# CodeHelp

## A Debugging Tool for Novice Programmers

Katie Hancock and Stefan D’Souza

# Context

- Novice programmers struggle with debugging their code for various reasons:
  - Syntax errors
  - Logic errors
  - Runtime errors
- Debugging can be frustrating and time-consuming for beginners: According to our survey, participants spend an average of 5.3 hours per week debugging their code (34% of total programming time)
- Many beginners give up on programming altogether because they feel overwhelmed and discouraged by the debugging process

# Our Goal

- We aim to make their coding journey smoother and more enjoyable
- Our goal is to provide:
  - Clear explanations
  - Efficient solutions
  - Helpful feedback
- Save users time and effort by automating some of the debugging tasks and providing efficient solutions
- Alleviate anxiety and make the learning process smoother and more enjoyable

# Our Solution

- CodeHelp is a Python script that can be run directly from the VSCode Terminal
- Utilizes LLM technology in order to analyze user code and produce insights
- CodeHelp can:
  - Detect and explain errors in the code
  - Optimize the code for improved performance
  - Explain the code in simple terminology

# Usability Research

- Usability tests were conducted to evaluate effectiveness and usability of CodeHelp
- Participants included novice programmers with varying levels of experience
- All participants said they appreciated the clear explanations and efficient solutions provided by CodeHelp, and that they would recommend the tool to other novice programmers

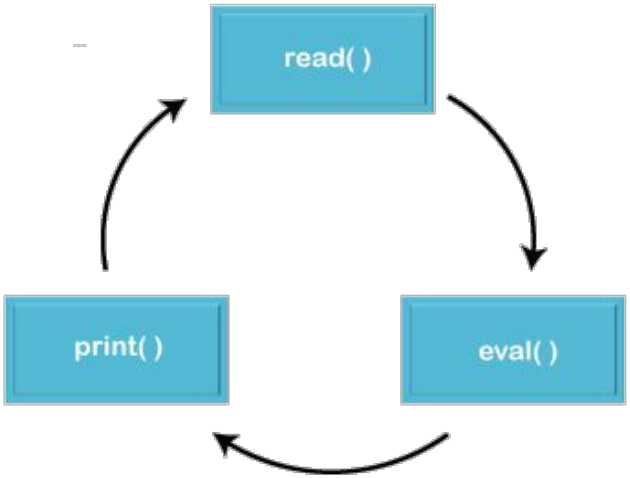
"Novice programmers could definitely use this to debug, especially in languages like Python that might not explicitly give you an error..." -Participant 1

"The explanation seems pretty helpful... I could probably just implement this directly" -Participant 2

# Implementation and Logistics

# How does CodeHelp work

- Python script run from the terminal
- Collects input file, attempts running it, and aggregates error logs
- Instantiates REPL loop with three key features:
  - Debug
  - Optimize
  - Explain
- Language-agnostic




Read-Eval-Print Loop (REPL)

