SNS COLLEGE OF TECHNOLOGY

Coimbatore-35
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF INFORMATION TECHNOLOGY

DATASTRUCTURES

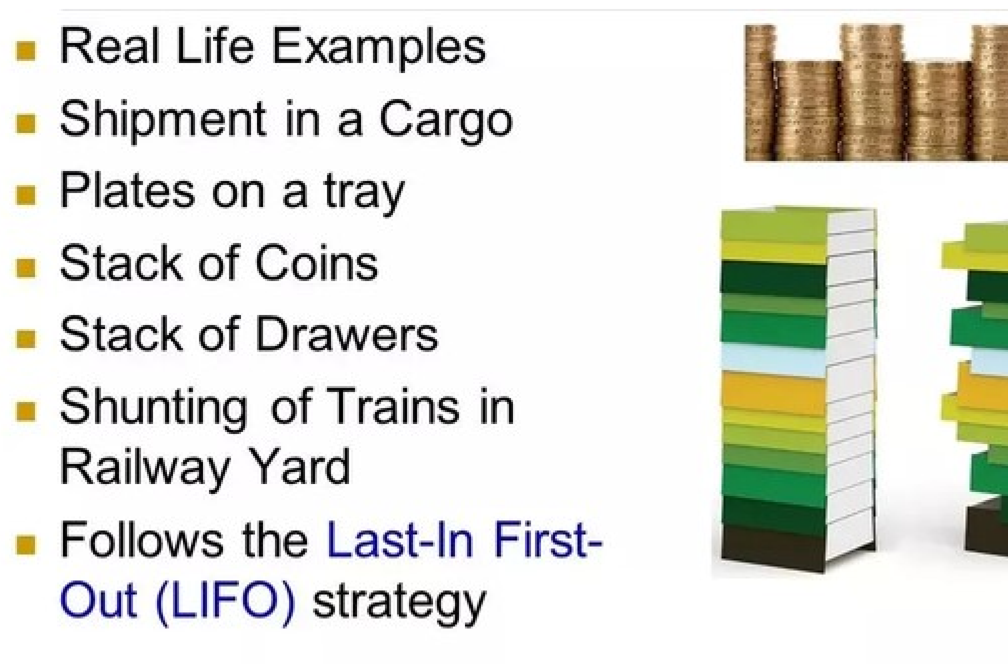
II YEAR III SEM

UNIT 1 –LINEAR STRUCTURES

TOPIC 6-APPLICATION OF STACK

Applications of Stack

- Infix to Postfix Conversion
- Balancing the Symbols
- Function Calls
- Evaluation of Postfix Expression



- Real Life Examples
- Shipment in a Cargo
- Plates on a tray
- Stack of Coins
- Stack of Drawers
- Shunting of Trains in Railway Yard
- Follows the Last-In First-Out (LIFO) strategy

