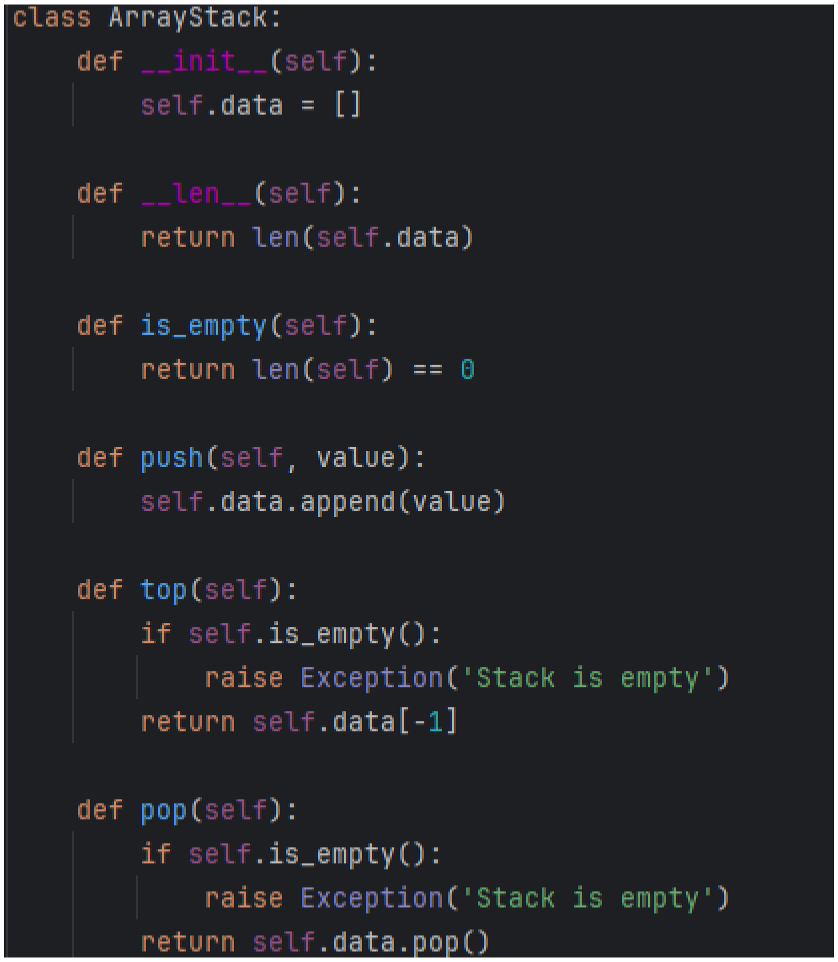
Charles Hanzel D. Gerardo
DSALGO
IDB2
October 18, 2024

Code (ArrayStack)

