1. Assembly and Disassembly Functions for SASM

File: `sasm_assembler.h`
Description: This file provides functions for assembling and disassembling programs in a custom assembly language.
Author: Soham Metha
Date: January 2025

1.1. Table of Contents

1.2. Functions

1.2.1. **Disassemble Bytecode into Program**

Declaration:

```
Program disassembleBytecodeIntoProgram(const char* filePath);
```

Description:
Takes the path to a binary file containing bytecode and disassembles it into a `Program` structure.

Parameters:

- `filePath`: Path to the file containing the bytecode.

Returns:
The disassembled `Program` structure.

1.2.2. **Assemble Program into Bytecode**

Declaration:

```
void assembleProgramIntoBytecode(const Program* prog, const char*
filePath);
```

Description:
Takes a `Program` structure, assembles it into bytecode, and writes it to a binary file.

Parameters:

- `prog`: Pointer to the `Program` structure to assemble.
- `filePath`: Path to the output binary file.

1.2.3. **Process Line**

Declaration:

```
Instruction processLine(String* line);
```

Description:
Processes a single line of assembly code and converts it into an `Instruction` structure.

Parameters:

- `line`: Pointer to the line of assembly code.

Returns:
The processed `Instruction` structure.

1.2.4. **Parse Assembly into Program**

Declaration:

```
Program parseAsmIntoProgram(String* src);
```

Description:
Parses a string of assembly code into a `Program` structure.

Parameters:

- `src`: Pointer to the string containing assembly code.

Returns:
The parsed `Program` structure.

1.2.5. **Load File into String**

Declaration:

```
String loadFileIntoString(const char* filePath);
```

Description:
Loads the contents of a file into a string.

Parameters:

- `filePath`: Path to the file.

Returns:
The loaded `String`.

1.2.6. **Write Program to File**

Declaration:

```
void writeProgramToFile(const Program* prog, const char* filePath);
```

Description:
Writes a `Program` structure to a non-binary file.

Parameters:

- `prog`: Pointer to the `Program` structure.
- `filePath`: Path to the output file.

1.2.7. **Assembly Mode Operations**

Declaration:

```
int assemblyMode(char* inputFile, char* outputFile);
```

Description:
Performs assembly mode operations. Reads assembly code from an input file, assembles it into bytecode, and writes it to an output file.

Parameters:

- `inputFile`: Path to the input file containing assembly code.
- `outputFile`: Path to the output file to store bytecode.

Returns:
`0` if successful.

1.2.8. **Disassembly Mode Operations**

Declaration:

```
int disassemblyMode(char* inputFile, char* outputFile);
```

Description:
Performs disassembly mode operations. Reads bytecode from an input file, disassembles it into assembly code, and writes it to an output file.

Parameters:

- `inputFile`: Path to the input file containing bytecode.
- `outputFile`: Path to the output file to store assembly code.

Returns:
`0` if successful.

1.3. References

- `sasm_instructions.h` for `Instruction` and `Program` structures.
- `univ_strings.h` for string handling utilities.

1.4. Example Usage

1.4.1. Assembling a Program

```
#include "sasm_assembler.h"

int main() {
    Program program = parseAsmIntoProgram(&assemblyCode);
    assembleProgramIntoBytecode(&program, "output.bytecode");
    return 0;
}
```

1.4.2. Disassembling Bytecode

```
#include "sasm_assembler.h"

int main() {
    Program program = disassembleBytecodeIntoProgram("input.bytecode");
    writeProgramToFile(&program, "output.asm");
    return 0;
}
```