Applications of Stack

Infix Expression

It follows the scheme of **<operand><operator><operand>**

E.g., **A+B**

Postfix Expression

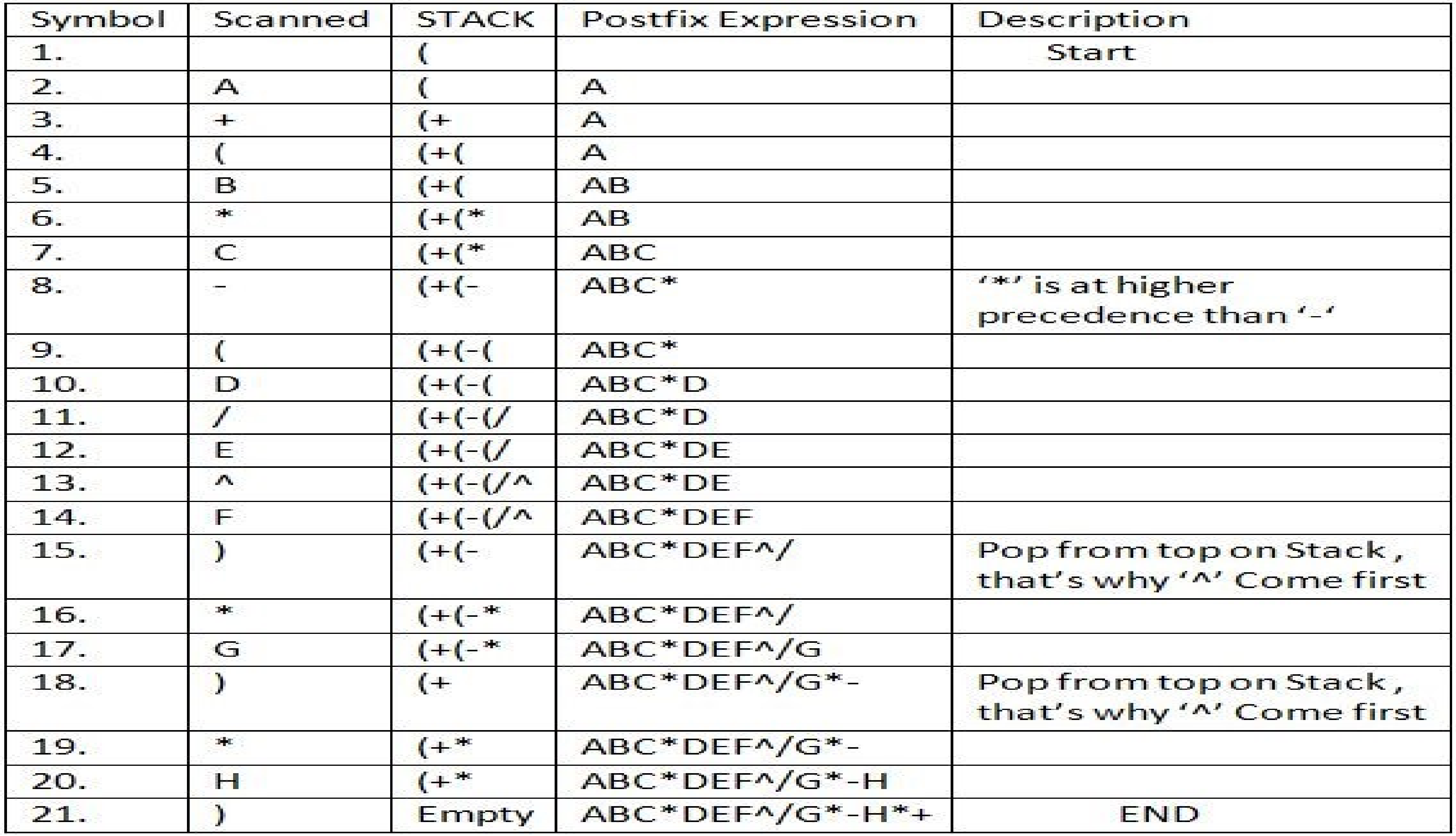
It follows the scheme of <operand><operand><operator> i.e. an <operator> is succeeded by both the <operand>. E.g., AB+

Applications of Stack-Infix to Postfix Conversion

- Push “(“onto Stack, and add “)” to the end of X.
- Scan X from left to right and repeat Step 3 to 6 for each element of X until the Stack is empty.
- If an operand is encountered, add it to Y.
- If a left parenthesis is encountered, push it onto Stack.
- If an operator is encountered ,then:
    - Repeatedly pop from Stack and add to Y each operator (on the top of Stack) which has the same precedence as or higher precedence than operator.
    - Add operator to Stack.
      [End of If]
- If a right parenthesis is encountered ,then:
    - Repeatedly pop from Stack and add to Y each operator (on the top of Stack) until a left parenthesis is encountered.
    - Remove the left Parenthesis.
      [End of If]
      [End of If]
- END.

