1. (20 pts) Show the deterministic FSA used by a lexical analyzer to recognize the keywords **packed**, **procedure**, **program**, and identifiers that start with a letter followed by only letters and digits. These are the only lexemes in the language. **Explain** the actions the system would perform if the input contains the text “prog2”.

```
prog2
1 vs prog22
```

```
pascal
X > Y
```

```
CS 480 Translators
Test I
```
2. (10 pts) Explain why a lexical analyzer needs either a look-ahead system or a push back buffer. Give an example to illustrate your answer.

3. (10 pts) What are the characteristics common to all types of translators.
4. (20 pts) Here is just a portion of the grammar for a programming language

\[ \text{type ::= id | type } \hat{\text{type}} \text{ [id]} \]

Is this grammar ambiguous? Why or why not?

5. (20 pts) Make the above grammar suitable for a recursive descent parser.

\[ \text{type } \rightarrow \text{ id} | \text{ type } \hat{\text{type}} \text{ id } \]

- 10 pts: Two trees with the same derivation
- 5 pts: Remove left recursion
6. (20 pts) Using subset construction, convert the following NFA to a DFA.