

Stack 2

Computer Science Department Eastern Washington University Yun Tian (Tony) Ph.D.



Recall

- Concept of Stack
 - Last-in First Out property
 - Operations restricted on one end of the container.
 - O(1)
 - Two ways of representation.
 - Demo of Array-based Stack Implementation

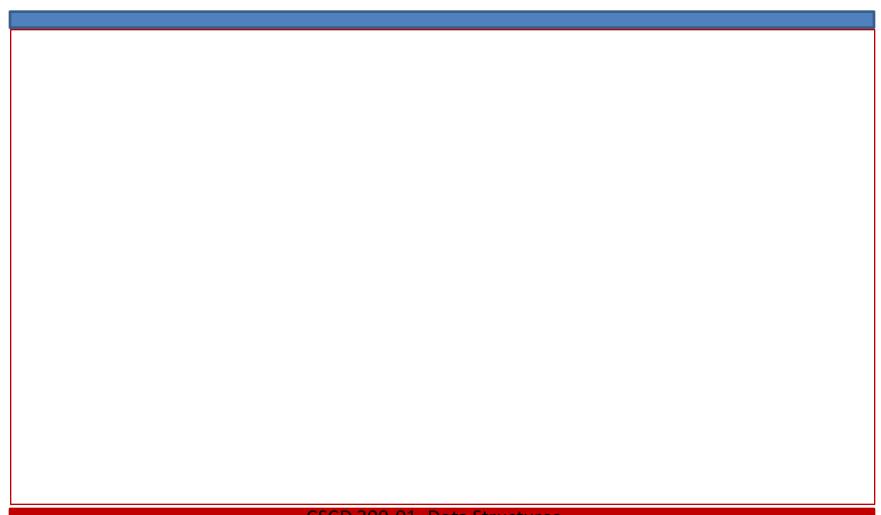


Today Class

- Demo of Linked List Stack
- Applications of Stack
 - Reverse List/Array
 - Decimal to Binary Number
 - Parenthesis Matching
 - Infix Expression and Postfix Expression



Demo of Linked List Stack



CSCD 300-01 Data Structures



Demo of Stack Applications

- Reverse List/Array
- Decimal to Binary Number
- Parenthesis Matching



Parentheses Matching

- Each "(", "{", or "[" must be paired with a matching ")", "}", or "]"
 - correct: ()(()){([()])}
 - incorrect: ((()(()){([()])}
 - incorrect:)(()){([()])}
 - incorrect: ({[])}
 - incorrect: (



Parentheses Matching

```
Algorithm ParenMatch(X,n):
Input: An array X of n tokens, each of which is either a grouping symbol, a
variable, an arithmetic operator, or a number
Output: true if and only if all the grouping symbols in X match
Let S be an empty stack
for i=0 to n-1 do
   if X[i] is an opening grouping symbol then
         S.push(X[i])
   else if X[i] is a closing grouping symbol then
         if S.isEmpty() then
                  return false {nothing to match with}
         if S.pop() does not match the type of X[i] then
                  return false {wrong type}
if S.isEmpty() then
   return true {every symbol matched}
else return false {some symbols were never matched}
```



Concept of Infix and Postfix Expressions

- Infix expressions
 - This is the common expression we use in programing.
 - It is a crucial task in Compiler.
 - Operators are in between operands.
 - You have walk though the whole expression before starting to evaluate.
 - E.g. 3 + 4 * 2,
 - You cannot do 3 + 4 first.



Concept of Infix and Postfix Expressions

- Postfix expressions,
 - Precedence and associativity are build-in to the expression.
 - So you can evaluate it from left to right.
 - -E.g 342*+



Concept of Infix and Postfix Expressions

- To Evaluate Infix Expression
 - We can first transfer infix to postfix.
 - Then we evaluate postfix expression.
- Both steps above need Stack structure.



Examples of Infix and Postfix

- Infix: 2 + 3 infix
 - Postfix: 2 3 +
- Infix: 2 * 3 + 4
 - Postfix: 2 3 * 4 +
- Infix: 2 + 3 * 4
 - Postfix: 2 3 4 * +
- Infix: (2 + 3) * 4
 - Postfix: 2 3 + 4 *



Postfix Evaluation algorithm

```
While there are items in the the postfix expression
          if the current item is an operand,
                    push it onto stack.
          else
                    pop stack and attach operand to right of the operator
                    pop stack again and attach operand to the left of the operator
                    evaluate and push the results
          advanced to next item in postfix expression,
if only one number left in stack,
         pop and this is the final result.
else
         postfix has error in it. (probably the original infix has syntax error)
```



Summary and Next Class

- Many Applications
 - Parens Matching
 - Reverse Array
 - Decimal to Binary Conversion
- Concept of infix and postfix expression
- How to evaluate postfix expression?



Next Class

- How to convert infix into postfix expression?
 - Then we know how to evaluate infix expression in compilers.