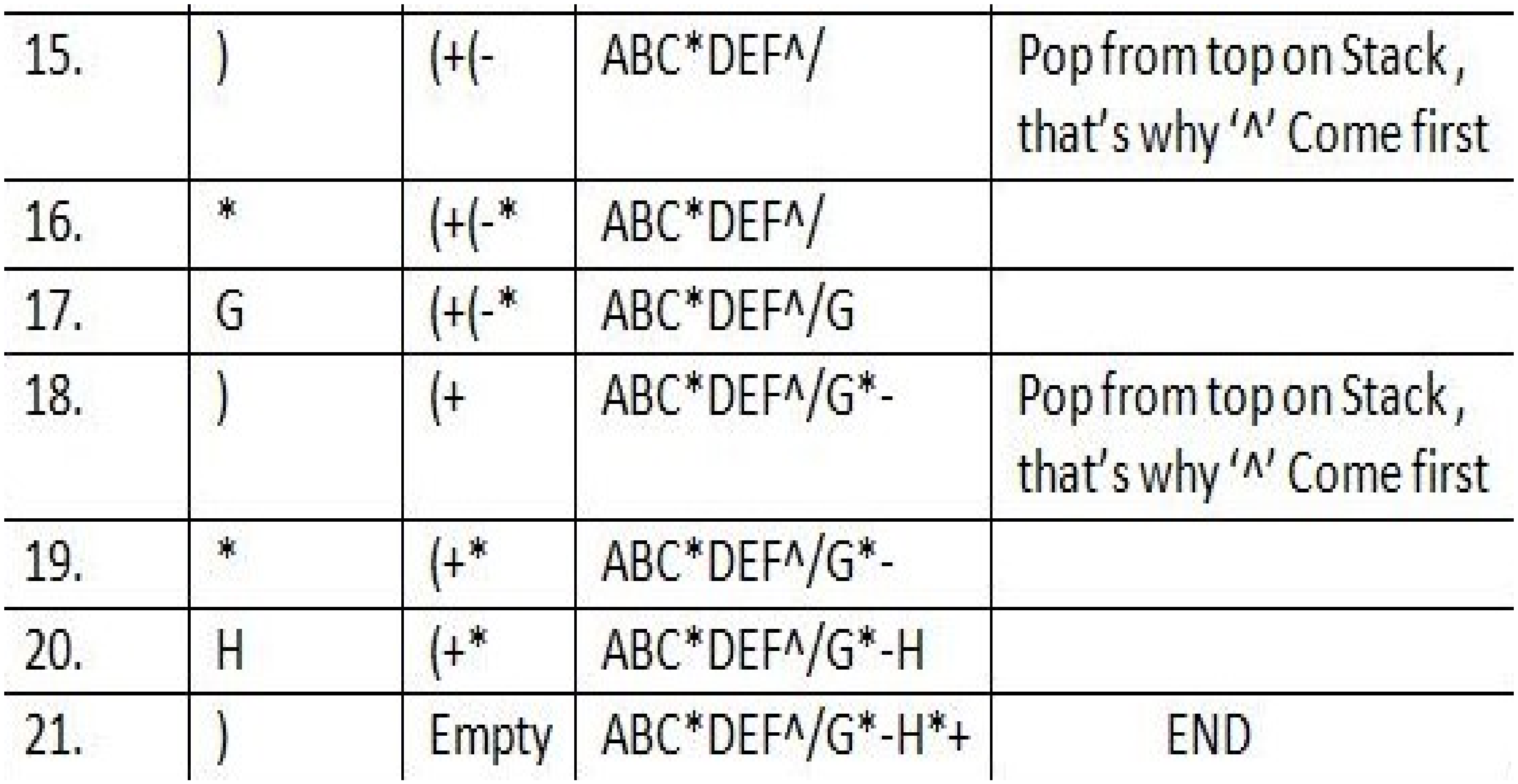
Infix to Postfix Conversion-Example

A+ (B*C-(D/E^F)*G)*H, where **^** is an exponential operator.



Symbol	Scanned	STACK	Postfix Expression	Description
1.		(		Start
2.	A	(	A	
3.	+	(+	A	
4.	(	(+(	A	
5.	B	(+(	AB	
6.	*	(+(*	AB	
7.	C	(+(*	ABC	
8.	-	(+(-	ABC*	'*' is at higher precedence than '-'
9.	(	(+(-(	ABC*	
10.	D	(+(-(	ABC*D	
11.	/	(+(-(/	ABC*D	
12.	E	(+(-(/	ABC*DE	
13.	^	(+(-(/^	ABC*DE	
14.	F	(+(-(/^	ABC*DEF	
15.	)	(+(-	ABC*DEF^/	Pop from top on Stack , that's why '^' Come first
16.	*	(+(-*	ABC*DEF^/	
17.	G	(+(-*	ABC*DEF^/G	
18.	)	(+	ABC*DEF^/G*-	Pop from top on Stack , that's why '^' Come first
19.	*	(+*	ABC*DEF^/G*-	
20.	H	(+*	ABC*DEF^/G*-H	
21.	)	Empty	ABC*DEF^/G*-H*+	END

