Study Guide for Final Exam

In this exam, you can use **one** cheat sheets of the standard US letter size.

The final exam will cover the following topics and skills:

Scanning (Week 11) [~11pts]:

1) List the tasks/responsibilities of a scanner
2) Describe the ad hoc approach to scanning. What are advantages of the ad hoc scanners? Which type of compilers use ad hoc scanners?
3) Describe the DFA approach to scanning. What are advantages of the DFA approach to scanning? When is the DFA approach often used?
4) How to convert a nondeterministic finite automaton (NFA) to a DFA.
5) Describe two approaches to representing a DFA.
6) Describe how a table-driven scanner works.

Context-Free Grammars (CFG) and Parsing (Week 13) [~12pts]:

1) Definition of context-free grammars, describe CFG in Backus-Naur Form (BNF) and extended BNF
2) Write derivations in CFG. What are left-most derivations? right-most derivations? What is a *sentential form*? What is the *yield* of a derivation?
3) What is the relationship between parse trees and derivations? What are ambiguous grammars?
4) What is the running time of the parsing algorithms that work for any CFG?
5) Distinguish between LL parsers and LR parsers. What are characteristics of LL parsers? What are characteristics of LR parsers?

LL(1) Parsers (Week 14) [10pts]:

1) Describe and compare the two approaches to **LL(1)** parsing: the recursive descent approach and the table-driven approach.
2) Describe how **LL(1)** parse stacks are maintained.
3) Describe the structures (indexes, entries) of **LL(1)** parse table.
4) What are the predict sets needed for? Apply the formula for calculating the PREDICT sets from the FIRST sets and FOLLOW sets. (Don't need to know how the FIRST sets and FOLLOW sets are calculated).
5) How to determine if a grammar is not **LL(1)**?

LR(1) Parsers (Week 15) [~7pts]:

1) Compare **LR(1)** parsers and the table-driven **LL(1)** parsers: What do they have in common? What are the differences between them.
2) Describe how LR(1) parse stacks (with states) are maintained. Describe how the shift and reduce actions are implemented as actions on the stack.
3) Describe the structures (indexes, entries) of **SLR(1)** parse table.
4) What kind of automata is used to represent the shift rules of an **LR(1)** parser, and of subclasses of **LR(1)** parsers?
5) Compare the parse table and the driver used by difference subclasses of **LR(1)**.

Racket [~20pts]: In this exam, you are expected to understand and apply the syntax of Racket for

1) Creating lists
2) Creating vectors
3) Creating transparent structure , defining constructor guard
4) Writing regular expressions (similar to the regular expression used in the `IP-address?` function that was provided in Homework 8)
5) Pre-defined functions: `map, apply, vector->list, make-vector, number?, circle, hc-append, regexp-match`