# Debug Feature

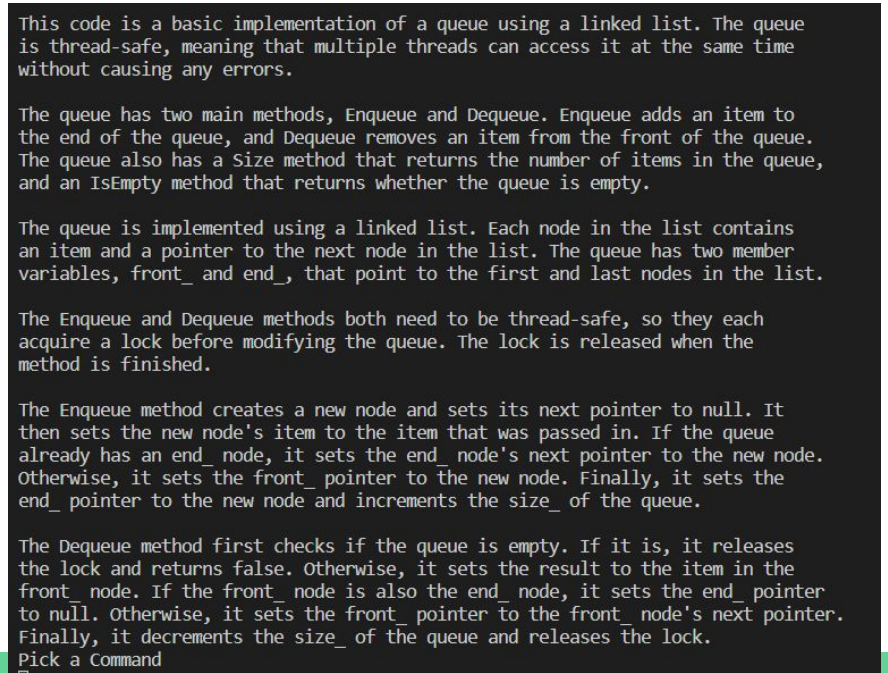
- Targeted at non-functional code
- Scans stderr in conjunction with file contents

```
The issue with the code is that it contains an unclosed string literal.
This is causing the code to suggest that it is missing a curly brace after the Run() method.
This can be solved by closing the string found on line 33
```

Example of an unclosed string literal in Java

# Explanation Feature

- Optimized for functional code
- Explanatory feature for programmers unfamiliar with the code base they're using



# Optimization Feature

- Works on functional and non-functional code
- Either serves as:
  - Optimization of working code
  - Suggested methodologies to change identified errors in non-functional code

```
# The code provided can be optimized by adding a check to see if the denominator is 0 before attempting the division.

def divide_numbers(numerator, denominator):
    if denominator == 0:
        print("Cannot divide by zero!")
        return
    return numerator / denominator

numerator = 10
denominator = 0
```

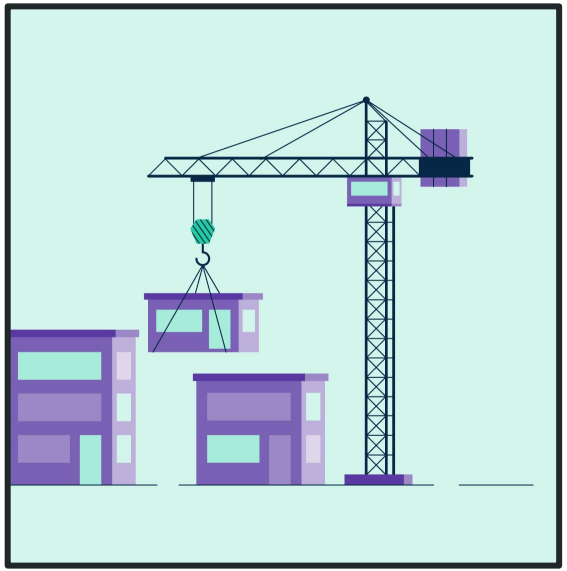
Example of an zero-division error in python

# Addressing Hallucinations

- We focus on preventing users from being negatively impacted by hallucinations
  - Avoiding time wastage is a priority
- Mechanical direct results when possible
- Fail fast when writing code
- Lowered focus on writing code blocks directly
- Soon to come - Integrated test suites!

# Implementation and Scalability

- Highly Scalable model - in a corporate setting we recommend releasing in waves in order to catch usability issues.
- Easy to access - currently a download but soon to be pip-installable
- Modular system - easy to track and apply usage.
    - Track usage through api keys- sharable or personal
- For novice programmers, our targeted installation method is pip install:
    - Currently working on testing a set-up user flow for inputting api keys



# Next Steps

- Conversational Query model
- Refactoring System
  - Integrating Test Suite
  - Autolinter
  - Leakcheck
- Test Generation

# So why use CodeHelp?

- Highly Scalable and immediately installable
- Convenient and un-intrusive interface
- Repl system - unique for most tools
    - More intuitive for novice programmers - allows for user to have incorrect entries and prompts for possible options
    - Flag based tools such as valgrind may be intuitive for other users
- Fast failing to mitigate hallucinations

# Thank you for your attention!