Applications of Stack-Balancing the Symbols

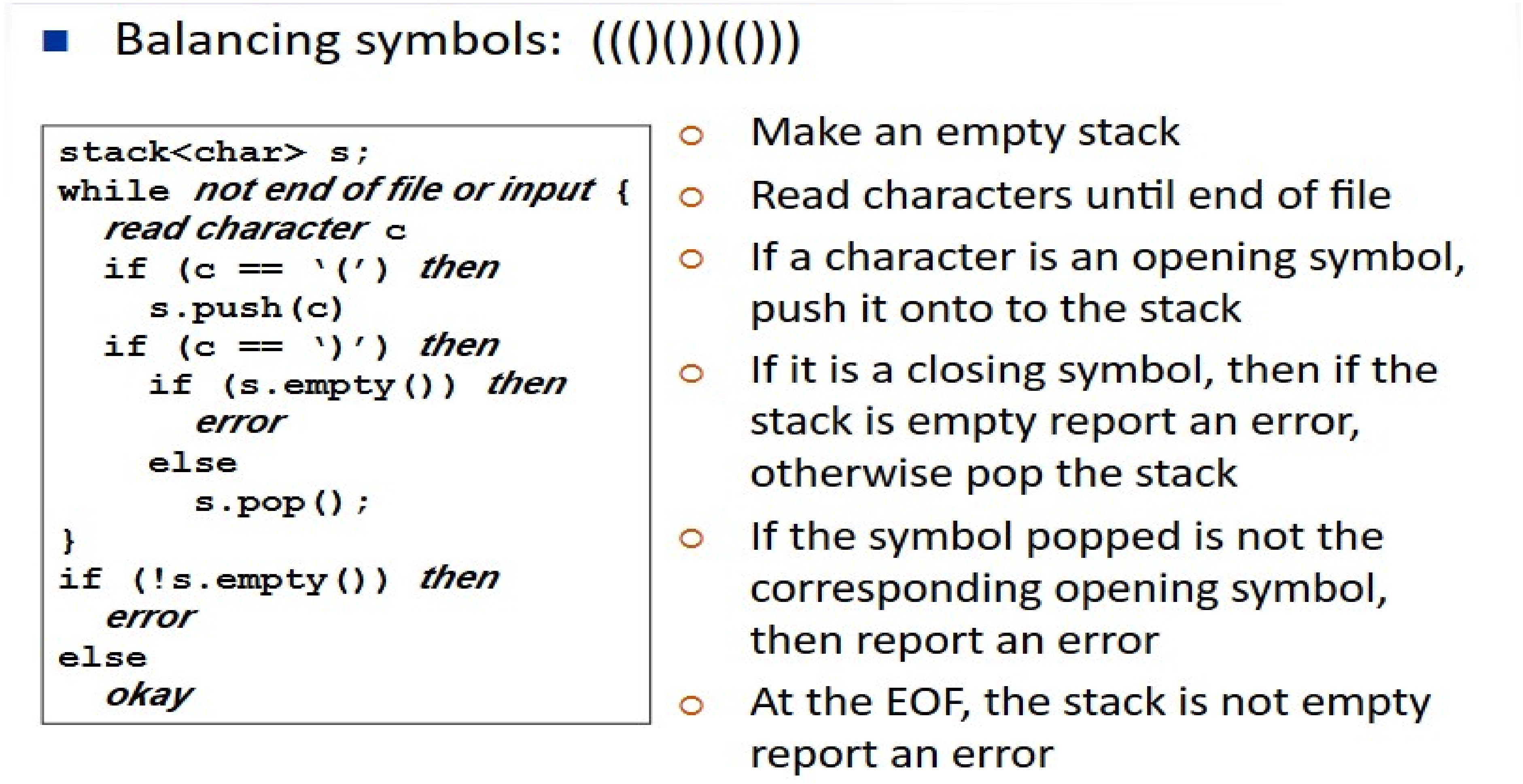


15.	)	(+(-	ABC*DEF^/	Pop from top on Stack , that's why '^' Come first
16.	*	(+(-*	ABC*DEF^/	
17.	G	(+(-*	ABC*DEF^/G	
18.	)	(+	ABC*DEF^/G*-	Pop from top on Stack , that's why '^' Come first
19.	*	(+*	ABC*DEF^/G*-	
20.	H	(+*	ABC*DEF^/G*-H	
21.	)	Empty	ABC*DEF^/G*-H*+	END

