Simple Compiler Project

A compiler implementation for a basic programming language that supports integer, float, and character types along with control flow structures.

Features

- **Type System**
  - Integer (int)
  - Float (float)
  - Character (char)
- **Operations**
  - Arithmetic: `+`, `-`, `*`, `/`
  - Comparison: `>`, `<`, `>=`, `<=`, `==`, `!=`
  - Assignment: `=`
- **Control Flow**
  - If-else statements
  - While loops
  - Block scoping with `{}`
- **Other Features**
  - Variable declarations with initialization
  - Print statements
  - Comments using `#`
  - Character escape sequences (`\n`, `\t`)
  - Division by zero protection

PROF

Project Structure

- `lexer.l` - Flex lexical analyzer
- `parser.y` - Bison parser grammar
- `ast.h/c` - Abstract Syntax Tree implementation
- `symbol_table.h/c` - Symbol table for variable tracking
- `build.sh` - Build script
- `clean.sh` - Cleanup script

Building the Project

To build the compiler, run:

```
./build.sh
```

This will:
This will:

- Generate lexer code with Flex
- Generate parser code with Bison
- Compile all source files into final executable

Usage

- Write your source code in `input.txt`
- Run the compiler:

```
./parser
```

- Check the results:
  - output.txt - Contains AST output
  - error.txt - Contains any compilation errors

Cleaning Up

```
./clean.sh
```

Requirements

- Flex
- Bison
- GCC