the sharpest drop because the task required participants to keep track of several structural relationships at once: tracking values, determining left–right positioning, and maintaining depth awareness.

The binary search application (T6) demonstrated high task accuracy, with participants achieving 100% accuracy across all search queries in moderate time (M=35.5s, SD=13.9). Our accuracy measure (AEI) reflects whether participants provided correct final answers, not whether they specifically employed binary search algorithms to reach those answers. Section 6.5 further examines participants' reasoning processes and strategies in detail to show how binary search was carried out in practice.

Confidence measures showed that participants felt highly confident with array understanding and navigation (M=4.88) but somewhat less confident throughout binary tree tasks (M=4.25-4.50). Participants expressed strong confidence in their binary search understanding (M=4.25-4.63). While confidence ratings alone do not equate to learning, when considered alongside participants' high task accuracy, they suggest that participants developed effective strategies for applying binary search.

While participants were required to use all representations in the initial learning phase, they could choose to use either a single representation or multiple representations when completing the tasks. With this option available, participants strongly preferred the tactile graphics over other representations throughout activities. Three participants used tactile representations exclusively for all tasks. Another three relied on tactile alone for at least three out of five tasks. Only one participant preferred a modality other than tactile: P6, who used tactile with navigable in T4, tactile alone in T5, and navigable alone in the remaining tasks. Overall, across the 40 task decisions (8 participants x 5 tasks), participants selected tactile 31 times, combined modalities 6 times (tabular + navigable + tactile in one case, tabular + tactile in three cases, and navigable + tactile in two cases), and non-tactile modalities only 3 times.

Participants consistently rated tactile graphics highest across all activities (M=5.0 for all tactile measures in Table 5), indicating strong overall preference for this representation. In contrast, ratings for tabular and navigable representations varied by task complexity. For arrays, both tabular and navigable representations received moderate ratings (M=4.25), but for binary trees, these ratings dropped substantially (tabular: M=3.38, SD=1.51; navigable: M=3.38, SD=0.52).

6.2 Representation Preferences & Affordances (RQ-Comprehension, RQ-Navigation)

The majority of participants preferred the tactile representations for all tasks. Some characterized their use of the (preferred) tactile representation as "cheating" or "shortcuts" because the representation made the tasks very easy to accomplish. Participants recognized the value of the representations, as each offered distinct affordances while imposing specific constraints on how participants could access and manipulate data structure information. As P4 put it, "I would be thrilled to only even have one."

The enthusiasm for the tactile representation reflected the intuitiveness of tactile access, across all data structures, especially for understanding structural and semantic details. As P1 explained, "the tactile was by far the best for the tree representation, it just made it intuitive to see or feel . . . how things were laid out." Participants emphasized that tactile graphics supported a fundamentally different kind of spatial understanding than digital alternatives. P6 described this distinction directly: "The answer to that is actually 5 because there's the spatial representation that's not just listening." P8 similarly described their preference for the tactile representation: "I definitely did like the tactile representations. I feel like I was able to really follow along . . . you know, literally at my fingertips." Others reinforced this enthusiasm with emphatic ratings, such as P2's declaration "Like a 10. Strongly agree" and P4's request to go beyond the scale: "Can I do a 5 1/2? I'll always go for tactile if it's available." Tactile graphics also demonstrated efficiency in navigation. Participants described being able to move fluidly across the representation using edge-following strategies and direct pathfinding, in contrast to slower, sequential digital approaches. While our navigable and tabular designs reduced some of the linearity of traditional ALT text, participants noted that they could not match the

Table 4: Quantitative measures of task performance. The row shown in *italics and red* is marked "T2" because it was excluded from analysis.

Task (Category)	Accuracy	Avg. Time (sec)	Std. Dev (sec)
T1 (Element Location)	91.67%	22.79	6.89
T2 (Order/Sorting)	*100%*	*2.13*	*1.55*
T3 (Parent/Child Identification)	100.00%	6.88	3.52
T4 (Leaf Node Identification)	87.50%	39.75	39.52
T5 (BST Property Check)	62.50%	62.50	52.09
T6 (Binary Search)	100.00%	35.54	13.87

Table 5: Likert-scale ratings (1-5 scale). Top scoring items begin with a * and scores are shown in bold (5). Items with scores below 4 or with standard deviation >= 1.00 begin with a - and scores are shown in *italic red*

Question	Avg. Rating	Std. Dev
I understand the overall structure of a array	4.88	0.35
I feel confident navigating an array using the representation(s) provided.	4.88	0.35
*The format(s) made it easy to locate specific nodes or values in the array	**5.00**	0.00
-The table made it easy to understand the array	4.25	*1.16*
The keyboard-friendly rep made it easy to understand the array	4.25	0.71
*The tactile graphic made it easy to understand the array	**5.00**	0.00
I understand the overall structure of a binary tree	4.50	0.93
I feel confident navigating a binary tree using the representation(s) provided	4.25	0.89
The format(s) made it easy to locate specific nodes or values in the binary tree	4.50	0.53
-The table made it easy to understand the binary tree.	*3.38*	*1.51*
-The keyboard-friendly rep made it easy to understand the binary tree.	*3.38*	0.52
*The tactile graphic made it easy to understand the binary tree.	**5.00**	0.00
I understood how binary search works after today's activities.	4.63	0.52
I could use binary search to solve a different problem or search a different tree if asked.	4.25	0.89
I would feel confident completing a new binary search task on my own.	4.13	0.83
-The table made it easy to complete binary search.	*2.75*	*1.04*
-The keyboard-friendly rep made it easy to complete binary search	*3.50*	0.53
*The tactile graphic made it easy to complete binary search	**5.00**	0.00

immediacy of tactile spatial access. P6 highlighted this difference bluntly: "This is where trying to solve it digitally is a stupid idea because it takes longer." This efficiency stemmed from the ability of tactile diagrams to preserve spatial layout and afford immediate, holistic access to structure.

Most significantly, tactile representations supported structural reasoning by making broader structural relationships perceptible at a glance (or touch). Rather than verifying properties node by node, participants could scan spatial layouts—such as left-to-right ordering of values—and detect if the BST property held. As P1 put it, "…the details, kind of, popped when I was going through it." This sense of details "popping" illustrates how maintaining semantic relationships in a tactile representation allowed participants to grasp structural properties rapidly and with confidence, directly linking tactile affordances to reasoning processes.

In contrast, performance of navigable representations was strongly connected to the specific data structure being explored. For example, some participants recognized the familiarity of the interaction model for tree structures but still found it difficult to understand hierarchical levels. P7 described this difficulty: "I have to figure out which one is the hierarchy first basically. Yeah. So that is the most difficult thing I work on." These struggles reflect the fundamental challenge of representing vertical hierarchy through screen reader-based tree interactions, where level awareness and local position is not always apparent. Participants appreciated the navigable list over the navigable tree because it mirrored the conceptual structure of arrays. P6 emphasized the scalability of this approach, explaining that "for definitely a larger number of elements one wants to have the keyboard friendly." However, they still preferred tactile representations and viewed the navigable list as a workable compromise rather than an optimal solution for supporting complex reasoning. P2's mixed assessment reflected this balance: "I'm going to [rate the list] 3 cause it's like I can see the value of it. But I don't think it's necessary [when the tactile format is available]"

The tabular representation was similarly helpful for arrays, as P4 describes: "The table to me is the clearest because I seem to understand things that way more easily." P1 describes how tabular representations preserved linear order while supporting multiple access pathways: "I can just follow through on either the values or the indices." Tables also helped participants organize information mentally. P2 explained, "In my head it keeps it organized better. So I can picture it." However, tables showed limitations with mapping the concept of a binary tree. P1 articulated this challenge, "I felt like the tabular version of the tree…the concepts didn't map onto

it very well. Because it really is just a 2D array, it doesn't behave like a tree."However, P1 still described them as acceptable fallback options: "For the binary tree .... If you don't have a choice and have to just do it in a linear format, the table did the job." P2 emphasized their conditional utility: "I think the table is of value ... once I would get familiar with the concepts and what I was looking for." Together, these reflections suggest that tables were well-suited for arrays but functioned only as partial substitutes for binary trees.