Assessment -1

1.Convert the expression into Postfix A+B*C/D

Applications of Stack-Balancing the Symbols

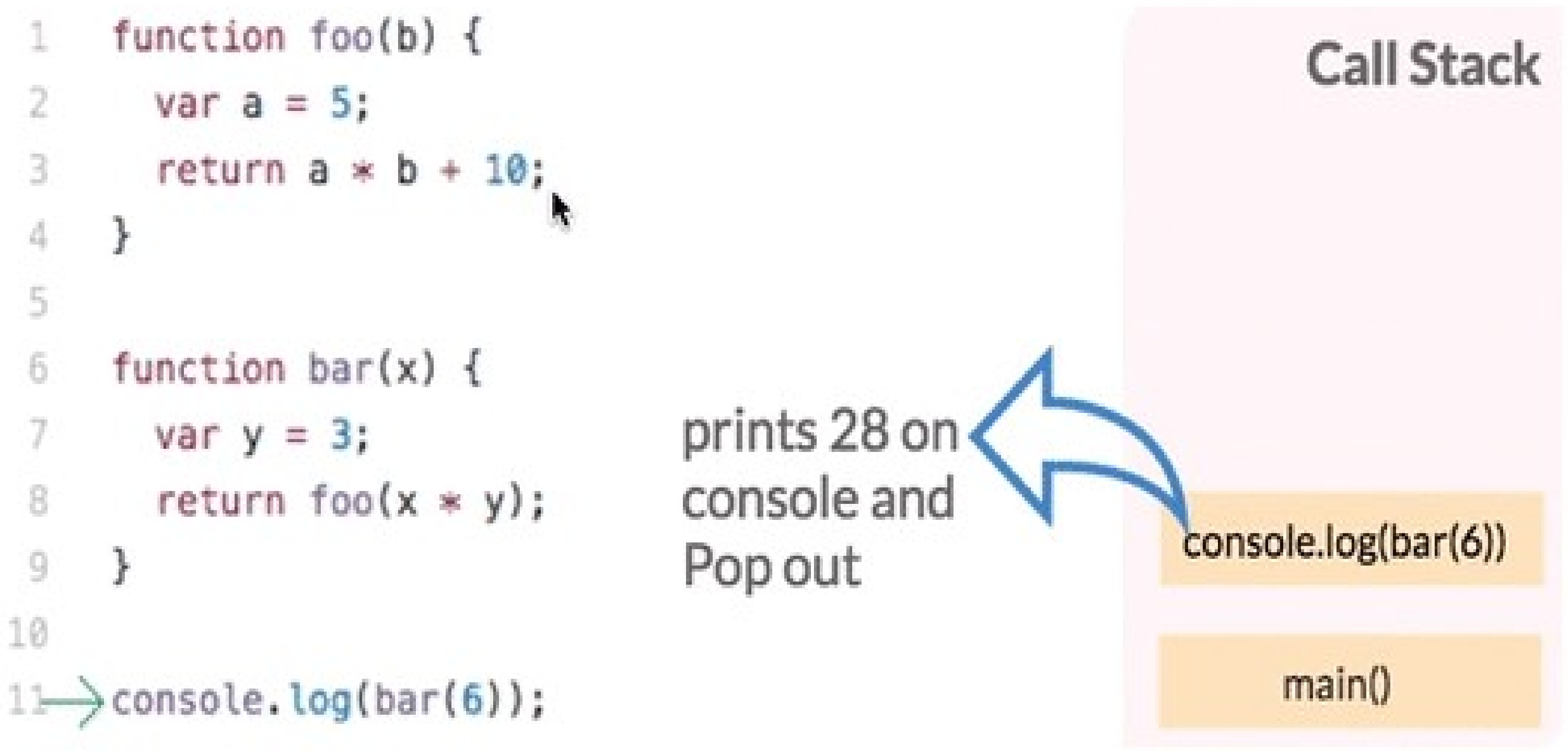


- Balancing symbols: ((()())(()))

```
stack<char> s;
while *not end of file or input* {
    *read character* c
    if (c == '(') *then*
        s.push(c)
    if (c == ')') *then*
        if (s.empty()) *then*
            *error*
        else
            s.pop();
}
if (!s.empty()) *then*
    *error*
else
    *okay*
```

