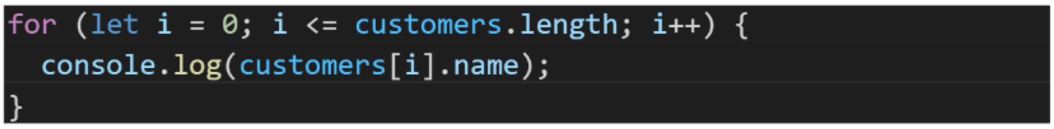
🐞 **Bug 1: Loop Output Issue**

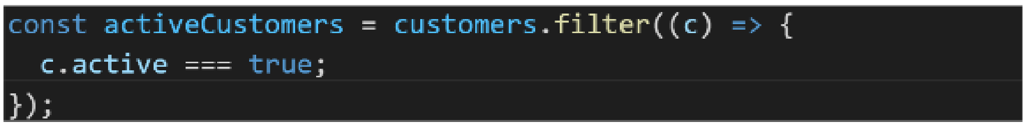


```
for (let i = 0; i <= customers.length; i++) {
  console.log(customers[i].name);
}
```

🔍 **Hint:** Array index & loop condition

```
for (let i = 0; i < customers.length; i++) {
  console.log(customers[i].name);
}
```

🐞 Bug 2: filter() Not Working

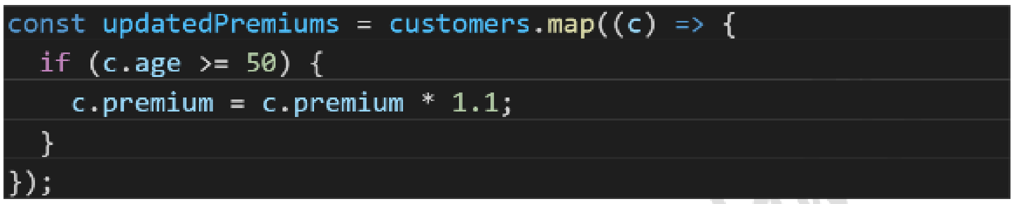


```
const activeCustomers = customers.filter((c) => {
  c.active === true;
});
```

🔍 **Hint:** Return value

```
const activeCustomers = customers.filter((c) => c.active === true);
```

🐞 Bug 3: Premium Increase Logic Broken

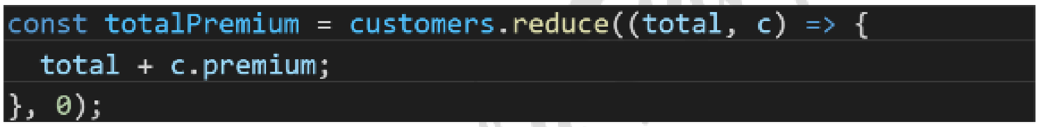


```
const updatedPremiums = customers.map((c) => {
  if (c.age >= 50) {
    c.premium = c.premium * 1.1;
  }
});
```

🔍 **Hint:** map return & immutability

```
const updatedPremiums = customers.map((c) => {
  if (c.age >= 50) return { ...c, premium: c.premium * 1.1 };
  return c;
});
```

🐞 Bug 4: Wrong Total Premium Calculation

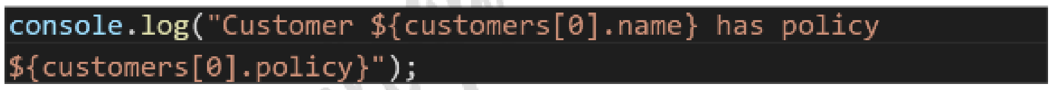


```
const totalPremium = customers.reduce((total, c) => {
  total + c.premium;
}, 0);
```

🔍 **Hint:** reduce return

```
const totalPremium = customers.reduce((total, c) => {
  return total + c.premium;
}, 0);
```

🐞 Bug 5: Template Literal Not Printing

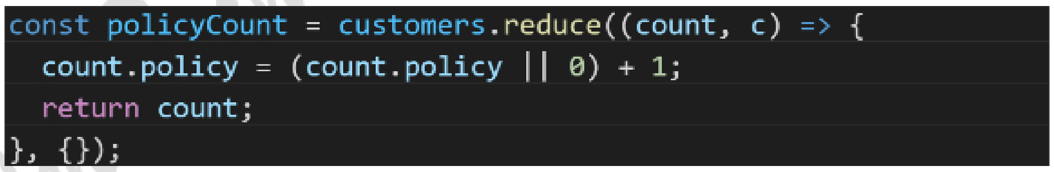


```
console.log("Customer ${customers[0].name} has policy
${customers[0].policy}");
```

🔍 **Hint:** Quotes type

console.log(`Customer ${customers[0].name} has policy ${customers[0].policy}`);

