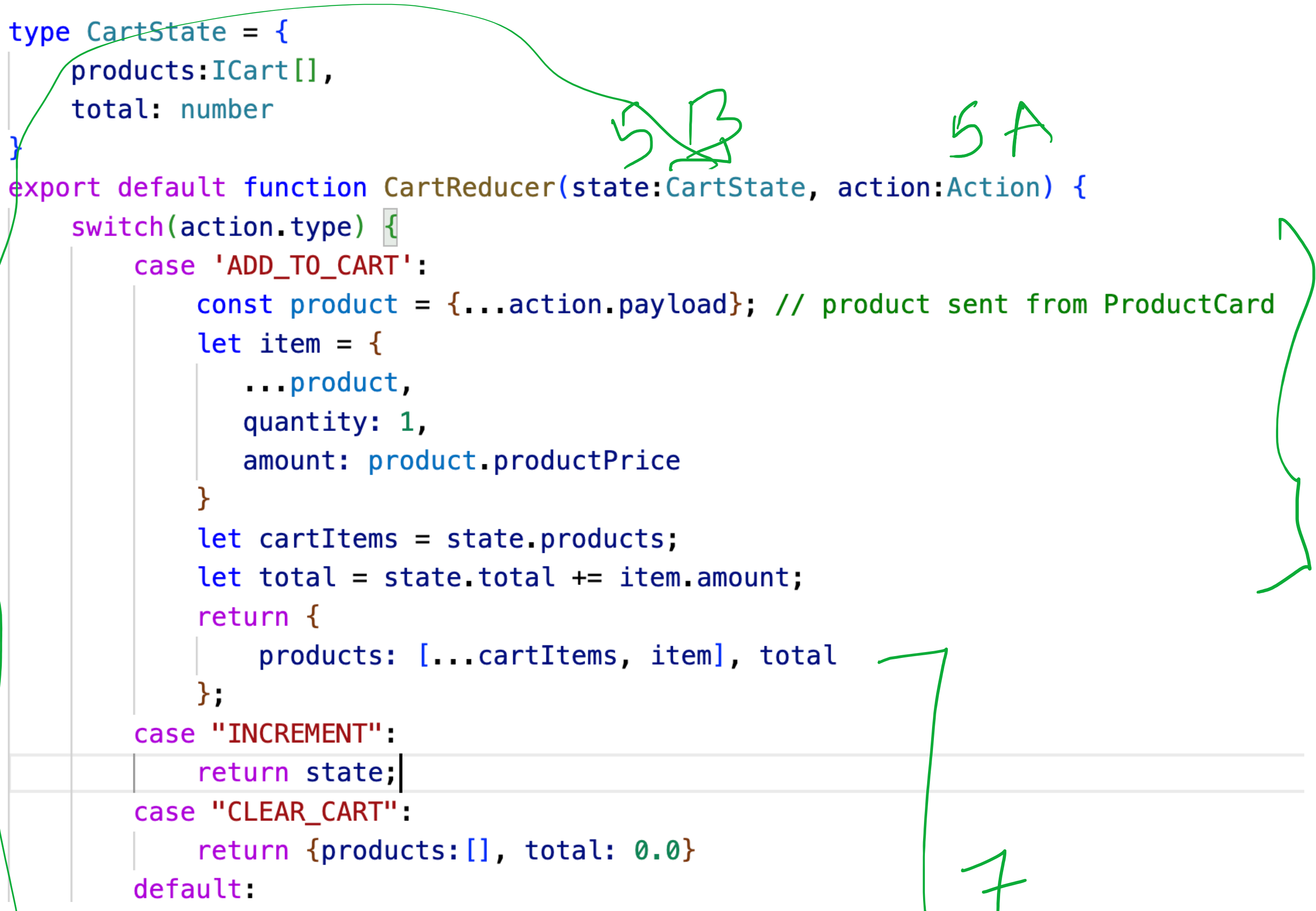
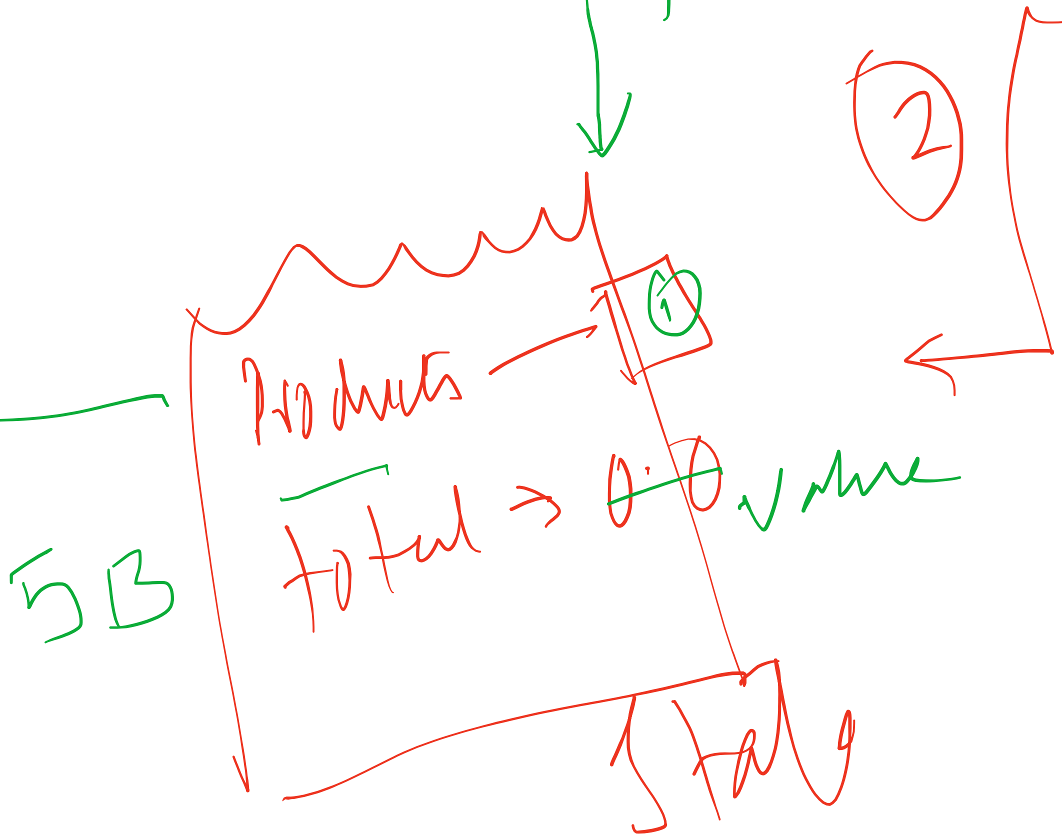
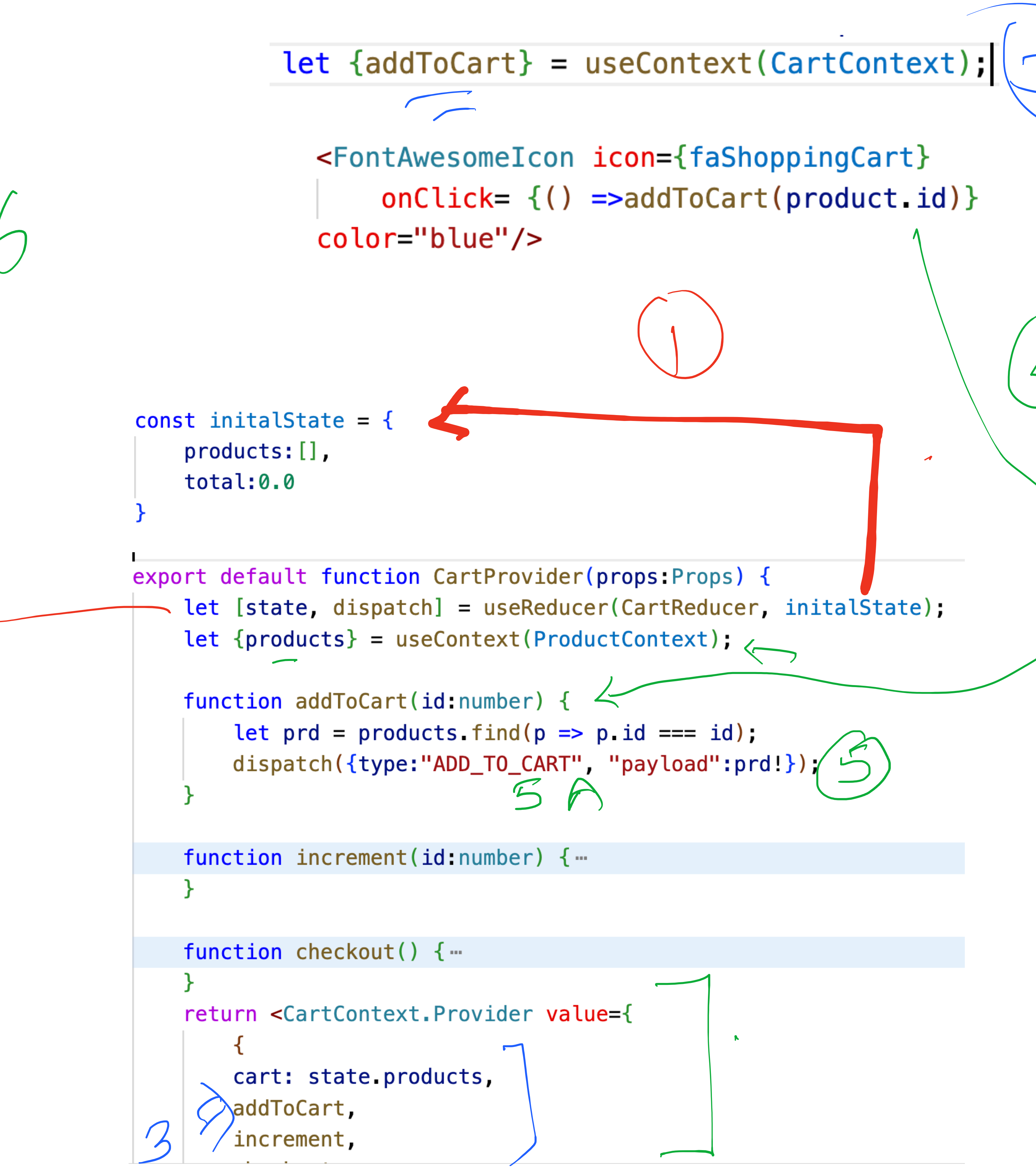


5B 5A

```
type CartState = {
    products:ICart[],
    total: number
}
export default function CartReducer(state:CartState, action:Action) {
    switch(action.type) {
        case 'ADD_TO_CART':
            const product = {...action.payload}; // product sent from ProductCard
            let item = {
                ...product,
                quantity: 1,
                amount: product.productPrice
            }
            let cartItems = state.products;
            let total = state.total += item.amount;
            return {
                products: [...cartItems, item], total
            };
        case "INCREMENT":
            return state;
        case "CLEAR_CART":
            return {products:[], total: 0.0}
        default:
```

7





```
let {addToCart} = useContext(CartContext);
```

```
<FontAwesomeIcon icon={faShoppingCart}
    onClick= {() =>addToCart(product.id)}
color="blue"/>
```

```
const initalState = {
    products:[],
    total:0.0
}

export default function CartProvider(props:Props) {
    let [state, dispatch] = useReducer(CartReducer, initalState);
    let {products} = useContext(ProductContext);

    function addToCart(id:number) {
        let prd = products.find(p => p.id === id);
        dispatch({type:"ADD_TO_CART", "payload":prd!});
    }

    function increment(id:number) {⋯
    }

    function checkout() {⋯
    }
    return <CartContext.Provider value={
        {
        cart: state.products,
        addToCart,
        increment,
```

3

4