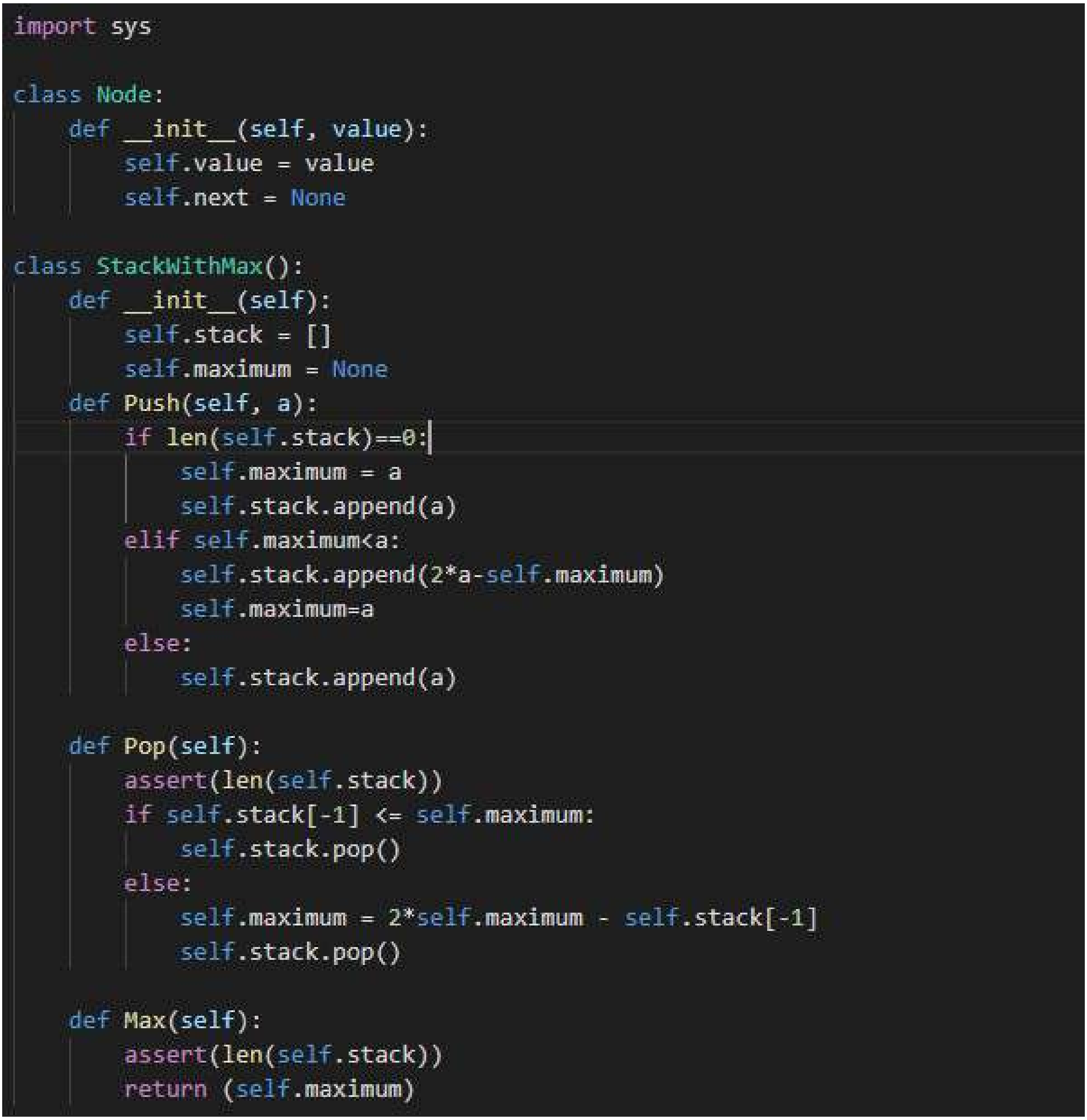
Extending stack interface

Stack with Max (Naive)



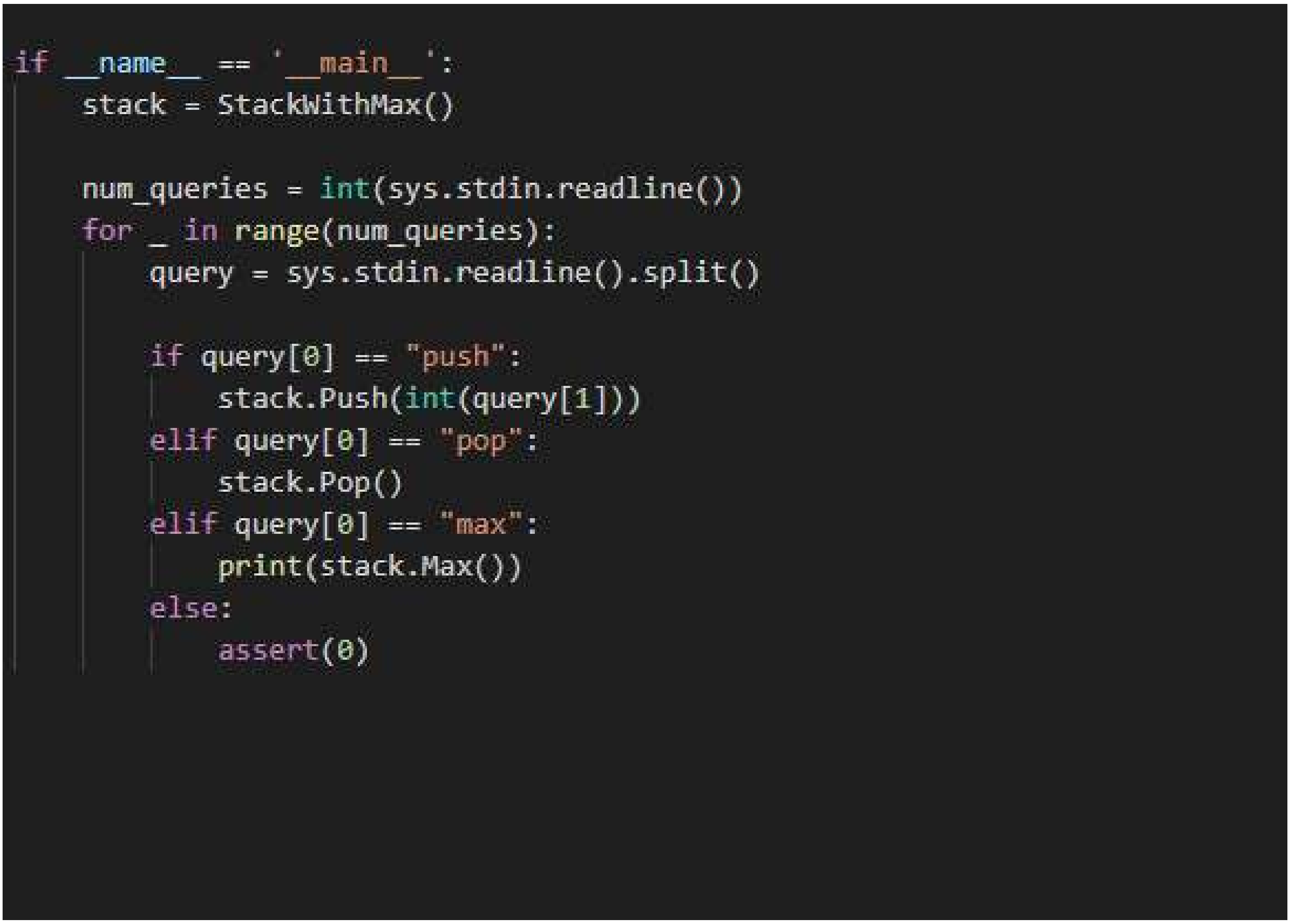
```python
import sys

class Node:
    def __init__(self, value):
        self.value = value
        self.next = None

class StackWithMax():
    def __init__(self):
        self.stack = []
        self.maximum = None
    def Push(self, a):
        if len(self.stack)==0:
            self.maximum = a
            self.stack.append(a)
        elif self.maximum<a:
            self.stack.append(2*a-self.maximum)
            self.maximum=a
        else:
            self.stack.append(a)

    def Pop(self):
        assert(len(self.stack))
        if self.stack[-1] <= self.maximum:
            self.stack.pop()
        else:
            self.maximum = 2*self.maximum - self.stack[-1]
            self.stack.pop()

    def Max(self):
        assert(len(self.stack))
        return (self.maximum)
```



```python
if __name__ == '__main__':
    stack = StackWithMax()

    num_queries = int(sys.stdin.readline())
    for _ in range(num_queries):
        query = sys.stdin.readline().split()

        if query[0] == "push":
            stack.Push(int(query[1]))
        elif query[0] == "pop":
            stack.Pop()
        elif query[0] == "max":
            print(stack.Max())
        else:
            assert(0)
```