

1 Examples

2 Balancing Parentheses

```
def balanced_parens(text):  
    """  
    Checks whether every ( has a matching ).  
  
    >>> balanced_parens('(()())')  
    True  
    >>> balanced_parens('(()((()((()()))))()((()'))')  
    True  
    >>> balanced_parens('))')((( '  
    False  
    >>> balanced_parens('(()()')  
    False  
    >>> balanced_parens('(()())')  
    False  
    """  
    stack = []  
    for i, symbol in enumerate(text):  
        if symbol == '(':  
            stack.append(symbol)  
        else:  
            if len(stack) == 0:  
                return False  
            stack.pop()  
    return len(stack) == 0
```

```

def balanced_parens2(text):
    """
    Checks whether all of the following types of parentheses are balanced: ([{

    >>> balanced_parens2('(()())')
    True
    >>> balanced_parens2('((()((()((())))))(()())')
    True
    >>> balanced_parens2('([[]{}])')
    True
    >>> balanced_parens2('([[]{}[]{}{(){}})')
    True
    >>> balanced_parens2('([{}])')
    False
    >>> balanced_parens2('))')
    False
    >>> balanced_parens2('(()()')
    False
    >>> balanced_parens2('(()())')
    False
    """
    stack = []
    for i, symbol in enumerate(text):
        if symbol in '([{':
            stack.append(symbol)
        else:
            if len(stack) == 0:
                return False
            if (stack[-1] == '(' and symbol == ')') or \
                (stack[-1] == '[' and symbol == ']') or \
                (stack[-1] == '{' and symbol == '}'):
                stack.pop()
            else:
                return False
    return len(stack) == 0

```