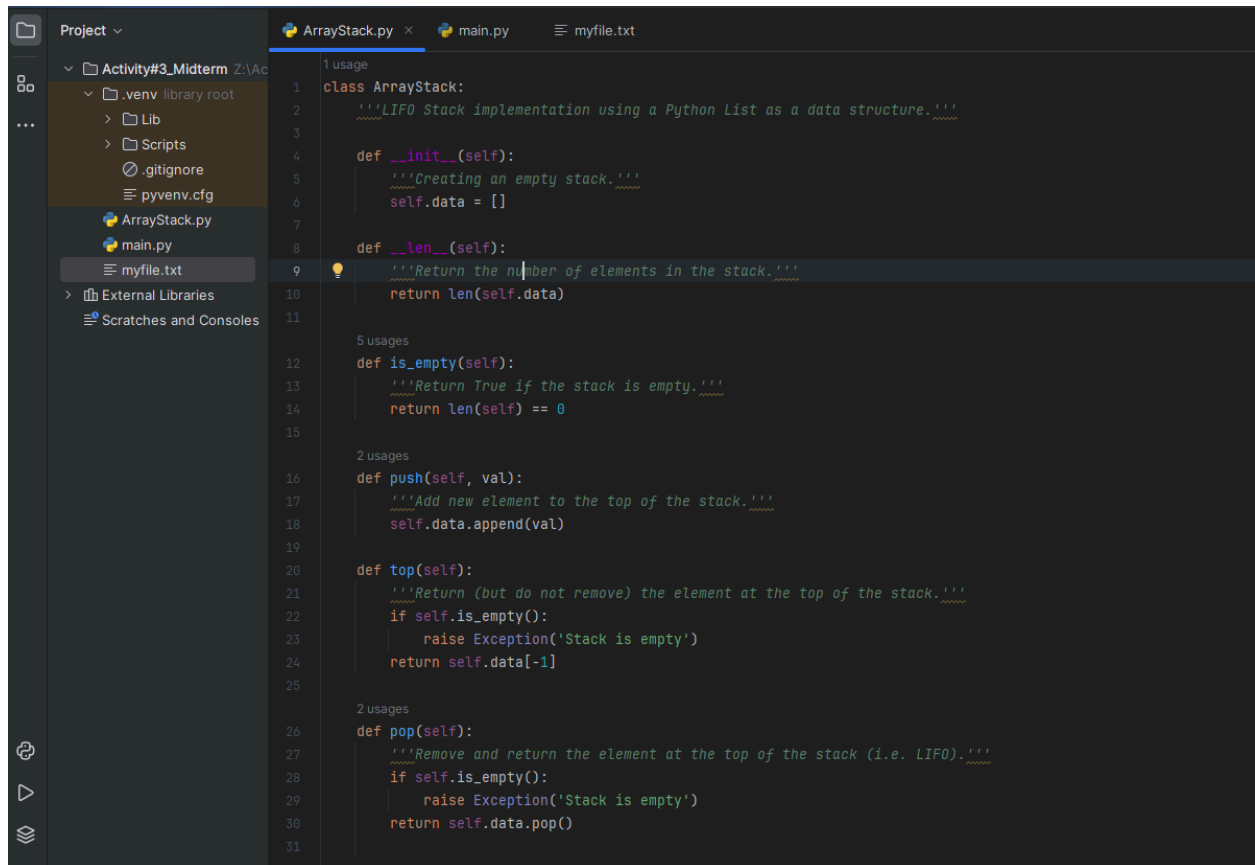


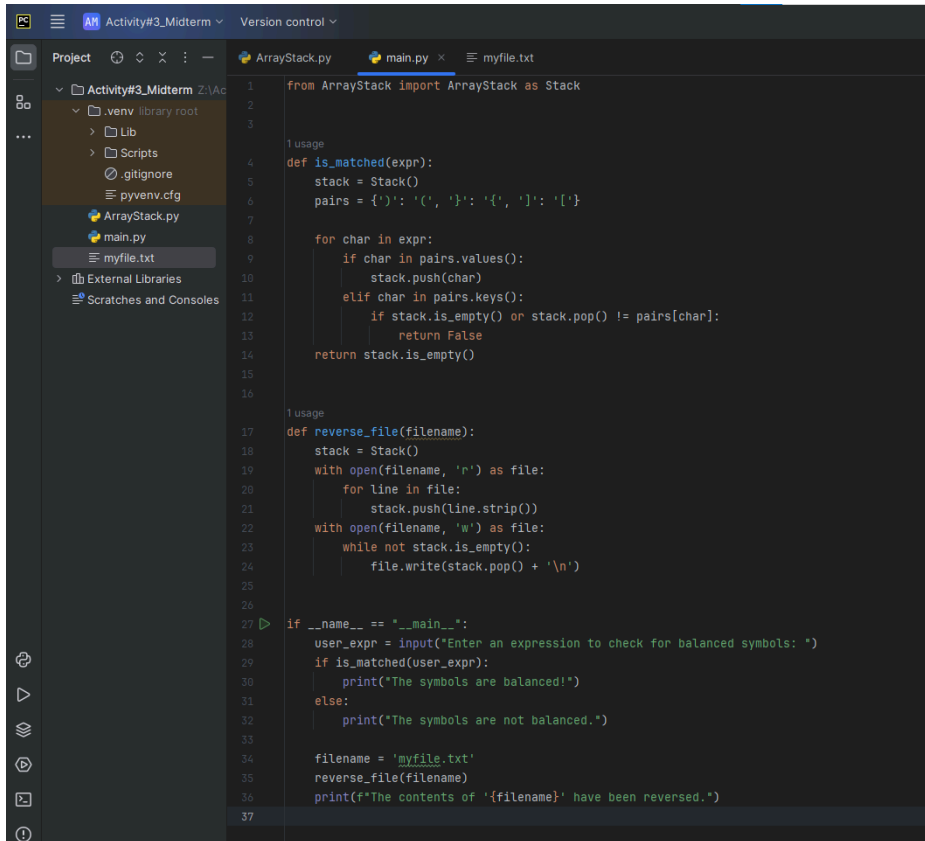
ARRAY STACK



The screenshot shows a code editor with a project sidebar on the left and a code editor on the right. The sidebar shows a project named 'Activity#3_Midterm' with a file structure including '.venv', 'Lib', 'Scripts', '.gitignore', 'pyvenv.cfg', 'ArrayStack.py', 'main.py', and 'myfile.txt'. The code editor shows the implementation of the 'ArrayStack' class in 'ArrayStack.py'.

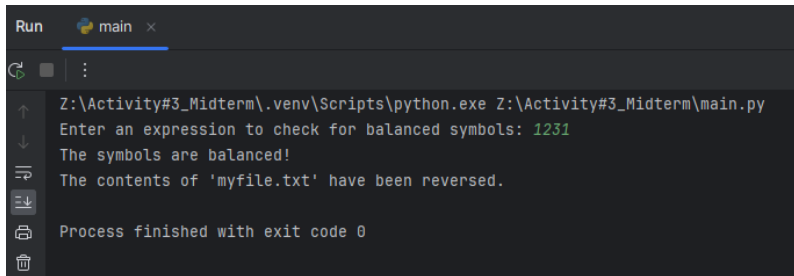
```
1 usage
2 class ArrayStack:
3     """LIFO Stack implementation using a Python List as a data structure."""
4
5     def __init__(self):
6         """Creating an empty stack."""
7         self.data = []
8
9     def __len__(self):
10        """Return the number of elements in the stack."""
11        return len(self.data)
12
13    5 usages
14    def is_empty(self):
15        """Return True if the stack is empty."""
16        return len(self) == 0
17
18    2 usages
19    def push(self, val):
20        """Add new element to the top of the stack."""
21        self.data.append(val)
22
23    def top(self):
24        """Return (but do not remove) the element at the top of the stack."""
25        if self.is_empty():
26            raise Exception('Stack is empty')
27        return self.data[-1]
28
29    2 usages
30    def pop(self):
31        """Remove and return the element at the top of the stack (i.e. LIFO)."""
32        if self.is_empty():
33            raise Exception('Stack is empty')
34        return self.data.pop()
```

MAIN.PY




```
1 from ArrayStack import ArrayStack as Stack
2
3
4 usage
5 def is_matched(expr):
6     stack = Stack()
7     pairs = {'(': ')', '[': ']', '{': '}'
8