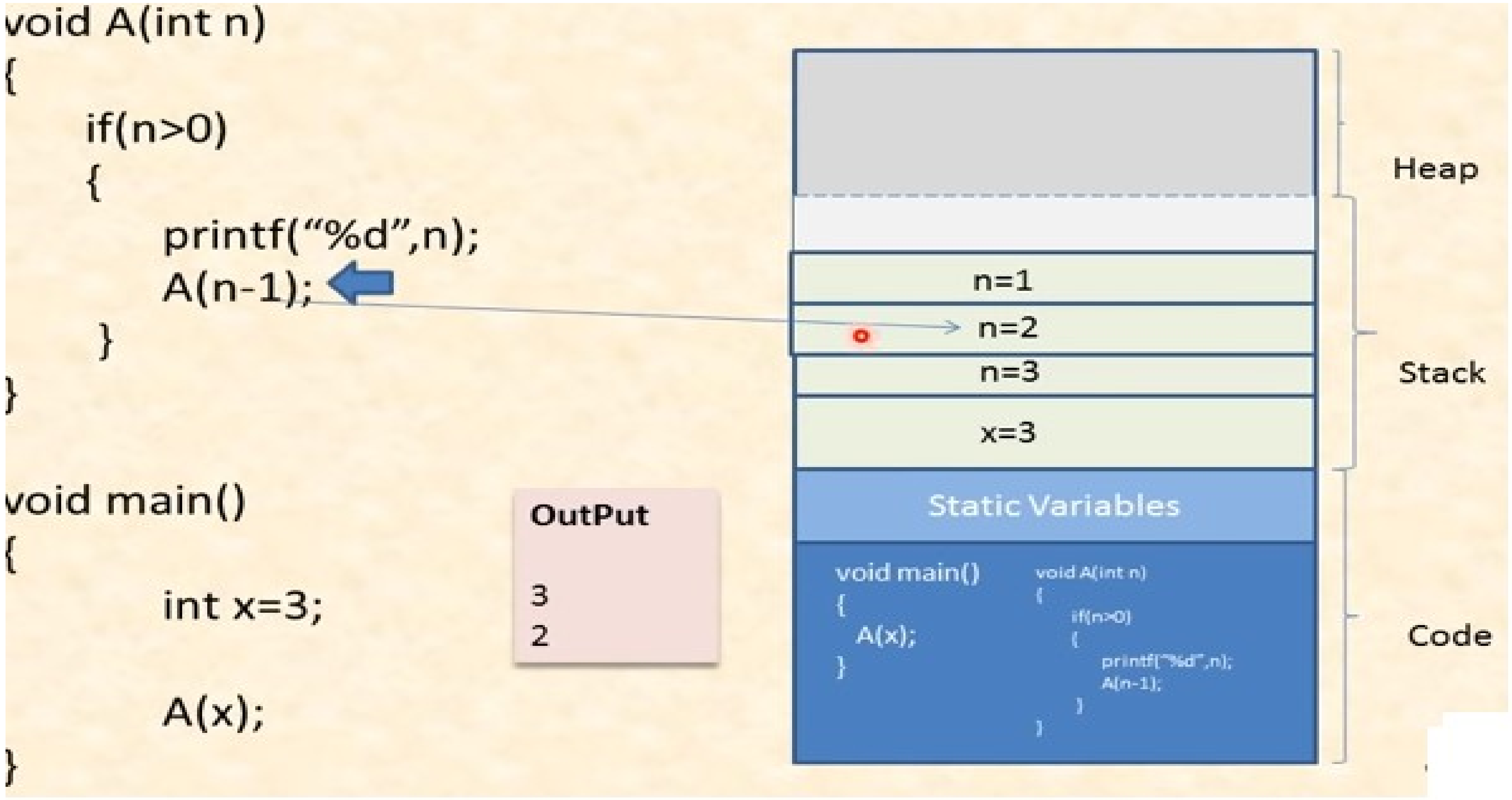
- Make an empty stack
- Read characters until end of file
- If a character is an opening symbol, push it onto to the stack
- If it is a closing symbol, then if the stack is empty report an error, otherwise pop the stack
- If the symbol popped is not the corresponding opening symbol, then report an error
- At the EOF, the stack is not empty report an error

Applications of Stack-Function Calls



```
function foo(b) {
  var a = 5;
  return a * b + 10;
}

function bar(x) {
  var y = 3;
  return foo(x * y);
}

console.log(bar(6));
```

