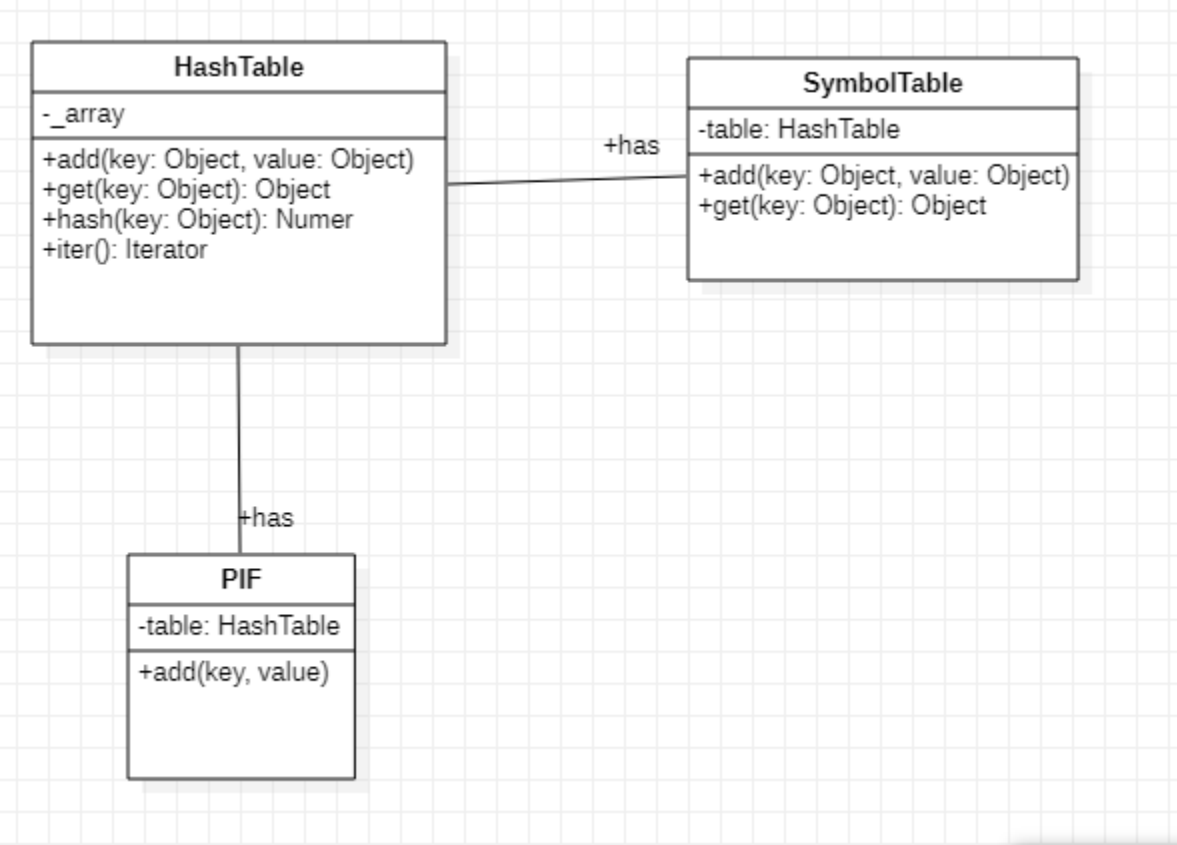
https://github.com/cinnamonbreakfast/flcd/tree/main/lab3

Lab 3 Documentation

Function	Pre-condition	Post-condition	Observations
Detect(program)	<program>: string – valid name for an input source file	<program>_PIF.out : contains PIF data <program>_ST.out : contains ST data	Using Regex to fetch the program, then send the tokens through filters
is_ident_const(cod)	cod : token	Boolean	Function checks if the current token is an identifier or constant
is_reserved(cod)	cod : token	Boolean	Checks if the current token is a reserved work or token.

Both ST and PIF are built on HashTable. The Analyzer is build using Functional Programming, using the functions documented above.



Run Example:

Code:

p1.in

```
entry {
    int number;

    number = 0;

    if(number < 5) {
        WRITE("RANDOMSTRING");
    }
}
```

p1_ST.out

```
Using HashTable for data representation

[]
[['5', 0]]
[]
[['number', 0], ['0', 0], ['"RANDOMSTRING"', 0]]
```

P1_PIF.out

```
('entry', 0)
('{', 0)
('int', 0)
('number', [3, 0])
(';', 0)
('=', 0)
('0', [3, 1])
(';', 0)
('if', 0)
('(', 0)
('<', 0)
('5', [1, 0])
(')', 0)
('{', 0)
('WRITE', 0)
('(', 0)
('"RANDOMSTRING"', [3, 2])
(')', 0)
(';', 0)
('}', 0)
('}', 0)
```

Error handling:

We'll take P1 and add some errors:

```
entry {
    int number;

    +number = -0;

    if(number < 5) {
        WRITE("RANDOMSTRING);
    }
}
```

Console:

Lexical error for +number at line 4

Lexical error for -0 at line 4

Lexical error for "RANDOMSTRING at line 7

ST:

```
Using HashTable for data representation

[]
[['number', 0]]
[['5', 0]]
[]
```

PIF:

```
('entry', 0)
('{', 0)
('int', 0)
('number', [1, 0])
(';', 0)
('=', 0)
(';', 0)
('if', 0)
('(', 0)
('<', 0)
('5', [2, 0])
(')', 0)
('{', 0)
('WRITE', 0)
('(', 0)
(')', 0)
(';', 0)
('}', 0)
('}', 0)
```