Research paper-(review)

1.1 Challenges in Learning Recursion

- Different approaches to learning recursion:
    - Lack of real-world examples forces learning from code examples, which is unnatural for human thinking.
    - Recursion takes many forms, making it hard to recognize patterns.
    - Debugging recursion requires recursion, adding confusion.
- Learning with previous programming experience:
    - Prior experience in iterative languages like BASIC can negatively transfer. Students make errors like using "goto", unnecessary nesting, etc.
- Learning with incorrect preconceptions:
    - Solving problems can reinforce misconceptions. Students need proper introduction before hands-on practice.
- Learning with algorithmic approach:
    - Comparing recursion and iteration helps, but has a learning curve. Students' comprehension improved over time.
- Misconception about base cases:
    - Common issues include ignoring base cases, using them incorrectly, and redundant base cases. Proper understanding is essential.

1.2 Assessment dimensions

- Assess comprehension: Describe concepts
- Evaluate: Trace recursive code flows
- Construction: Write recursive solutions

2.1 Overview of the tool

- Interactive interface to draw shapes using recursion in Elm
- User Interface:
    - Canvas to draw shapes
    - Buttons to zoom, enable challenges, copy code
    - Code panel to adjust variables
    - Menu of shape challenges
- Recursion Shapes:
    - Base shapes provided for each challenge
    - Increasing complexity from circles to coral
    - Observe recursion levels through shapes

2.2 Code Base

- Written in Elm using MVU (Model View Update) architecture
- Core Types:
    - StackFrame - recursive structure representing code
    - ElmExpr - wraps Elm types like Int to control UI
    - ElmShape - represents shapes and drawing logic
    - ControlState - manages state of adjustable UI controls
- Key Modules:
    - Render - renders shapes and code UI
    - Pretty - renders adjustable controls
    - Copiable - renders copyable code
    - AdjustConst - updates values after adjustments
    - HelperFunctions - reusable helper code
- MVU:
    - Model stores state like StackFrame
    - View renders UI using Render and Pretty
    - Update dispatches messages to AdjustConst to update Model

The Recursion Creator Tool provides an interactive Elm-based interface to draw recursive shapes. The key components include the recursive UI, ElmExpr wrappers, Render/Pretty modules, and MVU architecture.

3.1 Curriculum

- 3-day camp for students with Elm experience
- Review Elm Programming:
    - Covered syntax, functions, if/else
    - Example mapping steps to make pie to a function
- Introduce Recursion:
    - Recursive bedtime story example
    - Explain recursive structure via "divide-conquer-combine"
- Recursion shape challenges:
    - Students draw shapes like circles, coral using tool
    - Observe recursion levels in shapes
    - Adjust variables to change recursive behavior
- Assessments:
    - Students write recursive Elm solutions to problems
    - Peer review each other's code
    - Check for understanding of concepts

The camp curriculum leveraged the Recursion Creator tool to teach recursion interactively. It focused on Elm review, recursive stories, drawing challenges, and assessments of student code. The interactive tool enabled students to gain intuition about recursion by manipulating recursive shape drawings.

The Conclusion summarizes how the Recursion Creator tool developed in the report helps students learn recursion concepts:

- Learning recursion is challenging due to lack of real-world examples, difficulty tracing/debugging, and misconceptions about base cases.
- The Recursion Creator tool aims to address these challenges by:
- Providing intuitive recursive shape examples from the physical world
- Enabling tracing/debugging by adjusting variables and observing changes
- Illustrating the base case through the base shapes
- This helps students gain intuition about recursion by manipulating and visualizing recursive code.
- The curriculum and assessments in the camp build experience with writing recursive code.
- Future work is suggested:
- Develop assessments to evaluate learning outcomes quantitatively
- Conduct more research to add evidence-based features
- Fix bugs and upgrade the tool to a recursive maze game

In summary, the Recursion Creator tool helps students learn challenging recursion concepts by providing interactive visualization. The camp curriculum and assessments build intuition and experience. Future work will enhance the tool and learning process further.