■ Teaching Notes: Applications of Stack – Arithmetic Expressions

1. Introduction

• Stacks are widely used in expression evaluation and conversion.
• Arithmetic expressions can be represented in three forms:
1. Infix – Operator between operands (e.g., A + B)
2. Prefix (Polish Notation) – Operator before operands (e.g., + A B)
3. Postfix (Reverse Polish Notation) – Operator after operands (e.g., A B +)

2. Expression Notations

(a) Infix

• Operators are between operands.
• Example: (A + B) * C
• Human-friendly, but computers find it hard to evaluate directly (need precedence & associativity rules).

(b) Prefix

• Operators are written before operands.
• Example: * + A B C
• No need for parentheses, as order is fixed.

(c) Postfix

• Operators are written after operands.
• Example: A B + C *
• Easy for computers to evaluate using a stack.

3. Conversion between Expressions

(a) Infix → Postfix

• Use stack for operators.
• Rules:
1. Operands → Output directly
2. (→ Push to stack
3.) → Pop until (
4. Operator → Pop operators from stack if they have higher/equal precedence before pushing

Example: (A + B) * C
■ Postfix: A B + C *

(b) Infix → Prefix

• Two methods:
1. Reverse infix → Convert to postfix → Reverse result
2. Direct algorithm using stack

Example: (A + B) * C
■ Prefix: * + A B C

(c) Prefix ↔ Postfix

• Prefix → Postfix:
- Scan prefix from right to left
- Use stack for operands
- Pop 2 operands for operator, combine, push back

• Postfix → Prefix:
- Scan postfix from left to right
- Use stack for operands
- Pop 2 operands, combine with operator at front, push back

4. Evaluation using Stack

(a) Postfix Evaluation

• Scan left to right:
1. If operand → Push
2. If operator → Pop 2 operands, apply operator, push result

Example: 5 3 + 2 *
• Push 5, push 3
• +: Pop (5, 3) → 5+3=8 → Push 8
• Push 2
• *: Pop (8, 2) → 8*2=16 → Push 16
■ Result = 16

(b) Prefix Evaluation

• Scan right to left:
1. If operand → Push
2. If operator → Pop 2 operands, apply operator, push result

Example: * + 5 3 2
• Right to left → Push 2, push 3, push 5
• +: Pop (5, 3) → 8 → Push 8
• *: Pop (8, 2) → 16 → Push 16
■ Result = 16

5. Key Advantages of Stack in Expressions

• Removes ambiguity in operator precedence
• Eliminates need for parentheses
• Provides systematic method for conversion & evaluation

6. Quick Summary Table

Expression Type	Example (A+B)*C	Advantage
Infix	(A + B) * C	Easy for humans
Prefix	* + A B C	No parentheses, systematic
Postfix	A B + C *	Easiest for computers

■ *Teaching Tip:*

• Start with infix examples (students relate easily).
• Show how ambiguity arises without precedence.
• Demonstrate conversion step by step on the board.
• Finally, do a live dry-run evaluation with a stack diagram.