6.3 Multimodal Integration & Strategic Use (RQ-Confidence, RQ-Navigation)

While tactile representations overwhelmingly dominated task time (seven of eight participants used them for over 89% of recorded time), participants did not rely on tactile alone for all purposes. Three relied exclusively on the tactile format, while five supplemented it with at least one other representation. During the introduction of each data structure, participants often made comments such as "ahh that makes sense," reflecting how exploring all three representations deepened their understanding. Some participants noticed how once they used tactile, they understood the tabular and navigable better. Participants also switched between representations to support verification and discovered context-specific advantages that supported efficiency and comprehension. For examples, P3 first checked the tactile representation by scanning nodes for a value, then verified their answer in the tabular representation: "...I started with the tactile graphic, and I went by each [node]...for each one of [the nodes], I would check to find the value. And then I went back to the table and looked for the value."

Multi-modal verification. During guided learning, and occasionally task completion, participants valued alternative representations as a way to verify their answers. When being introduced to the binary tree navigable representation, P1 cross-checked details using the tactile representation. For example, they misidentified a node as the left child in the navigable representation, then reached over to the tactile representations to confirm: "I'm reaching over to check [tactile]...I want to check on the, uh...okay, no. 2 is the right child of one." P3 also expressed interest while learning the binary tree navigable representation to verify with the tactile graphic: "So what you might have seen. But after I looked at using the keyboard friendly, I then wanted to do a double check with the tactile graphic." We saw this strategy throughout the session, where participants would switch to an alternative representation to formalize and verify before confirming their answer.

Synergistic Effects. As P6 described, having access to multiple representations was especially important during the introduction and practice phase, when participants were still learning how to interpret them: "The graphical table one or Braille one, tactile one and then having the tree one and then OK, here's your table. Here's what to listen for. I would use those as like building blocks. Oh my God. It was essential." Moving between representations reinforced participants' understanding of both the data structures and the other modalities. As P2 explained, "So like I since they were building on each other and if I only had access to one [representation], once I've had the terminology down and what I was looking for, I think that either of these three would work." Access to multiple representations worked synergistically throughout the session, helping participants build data structure understanding.

Context-Dependent Use. A few participants, such as P5, saw switching between representations as data structure-specific; modality switching was beneficial in learning arrays, but not in binary trees: "OK, switching helped with the arrays, but then with the tree just confused me." P1 also noted that their preference might depend on what device they used: "if I was on a phone, I might prefer the list. It's just a simpler thing for the array, at least...versus the table, you know, if I...If my finger's a quarter inch off, it's gonna be saying that value instead of the index, or something like that." These reflections underscore the need for accessible representation design to be cognizant of how individuals might use them in varying contexts.

6.4 Learning Processes & Mental Model Development (RQ-Comprehension, RQ-Application)

Participants developing understanding of data structures revealed complex cognitive processes involving mental model construction, strategic adaptation to barriers, and sophisticated metacognitive awareness. This theme examines the mechanisms through which participants overcame representational challenges and built understanding across data structures. These approaches revealed how participants actively construct meaning when working with accessible representations.

Mental Model Construction. Participants developed mental models of data structures by combining multiple accessible representations. Mental model construction was most effective when participants integrated insights across representations. P2 described this synergistic process, highlighting the critical importance of the tactile representation: "I felt like maybe the table allowed me to have a little bit of a picture. And then like it was just great to have the picture in the tactile form. So that just kind of really put it together." Participants emphasized that tactile graphics provided an initial anchor that strengthened their understanding across modalities. As P8 described, if they began with "either the table or the keyboard representation," the tactile version "kind of amplified my understanding... put an image in my head... so I could follow along," ultimately showing how the tactile representation made the digital representations easier to interpret. This sequential building process, where initial understanding from one representation provided foundation for deeper comprehension through another, demonstrates how multimodal access supports mental model development. P3 characterized this as having "a smaller way of presenting the picture, therefore you had something to build upon," highlighting how each representation contributed scaffolding for more complete understanding.

Analogical Reasoning. Participants often connected data structures to familiar real-world experiences to make abstract concepts concrete. P1 used economic metaphors to understand algorithmic efficiency and the value of the binary search algorithm: "If you imagine those as safety deposit boxes, and you actually have to open them, and it takes some cost to opening up each one...the binary tree is clearly better." P4 drew on everyday technology to

situate their understanding of the tree navigation, reminding themselves: "Wait a minute. This is just like you're in File Explorer." P3 described hierarchical levels in generational terms: "I'm thinking the grandparent…the mom, and then the child underneath." These analogies revealed how participants actively connected abstract computational concepts to familiar experiences and structures, making them more intuitive to navigate.

Metacognitive strategies. Metacognitive strategies proved central to participants' success, involving both active self-monitoring and strategic management of their own cognitive resources. P2 demonstrated explicit reasoning checks: "So that would meet that [BST property]. Would the only one on the left not meet [the BST property]. That would be the one that goes from [node] 2 to [node] 3. Am I thinking of that correctly or not?" In addition, participants showed metacognitive awareness of their own memory limitations, as described by P4: "My brain works…I can remember a column-line correlation better than I can remember as I'm moving through something, better than I can remember the other type of correlation." Beyond these moment-to-moment checks, participants also engaged in metacognitive reflection about whether their use of a representation aligned with the intended learning objectives. "If I'm using this table like this, then I'm really just looking at it as an array. I'm not using the tree…I feel like that spoils the objective of learning about it as a tree." This illustrates how participants not only monitored their reasoning but also evaluated whether their strategy preserved the conceptual integrity of the task.

6.5 Conceptual Understanding & Algorithm Application

Across activities, participants demonstrated varying degrees of conceptual understanding and algorithm application. Their approaches highlighted how prior experience, representational affordances, and terminology shaped both comprehension and problem-solving.

Impact of Prior CS Experience on Understanding. Participants' backgrounds in computer science shaped how they approached the representations. While participants' prior computing experience provided some scaffolding—as P7 noted, "I'm already familiar with array structure because I work on computers"—this familiarity didn't automatically transfer to representation-specific skills. In fact, P7 reflected on how ease of understanding might differ for those without such exposure: "So people who would not be familiar with this array structure…I'm not sure how easy it would be, but you know, at least to me it is easy." While no participants were previously familiar with binary trees, all were familiar with trees as a concept and as a navigation pattern, having encountered them in screen reader contexts. Terminology gaps also impacted understanding, as P5 noted: "I don't know how to explain the relationship between the elements, like the first column header." These were particularly salient when participants had partial understanding but lacked precise language to describe relationships.

Semantics. Participants developed strategies for identifying and understanding semantic roles across different data structures. When working with tree structures, participants demonstrated clear comprehension of hierarchical relationships, as P3 explained: "On the tactile format, having the representation with the actual lines that showed the parent to child, and it delineated those." Identifying leaf nodes illustrated how participants applied semantic concepts in problem-solving, as shown by P2: "There's only three [leaf nodes]. Because I don't have any lines going from there." This conceptual grasp showed value during algorithmic reasoning, as evidenced when P6 performed binary search: "We've got 3. So 5 is greater than 3. So we gotta go down the right one and then we're up at 6. So 5 is less than six. It's going to be down here. Instead of four, it's a leaf, so there's no 5." This example shows how semantic understanding of leaf nodes informed decision-making. However, participants sometimes encountered conceptual challenges when working to establish semantic clarity. P1 noted how certain formats required additional cognitive effort to distinguish semantic roles: "For the list, just having each one of them be a separate entity, like both the index and the value, I didn't think it was as obvious." These moments reveal the active cognitive work participants engaged in to construct semantic understanding across different representations.

Structural Understanding. Conceptual understanding also depended on participants' ability to reason about structure—how elements were positioned and connected in ways that supported algorithmic procedures. As GP3 emphasizes, this requires access to topological properties such as order, hierarchy, adjacency, and connectivity. For example, P4 reflected on how tree traversal required awareness of depth: "…you're down here somewhere at like the fourth level in the tree, and somebody's looking for something that's over here in the third level of the tree, not on a different branch…" Tactile cues helped with this, as P1 explained: "Alright, I'm on 5…6…ought to be down and to the right, and it is. And one step took me there." Reasoning also involved reconciling different formats. For example, P1 expected arrays to be laid out vertically, yet tactile versions were often horizontal: "The tactile one is horizontal, whereas…the ones on the screen are vertical. I don't know if that's actually how they're visually laid out."

