Joel - 16 October 2025

Review 12 Assessment Report (Resubmission)

Overall Score: 95%

Detailed Review

Function 1: canWatchTv (33/33 marks)

Your Code:

```
function canWatchTv(isHoliday, isWeekday){
    return(isHoliday || !isWeekday )
}
```

Evaluation: - **Logic is Perfect**: Correctly returns true if it's a holiday OR not a weekday - **Uses FoundationScript**: Proper function declaration syntax - **Excellent Simplification**: Direct return statement - professional style! - **Correct Parameters**: Matches specification exactly - **Good Comment**: Clear explanation of what the function does - **Has Summary**: Provided excellent 2-sentence explanation - **Test Case**: Includes verification

Your Summary: "My approach to this function was to use the OR operator to check if it's a holiday or not a weekday. I used a simple return statement with the condition to return true or false based on the inputs."

Perfect! Clear explanation showing understanding of both the logic and the implementation approach.

Function 2: doTheyAgree (33/33 marks)

Your Code:

```
function doTheyAgree(partner1Decision, partner2Decision){
    return(partner1Decision === partner2Decision)
}
```

Evaluation: - **Logic is Perfect**: Correctly checks if both decisions are equal - **Uses FoundationScript**: Proper function declaration - **Excellent Simplification**: Direct return statement - **Correct Parameters**: Matches specification exactly - **Good Comment**: Clear explanation - **Has Summary**: Provided excellent explanation - **Test Case**: Includes verification

Your Summary: "My approach to this function was to use the strict equality operator to check if both partners made the same decision. I used a simple return statement with the condition to return true or false based on the input."

Excellent! Shows understanding of strict equality and why it's used.

Function 3: isOpen (29/33 marks)

Your Code:

```
function isOpen(weekday, month){
    return(weekday != "monday" && month != "july" )
}
```

Issues Identified: - **Loose Equality Warning**: Uses `!=` (loose inequality) instead of `!==` (strict inequality) - While this works correctly for string comparison, best practice is strict inequality - **Logic is Perfect**: Correctly returns true if NOT Monday AND NOT July - **Uses FoundationScript**: Proper function declaration - **Correct Parameters**: Matches specification exactly - **Correct Case**: Uses lowercase "monday" and "july" - **Good Comment**: Clear explanation - **Has Summary**: Provided excellent explanation - **Test Case**: Includes verification

Why This Matters:

```
// Loose equality (!=):
"5" != 5    // false (converts types)

// Strict inequality (!==):
"5" !== 5   // true (different types)
```

For strings comparing to strings, both work the same. But strict (`!==`) is safer and considered best practice.

Your Summary: "My approach to this function was to use the AND operator to check if it's not Monday and not July. Again I used a simple return statement with the condition to return true or false based on the inputs."

Excellent explanation!

Summary Assessment

Strengths: - **All Logic is Perfect**: Every function works exactly as specified - **Proper FoundationScript Syntax**: Used standard function declarations throughout - **Excellent Code Quality**: Direct return statements show professional coding style - **Outstanding Summaries**: All functions have clear, detailed 2-sentence explanations - **Good Testing**: Each function includes verification - **Clean Code**: Very readable and well-structured - **Shows Understanding**: Summaries explain both the "what" and the "how" - **Consistent Style**: All three functions use the same clean approach

Only Minor Issue: - **Function 3**: Uses `!=` instead of `!==` - Works correctly for this case - But `!==` is best practice (strict comparison)

What You Did Exceptionally Well:

1. **Perfect Logic**: All three functions work flawlessly
2. **Code Simplification**: Direct return statements throughout - professional style
3. **Excellent Summaries**: Each summary explains:
   - What operator/approach you used (OR, strict equality, AND)
   - Why you chose that approach (check conditions, return based on inputs)
4. **Clear Comments**: Every function has descriptive comments
5. **Consistent Quality**: All three functions maintain the same high standard

---

Comparison to First Submission

First Submission (95%): - All logic correct - Proper FoundationScript syntax - Excellent summaries - Wrong function name (doPartnersAgree vs doTheyAgree) - Could simplify if/else to direct returns

Second Submission (95%): - All logic correct - Correct function names - Excellent summaries - Direct return statements (improvement!) - Uses `!=` instead of `!==` in Function 3

Note: Same score, but different minor issues. You fixed the naming and simplified the code, which is excellent! The only new issue is using loose equality.

Code Quality Analysis

Readability: 10/10 - Clear variable names - Consistent style - Well-commented

Correctness: 10/10 - All logic perfect - All test cases would pass - No edge case issues

Efficiency: 10/10 - Direct returns (no unnecessary if/else) - Simple, elegant solutions - Optimal implementations

Documentation: 10/10 - Excellent summaries - Good comments - Clear test cases

Best Practices: 9/10 - Minor deduction for loose equality in Function 3

Loose vs Strict Equality

What You Used:

```
weekday != "monday"  // loose inequality
```

Best Practice:

```
weekday !== "monday"  // strict inequality
```

Why It Matters:

```
// Loose equality converts types:
"5" == 5    // true (converts string to number)
"" == 0     // true (empty string to number)
false == 0  // true (boolean to number)

// Strict equality doesn't:
"5" === 5   // false (different types)
"" === 0    // false (different types)
false === 0 // false (different types)
```

For your case (string to string), both work identically. But using strict is: - Safer (no unexpected type conversions) - Considered best practice - More predictable behavior

---

What Sets Your Code Apart

1. **Excellent Summaries**: You don't just say "this function works" - you explain:
   - What approach you used
   - Why it works
   - How it achieves the goal
2. **Professional Style**: Direct return statements show you understand clean code principles
3. **Consistent Quality**: All three functions maintain the same high standard
4. **Shows Learning**: You improved the code structure from first to second submission

---

Minor Suggestion

Change Function 3 to use strict inequality:

Current:

```
return(weekday != "monday" && month != "july" )
```

Suggested:

```
return(weekday !== "monday" && month !== "july" )
```

This is the only "issue" and it's very minor!

Final Thoughts

Outstanding work! This is exemplary code that demonstrates: - Deep understanding of boolean logic - Professional coding practices - Excellent documentation skills - Clean, maintainable code - Attention to specification details

Your summaries are particularly strong - they show you don't just know *what* the code does, but *why* you wrote it that way. This is excellent engineering practice.

The only deduction is for using loose equality (`!=`) instead of strict (`!==`) in Function 3. This is a minor best-practice issue that doesn't affect functionality.

Keep up this exceptional standard!

Score Breakdown: - Function 1: 33/33 (Perfect) - Function 2: 33/33 (Perfect) - Function 3: 29/33 (Perfect logic, uses `!=` instead of `!==`) - **Total: 95/99 = 95%**

Grade: Outstanding (A)

Achievement: Maintained excellent standard!