Applications of Stack-Function Calls



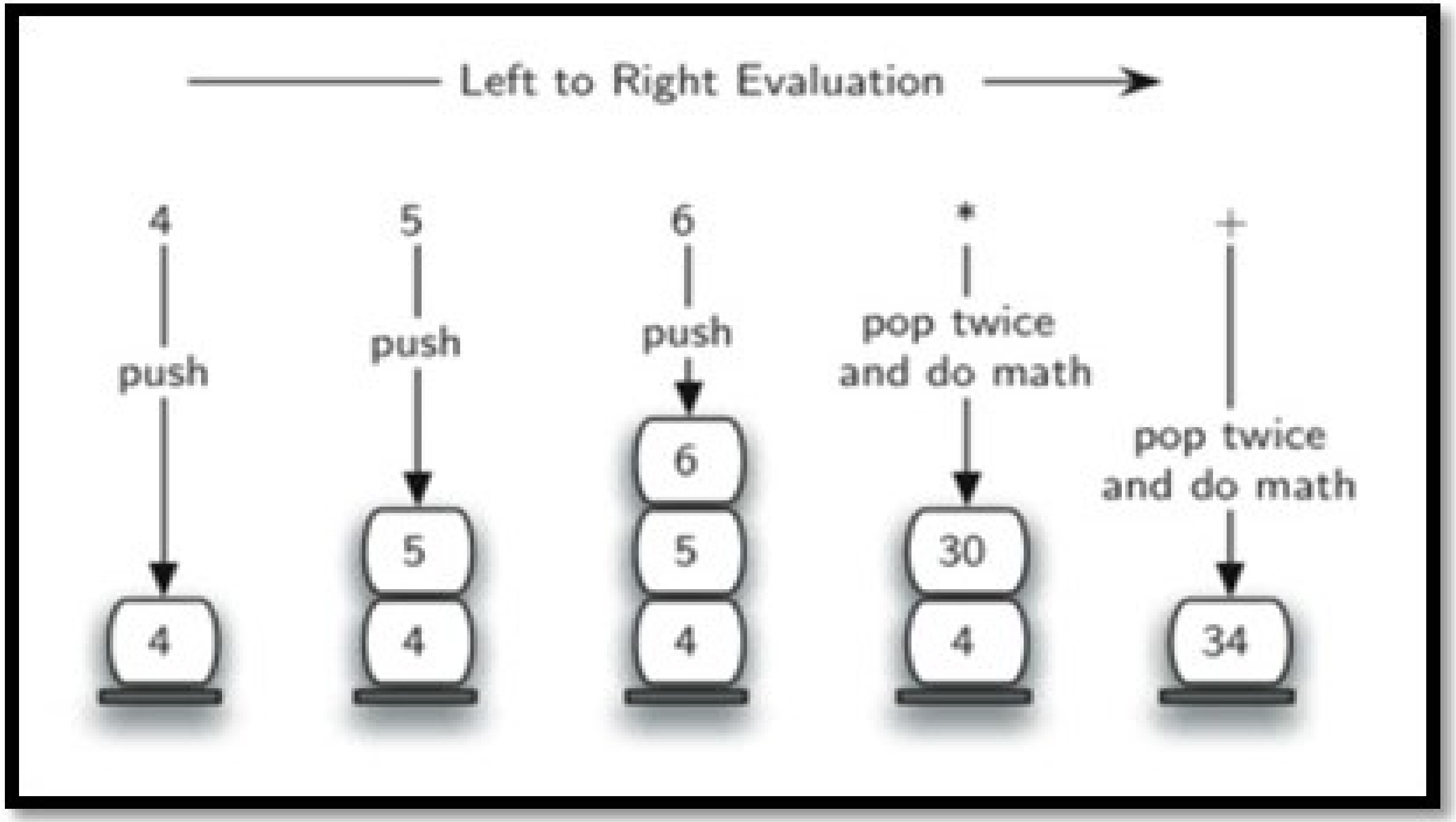
```
void A(int n)
{
     if(n>0)
     {
          printf("%d",n);
          A(n-1);
     }
}

void main()
{
          int x=3;

          A(x);
}
```

```
OutPut

3
2
```

```
void main()
{
  A(x);
}

void A(int n)
{
     if(n>0)
     {
          printf("%d",n);
          A(n-1);
     }
}
```