Algorithmic Reasoning. Participants demonstrated algorithmic reasoning across representations, applying comparative rules and checking structural constraints. P3 explicitly checked BST properties: "So let me…before we do that, it has to have. The left child is going to be smaller than the parent. And if the right…if there is a right child, that child's going to be larger than the parent." Both novice and experienced participants executed binary search algorithms through systematic rule application. P7 demonstrated methodical traversal logic: "Zero is less than 5, so go to the left. Node 0 is less than three. Go to the left, node 0 is less than two. Go to the left…and zero is less than one, but there's no left node there after that." Such careful narrations show how participants internalized the comparative logic of binary search while maintaining awareness of structural constraints like leaf nodes. Verbalization also served as an important strategy for scaffolding reasoning and checking errors.

Participants further developed nuanced awareness of how different representations shaped algorithmic reasoning. Some formats naturally encouraged systematic approaches, as P1 observed: "I did, at least with the electronic version of the tree form…it does kind of push me to use the algorithm so I don't just end up reading every value." Others recognized when representational constraints conflicted with algorithmic logic. P7 noted: "But if I wanna apply

binary search, tree view wouldn't work very well because you have to figure which one is left node, which one is right node…I go back and forth and it keeps interrupting my workflow."

6.6 Accessible Education Implications

Access to high-quality instruction and accessible materials can fundamentally shape learning trajectories, while persistent systemic barriers limit equitable participation in computing education. Although we did not explicitly ask participants about their broader educational histories, this theme emerged through our thematic analysis. Participants reflected on the role that accessible representations played—or could have played—in shaping their educational experiences and highlighted the potential impact of Arboretum.

Participants had varied access to tactile graphics in the context of CS education. P1 recalled that: "Having good computer science instruction…was so valuable to keeping me interested…I was very lucky in high school computer science, I did have tactile graphics to represent these concepts in my textbook." This combination of effective pedagogy and accessible materials sparked sustained interest and self-directed learning that extended far beyond formal coursework. Other participants reflected on how their educational experiences might have differed with better access to accessible representations. P4 expressed regret about missed opportunities: "I wish I'd had this kind of stuff when I took that one class in computer programming." P1 connected this need to self-directed learning: "My training past high school…was self-taught, and so having these representations in some of those textbooks would have been great." These experiences shaped long-term engagement, enabling self-directed learning that extended beyond formal instruction. Participants with this prior exposure also articulated a clear sense of tactile's limits: tactile graphics were foundational for understanding structure, but often arrived slowly or were impractical to produce at scale. As P1 explained, they would still "prefer one of the electronic versions over waiting… to get a hard copy," highlighting a pragmatic shift toward digital formats once a tactile mental model was firmly in place.

These individual experiences reflect broader institutional challenges. P6 described how educators regularly encounter these gaps: "This one programmer list I was on, one of the things that happens is they'll be a teacher who's like, I've got a blind student in my class. How do I do data structures?" Even for well-intentioned instructors, there is a lack of appropriate resources and training. There are also logistical challenges of relying on tactile graphics, as P1 described: "I think people are gonna be…in a lot of cases, stuck with the electronic one if they're remote and they don't have the tools or the agency they're working with doesn't have the resources to mail things." Participants who primarily used magnification software or digital tools similarly emphasized that electronic formats were often more reliable in day-to-day coursework, reinforcing the need for accessible digital modalities alongside tactile ones.

P6 highlighted that, "…not everybody has access to like a Braille display or you know [other assistive technologies]." Also, even when technologies were available, participants noted that screen reader settings and conventions sometimes complicated access. P1 explained, "I have mine in advanced mode, so it only says the number. For the level, it doesn't actually say level." In this case, P1 was describing how customized screen reader verbosity settings change the output: instead of announcing "level 2," the screen reader only announced "2." While efficient for experienced users, this shorthand could obscure structural cues (e.g., hierarchy depth in a tree) that are critical for navigating educational representations.

Participants also noted that accessibility barriers extended beyond blindness, reflecting broader educational challenges. One participant with a hearing disability emphasized the limits of relying solely on auditory information: "Because of the hearing disability, solely relying on speech is not necessarily giving me the most accurate information, so having the tactile again gives me greater independence and ability to manipulate the information." Similarly, low-vision participants described how magnification tools often created inefficiencies: "It's more efficient to not work visually, 'cause at my level of magnification it's off the right side of the screen" (P6). Together, these accounts highlight the importance of designing representations that are not only conceptually clear but also broadly usable across disabilities and contexts.

7 DISCUSSION

Traditional accessibility approaches treat data structure diagrams as visual artifacts to be described, rather than as structured information whose semantics—relationships, ordering, and roles—must be made directly accessible. As a result, standards such as WCAG's "complex graphics" guidance [75] often reinforce surface-level translations of visual diagrams instead of exposing the computational properties that nonvisual learners rely on.

We discuss our findings into two areas. First, we connect our findings to the four foundational principles of WCAG—that a document is *Perceivable*, *Operable*, *Understandable*, and *Robust*. In doing so, we formalize what accessible, structure-first representations must provide. Second, we discuss implications for scaling accessible diagram systems, focusing on how educators can generate, adapt, and integrate representations in practice.

7.1 Design Principles

In this section, we consolidate the five design requirements (**DR1**–**DR5**) into four broader design principles that define what structure-aligned accessibility demands. These principles build on WCAG's perceivable, operable, understandable, and robust (POUR) framework [75], but extend it to contexts where the underlying structure—not the visual layout—is the primary object of reasoning. These principles offer a foundation for accessible representations across data structures and other diagram types.

DP1. Perceivable - Explicit Semantic Roles and Relationships. The *Perceivable* principle from the WCAG framework requires that all relevant information be present in a modality that users can receive [75]. Prior accessibility work shows that explicit semantic structure supports efficient nonvisual navigation [22, 26]. However, past work is premised on an assumption that an element's role, grouping, and position are fixed—an assumption that breaks down in many diagrams. Even when these properties are stable, their interpretation is often ambiguous without additional semantic context, as a single element can occupy multiple functional roles at once. For example, a node may simultaneously be the left child of its parent, the parent of its own subtree, and a leaf candidate, with

these identities shifting as the structure is traversed or modified. When semantic roles are multi-role and dynamic, relationships are often implicit, multi-functional, and dependent on spatial context.

Our Wizard-of-Oz study indicated that no single representation provided enough information for participants to determine these relationships on its own (**DR3**). Participants faced difficulties identifying "which nodes were attached to which edges," reflecting that the representations presented the elements but did not always make their structural roles clear. Our second study showed a similar pattern. Within verifying the BST property, participants often had to reconsider a node's role at each step to determine whether it satisfied the property rules. When the representation listed a node's parent, children, or value explicitly participants used this information to guide their reasoning through analogies that also reflected dynamic roles, such as describing a node as a "grandparent, mom, and child underneath."

DP2. Operable - Structure-Aligned Navigation. Navigation is central to accessible content consumption. WCAG's *Operable* principle states that users must be able to navigate using a their preferred access technology [75]. Efficient nonvisual navigation benefits from familiar interaction patterns (**DR1**), and prior research has demonstrated that screen reader navigation improves efficiency and orientation within coding environments [4, 56, 58] and accessing visualizations [7, 71, 87]. However, screen reader interactions typically rely on tree-based navigation models [7], which are often incompatible with relational, spatial, diagrammatic, or geographic structures [23, 49]. When underlying structure diverges from a simple tree, familiar interactions such as WAI-ARIA tree navigation, break down—leaving many visualization types unsupported [23]. This mismatch has motivated the development of new interaction paradigms for graphs and networks, such as Benthic [49], that support navigation across both hierarchical and adjacent relationships.

