```
1
2   # HSLU / ICS/AIML : Modul ADS : Algorithmen & Datenstrukturen
3   # Path   : uebung02/al/aufgabe03
4   # Version: Mon Sep 25 19:11:42 CEST 2023
5
6   from uebung02.al.aufgabe03.empty_stack_exception import EmptyStackException
7   import sys
8
9
10  class StackImplementation:
11    '''
12    Stack: a collection of objects that are inserted and removed according
13    to the last-in first-out principle.
14    '''
15    # --- nested _Node class: -----------------
16    class _Node:
17
18      def __init__(self, elem):
19        self._element = elem
20        self._next = None
21
22      def append_node(self, nextNode):
23        self._next = nextNode
24
25      def get_next(self):
26        return self._next
27
28      def get_element(self):
29        return self._element
30
31
32    # --- stack methods: -----------------
33
34    def __init__(self):
35      self._top = None
36      self._size = 0
37
38    def __len__(self):
39      return self._size
40
41    def size(self):
42      # TODO: Implement here...
43      pass
44
45    def is_empty(self):
46      # TODO: Implement here...
47      return True
48
49    def top(self):
50      # TODO: Implement here...
51      return None
52
53    def push(self, element):
54      # TODO: Implement here...
55      pass
56
57    def pop(self):
58      # TODO: Implement here...
59      return None
60
61    def printout(self):
62      print("Stack: (", self._to_string(self._top, ""), ")")
63
64    def _to_string(self, node, content):
65      if node == None:
66        return content
67      if not content == "":
68        content += ", "
69      content += str(node.get_element())
70      return self._to_string(node.get_next(), content)
71
```



```
72
73  if __name__ == '__main__':
74    stack = StackImplementation()
75    stack.printout()
76    TEST_SIZE = 4
77    for i in range(TEST_SIZE):
78      stack.push(i)
79      stack.printout()
80      if stack.size() != i+1:
81        print("ERROR: Stack.size() != ", i+1)
82        sys.exit()
83      if stack.top() != i:
84        print("ERROR: Stack.top() != ", i)
85        sys.exit()
86    for i in range(TEST_SIZE-1, 0, -1):
87      if stack.pop() != i:
88        print("ERROR: Stack.pop() != ", i)
89        sys.exit()
90      stack.printout()
91      if stack.size() != i:
92        print("ERROR: Stack.size() != ", i)
93        sys.exit()
94      if stack.top() != i-1:
95        print("ERROR: Stack.top() != ", i-1)
96        sys.exit()
97    if stack.pop() != 0:
98      print("ERROR: Stack.pop() != ", 0)
99      sys.exit()
100   stack.printout()
101   if not stack.is_empty():
102     print("ERROR: Stack.empty() != true")
103     sys.exit()
104   if stack.size() != 0:
105     print("ERROR: Stack.size() != 0")
106     sys.exit()
107   try:
108     stack.top()
109     print("ERROR: no EmptyStackException for stack.top()!")
110     sys.exit()
111   except EmptyStackException:
112     pass
113   try:
114     stack.pop()
115     print("ERROR: no EmptyStackException for stack.pop()!")
116     sys.exit()
117   except EmptyStackException:
118     pass
119
120
121 """ Session-Log:
122
123 Stack: (  )
124 Stack: ( 0 )
125 Stack: ( 1, 0 )
126 Stack: ( 2, 1, 0 )
127 Stack: ( 3, 2, 1, 0 )
128 Stack: ( 2, 1, 0 )
129 Stack: ( 1, 0 )
130 Stack: ( 0 )
131 Stack: (  )
132
133 """
134
```

```
1
2  # HSLU / ICS/AIML : Modul ADS : Algorithmen & Datenstrukturen
3  # Path   : uebung02/al/aufgabe03
4  # Version: Mon Sep 25 19:11:42 CEST 2023
5
6  class EmptyStackException(Exception):
7
8      def __init__(self, err):
9        super().__init__(err)
10
11
```