🐞 Bug 6: Policy Count Incorrect

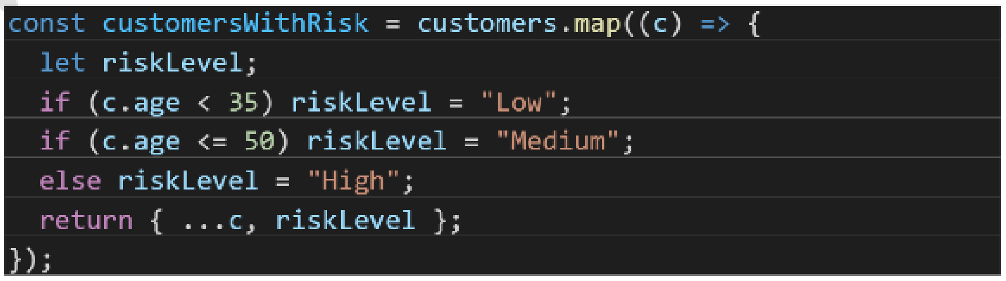


```
const policyCount = customers.reduce((count, c) => {
  count.policy = (count.policy || 0) + 1;
  return count;
}, {});
```

🔍 **Hint:** Dynamic object key

```
const policyCount = customers.reduce((count, c) => {
  count[c.policy] = (count[c.policy] || 0) + 1;
  return count;
}, {});
```

🐞 Bug 7: Risk Level Always Undefined

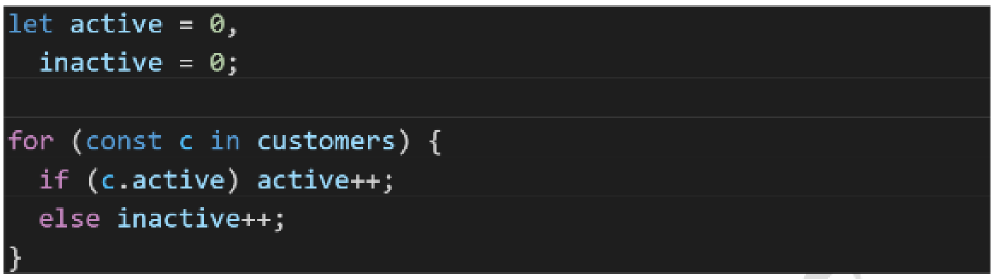


```
const customersWithRisk = customers.map((c) => {
  let riskLevel;
  if (c.age < 35) riskLevel = "Low";
  if (c.age <= 50) riskLevel = "Medium";
  else riskLevel = "High";
  return { ...c, riskLevel };
});
```

🔍 **Hint:** Condition chaining

```
const customersWithRisk = customers.map((c) => {
  let riskLevel;
  if (c.age < 35) riskLevel = "Low";
  else if (c.age <= 50) riskLevel = "Medium";
  else riskLevel = "High";
  return { ...c, riskLevel };
});
```

🐞 Bug 8: Active vs Inactive Count Wrong

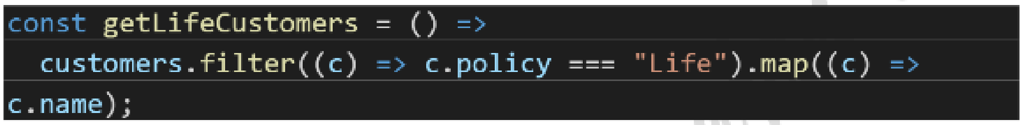


```
let active = 0,
  inactive = 0;

for (const c in customers) {
  if (c.active) active++;
  else inactive++;
}
```

🔍 **Hint:** for...in vs for...of

```
let active = 0, inactive = 0;


for (const c of customers) {
  if (c.active) active++;
  else inactive++;
}
```

🐞 Bug 9: Arrow Function Syntax Error

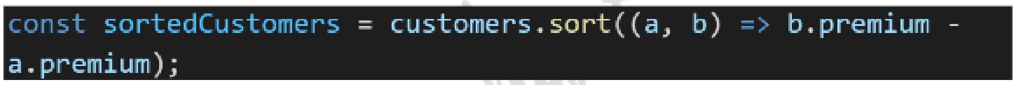


```
const getLifeCustomers = () =>
  customers.filter((c) => c.policy === "Life").map((c) =>
c.name);
```

🔍 **Hint:** Arrow function return

```
const getLifeCustomers = () =>
  customers
    .filter((c) => c.policy === "Life")
    .map((c) => c.name);
```

🐞 Bug 10: Sorting Mutates Original Array



```
const sortedCustomers = customers.sort((a, b) => b.premium -
a.premium);
```

🔍 **Hint:** Array mutation

```
const sortedCustomers = [...customers].sort((a, b) => b.premium - a.premium);
```