Although accessible visualization and interface designers often begin with standard interactions set by WAI-ARIA, these interaction patterns do not always correspond to the structural relationships that data structures express. Across both studies, participants encountered cases where navigation behavior implied one structural meaning while the diagram required another. For example, the right-arrow key, conventionally interpreted as "move right" or "expand", navigated to a node's left child due to the collapse/expand interactions in trees. Participants described this as confusing because interaction semantics (e.g., arrow keys as spatial movement) conflicted with the intended traversal semantics. For data structures with multiple meaningful traversal orders, a single navigation paradigm is unlikely to support all reasoning tasks. A structure-aligned approach should therefore expose more than one traversal logic when appropriate. For instance, enabling a user to move through a tree using depth-first or breadth-first logic, depending on the reasoning task at hand.

DP3. Understandable - Supporting Structural Reasoning. WCAG's *Understandable* principle emphasizes clear and comprehensible content. Prior work on auditory information seeking [85] and code navigation [4, 56, 58] shows that nonvisual learners rely on multiple access strategies—gists, overviews, orientation cues, and contextual retrieval—to keep users oriented and reason effectively. This work also identifies a recurring tradeoff: interfaces that prioritize faster movement can make it harder to perceive underlying structural relationships, whereas more structured designs may slow navigation but clarify how elements relate [4, 49, 58]. Structural reasoning requires understanding how elements in a data structure relate (*e.g.,* parent–child links, left–right orientation, depth, adjacency) and applying those relationships to verify data structure properties or simulate algorithms.

Our Wizard-of-Oz study revealed that BVI learners can only maintain positional awareness—and thus reason accurately about these relationships—when overviews are clearly separated from structural detail (**DR4**). When data structures are presented as diagrams, they encode computational properties that ALT text alone cannot convey, as screen readers typically only access them as static, non-interactive text. Tasks such as verifying BST properties require simultaneously tracking a node's value, its left and right children, and its depth—relationships that are difficult to reconstruct without explicit positional cues. In the second study, several participants preferred tactile graphics for tree-based tasks because the tactile layouts made positional relationships easier to interpret, whereas some digital formats made those cues harder to identify. Accordingly, accessible representations should preserve explicit structural information even if doing so increases navigational effort. Explicit context helps users understand how a data structure is organized. In practice, this includes consistent spatial layouts in tactile forms, explicit positional cues in digital representations (e.g., "left child of," "parent of," rather than only "connected to"), and maintaining a clear separation between structural specification and explanatory commentary.

DP4. Robust - Consistency Across Modalities. WCAG's *Robust* principle emphasizes that content should remain consistent and reliable across assistive technologies, rather than imposing a specific access modality (or not supporting any). Most prior work on accessible data structures has largely focused on *translation*—converting visual diagrams directly into tactile graphics through improved spacing, labeling, or layout standards [17]. Similar patterns appear in visualization accessibility, where modalities are typically treated as one-to-one conversions (*e.g.,*visual-to-audio, visual-to-tactile) [39, 81]. These approaches improve access, but they inherit the assumptions and constraints of the visual source because each new modality begins with the diagram's appearance rather than its underlying structure.

Our first study showed that individual prototypes conveyed different subsets of the structure (**DR5**). In the second study, several participants moved deliberately across representations—checking spatial relationships in tactile form, comparing them to digital navigation, and using each modality to clarify what another left uncertain. The representations did not need to be identical; rather, learners benefited when modalities expressed the same underlying relationships, even if said relationships were presented differently; participants emphasized that the representations "built on each other," which helped them reconcile gaps across formats. Our findings suggest that the central issue is not consistency across modalities by itself, but the starting point for translation. When translation begins from a visual diagram, each modality can diverge slightly because it interprets the visual layout differently. Accessible representations can be more robust when they originate from a

structural description (*i.e.* a diagram specification or intermediate representation) rather than the visual form. A structure-first model ensures that all modalities draw from the same semantics, reducing opportunities for divergence as new formats are introduced.

7.2 Designing for Scalable, Instructor-Led Accessibility

A persistent gap in accessible CS education is that most prior work addresses learner-facing accessibility—supporting BVI students directly—without equipping educators to author accessible materials themselves [8, 36, 41, 42, 62]. This divide is increasingly consequential as a 2024 Department of Justice rule about Title II of the Americans with Disabilities Act places institutional responsibility on instructors to deliver accessible content at scale [74], despite limited expertise among instructors [10]. Arboretum lowers adoption barriers and treats accessibility as an extension of existing curricular practices. This design choice—meeting educators where they are—suggests a broader opportunity: accessibility systems that leverage disciplinary conventions can make accessible authoring feel seamless.

Arboretum's design promotes consistent data structure diagrams across large instructional teams by providing a unified interface for learners, regardless of the diagram specification instructors use. Because instructors and teaching assistants often create bespoke diagrams with disparate tools, variability in structure and accessibility is common. Standardizing output representations is therefore essential, as existing tools rarely preserve the computational properties of data structures or support sufficient accessible alternatives. By relying on familiar specification languages, Arboretum also offers a low barrier to entry: instructors and TAs who know these languages can begin authoring diagrams immediately and adapt the tool to their workflows with minimal overhead. Arboretum also supports rapid authoring in ad-hoc settings such as office hours, where instructors can generate examples on the fly—for instance, creating a custom binary tree to illustrate the BST property for a student.

Our multimodal output strategy further supports scalability across learner needs and institutional contexts. By generating multiple representations from a single source, Arboretum can support novices—who can access the representations in combination to form a strong mental model—and more advanced learners who can efficiently utilize the representation that best suits them once spatial relationships become internalized.

Arboretum's design highlights broader principles: scalable accessibility in CS education emerges not from perfecting single-modality representations, but from systems that (1) minimize the instructional workload required to create accessible content and (2) support learners across expertise levels through synchronized multimodal representations.

7.3 Limitations

While Arboretum demonstrates the feasibility of generating multiple accessible representations of data structures, several limitations point to directions for future work. Our evaluation was necessarily introductory, focusing on foundational concepts rather than complex algorithmic tasks. Future studies should examine how these principles extend to advanced data structures such as linked lists, graphs, and multidimensional arrays, as well as to more sophisticated reasoning tasks. In addition, our remote study design limited direct observation of tactile interactions, underscoring the value of in-person studies that can capture the nuanced ways learners engage with physical representations.

8 FUTURE WORK AND CONCLUSIONS

Diagrams are foundational to computing education, but they remain difficult to access nonvisually because accessibility practices are often grounded in visual renderings rather than the underlying structural information in the diagram. Our work reframes this challenge by identifying the loss of computational properties—such as roles, relationships, and ordering— as a key barrier when diagrams are presented as static images or text descriptions. Through two studies, we examined where nonvisual representations fall short and derived design principles that specify what structure-aligned accessibility should provide. Arboretum operationalizes these principles by starting from a diagram specification rather than a visual form. By generating synchronized tabular, navigable, and tactile representations from a shared structural model, it supports educators in automatically generating multimodal representations for their data structure diagrams. Because all formats draw from the same underlying model, updates made once can be reflected across every representation, supporting more consistent materials and making it feasible to create, maintain, and update accessible diagrams at the pace required for classes and instructional workflows.

Looking forward, our principles suggest opportunities to extend structure-aligned accessibility beyond data structures. Many visually-intensive domains with complex graphics (*e.g.*, graphs, process diagrams, scientific visualizations) share the same challenges of implicit semantics, spatial reasoning, and dependence on visual layout. Future work can explore how structural model authoring can support multimodal access across these domains.

Ultimately, accessible learning at scale requires tools that integrate accessibility into instructional design itself, rather than layering accommodations on top of visual materials. By grounding each representation in a shared structural model rather than the visual form of the diagram, Arboretum demonstrates how accessibility can be made systematic, scalable, and integral to computing education.

ACKNOWLEDGMENTS

We thank Ather Sharif, Sean Mealin, and Richard E. Ladner for formative contributions to this work, and Gaby de Jongh and Scott Ferguson at the University of Washington's Access Technology Center for their guidance. We also thank the UW CSE 12X instructors, teaching assistants, and students who motivated this research. This work was supported in part by the NSF BPC Alliance Access-Computing (2137312), the Paul G. Allen School of Computer Science & Engineering, and a Google PhD Fellowship, as well as a grant from the National Institute on Disability, Independent Living, and Rehabilitation Research (NIDILRR grant number 90REGE0026-01-00) funding the Center for Research and Education on Accessible Technology and Experiences (CREATE). NIDILRR is a Center within the Administration for Community Living (ACL), Department of

Health and Human Services (HHS). The contents of this work do not necessarily represent the policy of NIDILRR, ACL, HHS, and you should not assume endorsement by the Federal Government.

