University of Central Punjab (UCP)

Academic Year 2022-2026

Department: Computer Science

Project: Compiler_Construction

Submitted To: Prof Asif Farooq

Full Name: **M** Kamran Saleem **Roll No:**0613

Section: "G-1" **Subject:** CC

Date of Submission: 09/11/202

Student Sign: **Professor Sign:**

PHASE 1 DELIVERABLES CHECKLIST

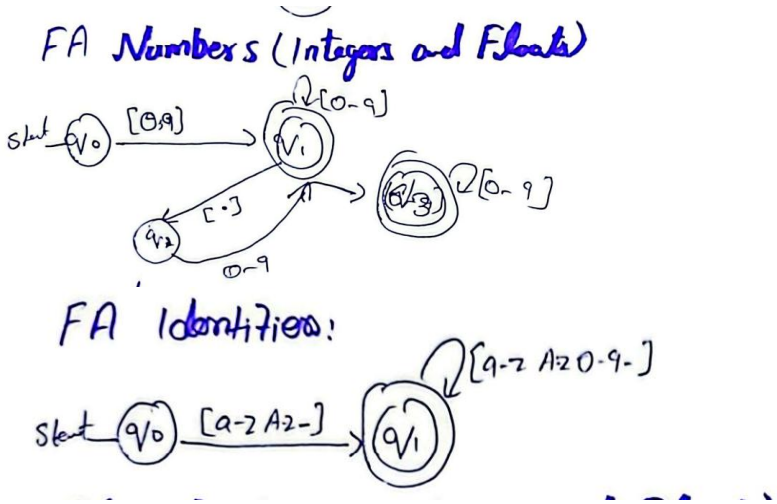
Regex definitions table

Token Type	Regular Expression	Examples
Token Type	Regular Expression	Examples
Identifier	[a-zA-Z_][a-zA-Z0-9_]*	my_var, trade123, _foo
Keyword	strategy|asset|when|otherwise|monitor|buy|sell|hold|stop_loss|take_profit|log|get_price|get_volume|portfolio|shares	buy, when, log
Integer	[0-9]+	100, 42
Float	[0-9]+\.[0-9]{1,2}	10.5, 150.00
Operator	=, >, <, *, \+	=, >, <, *, +
Punctuation	;, $, $, \{, \}, \"	;, (,), {, }, "
String	"[^"]*"	"AAPL", "Buy now"

Explanation:

- Each row describes a valid token in the TradeScript language, with its regular expression and examples.
- This table is typically included in the theoretical portion or as an appendix in your compiler project documentation. It shows clearly how your lexer recognizes each token class.

Finite Automata Diagrams (Draw by Hand)



Keyword Specifications & Explanation

TradeScript Keyword Table with Explanations

This table lists all **15 unique keywords** from the TradeScript language, along with a brief explanation for each, focusing on their relevance to financial trading and programming logic.

Keyword	Meaning/Role	Brief Explanation and Domain Relevance
strategy	Function/Strategy definition	Starts a new trading strategy block, like a function for automated trading rules. Makes code modular and clear.
asset	Asset declaration	Specifies which market symbol (e.g., stock, commodity) the strategy operates on. Essential for precise targeting.
when	Conditional 'if'	Checks if a condition (like price < 100) is met. Enables rules-based logic directly using trading criteria.

otherwise	'else' case	Specifies alternative actions if the 'when' condition fails. Improves code readability for handling both outcomes.
monitor	Continuous check loop	Repeatedly evaluates market conditions (e.g., market open) for real-time trading automation.
buy	Purchase action	Issues a buy order for a given asset/quantity. Direct, domain-specific, and intuitive for traders.
sell	Sell action	Issues a sell order for an asset/shares. Core trading operation, making code clear for non-programmers.
hold	No action (wait)	Directs the strategy to not act — reflects a common real-world trading stance.
stop_loss	Set loss-exit condition	Defines automatic exit point to cap potential losses. Increases safety and aligns with trading best practices.
take_profit	Set profit-exit condition	Sets a target to close a position when a desired profit is reached. Automated risk/reward management.
log	Print message	Outputs messages for tracking or debugging strategy behavior. Useful for understanding strategy runs.
get_price	Fetch latest price	Pulls live price data for any specified asset. Enables dynamic, data-driven strategies and rules.
get_volume	Fetch latest volume	Gets traded volume data, often a key market indicator. Useful for advanced trade triggers.

portfolio	Account/Portfolio namespace	Lets the script access user's cash, assets, or positions, essential for real-world trade handling.
shares	Quantity unit	Used to specify the number of shares/contracts for buy/sell.

GitHub repository link

Repository must contain:

- Lex/scanner.l file
- Your unique sample test program
- tokens.txt output file
- error log (if errors exist)

Sample execution video (max 5 mins) must be in the GitHub repo.

Link: https://github.com/ks-bhatti/Tradescript-Compiler-Phase1