**Tactile Programming – Exploratory Design Evaluation Outline**

**Spring – Summer, 2021**

*Research Questions*

**RQ1**: Are there any usability issues that impede students of mixed visual ability from reading code?

**RQ2**: Are there any usability issues that impede students of mixed visual ability from writing code?

**RQ3**: How do students of mixed visual ability trace programs?

**RQ4**: How do students of mixed visual ability create programs?

*Participants*

Target population characteristics:

* **Age**: 18 – 23 *(Typical age range of college students)*
* **Major(s)/ Background**: Computer Science
* **Programming Experience**: Participants must identify as having programming experience in some form *(No strict definition used here)*
* **Visual Ability**: Sighted *(It should be noted that, roughly speaking, a single degree of sightedness should be considered, as this may impact the exploratory/ scanning abilities demonstrated by participants; sightedness controlled using blindfold)*

*Data Capture and Analysis*

Collection strategies:

* **Video**
  + Camera positioned directly above programming device, facing downward onto device surface.
  + Capture participant hand movements and exploratory motions.
  + No audio capture necessary.
  + Annotations
  + QR code hand tracker
* **Compiled programs**
  + Store raw block code, compilation success or failure flag, and error message (if any)
    - Per-participant entry created following compilation

*Tracing Programs*

Working “Trace” Definition: Any physical, verbal, or visual way of describing program execution.

Tracing Sequences

* **Purpose**
  + Examine ability of users to interpret basic statements.
  + Consider how line continuation may impact readability.
    - Examine continuous assignment statement (all on one line) and statement that is broken across the two lines.
* Performance test:
  + Provided completed code, ask participants to describe the program execution
    - Level 1: “Assign 7 to V1”
    - Level 2: “Assign 2 to V1. Assign sum of 3 and V1 to V2”
* Time Estimate: ~ 5 minutes

Tracing Branching Logic

* **Purpose**:
  + Examine ability of participants to interpret branch conditions
  + Examine ability of participants to determine which branch will execute given a condition to be evaluated
* Performance test
  + Provided completed code, ask participants to describe the program execution
    - Level 1: “Assign 3 to V1. If V1 greater than 4, assign 8 to V1. Else, assign 5 to V1.”
    - Level 2: “Assign 6 to V1. If V1 less than 4, … Else if V1 less than 7, … Else, …”
* Time Estimate: ~ 5 minutes

Tracing Loops

* **Purpose**:
  + Examine ability of participants to interpret loop conditions
  + Examine ability of participants to trace looping over several iterations
* Performance test
  + Provided completed code, ask participants to describe the program execution
    - Level 1: “Assign 1 to V1. Loop 5 times, incrementing V1 by 1 each time”
    - Level 2: “Assign 4 to V1. Loop 8 times, incrementing V1 by 2 each time”
* Time Estimate: ~ 5 minutes

Tracing to Debug

* **Purpose**:
  + Examine ability of participants to find and fix program bugs, both syntactic and semantic
  + Sheds light on helpfulness of auditory feedback (error messages) and the exploratory strategies used by
* NO performance test here, just something to observe/ take note of
  + Specifically, pay special attention to annotate comments/ strategies used by participants to fix problems, also any issues that impede problem solving

*Creating Programs*

Create sequence

* **Purpose**:
  + Examine sequence-building patterns of students with mixed visual ability
* Performance test
  + Provided necessary blocks and high-level description, ask participants to write code

Create branching logic

* **Purpose**:
  + Examine branching-logic-building patterns of students with mixed visual ability
* Performance test
  + Provided necessary blocks and high-level description, ask participants to write code

Create loop

* **Purpose**:
  + Examine loop building patterns of students with mixed visual ability
* Performance test
  + Provided necessary blocks and high-level description, ask participants to write code

*User Evaluation*

• Evaluate characteristics of interest for tactile learning tools (refer to Inclusive Education article)

• Request that participants complete the evaluation following all performance testing

|  |  |  |
| --- | --- | --- |
| **Perception** | **Question** | **Description** |
| Joy | How fun was the programming environment? | Low values boring, high values fun |
| Weariness | How tired did you feel when using the programming environment? | Low values tired, high values comfortable |
| Usefulness | How useful did you find the programming environment? | Low values useless, high values useful |
| Accessibility | How accessible did you find the programming environment? | Low values inaccessible, high values accessible |
| Learning | Is the programming environment a good learning tool? | Low values considered bad learning tool, high values considered good learning tool |
| Potential Use | Do you think this system would have been helpful for you when first learning about topics such as instruction sequences, branching logic, and looping? | Low values indicate would like to use system to learn about basic programming, high values indicate would like to use system to learn about basic programming |
| Didactic | How didactic (learning with pleasure and entertainment) was the programming environment? | Low values indicate stultifying (stifle enthusiasm, freedom), high values indicate system conveyed meaning and pleasure |
| Easiness | Did you find the programming environment easy to use? | Low values indicate difficult to use system, high values indicate easy to use system |

*References*

• [Torino](https://www.tandfonline.com/doi/pdf/10.1080/07370024.2018.1512413?casa_token=PJHrWtlTyTAAAAAA:fEpqx3yobWuH0z9fhcaMrFUuh8H7TBZO2tHx2C1I3kxpzaJ8J851cAsXv85Fjp_jVQi4LGFsLy-S7A)

• Virtual Haptic Perception as an Educational Assistive Technology: A Case Study in Inclusive Education