```
//cs2410 assn1

console.log(transactions);
console.log(customers);

//functions
function filter(data, predicate) {
    let result = [];
    for (let item of data) {
        if (predicate(item)) result.push(item);
    }
    return result;
}

function pairIf(data1, data2, predicate) {
    let result = [];
    for (let item1 of data1) {
        for (let item2 of data2) {
            if (predicate(item1, item2)) {
                result.push([item1, item2]);
            }
        }
    }
    return result;
}

function findLast(data, predicate) {
    for (let i = data.length - 1; i >= 0; i--) {
        if (predicate(data[i])) return data[i];
    }
    return null;
}

function reduce(data, reducer, initialValue) {
    let accumulatedResult = initialValue;
    for (let value of data) {
        accumulatedResult = reducer(value, accumulatedResult);
    }
    return accumulatedResult;
}

function map(data, callback) {
    let result = [];
    for (let item of data) {
        result.push(callback(item));
    }
    return result;
}



//figs questions

//invalid t
//let invalidTransactions = filter(transactions, t => !t.amount || t.amount === 0
|| ![“FIG_JAM”, “FIG_JELLY”, “SPICY_FIG_JAM”,
“ORANGE_FIG_JELLY”].includes(t.product));
//console.log("Number of invalid transactions:", invalidTransactions.length);
```

```
let invalidTransactions = filter(transactions, t => 0 || !["FIG_JAM", "FIG_JELLY",
"SPICY_FIG_JAM", "ORANGE_FIG_JELLY"].includest(t.product));
console.log("Number of Invalid Transactions: ", invalidTransactions.length);

//duplicate customers
let duplicatePairs = pairIf(customers, customers, (c1, c2) => c1.id !== c2.id &&
c1.emailAddress === c2.emailAddress);
console.log("Number of duplicate customers:", duplicatePairs.length / 2);


//recent 200
let recentLargeTransaction = findLast(transactions, t => t.amount > 200);
console.log("Most recent transaction over $200:", `$$
{recentLargeTransaction.amount.toFixed(2)}`);

//small, med, large
let transactionSizes = reduce(transactions, (t, acc) => {
    if (t.amount < 25) acc.small++;
    else if (t.amount >= 25 && t.amount < 75) acc.medium++;
    else if (t.amount >= 75) acc.large++;
    return acc;
}, { small: 0, medium: 0, large: 0 });
console.log("Number of small transactions:", transactionSizes.small);
console.log("Number of medium transactions:", transactionSizes.medium);
console.log("Number of large transactions:", transactionSizes.large);

//customers w/ t over 200
let largeTransactions = filter(transactions, t => t.amount > 200);
let customerTransactions = pairIf(customers, largeTransactions, (c, t) => c.id ===
t.customerId);
let uniqueCustomers = reduce(customerTransactions, (pair, acc) => {
    if (!acc.includes(pair[0])) acc.push(pair[0]);
    return acc;
}, []);
console.log("Customers with transactions over $200:", uniqueCustomers);
let customerNames = map(uniqueCustomers, c => `${c.firstName} ${c.lastName}`);
console.log("Names of customers with transactions over $200:", customerNames);
```