REFERENCES

[1] AccessComputing. 2025. AccessComputing. https://www.washington.edu/accesscomputing/

[2] Sara Alzalabny, Omar Moured, Karin Müller, Thorsten Schwarz, Bastian E. Rapp, and Rainer Stiefelhagen. 2024. Touch for Accessibility: Haptic SVG Diagrams for Visually Impaired and Blind Individuals. In IEEE Haptics Symposium, HAPTICS 2024, Long Beach, CA, USA, April 7-10, 2024. IEEE, 79–84. https://doi.org/10.1109/HAPTICS59260.2024.10520858

[3] Catherine M. Baker, Cynthia L. Bennett, and Richard E. Ladner. 2019. Educational Experiences of Blind Programmers. In Proceedings of the 50th ACM Technical Symposium on Computer Science Education, SIGCSE 2019, Minneapolis, MN, USA, February 27 - March 02, 2019, Elizabeth K. Hawthorne, Manuel A. Pérez-Quiñones, Sarah Heckman, and Jian Zhang (Eds.). ACM, 759–765. https://doi.org/10.1145/3287324.3287410

[4] Catherine M. Baker, Lauren R. Milne, and Richard E. Ladner. 2015. StructJumper: A Tool to Help Blind Programmers Navigate and Understand the Structure of Code. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (Seoul, Republic of Korea) (CHI '15). Association for Computing Machinery, New York, NY, USA, 3043–3052. https://doi.org/10.1145/2702123.2702589

[5] Suzanne Balik, Sean P. Mealin, Matthias F. Stallmann, and Robert D. Rodman. 2013. GSK: universally accessible graph sketching. In The 44th ACM Technical Symposium on Computer Science Education, SIGCSE 2013, Denver, CO, USA, March 6-9, 2013, Tracy Camp, Paul T. Tymann, J. D. Dougherty, and Kris Nagel (Eds.). ACM, 221–226. https://doi.org/10.1145/2445196.2445266

[6] Suzanne P. Balik, Sean P. Mealin, Matthias F. Stallmann, and Robert D. Rodman. 2013. GSK: universally accessible graph sketching. In Proceeding of the 44th ACM Technical Symposium on Computer Science Education (Denver, Colorado, USA) (SIGCSE '13). Association for Computing Machinery, New York, NY, USA, 221–226. https://doi.org/10.1145/2445196.2445266

[7] Matt Blanco, Jonathan Zong, and Arvind Satyanarayan. 2022. Olli: An Extensible Visualization Library for Screen Reader Accessibility. In IEEE VIS Posters. https://vis.csail.mit.edu/pubs/olli

[8] Brianna Blaser, Richard E. Ladner, and Sheryl Burgstahler. 2018. Including Disability in Diversity. In 2018 Research on Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT). 1–4. https://doi.org/10.1109/RESPECT.2018.8491717

[9] Paul Blenkhorn and D. Gareth Evans. 1998. Using speech and touch to enable blind people to access schematic diagrams. J. Netw. Comput. Appl. 21, 1 (1998), 17–29. https://doi.org/10.1006/JNCA.1998.0060

[10] Way Kiat Bong and Weiqin Chen. 2024. Increasing faculty's competence in digital accessibility for inclusive education: a systematic literature review. International Journal of Inclusive Education 28, 2 (2024), 197–213.

[11] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. Qualitative research in psychology 3, 2 (2006), 77–101.

[12] Robert G Brookshire. 2006. Teaching uml database modeling to visually impaired students. Issues in Information Systems 7, 1 (2006), 98–101.

[13] Matt Calder, Robert F. Cohen, Jessica A. Lanzoni, and Yun Xu. 2006. PLUMB:: an interface for users who are blind to display, create, and modify graphs. In Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS 2006, Portland, Oregon, USA, October 23-25, 2006, Simeon Keates and Simon Harper (Eds.). ACM, 263–264. https://doi.org/10.1145/1168987.1169046

[14] Mauro Cherubini, Gina Venolia, Robert DeLine, and Amy J. Ko. 2007. Let's go to the whiteboard: how and why software developers use drawings. In Proceedings of the 2007 Conference on Human Factors in Computing Systems, CHI 2007, San Jose, California, USA, April 28 - May 3, 2007, Mary Beth Rosson and David J. Gilmore (Eds.). ACM, 557–566. https://doi.org/10.1145/1240624.1240714

[15] Robert F. Cohen, Arthur Meacham, and Joelle Skaff. 2006. Teaching graphs to visually impaired students using an active auditory interface. In Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education (Houston, Texas, USA) (SIGCSE '06). Association for Computing Machinery, New York, NY, USA, 279–282. https://doi.org/10.1145/1121341.1121428

[16] Robert F. Cohen, Rui Yu, Arthur Meacham, and Joelle Skaff. 2005. PLUMB: displaying graphs to the blind using an active auditory interface. In Proceedings of the ACM SIGACCESS Conference on Computers and Accessibility, ASSETS 2005, Baltimore, MD, USA, October 9-12, 2005, Andrew Sears and Enrico Pontelli (Eds.). ACM, 182–183. https://doi.org/10.1145/1090785.1090820

[17] April R. Crockett and Gerald C. Gannod. 2020. Improving Understanding of Data Structures for the Blind with Tactile Media and a User-Centered Iterative Approach. In IEEE Frontiers in Education Conference, FIE 2020, Uppsala, Sweden, October 21-24, 2020. IEEE, 1–8. https://doi.org/10.1109/FIE44824.2020.9273978

[18] Charlie Cross, Deniz Cetinkaya, and Huseyin Dogan. 2020. Transforming Diagrams' Semantics to Text for Visually Impaired. In Design, User Experience, and Usability. Interaction Design - 9th International Conference, DUXU 2020, Held as Part of the 22nd HCI International Conference, HCII 2020, Copenhagen, Denmark, July 19-24, 2020, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 12200), Aaron Marcus and Elizabeth Rosenzweig (Eds.). Springer, 339–350. https://doi.org/10.1007/978-3-030-49713-2_24

[19] Nils Dahlbäck, Arne Jönsson, and Lars Ahrenberg. 1993. Wizard of Oz studies: why and how. In Proceedings of the 1st international conference on Intelligent user interfaces. 193–200.

[20] Brad Doherty and Betty H. C. Cheng. 2015. UML Modeling for Visually-Impaired Persons. In Proceedings of the First International Workshop on Human Factors in Modeling co-located with ACM/IEEE 18th International Conference on Model Driven Engineering Languages and Systems (MoDELS 2015), Ottawa, Canada, September 28, 2015 (CEUR Workshop Proceedings, Vol. 1522), Harald Störrle, Vasco Amaral, Michel Chaudron, and Miguel Goulão (Eds.). CEUR-WS.org, 4–10. https://ceur-ws.org/Vol-1522/Doherty2015HuFaMo.pdf

[21] Md Ehtesham-Ul-Haque and Syed Masum Billah. 2025. ToPSen: Task-Oriented Priming and Sensory Alignment for Comparing Coding Strategies Between Sighted and Blind Programmers. In Proceedings of the 2025 ACM Designing Interactive Systems Conference (DIS '25). Association for Computing Machinery, New York, NY, USA, 2103–2116. https://doi.org/10.1145/3715336.3735839

[22] Md Ehtesham-Ul-Haque, Syed Mostofa Monsur, and Syed Masum Billah. 2022. Grid-Coding: An Accessible, Efficient, and Structured Coding Paradigm for Blind and Low-Vision Programmers. In Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology (Bend, OR, USA) (UIST '22). Association for Computing Machinery, New York, NY, USA, Article 44, 21 pages. https://doi.org/10.1145/3526113.3545620

[23] Frank Elavsky, Lucas Nadolskis, and Dominik Moritz. 2024. Data Navigator: An Accessibility-Centered Data Navigation Toolkit. IEEE Trans. Vis. Comput. Graph. 30, 1 (2024), 803–813. https://doi.org/10.1109/TVCG.2023.3327393