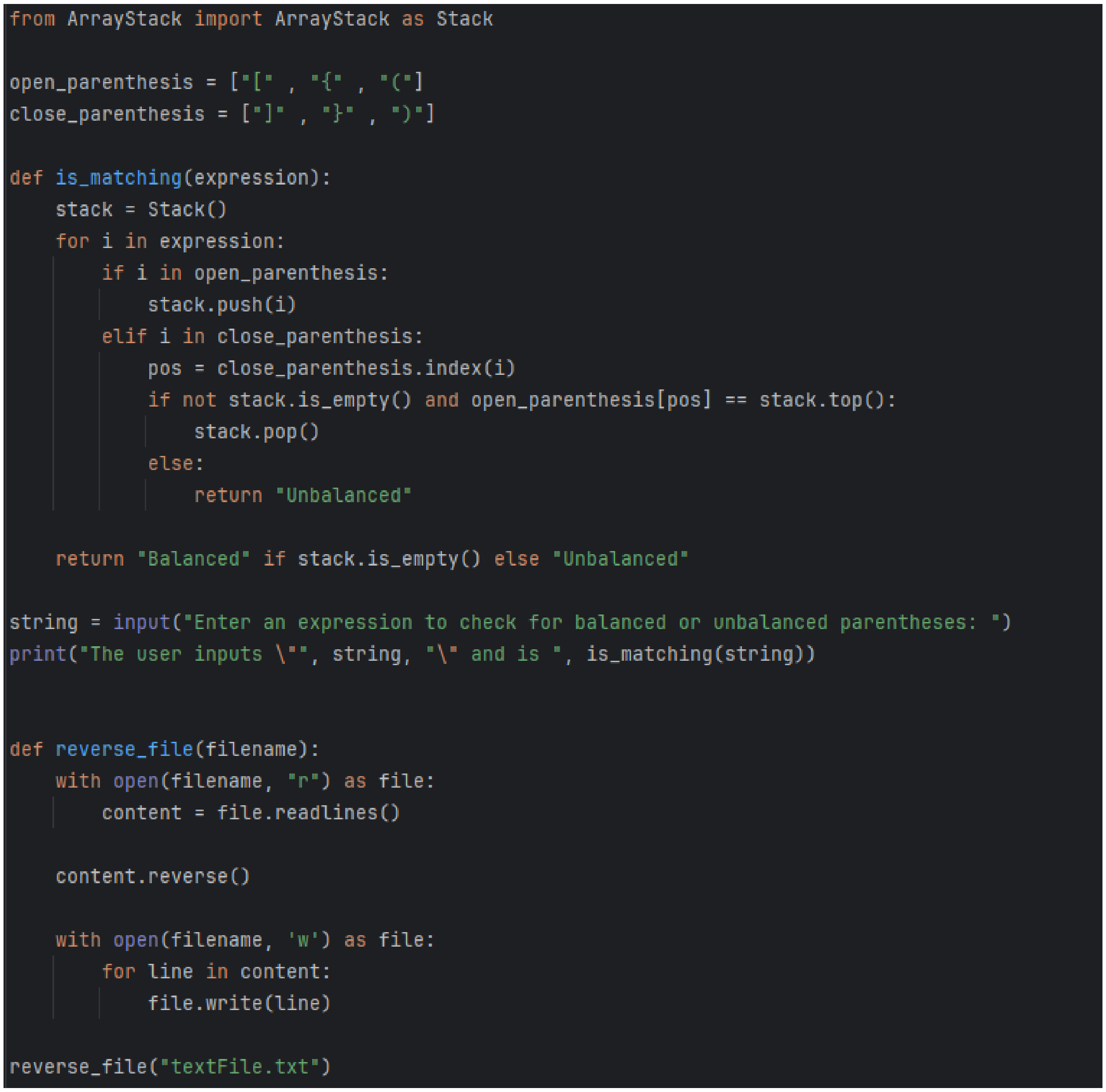


```python
class ArrayStack:
    def __init__(self):
        self.data = []

    def __len__(self):
        return len(self.data)

    def is_empty(self):
        return len(self) == 0

    def push(self, value):
        self.data.append(value)

    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()
```

```python
class ArrayStack:
    def __init__(self):
        self.data = []

    def __len__(self):
        return len(self.data)

    def is_empty(self):
        return len(self) == 0

    def push(self, value):
        self.data.append(value)

    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()
```

