1 Introduction

This document serves as a guide to understanding a Haskell code implementation for parsing and calculating the limit of a polynomial function at a specified point. The provided code utilizes Haskell's functional programming paradigm and leverages the `Text.ParserCombinators.Parsec` library for parsing expressions.

The code consists of three primary functions:

1. **Parsing Polynomials:** The `parsePolynomial` function takes a string representation of a polynomial as input and parses it into a structured form for further manipulation. It utilizes the `polynomialParser` function, which, in turn, utilizes `termParser` to parse individual terms within the polynomial expression.

2. **Calculating Limits:** The `calculateLimit` function computes the limit of a polynomial function for a given value of x. It evaluates the polynomial expression at the specified point x by summing the contributions from each term in the polynomial.

3. **Main Function:** The `main` function orchestrates the interaction with the user. It prompts the user to input a polynomial expression, parses it, prompts for the value of x, and then calculates and displays the limit of the polynomial function at that x value.

This document will provide a detailed explanation of each component of the code, including how polynomials are parsed, terms are extracted, and limits are calculated. Additionally, it will cover the usage of Haskell's monadic parsing constructs and functional programming techniques employed in the implementation.

By the end of this document, readers will have a comprehensive understanding of how the provided Haskell code parses polynomial expressions and calculates their limits, empowering them to utilize and extend the functionality as needed.

... In process ...