[24] Christin Engel and Gerhard Weber. 2018. A User Study to Evaluate Tactile Charts with Blind and Visually Impaired People. In Computers Helping People with Special Needs - 16th International Conference, ICCHP 2018, Linz, Austria, July 11-13, 2018, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 10897), Klaus Miesenberger and Georgios Kouroupetroglou (Eds.). Springer, 177–184. https://doi.org/10.1007/978-3-319-94274-2_24

[25] Daqing Fan, Katya Glazko, and Steven Follmer. 2022. Accessibility of Linked-Node Diagrams on Collaborative Whiteboards for Screen Reader Users: Challenges and Opportunities. In Design Thinking Research: Understanding Innovation, Christoph Meinel and Larry Leifer (Eds.). Springer, Cham, 95–110. https://doi.org/10.1007/978-3-031-09297-8_6

[26] António Ramires Fernandes, Alexandre Carvalho, J. J. Almeida, and Alberto Simões. 2006. Transcoding for Web Accessibility for the Blind: Semantics from Structure. In Proceedings of the International Conference on Electronic Publishing (ELPUB 2006), Bob Martens and Milena Dobreva (Eds.). Bansko, Bulgaria, 123–134. https://hdl.handle.net/1822/5448

[27] Sally Fincher, Johan Jeuring, Craig S. Miller, Peter Donaldson, Benedict du Boulay, Matthias Hauswirth, Arto Hellas, Felienne Hermans, Colleen M. Lewis, Andreas Mühling, Janice L. Pearce, and Andrew Petersen. 2020. Notional Machines in Computing Education: The Education of Attention. In Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education, ITiCSE-WGR 2020, Trondheim, Norway, June 15-19, 2020, Michail N. Giannakos, Guttorm Sindre, Andrew Luxton-Reilly, and Monica Divitini (Eds.). ACM, 21–50. https://doi.org/10.1145/3437800.3439202

[28] Eric Fouh, Monika Akbar, and Clifford A. Shaffer. 2012. The Role of Visualization in Computer Science Education. Computers in the Schools 29, 1–2 (2012), 95–117. https://doi.org/10.1080/07380569.2012.651422

[29] John A. Gardner. 1996. Tactile graphics: an overview and resource guide. Information Technology and Disabilities 3 (1996). https://api.semanticscholar.org/CorpusID:58659489

[30] Cagatay Goncu, Kim Marriott, and John Hurst. 2010. Usability of Accessible Bar Charts. In Diagrammatic Representation and Inference, 6th International Conference, Diagrams 2010, Portland, OR, USA, August 9-11, 2010. Proceedings (Lecture Notes in Computer Science, Vol. 6170), Ashok K. Goel, Mateja Jamnik, and N. Hari Narayanan (Eds.). Springer, 167–181. https://doi.org/10.1007/978-3-642-14600-8_17

[31] Devamardeep Hayatpur, Daniel Wigdor, and Haijun Xia. 2023. CrossCode: Multi-level Visualization of Program Execution. In Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, CHI 2023, Hamburg, Germany, April 23-28, 2023, Albrecht Schmidt, Kaisa Väänänen, Tesh Goyal, Per Ola Kristensson, Anicia Peters, Stefanie Mueller, Julie R. Williamson, and Max L. Wilson (Eds.). ACM, 593:1–593:13. https://doi.org/10.1145/3544548.3581390

[32] Thomas Hermann, Andy Hunt, and John G. Neuhoff. 2011. The Sonification Handbook. Logos Publishing House. 586 pages. https://sonification.de/handbook/

[33] Md. Naimul Hoque, Md Ehtesham-Ul-Haque, Niklas Elmqvist, and Syed Masum Billah. 2023. Accessible Data Representation with Natural Sound. In Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, CHI 2023, Hamburg, Germany, April 23-28, 2023, Albrecht Schmidt, Kaisa Väänänen, Tesh Goyal, Per Ola Kristensson, Anicia Peters, Stefanie Mueller, Julie R. Williamson, and Max L. Wilson (Eds.). ACM, 826:1–826:19. https://doi.org/10.1145/3544548.3581087

[34] Mirko Horstmann, Martin Lorenz, A. Watkowski, George T. Ioannidis, Otthein Herzog, Alasdair King, D. Gareth Evans, Cornelius Hagen, Christoph Schlieder, Anne-Marie Burn, Neil King, Helen Petrie, Sijo Dijkstra, and David Crombie. 2004. Automated interpretation and accessible presentation of technical diagrams for blind people. New Rev. Hypermedia Multim. 10, 2 (2004), 141–163. https://doi.org/10.1080/13614560512331326017

[35] Earl W. Huff Jr., Kwajo Boateng, Makayla Moster, Paige Rodeghero, and Julian Brinkley. 2021. Exploring the Perspectives of Teachers of the Visually Impaired Regarding Accessible K12 Computing Education. In SIGCSE '21: The 52nd ACM Technical Symposium on Computer Science Education, Virtual Event, USA, March 13-20, 2021, Mark Sherriff, Laurence D. Merkle, Pamela A. Cutter, Alvaro E. Monge, and Judithe Sheard (Eds.). ACM, 156–162. https://doi.org/10.1145/3408877.3432418

[36] Ritesh Kanchi, Miya Natsuhara, and Matt X Wang. 2025. Systems for Scaling Accessibility Efforts in Large Computing Courses. arXiv preprint arXiv:2510.25964 (2025). https://doi.org/10.1145/3770762.3772648 arXiv:2510.25964 [cs.CY]

[37] Mateusz Kawulok, Stella Mackowska, Michal Mackowski, and Dominik Spinczyk. 2025. Reducing Blind Students' Learning Effort with an Audio-Tactile Approach to Understanding Basic Computer Algorithms. In Proceedings of the 30th ACM Conference on Innovation and Technology in Computer Science Education V. 1, ITiCSE 2025, Nijmegen, The Netherlands, 27 June 2025 - 2 July 2025, Erik Barendsen, Floor Binkhorst, J. Ángel Velázquez-Iturbide, Jaime Urquiza-Fuentes, James Paterson, and Keith Quille (Eds.). ACM, 472–478. https://doi.org/10.1145/3724363.3729098

[38] Andrea R. Kennel. 1996. Audiograf: A Diagram-Reader for the Blind. In Proceedings of the Second Annual ACM Conference on Assistive Technologies, ASSETS 1996, Vancouver, BC, Canada, April 11-12, 1996, Ephraim P. Glinert and David L. Jaffe (Eds.). ACM, 51–56. https://doi.org/10.1145/228347.228357

[39] Nam Wook Kim, Samantha C. Joyner, Alexandra Riegelhuth, and Younghoon Kim. 2021. Accessible Visualization: Design Space, Opportunities, and Challenges. Computer Graphics Forum 40, 3 (2021), 173–188. https://doi.org/10.1111/cgf.14298

[40] Alasdair King, Paul Blenkhorn, David Crombie, Sijo Dijkstra, Gareth Evans, and John Wood. 2004. Presenting UML Software Engineering Diagrams to Blind People. In Computers Helping People with Special Needs, 9th International Conference, ICCHP 2004, Paris, France, July 7-9, 2004, Proceedings (Lecture Notes in Computer Science, Vol. 3118), Joachim Klaus, Klaus Miesenberger, Wolfgang L. Zagler, and Dominique Burger (Eds.). Springer, 522–529. https://doi.org/10.1007/978-3-540-27817-7_76

[41] Richard E. Ladner and Sheryl Burgstahler. 2015. Increasing the participation of individuals with disabilities in computing. Commun. ACM 58, 12 (2015), 33–36. https://doi.org/10.1145/2835961

[42] Richard E. Ladner and Maya Israel. 2016. "For all" in "computer science for all". Commun. ACM 59, 9 (2016), 26–28. https://doi.org/10.1145/2971329

[43] Jill H. Larkin and Herbert A. Simon. 1987. Why a Diagram is (Sometimes) Worth Ten Thousand Words. Cogn. Sci. 11, 1 (1987), 65–100. https://doi.org/10.1111/J.1551-6708.1987.TB00863.X

[44] Jonathan Lazar, Jinjuan Feng, and Harry Hochheiser. 2017. Research Methods in Human-Computer Interaction, 2nd Edition. Morgan Kaufmann. https://www.sciencedirect.com/science/book/9780128053904

[45] Henry R. Lewis and Larry Denenberg. 1991. Data Structures and Their Algorithms. HarperCollins Publishers.