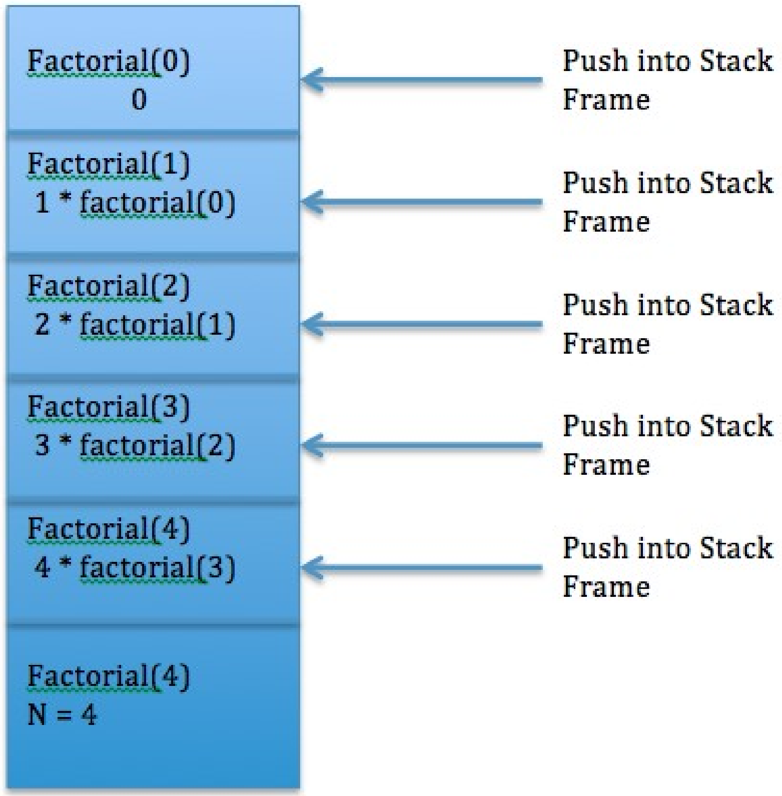
Applications of Stack- Evaluation of Postfix Expression

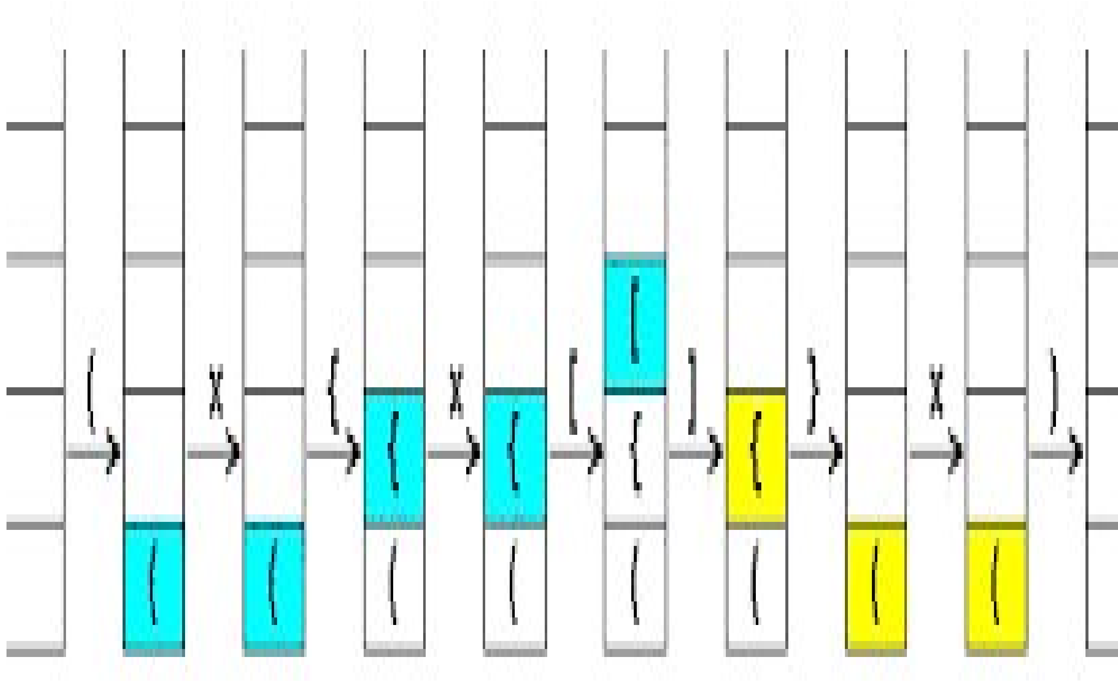
- **Expression: 456*+**



ASSESMENT 2

Guess this Application where the concept of stack is used.





References

1. M. A. Weiss, “Data Structures and Algorithm Analysis in C”, Pearson Education, 2nd Edition, 2002.
2. A. V. Aho, J. E. Hopcroft and J. D. Ullman, “Data Structures and Algorithms”, Pearson Education, 2nd Edition, 2007
3. Ashok Kamthane, " Data Structures Using C ", Pearson Education, 2nd Edition, 2012.
4. Sahni Horowitz, “Fundamentals of Data Structures in C”Universities Press; Second edition 2008.

Thank You