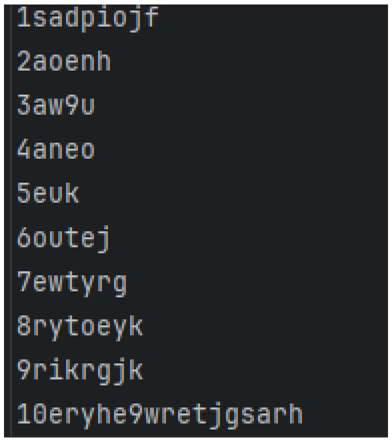
Code(main)



```
from ArrayStack import ArrayStack as Stack

open_parenthesis = ["[" , "{" , "("]
close_parenthesis = ["]" , "}" , ")"]

def is_matching(expression):
    stack = Stack()
    for i in expression:
        if i in open_parenthesis:
            stack.push(i)
        elif i in close_parenthesis:
            pos = close_parenthesis.index(i)
            if not stack.is_empty() and open_parenthesis[pos] == stack.top():
                stack.pop()
            else:
                return "Unbalanced"

    return "Balanced" if stack.is_empty() else "Unbalanced"

string = input("Enter an expression to check for balanced or unbalanced parentheses: ")
print("The user inputs \"", string, "\" and is ", is_matching(string))


def reverse_file(filename):
    with open(filename, "r") as file:
        content = file.readlines()

    content.reverse()

    with open(filename, 'w') as file:
        for line in content:
            file.write(line)

reverse_file("textFile.txt")
```

```
from ArrayStack import ArrayStack as Stack

open_parenthesis = ["[" , "{" , "("]
close_parenthesis = ["]" , "}" , ")"]

def is_matching(expression):
    stack = Stack()
    for i in expression:
        if i in open_parenthesis:
            stack.push(i)
        elif i in close_parenthesis:
            pos = close_parenthesis.index(i)
            if not stack.is_empty() and open_parenthesis[pos] == stack.top():
                stack.pop()
            else:
                return "Unbalanced"

    return "Balanced" if stack.is_empty() else "Unbalanced"

string = input("Enter an expression to check for balanced or unbalanced parentheses: ")
print("The user inputs \"", string, "\" and is ", is_matching(string))



def reverse_file(filename):
    with open(filename, "r") as file:
        content = file.readlines()

    content.reverse()

    with open(filename, 'w') as file:
        for line in content:
            file.write(line)

reverse_file("textFile.txt")
```

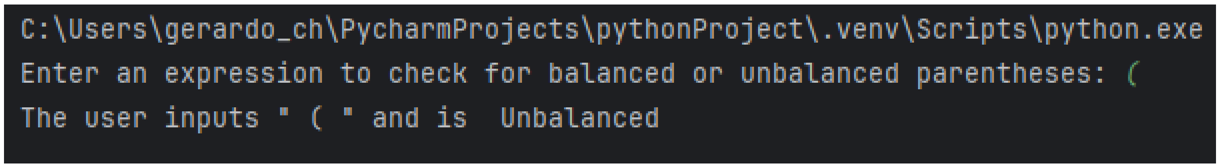
Text file (textFile)



```
1sadpiojf
2aoenh
3aw9u
4aneo
5euk
6outej
7ewtyrg
8rytoeyk
9rikrgjk
10eryhe9wretjgsarh
```

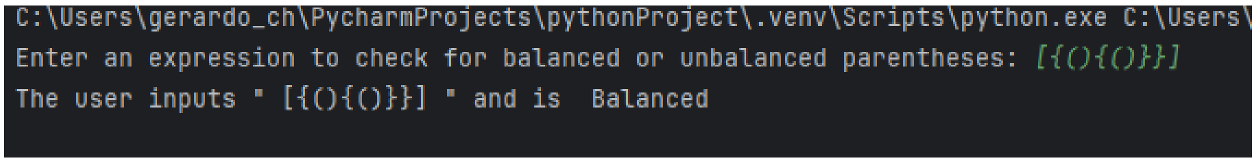
1sadpiojf
2aoenh
3aw9u
4aneo
5euk
6outej
7ewtyrg
8rytoeyk
9rikrgjk
10eryhe9wretjgsarh

Output



```
C:\Users\gerardo_ch\PycharmProjects\pythonProject\.venv\Scripts\python.exe
Enter an expression to check for balanced or unbalanced parentheses: (
The user inputs " ( " and is  Unbalanced
```

```
10eryhe9wretjgsarh
9rikrgjk
8rytoeyk
7ewtyrg
6outej
5euk
4aneo
3aw9u
2aoenh
1sadpiojf
```



```
C:\Users\gerardo_ch\PycharmProjects\pythonProject\.venv\Scripts\python.exe C:\Users\
Enter an expression to check for balanced or unbalanced parentheses: [{(){()}}]
The user inputs " [{(){()}}] " and is  Balanced
```

```
1sadpiojf
2aoenh
3aw9u
4aneo
5euk
6outej
7ewtyrg
8rytoeyk
9rikrgjk
10eryhe9wretjgsarh
```