[46] Leandro Luque, Leônidas de Oliveira Brandão, Elisabeti Kira, and Anarosa Alves Franco Brandão. 2017. Inclusion in computing and engineering education: Perceptions and learning in diagram-based e-learning classes with blind and sighted learners. In 2017 IEEE Frontiers in Education Conference, FIE 2017, Indianapolis, IN, USA, October 18-21, 2017. IEEE Computer Society, 1–8. https://doi.org/10.1109/FIE.2017.8190513

[47] L Luque, E d S Veriscimo, G d C Pereira, and LVL Filgueiras. 2014. Can we work together? on the inclusion of blind people in uml model-based tasks. In Inclusive Designing, Patrick Langdon, Jonathan Lazar, Ann Heylighen, and Hua Dong (Eds.). Springer, Cham. https://doi.org/10.1007/978-3-319-05095-9_20

[48] Jennifer Mankoff and Kelly A. Mack. 2025. Teaching Accessible Computing. Bookish, Chapter Foundations: Teaching Inclusively. https://bookish.press/tac/TeachingInclusively Retrieved July 1, 2025.

[49] Catherine Mei*, Josh Pollock*, Daniel Hajas, Jonathan Zong, and Arvind Satyanarayan. 2025. Benthic: Perceptually Congruent Structures for Accessible Charts and Diagrams. In Proceedings of the 27th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '25). Association for Computing Machinery, New York, NY, USA, Article 27, 17 pages. https://doi.org/10.1145/3663547.3746342

[50] Aboubakar Mountapmbeme, Obianuju Okafor, and Stephanie Ludi. 2022. Addressing Accessibility Barriers in Programming for People with Visual Impairments: A Literature Review. ACM Trans. Access. Comput. 15, 1 (2022), 7:1–7:26. https://doi.org/10.1145/3507469

[51] Helen Petrie, Neil King, Anne-Marie Burn, and Peter Pavan. 2006. Providing interactive access to architectural floorplans for blind people. British journal of visual impairment 24, 1 (2006), 4–11.

[52] Helen Petrie, Christoph Schlieder, Paul Blenkhorn, Gareth Evans, Alasdair King, Anne-Marie O'Neill, George T. Ioannidis, Blaithin Gallagher, David Crombie, Rolf Mager, and Maurizio Alafaci. 2002. TeDUB: A System for Presenting and Exploring Technical Drawings for Blind People. In Computers Helping People with Special Needs, 8th International Conference, ICCHP 2002, Linz, Austria, July 15-20, Proceedings (Lecture Notes in Computer Science, Vol. 2398), Klaus Miesenberger, Joachim Klaus, and Wolfgang L. Zagler (Eds.). Springer, 537–539. https://doi.org/10.1007/3-540-45491-8_102

[53] Josh Pollock, Catherine Mei, Grace Huang, Elliot Evans, Daniel Jackson, and Arvind Satyanarayan. 2024. Bluefish: Composing Diagrams with Declarative Relations. In Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology (Pittsburgh, PA, USA) (UIST '24). Association for Computing Machinery, New York, NY, USA, Article 23, 21 pages. https://doi.org/10.1145/3654777.3676465

[54] Stephan Pölzer and Klaus Miesenberger. 2014. A Tactile Presentation Method of Mind Maps in Co-located Meetings. In Proceedings of the International Workshop on Tactile/Haptic User Interfaces for Tabletops and Tablets, held in conjunction with ACM ITS 2014, Dresden, Germany, November 16, 2014 (CEUR Workshop Proceedings, Vol. 1324), Limin Zeng and Gerhard Weber (Eds.). CEUR-WS.org. https://ceur-ws.org/Vol-1324/paper_8.pdf

[55] Stephan Pölzer, Dirk Schnelle-Walka, Daniel Pöll, Peter Heumader, and Klaus Miesenberger. 2013. Making brainstorming meetings accessible for blind users. In Assistive Technology: From Research to Practice. IOS Press, 653–658.

[56] Venkatesh Potluri, Priyan Vaithilingam, Suresh Iyengar, Y. Vidya, Manohar Swaminathan, and Gopal Srinivasa. 2018. CodeTalk: Improving Programming Environment Accessibility for Visually Impaired Developers. In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (Montreal QC, Canada) (CHI '18). Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3173574.3174192

[57] Denise Prescher, Jens Bornschein, and Gerhard Weber. 2014. Production of Accessible Tactile Graphics. In Computers Helping People with Special Needs (Lecture Notes in Computer Science, Vol. 8548), Klaus Miesenberger, Deborah Fels, Dominique Archambault, Petr Peňáz, and Wolfgang L. Zagler (Eds.). Springer, Cham. https://doi.org/10.1007/978-3-319-08599-9_5

[58] Emmanuel Schanzer, Sina Bahram, and Shriram Krishnamurthi. 2019. Accessible AST-Based Programming for Visually-Impaired Programmers. In Proceedings of the 50th ACM Technical Symposium on Computer Science Education (Minneapolis, MN, USA) (SIGCSE '19). Association for Computing Machinery, New York, NY, USA, 773–779. https://doi.org/10.1145/3287324.3287499

[59] Raymond Scupin. 1997. The KJ method: A technique for analyzing data derived from Japanese ethnology. Human organization 56, 2 (1997), 233–237.

[60] Ather Sharif, Olivia H. Wang, Alida T. Muongchan, Katharina Reinecke, and Jacob O. Wobbrock. 2022. VoxLens: Making Online Data Visualizations Accessible with an Interactive JavaScript Plug-In. In CHI '22: CHI Conference on Human Factors in Computing Systems, New Orleans, LA, USA, 29 April 2022 - 5 May 2022, Simone D. J. Barbosa, Cliff Lampe, Caroline Appert, David A. Shamma, Steven Mark Drucker, Julie R. Williamson, and Koji Yatani (Eds.). ACM, 478:1–478:19. https://doi.org/10.1145/3491102.3517431

[61] Ann C. Smith, Joan M. Francioni, and Sam D. Matzek. 2000. A Java programming tool for students with visual disabilities. In Proceedings of the Fourth International ACM Conference on Assistive Technologies (Arlington, Virginia, USA) (Assets '00). Association for Computing Machinery, New York, NY, USA, 142–148. https://doi.org/10.1145/354324.354356

[62] Melinda R. Snodgrass, Maya Israel, and George C. Reese. 2016. Instructional Supports for Students with Disabilities in K–5 Computing: Findings from a Cross-Case Analysis. Computers & Education 100 (2016), 1–17. https://doi.org/10.1016/j.compedu.2016.04.011

[63] Abigale Stangl, Ann Cunningham, Lou Ann Blake, and Tom Yeh. 2019. Defining Problems of Practices to Advance Inclusive Tactile Media Consumption and Production. In The 21st International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS 2019, Pittsburgh, PA, USA, October 28-30, 2019, Jeffrey P. Bigham, Shiri Azenkot, and Shaun K. Kane (Eds.). ACM, 329–341. https://doi.org/10.1145/3308561.3353778

[64] Andreas Stefik, Willliam Allee, Gabriel Contreras, Timothy Kluthe, Alex Hoffman, Brianna Blaser, and Richard Ladner. 2024. Accessible to Whom? Bringing Accessibility to Blocks. In Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1 (Portland, OR, USA) (SIGCSE 2024). Association for Computing Machinery, New York, NY, USA, 1286–1292. https://doi.org/10.1145/3626252.3630770

[65] Andreas Stefik and Ed Gellenbeck. 2011. Empirical studies on programming language stimuli. Software Quality Journal 19, 1 (March 2011), 65–99. https://doi.org/10.1007/s11219-010-9106-7

[66] Andreas Stefik, Christopher D. Hundhausen, and Derrick W. Smith. 2011. On the design of an educational infrastructure for the blind and visually impaired in computer science. In Proceedings of the 42nd ACM technical symposium on Computer science education, SIGCSE 2011, Dallas, TX, USA, March 9-12, 2011, Thomas J. Cortina, Ellen Lowenfeld Walker, Laurie A. Smith King, and David R. Musicant (Eds.). ACM, 571–576. https://doi.org/10.1145/1953163.1953323

