

Problem 5: Implement Min Stack | $O(2N)$ and $O(N)$ Space Complexity. Design a stack that supports push, pop, top, and retrieving the minimum element in constant time.

```
class MinStack:
```

```
    def __init__(self):
```

```
        self.stack = []
```

```
        self.min_stack = []
```

```
    def push(self, val):
```

```
        self.stack.append(val)
```

```
        if not self.min_stack or val <= self.min_stack[-1]:
```

```
            self.min_stack.append(val)
```

```
    def pop(self):
```

```
        if self.stack:
```

```
            val = self.stack.pop()
```

```
            if val == self.min_stack[-1]:
```

```
                self.min_stack.pop()
```

```
    def top(self):
```

```
        if self.stack:
```

```
            return self.stack[-1]
```

```
    def getMin(self):
```

```
        if self.min_stack:
```

```
        return self.min_stack[-1]

min_stack = MinStack()

print(min_stack.push(-2))

print(min_stack.push(0))

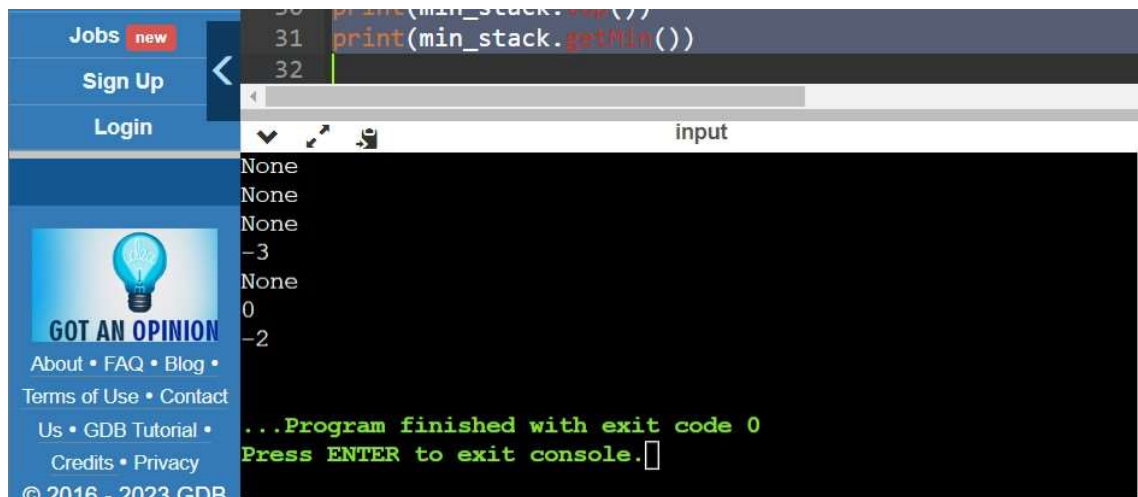
print(min_stack.push(-3))

print(min_stack.getMin())

print(min_stack.pop())

print(min_stack.top())

print(min_stack.getMin())
```



The screenshot shows a web browser on the left and a terminal window on the right. The browser has a blue sidebar with links: 'Jobs new', 'Sign Up', 'Login', 'GOT AN OPINION' (with a lightbulb icon), 'About • FAQ • Blog •', 'Terms of Use • Contact', 'Us • GDB Tutorial •', 'Credits • Privacy', and '© 2016 - 2023 GDB'. The terminal window shows the execution of a Python program. The code being executed is:

```
30 print(min_stack.top())
31 print(min_stack.getMin())
32
```

 The terminal output shows:

```
None
None
None
-3
None
0
-2
...Program finished with exit code 0
Press ENTER to exit console.
```