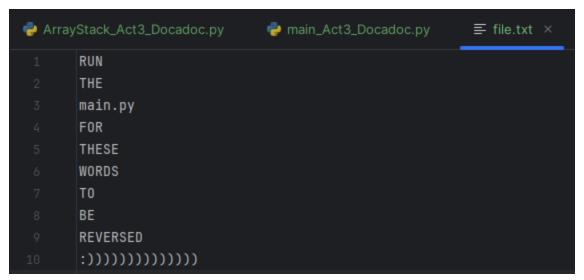
```
∨ class ArrayStack:

          def __init__(self):
              self.data = []
          def __len__(self):
              return len(self.data)
          def is_empty(self):
              return len(self.data) == 0
          def push(self, val):
              self.data.append(val)
          def top(self):
              if self.is_empty():
                  raise Exception('Stack is empty')
              return self.data[-1]
          def pop(self):
              if self.is_empty():
                  raise Exception('Stack is empty')
              return self.data.pop()
26 ▷ ∨ if __name__ == "__main__":
          stack = ArrayStack()
          stack.push(1)
          stack.push(2)
          print("Top element:", stack.top())
          print("Stack size:", len(stack))
          print("Popped element:", stack.pop())
          print("Stack is empty:". stack.is empty())
```

```
ArrayStack_Act3_Docadoc.py
                              main_Act3_Docadoc.py ×
      from ArrayStack_Act3_Docadoc import ArrayStack
      def correct(expression):
          stack = ArrayStack()
          matching_pairs = {')': '(', '}': '{', ']': '['}
          for char in expression:
              if char in '({[':
                  stack.push(char)
              elif char in ')}]':
                  if stack.is_empty() or stack.pop() != matching_pairs[char]:
                      return False
          return stack.is_empty()
      def reverse(file0):
          stack = ArrayStack()
          with open(file0, 'r') as file:
              for line in file:
                  stack.push(line.rstrip('\n'))
          with open(file0, 'w') as file:
              while not stack.is_empty():
                  file.write(stack.pop() + '\n')
      def main():
          user_input = input("Enter an expression to see if it is balanced: ")
          result = correct(user_input)
          print(f"Expression: {user_input} is {'correct' if result else 'incorrect'}")
          file1 = 'file.txt'
          reverse(file1)
          print(f"The input per line in {file1} has been reversed.")
36 ▶ if __name__ == "__main__":
          main()
```



```
Z:\DSAL601-1DB2\venv\Scripts\python.exe Z:\DSAL601-1DB2\MIDTERMS\main_Act3_Docadoc.py
Enter an expression to see if it is balanced: ()()
Expression: ()() is correct
The input per line in file.txt has been reversed.

Process finished with exit code 0
```