9     for char in expr:
10         if char in pairs.values():
11             stack.push(char)
12         elif char in pairs.keys():
13             if stack.is_empty() or stack.pop() != pairs[char]:
14                 return False
15     return stack.is_empty()
16
17 usage
18 def reverse_file(filename):
19     stack = Stack()
20     with open(filename, 'r') as file:
21         for line in file:
22             stack.push(line.strip())
23     with open(filename, 'w') as file:
24         while not stack.is_empty():
25             file.write(stack.pop() + '\n')
26
27 if __name__ == "__main__":
28     user_expr = input("Enter an expression to check for balanced symbols: ")
29     if is_matched(user_expr):
30         print("The symbols are balanced!")
31     else:
32         print("The symbols are not balanced.")
33
34     filename = 'myfile.txt'
35     reverse_file(filename)
36     print(f"The contents of '{filename}' have been reversed.")
37
```

OUTPUT:



```
Run main x
Z:\Activity#3_Midterm\.venv\Scripts\python.exe Z:\Activity#3_Midterm\main.py
Enter an expression to check for balanced symbols: 1231
The symbols are balanced!
The contents of 'myfile.txt' have been reversed.
Process finished with exit code 0
```



```
Run main x
Z:\Activity#3_Midterm\.venv\Scripts\python.exe Z:\Activity#3_Midterm\main.py
Enter an expression to check for balanced symbols: ]][[[[
The symbols are not balanced.
The contents of 'myfile.txt' have been reversed.
Process finished with exit code 0
```

```
ArrayStack.py  main.py  myfile.txt x
1  Y
2  A
3  U
4  G
5  -
6  Y
7  E
8  L
9  T
10 S
11 E
12 R
13
```

MYFILE.TXT

```
ArrayStack.py  main.py  myfile.txt x
1  R
2  E
3  S
4  T
5  L
6  E
7  Y
8  -
9  G
10 U
11 A
12 Y
13
```