[67] Andreas Stefik and Richard Ladner. 2017. The Quorum Programming Language (Abstract Only). In Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education (Seattle, Washington, USA) (SIGCSE '17). Association for Computing Machinery, New York, NY, USA, 641. https://doi.org/10.1145/3017680.3022377

[68] Andreas Stefik, Richard E. Ladner, William Allee, and Sean Mealin. 2019. Computer Science Principles for Teachers of Blind and Visually Impaired Students. In Proceedings of the 50th ACM Technical Symposium on Computer Science Education, SIGCSE 2019, Minneapolis, MN, USA, February 27 - March 02, 2019, Elizabeth K. Hawthorne, Manuel A. Pérez-Quiñones, Sarah Heckman, and Jian Zhang (Eds.). ACM, 766–772. https://doi.org/10.1145/3287324.3287453

[69] Poonam Syal, Shantanu Chatterji, and Harish Kumar Sardana. 2016. DiGVis: A system for comprehension and creation of directed graphs for the visually challenged. Univers. Access Inf. Soc. 15, 2 (2016), 199–217. https://doi.org/10.1007/S10209-014-0387-7

[70] TeachAccess. 2025. TeachAccess. https://www.teachaccess.org/

[71] John R. Thompson, Jesse J. Martinez, Alper Sarikaya, Edward Cutrell, and Bongshin Lee. 2023. Chart Reader: Accessible Visualization Experiences Designed with Screen Reader Users. In Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, CHI 2023, Hamburg, Germany, April 23-28, 2023, Albrecht Schmidt, Kaisa Väänänen, Tesh Goyal, Per Ola Kristensson, Anicia Peters, Stefanie Mueller, Julie R. Williamson, and Max L. Wilson (Eds.). ACM, 802:1–802:18. https://doi.org/10.1145/3544548.3581186

[72] Christine D Tippett. 2016. What recent research on diagrams suggests about learning with rather than learning from visual representations in science. International Journal of Science Education 38, 5 (2016), 725–746.

[73] Márcio Josué Ramos Torres and Regina Barwaldt. 2019. Approaches for diagrams accessibility for blind people: a systematic review. In IEEE Frontiers in Education Conference, FIE 2019, Cincinnati, OH, USA, October 16-19, 2019. IEEE, 1–7. https://doi.org/10.1109/FIE43999.2019.9028522

[74] U.S. Department of Justice. 2024. Nondiscrimination on the Basis of Disability; Accessibility of Web Information and Services of State and Local Government Entities. https://www.ada.gov/assets/pdfs/web-rule.pdf. Final rule implementing Title II of the ADA.

[75] W3C. 2018. Web Content Accessibility Guidelines (WCAG) 2.1. https://www.w3.org/TR/WCAG21/. W3C Recommendation.

[76] W3C. 2021. Accessible Rich Internet Applications (WAI-ARIA) Overview. https://www.w3.org/WAI/standards-guidelines/aria/. W3C Web Accessibility Initiative.

[77] Yanan Wang, Arjun Srinivasan, and Yea-Seul Kim. 2024. Enabling Tabular Data Exploration for Blind and Low-Vision Users. In Proceedings of the 2024 ACM Designing Interactive Systems Conference (Copenhagen, Denmark) (DIS '24). Association for Computing Machinery, New York, NY, USA, 1218–1233. https://doi.org/10.1145/3643834.3661609

[78] Tetsuya Watanabe and Hikaru Mizukami. 2018. Effectiveness of Tactile Scatter Plots: Comparison of Non-visual Data Representations. In Computers Helping People with Special Needs - 16th International Conference, ICCHP 2018, Linz, Austria, July 11-13, 2018, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 10896), Klaus Miesenberger and Georgios Kouroupetroglou (Eds.). Springer, 628–635. https://doi.org/10.1007/978-3-319-94277-3_97

[79] Tetsuya Watanabe, Toshimitsu Yamaguchi, and Masaki Nakagawa. 2012. Development of Software for Automatic Creation of Embossed Graphs - Comparison of Non-visual Data Presentation Methods and Development Up-to-Date. In Computers Helping People with Special Needs - 13th International Conference, ICCHP 2012, Linz, Austria, July 11-13, 2012, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 7382), Klaus Miesenberger, Arthur I. Karshmer, Petr Penáz, and Wolfgang L. Zagler (Eds.). Springer, 174–181. https://doi.org/10.1007/978-3-642-31522-0_25

[80] Fabian Wildhaber, Nadim Salloum, Marcel Gygli, and Andrea Kennel. 2020. Self-Directed Creation and Editing of UML Class Diagrams on Mobile Devices for Visually Impaired People. In 10th IEEE International Model-Driven Requirements Engineering, MoDRE@RE 2020, Zurich, Switzerland, August 31, 2020. IEEE, 49–57. https://doi.org/10.1109/MODRE51215.2020.00012

[81] Brianna Lynn Wimer, Laura South, Keke Wu, Danielle Albers Szafir, Michelle A. Borkin, and Ronald A. Metoyer. 2024. Beyond Vision Impairments: Redefining the Scope of Accessible Data Representations. IEEE Transactions on Visualization and Computer Graphics 30, 12 (2024), 7619–7636. https://doi.org/10.1109/TVCG.2024.3356566

[82] Yalong Yang, Kim Marriott, Matthew Butler, Cagatay Goncu, and Leona Holloway. 2020. Tactile Presentation of Network Data: Text, Matrix or Diagram?. In CHI '20: CHI Conference on Human Factors in Computing Systems, Honolulu, HI, USA, April 25-30, 2020, Regina Bernhaupt, Florian 'Floyd' Mueller, David Verweij, Josh Andres, Joanna McGrenere, Andy Cockburn, Ignacio Avellino, Alix Goguey, Pernille Bjøn, Shengdong Zhao, Briane Paul Samson, and Rafal Kocielnik (Eds.). ACM, 1–12. https://doi.org/10.1145/3313831.3376367

[83] Koji Yatani, Eunyoung Chung, Carlos Jensen, and Khai N. Truong. 2009. Understanding how and why open source contributors use diagrams in the development of Ubuntu. In Proceedings of the 27th International Conference on Human Factors in Computing Systems, CHI 2009, Boston, MA, USA, April 4-9, 2009, Dan R. Olsen Jr., Richard B. Arthur, Ken Hinckley, Meredith Ringel Morris, Scott E. Hudson, and Saul Greenberg (Eds.). ACM, 995–1004. https://doi.org/10.1145/1518701.1518853

[84] Wai Yu and Stephen A. Brewster. 2002. Multimodal virtual reality versus printed medium in visualization for blind people. In Proceedings of the ACM Conference on Assistive Technologies, ASSETS 2002, Edinburgh, Scotland, UK, July 8-10, 2002, Vicki L. Hanson and Julie A. Jacko (Eds.). ACM, 57–64. https://doi.org/10.1145/638249.638261

[85] Haixia Zhao, Catherine Plaisant, Ben Shneiderman, and Ramani Duraiswami. 2004. Sonification of Geo-Referenced Data for Auditory Information Seeking: Design Principle and Pilot Study.

[86] Yichun Zhao, Miguel A. Nacenta, Mahadeo A. Sukhai, and Sowmya Somanath. 2024. TADA: Making Node-link Diagrams Accessible to Blind and Low-Vision People. In Proceedings of the CHI Conference on Human Factors in Computing Systems, CHI 2024, Honolulu, HI, USA, May 11-16, 2024, Florian 'Floyd' Mueller, Penny Kyburz, Julie R. Williamson, Corina Sas, Max L. Wilson, Phoebe O. Toups Dugas, and Irina Shklovski (Eds.). ACM, 45:1–45:20. https://doi.org/10.1145/3613904.3642222

[87] Jonathan Zong, Crystal Lee, Alan Lundgard, JiWoong Jang, Daniel Hajas, and Arvind Satyanarayan. 2022. Rich Screen Reader Experiences for Accessible Data Visualization. Comput. Graph. Forum 41, 3, 15–27. https://doi.org/10.1111/CGF.14519