

CSC148 Worksheet 9 Solution

Hyungmo Gu

April 21, 2020

Question 1

Expression 1:

~~(~~ 6
+
) 8
*
9
-
1
(

Stack

Were the parentheses balanced?

Yes

☒ No

Expression 2:

~~5~~ a
~~2~~)
b
-
2
/
)
)
)
b
+
)
3
*
~~1~~
~~3~~
~~4~~ (

Stack

Were the parentheses balanced?

☒ Yes

No

Expression 3 :

(

6
)
+
4

Stack

Were the parentheses balanced?

Yes ☒ No

Question 2

- a. For each character being received,
 1. If the character is left parenthesis, then we need to store it in stack using *push()* method
 2. If the character is right parenthesis
 1. First, check for the non-emptiness of stack.
 2. If the list is not empty, then we need to pop an element form stack.
 3. If list is empty, then we need to raise error.
 3. If the character is other than left or right parenthesis, then pass the character.
- b. We will know the parenthesis are balanced when the number of elements in stack is zero after traversing a string.

Question 3

```
1 def is_balanced(line: str) -> bool:
2     """Return whether <line> contains balanced parentheses.
3
4     Ignore square and curly brackets.
5
```

```

6     >>> is_balanced('(a * (3 + b))')
7     True
8     >>> is_balanced('(a * (3 + b]]') # Note that the two ']'s don't
matter
9     False
10    >>> is_balanced('1 + 2(x-y)}') # Note that the '}' doesn't matter
11    True
12    >>> is_balanced('3 - (x')
13    False
14    """
15    parenthesis_stack = Stack()
16
17    for character in line:
18        # If the character is left parenthesis,
19        if character == '(':
20            # Store it in stack
21            parenthesis_stack.push('(')
22        # If the character is right parenthesis,
23        elif character == ')':
24            # Check for the non-emptiness of stack.
25            if parenthesis_stack.is_empty():
26                # if empty, return false.
27                return False
28
29            # If the list is not empty, then pop an element form stack
30            .
31            parenthesis_stack.pop()
32
33        # Check parenthesis are balanced by checking stack is empty.
34        if not parenthesis_stack.is_empty():
35            return False
36
37    return True

```

Listing 1: worksheet_9_q3_solution.py

Question 4

```

1    def is_balanced(line: str) -> bool:
2        """Return whether <line> contains balanced parentheses.
3
4        >>> is_balanced('abc')
5        True
6        >>> is_balanced('(a * (3 + b))')
7        True
8        >>> is_balanced('(a * (3 + b]]')
9        False
10       >>> is_balanced('(a * [3 + b]]')
11       True
12       >>> is_balanced('1 + 2(x-y)}')
13       False
14       >>> is_balanced('{3 + [2 * 4(x-y)]}')
15       True
16       >>> is_balanced('3 - (x')

```

```

17     False
18     """
19     brackets_stack = Stack()
20
21     for character in line:
22         # If the character is one of '[', '{', or '(',
23         if (character == '(' or
24             character == '[' or
25             character == '{'):
26             # Store it in stack
27             brackets_stack.push(character)
28         # If the character is one of ']', '}', or ')',
29         elif (character == ')' or
30             character == ']' or
31             character == '}'):
32             # Check for the non-emptiness of stack.
33             if brackets_stack.is_empty():
34                 # if empty, return false.
35                 return False
36
37             # If the list is not empty, then pop an element form stack
38             .
39             left_bracket = brackets_stack.pop()
40
41             # If popped bracket doesn't match, then return false
42             if ((left_bracket == '(' and character != ')') or
43                 (left_bracket == '[' and character != ']') or
44                 (left_bracket == '{' and character != '}')):
45
46                 return False
47
48         # Check parenthesis are balanced by checking stack is empty.
49         if not brackets_stack.is_empty():
50             return False
51
52     return True

```

Listing 2: worksheet_9_q4_solution.py