Name: **PID:**

Quiz 2

Which class of grammars is usually used to describe the lexical structure of a programming language? [1 point]

Regular grammars.

Which class of grammars is usually used to describe the syntactical structure of a programming language? [1 point]

Context-free grammars.

Why are two distinct classes used in practice? Provide two reasons. [2 points]

Regular grammars are used for lexical analysis because they are sufficient to describe most token types, and because finite automata can be implemented more efficiently than parsers for context-free grammars.

Context-free grammars are used for syntax analysis because regular grammars cannot express recursive structures such as balanced parenthesis and arithmetic expressions.

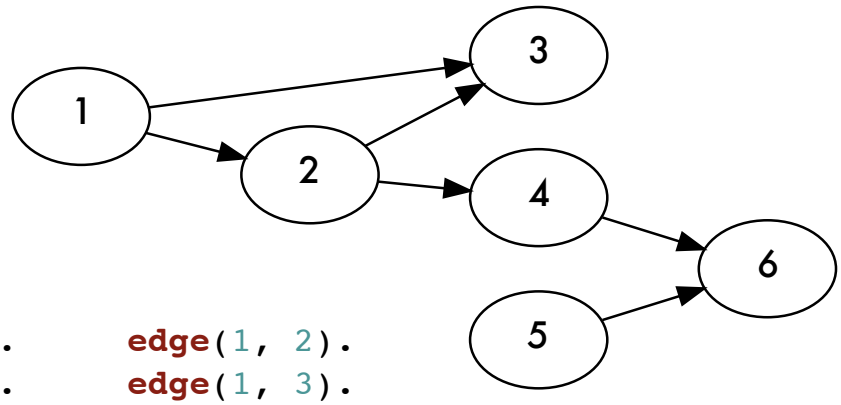
Is the following grammar a LL(1) grammar? [1 point]
Briefly explain how you arrived at your conclusion.

zzz → *xxx* .
xxx → *aaa yyy*
yyy → ε
yyy → *bbb xxx*
aaa → 1 | 2 | 3 | 4 | 5
bbb → & | % | $ | # | @

(Terminals are underlined.)

It's a LL(1) grammar.
It is not left-recursive, and does not have any common prefixes.
In particular, the predict sets for the two *yyy* productions are disjoint.

Write a Prolog knowledge base that represents the below graph using the clauses `vertex/1` and `edge/2`. [2 points]




```
vertex(1).      edge(1, 2).
vertex(2).      edge(1, 3).
vertex(3).      edge(2, 3).
vertex(4).      edge(2, 4).
vertex(5).      edge(4, 6).
vertex(6).      edge(5, 6).
```

Based on `vertex/1` and `edge/2`, write a Prolog clause `reachable/2` that is satisfied if there exists a path from the first argument to the second. Ensure that both arguments are vertices. [2 points]

```
reachable(X, Y) :-
     vertex(X), vertex(Y),
     edge(X, Y).
reachable(X, Y) :-
     vertex(Y), vertex(Z),
     edge(Z, Y),
     reachable(X, Z).
```

Write a Prolog query to find all vertices that are reachable from vertex 1. [1 point]

```
?-   reachable(1, Y).
```