

### Problem 3: Implement a Stack using a single Queue.

```
class Stack:
```

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

```
        self.queue = []
```

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

```
        self.queue.append(value)
```

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

```
        if not self.is_empty():
```

```
            size = len(self.queue)
```

```
            for _ in range(size - 1):
```

```
                self.queue.append(self.queue.pop(0))
```

```
            return self.queue.pop(0)
```

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

```
        if not self.is_empty():
```

```
            size = len(self.queue)
```

```
            for _ in range(size - 1):
```

```
                self.queue.append(self.queue.pop(0))
```

```
            return self.queue[0]
```

```
    def size(self):
```

```
        return len(self.queue)
```

```
    def is_empty(self):
```

```
        return len(self.queue) == 0
```

```
    def print_stack(self):
```

```
        print("Stack:", self.queue)
```

```
stack = Stack()
```

```
stack.push(1)
```

```
stack.push(2)
stack.push(3)
stack.print_stack()
print("Size:", stack.size())
print("Top:", stack.top())
print("Pop:", stack.pop())
stack.print_stack()
```



The screenshot shows a code editor with the following Python code:

```
34 stack.push(3)
35 stack.print_stack()
36 print("Size:", stack.size())
37 print("Top:", stack.top())
38 print("Pop:", stack.pop())
39 stack.print_stack()
40
```

Below the code editor is a terminal window titled "input". The terminal output is as follows:

```
Stack: [1, 2, 3]
Size: 3
Top: 3
Pop: 2
Stack: [3, 1]
...Program finished with exit code 0
Press ENTER to exit console.
```

On the left side of the terminal window, there is a sidebar with the following text: "UNION", "Blog", "Contact", "utorial", "vacy", and "3 GDB".