Large Language Models (LLMs) are AI-based tools that can understand and reproduce human language. LLMs can generate more accurate responses as they are trained on additional data, making them more popular. These models often work in a chat-based prompt-response system where the LLM produces an output in response to the user's input. Our project is focused on investigating the LLMs' ability to produce high-level language code given a prompt in natural language that describes the code intent. We evaluated the LLMs' ability to generate correct code as many people are currently using these tools for code synthesis by using the following cycle:

**Objective & Impact of Professor's Research**

Professor Mukund Raghothaman's research is mainly concerned with the inner workings of computer programming languages. Using machine learning and formal verification techniques, Professor Raghothaman works to solve various problems in the field, like improving program synthesis methods and fault localization.

- Machine Learning is an extremely popular field of computer science, concerning teaching a computer how to do a task well by feeding it large amounts of data.
- Formal Verification is determining whether a program completes a job by verifying if it passes a set of mathematical benchmarks.

**Introduction**

Larger Language Models (LLMs) are AI-based tools that can understand and reproduce human language. LLMs can generate more accurate responses as they are trained on additional data, making them more popular. These models often work in a chat-based prompt-response system where the LLM produces an output in response to the user's input. Our project is focused on investigating the LLMs' ability to produce high-level language code given a prompt in natural language that describes the code intent. We evaluated the LLMs' ability to generate correct code as many people are currently using these tools for code synthesis by using the following cycle:

**Methods & Results**

**Procedures**

1. Gather prompts from human-eval-x, a dataset containing original handmade programming questions publicly available LLMs are unlikely to be trained on.
2. Using test cases provided in the human-eval dataset evaluate the results of code.
3. Phase 1: Apply prompts to LLM and gather generated code. Using test cases provided in the human-eval dataset, evaluate the results of code.

**Results**

- 122 Passed, 42 Failed
- 18 Passed, 24 Failed
- 9 Passed, 15 Failed

**Conclusion**

Large Language Models are quickly becoming large parts of people's lives across the globe. With that increase of use, more transparency about the accuracy of such models is important. With regard to code generation it is evident that Chat GPT is extremely successful, despite requiring assistance with some more challenging problems. Our research could be expanded by evaluating various other Large Language Models using a different category of questions other than code synthesis.

**Citations**

Yetiştiren Burak... "Evaluating the Code Quality of AI-Assisted Code Generation Tools: An Empirical Study on GitHub Copilot, Amazon CodeWhisperer, and ChatGPT". The authors of this paper used a dataset of natural language questions other than code synthesis.

**Acknowledgments**

We have to give thanks to Professor Mukund Raghothaman and Matin Amini and Sara Baradaran, our Ph.D. student mentors. We would also like to thank the SHINE team, Jayron, Marcus and Monica for helping us along the way outside of the project.