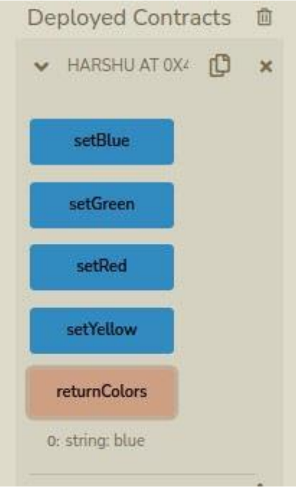
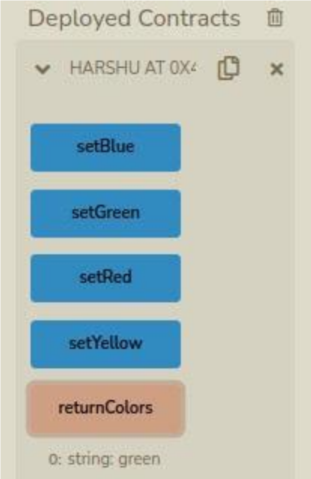
Practical 3:

Enums in Solidity

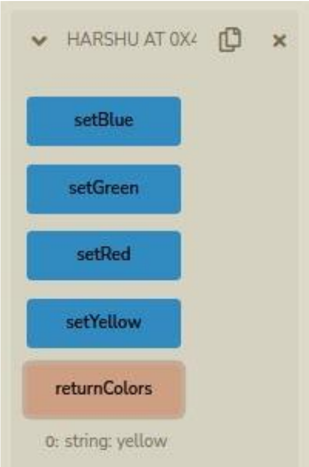
```
pragma solidity ^0.5.0;
contract harshu
{
    enum Colors{red,blue,green,yellow}
    Colors s;
    function setRed() public{
        s = Colors.red;
    }
    function setBlue() public{
        s = Colors.blue;
    }
    function setGreen() public{
        s = Colors.green;
    }function setYellow() public{
        s = Colors.yellow;
    }
    function returnColors() external view returns(string memory)
    {
        Colors x = s;
        if(x==Colors.red) return "red";
        if(x==Colors.blue) return "blue";
        if(x==Colors.green) return "green";
        if(x==Colors.yellow) return "yellow";
        return "";
    }
}
```



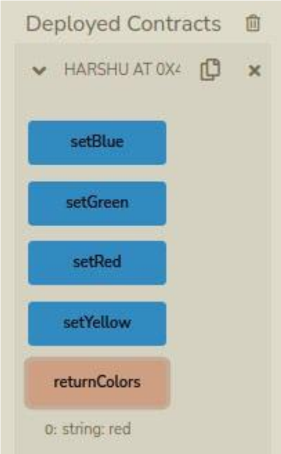
```
0: string: blue
```



```
0: string: green
```



```
0: string: yellow
```

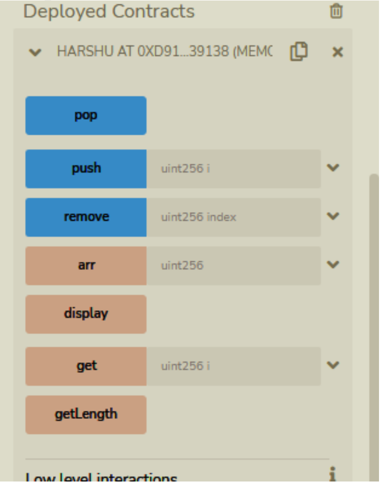


```
0: string: red
```

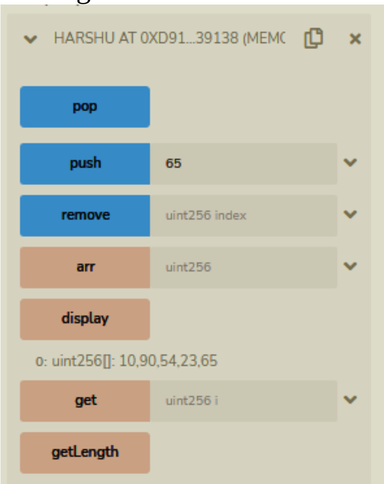
Arrays in Solidity

```
pragma solidity ^0.8.0;
/*
    --operations--
    1. push
    2. pop
    3. delete
    4. display
*/
contract harshu {
    uint[] public arr;
    function get(uint i) public view returns (uint) {
        return arr[i];
    }
    function display() public view returns (uint[] memory) {
        return arr;
    }
    function push(uint i) public {
        arr.push(i);
    }
    function pop() public {
        arr.pop();
    }
    function getLength() public view returns (uint) {
        return arr.length;
    }
    function remove(uint index) public {
        delete arr[index];
    }
}
```

Output



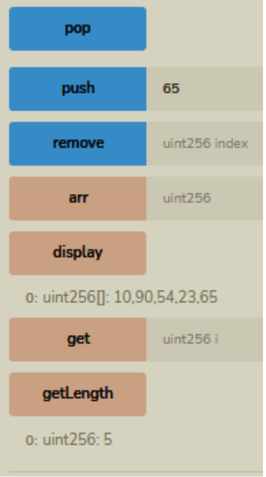
adding elements



push 65

```
0: uint256[]: 10,90,54,23,65
```

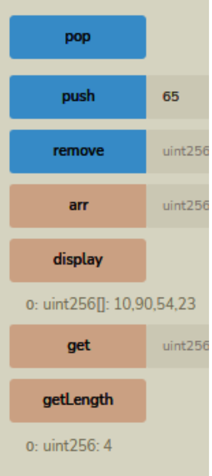
getting length



push 65

```
0: uint256[]: 10,90,54,23,65
0: uint256: 5
```

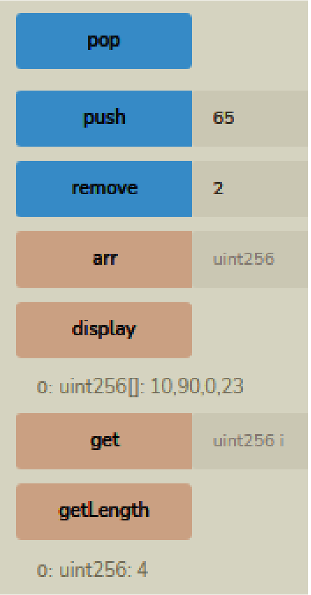
popping out element



push 65

```
0: uint256[]: 10,90,54,23
0: uint256: 4
```

Removing by index



0: uint256[]: 10,90,0,23

0: uint256: 4