

With ChatGPT as the LLM

-A BNF created by him-

Could you create a BNF, in the form of a mini-language, which has the purpose of helping teachers to check the final situation for each student at the end of the semester.

BNF Grammar

go

Copy code

```
<program>      ::= { <student_record> }

<student_record> ::= "student" <student_id> "{" <student_body> "}"

<student_body>  ::= <name> <courses> <attendance> <remarks>

<student_id>    ::= <identifier>

<name>          ::= "name:" <string>

<courses>      ::= "courses:" "{" { <course_record> } "}"

<course_record> ::= <course_name> ":" <grade>

<course_name>   ::= <identifier>

<grade>         ::= <letter_grade> | <numeric_grade>

<letter_grade>  ::= "A" | "B" | "C" | "D" | "F"

<numeric_grade> ::= <digit> { <digit> } [ "." <digit> { <digit> } ]

<attendance>    ::= "attendance:" <percentage>

<percentage>    ::= <digit> { <digit> } "%"

<remarks>      ::= "remarks:" <string>

<identifier>    ::= <letter> { <letter> | <digit> | "_" }

<string>        ::= '"' { any_character_except_quote } '"'

<digit>         ::= "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"

<letter>        ::= "A" | ... | "Z" | "a" | ... | "z"
```

Samples:

```
sfs Copy code

student S001 {
  name: "Alice Smith"
  courses: {
    Math: A
    History: 85.5
    Physics: B
  }
  attendance: 92%
  remarks: "Excellent performance overall"
}

student S002 {
  name: "Bob Johnson"
  courses: {
    Math: C
    History: 70
    Physics: B
  }
  attendance: 78%
  remarks: "Needs improvement in Math"
}
```

Changes into json like format, good for keeping track of all details, but there is no way to calculate the student grade / attendance track, you have to correctly put it.

-Checking for validation / improvement-

Prompt (It also says to create the bnf by himself, but he mostly copied mine, also forgot to put the bnf in that prompt so I did afterward) :

I created the following bnf, in the form of a mini-language, which has the purpose of helping teachers to check the final situation for each student at the end of the semester, could you do the following 2 things, first, try to generate one on your own, just by the idea that I have provided you, and second, check mine to see if i have any errors, or small things that I might have forgotten. And actually even a third thing, suggest improvements to it. Here is the bnf

Issues found

⚠ Issues / Minor Errors		
Issue	Description	Suggestion
Empty alternative in <code><value></code>	You have <code><value> ::=</code>	<code>"1"</code>
<code><identifier></code> recursion redundant	<code><identifier></code> and <code><ident_rest></code> recursion is a bit tangled; it allows empty strings and duplicates letter/digit branches.	Simplify to <code><id> ::= <letter> {<letter>}</code>
<code><if></code> recursion	<code><elif></code> structure works but is awkward (<code>"</code>	<code>"elif" ...</code>).
<code><output></code> rule detached	The <code><output></code> statement is not clearly defined as a top-level command with a keyword.	Either rename to <code>"output" <output></code> or merge with <code><print></code> .
String grammar incomplete	Your <code><string></code> rule doesn't allow spaces or escape sequences.	Add <code>" "</code> and optional escaped characters to <code><symbol></code> or <code><character></code> .
<code><grades></code> limited to 1-10	This assumes only integer grades from 1-10; might be too strict.	Change <code><value></code> to allow decimals or identifiers (<code><number></code>).
<code><if></code> and <code><for></code> lack punctuation	No <code>:</code> or clear start marker for the body. Could confuse parsing.	Add <code>":"</code> after the condition for readability and parsing ease.
Missing semicolon or newline separator	<code><statement></code> s are just concatenated; ambiguous where one ends.	Add a separator (newline or semicolon).
No "compute" or "assign"	There's no general way to compute or assign new values except via functions.	Add something like <code><assign> ::= <id> "=" <expr> .</code>

I did have an empty alternative in value

```
3 <value> ::= | "1" | "2" | "3"  
  | "4" | "5" | "6" | "7" | "8"  
  | "9" | "10"
```

In examples I use “output x” or “output ‘word’” but in the bnf forgot the keyword

```
<bool> ::= "true" | "false"  
<output> ::= <string> | <id>  
<string> ::= ""<string_content>
```

Improvements

Added the assign statement

```
3. Allow general expressions in assignment or conditionals.  
Add:  
bash  
<assign> ::= <id> "=" <expr>
```

Added comments by suggestion

```
6. Add comments (useful for a teacher scripting language):  
go  
<comment> ::= "#" <string_content>
```