```
Object.pick(obj[, pickedKeys | predictedFunction(currentValue[, key[, object]])[, thisArg])

Object.omit(obj[, omittedKeys | predictedFunction(currentValue[, key[, object]])[, thisArg])
```

How about the below for Stage-1?

O

b

j

C

t

fi

t

e

r

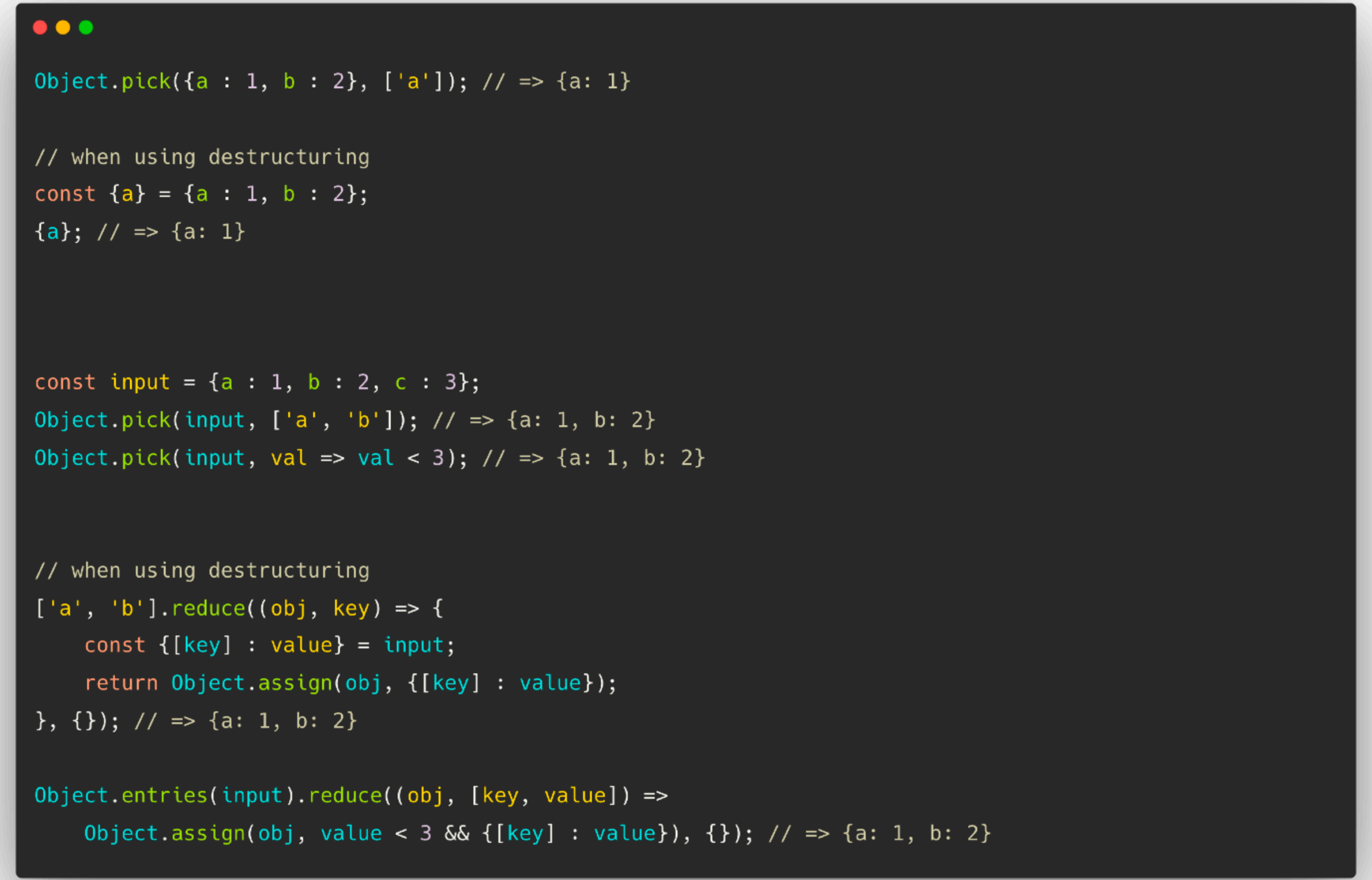
?

How about the below for Stage-1?

Object.pick(obj[, pickedKeys | predictedFunction(currentValue[, key[, object]])[, thisArg])

Object.omit(obj[, omittedKeys | predictedFunction(currentValue[, key[, object]])[, thisArg])

Object.filter()?


```
Object.pick({a : 1, b : 2}, ['a']); // => {a: 1}

// when using destructuring
const {a} = {a : 1, b : 2};
{a}; // => {a: 1}



const input = {a : 1, b : 2, c : 3};
Object.pick(input, ['a', 'b']); // => {a: 1, b: 2}
Object.pick(input, val => val < 3); // => {a: 1, b: 2}


// when using destructuring
['a', 'b'].reduce((obj, key) => {
    const {[key] : value} = input;
    return Object.assign(obj, {[key] : value});
}, {}); // => {a: 1, b: 2}

Object.entries(input).reduce((obj, [key, value]) =>
    Object.assign(obj, value < 3 && {[key] : value}), {}); // => {a: 1, b: 2}
```