exercise: dealing with ambiguity

Exercise:

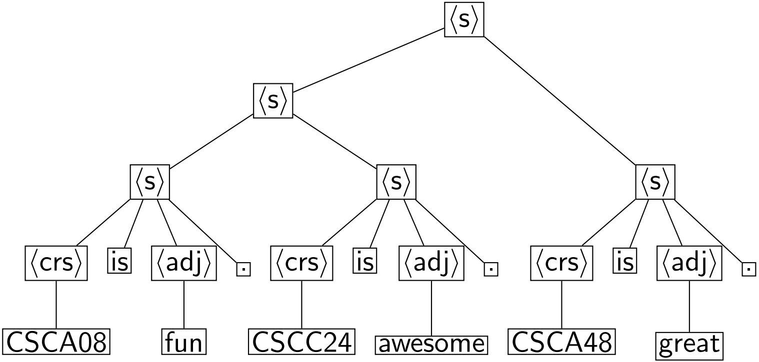
```
<sentence> ::= \empty |
               <course> is <adjective>. |
               <sentence><sentence>
<course> ::= CSCA08 | CSCA48 | CSCB07 | CSCB09 | CSCC24
<adjective> ::= great | fun | awesome
```

where `\empty` stands for the empty string.

- Demonstrate that the CFG is ambiguous.
- Provide a grammar that generates exactly the same language as above and is not ambiguous.

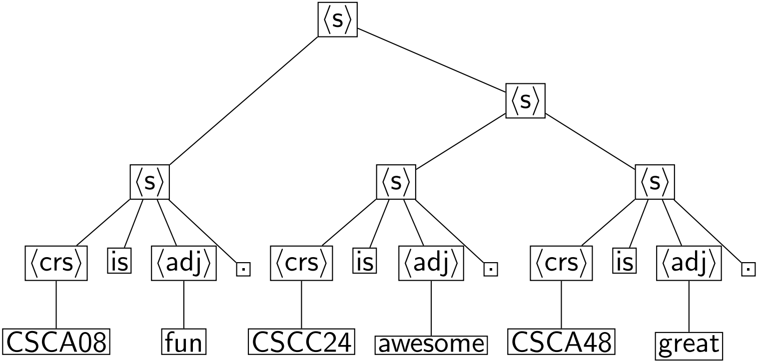
exercise: dealing with ambiguity

```
<sentence> ::= \empty |
               <course> is <adjective>. |
               <sentence><sentence>
<course> ::= CSCA08 | CSCA48 | CSCB07 | CSCB09 | CSCC24
<adjective> ::= great | fun | awesome
```



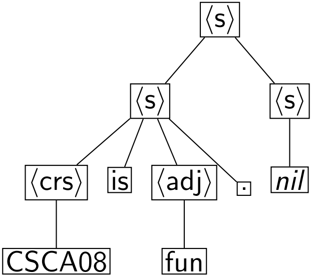
exercise: dealing with ambiguity

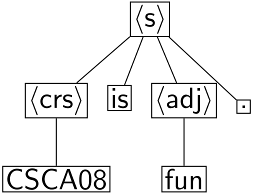
```
<sentence> ::= \empty |
               <course> is <adjective>. |
               <sentence><sentence>
<course> ::= CSCA08 | CSCA48 | CSCB07 | CSCB09 | CSCC24
<adjective> ::= great | fun | awesome
```



exercise: dealing with ambiguity

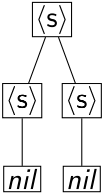
```
<sentence> ::= \empty |
               <course> is <adjective>. |
               <sentence><sentence>
<course> ::= CSCA08 | CSCA48 | CSCB07 | CSCB09 | CSCC24
<adjective> ::= great | fun | awesome
```

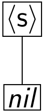




exercise: dealing with ambiguity

```
<sentence> ::= \empty |
               <course> is <adjective>. |
               <sentence><sentence>
<course> ::= CSCA08 | CSCA48 | CSCB07 | CSCB09 | CSCC24
<adjective> ::= great | fun | awesome
```





exercise: dealing with ambiguity

```
<sentence> ::= \empty | <sentences>
<sentences> ::= <course> is <adjective>. |
                <course> is <adjective>. <sentences>
<course> ::= CSCA08 | CSCA48 | CSCB07 | CSCB09 | CSCC24
<adjective> ::= great | fun | awesome
```

```
<sentence> ::= \empty |
               <course> is <adjective>. <sentence>
<course> ::= CSCA08 | CSCA48 | CSCB07 | CSCB09 | CSCC24
<adjective> ::= great | fun | awesome
```

dealing with ambiguity

1. Can't always remove an ambiguity from a grammar by restructuring productions.
2. An inherently ambiguous language does not possess an unambiguous grammar.

Question. Is there an algorithm that can examine an arbitrary context-free grammar and tell if it is ambiguous?

an inherently ambiguous language

Suppose we want to generate the following language:

$$\mathcal{L} = \{a^i b^j c^k \mid i, j, k \geq 1, i = j \text{ or } j = k\}$$

Grammar:

Two parse trees for $a^i b^i c^i$.

limitations of CFGs

CFGs are not powerful enough to describe some languages.
Examples:

- $\{ a^i b^i c^i \mid i \geq 1 \}$.
- $\{ a^m b^n c^m d^n \mid m, n \geq 1 \}$.

Question: Is there an algorithm that can examine two arbitrary CFGs and determine if they generate the same language?

translation process summary

1. Lexical Analysis:
   Converts source code into sequence of tokens.
   We use regular grammars and finite state automata (recognizers).

2. Syntactic Analysis:
   Structures tokens into initial parse tree.
   We use CFGs and parsing algorithms.

3. Semantic Analysis:
   Annotates parse tree with semantic actions.

4. Code Generation:
   Produces final machine code.

more on this...

Take Compilers & Interpreters!