DEBUGGING QUESTIONS:

1.
   ```
   for (let i = 0; i < customers.length; i++) {
    console.log(customers[i].name);
    }
   ```

Explanation:

The error lied in i value equal to which made the loop to run one extra time.

2.
   ```
   const activeCustomers = customers.filter(c => c.active === true);
   ```

Explanation:

The error lied in not returning the condition inside filter(), so it always returned undefined.

3.
   ```
   const updatedPremiums = customers.map(c => {
   if (c.age >= 50) {
   return { ...c, premium: c.premium * 1.1 };
    }
    return c;});
   ```

Explanation:
The error lied in not returning a value from map() and directly modifying the original object.

4.
   ```
   const totalPremium = customers.reduce((total, c) => {
   return total + c.premium;
   }, 0);
   ```

Explanation:
The error lied in not returning the accumulated value inside the reduce() function.

5.
   ```
   console.log(`Customer ${customers[0].name} has policy ${customers[0].policy}`);
   ```

Explanation:
The error lied in using double quotes instead of backticks for template literals.

6.
   ```
   const policyCount = customers.reduce((count, c) => {
   ```

```
   count[c.policy] = (count[c.policy] || 0) + 1;
   return count;
   }, {});
```

Explanation:

The error lied in using a fixed key instead of a dynamic object key.

7.
```
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 };
   });
```

Explanation:

The error lied in overlapping conditions which caused the risk value to be overwritten.

8.
```
let active = 0, inactive = 0;
 for (const c of customers) {
   if (c.active) active++;
   else inactive++;
}
```

Explanation:

The error lied in using for...in which iterates over indexes instead of objects.

9.
```
const getLifeCustomers = () =>
   customers.filter(c => c.policy === "Life").map(c => c.name);
```

Explanation:

The error lied in incorrect arrow function return formatting.

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

Explanation:

The error lied in using sort() directly which modified the original array.