Great! Based on your description, **SimpleLang** is a minimal, educational high-level language tailored for an 8-bit CPU, and includes:

✅ Supported Features:

- **Variable Declarations**
- **Assignments**
- **Arithmetic Operations (+, -)**
- **Conditional Statements (if)**

Not Included:

- **Loops**
- **Functions**
- **Data types beyond int**
- **I/O operations**

Let me expand and summarize your syntax definitions clearly for documentation or implementation purposes.

SimpleLang Syntax Reference

1. Variable Declaration

Declares a variable of type `int`.

int <identifier>;

Example:

```
int a;
int b;
```

2. Assignment

Assigns the result of an expression to a variable.

Syntax:

```
<identifier> = <expression>;
```

Expression can include:

- constants (e.g., `5`)
- variables (e.g., `b`)
- arithmetic operations (`+`, `-`)
- **Examples:**
- `a = b;`
- `a = b + c;`

- `a = b - 5;`

- 3. Arithmetic Operations
- Only addition and subtraction are supported.
- **Syntax (within assignment):**
- `<identifier> = <term> (+|-) <term>;`
- **Examples:**
- a = b + 1;
- x = y - z;

4. Conditional Statements

Only equality conditions (==) are supported.

Syntax:

```
if (<identifier> == <identifier|constant>) {
    <statement>;
    ...
}
if (a == b) {
    a = a + 1;
}
```

Nested conditionals are not mentioned, so it's likely they are either unsupported or need to be kept simple.

Example SimpleLang Program:

```
int a;
int b;
int c;

a = 5;
b = 3;
c = a - b;

if (c == 2) {
    a = a + 1;
}
```

Would you like help building a **compiler**, **interpreter**, or **transpiler** for SimpleLang targeting your 8-bit CPU? Or do you want a grammar (e.g., in BNF/EBNF) to formalize it further?