

# Code Documentation: Quadrilateral\_Substitution

UMEDCTA Repository

December 3, 2025

## Contents

1	Quadrilateral_Substitution/brandom_lesson.js	2
2	Quadrilateral_Substitution/brandom_styles.css	15
3	Quadrilateral_Substitution/inferential_strength.html	23
4	Quadrilateral_Substitution/inferential_strength_styles.css	32

# 1 Quadrilateral\_Substitution/brandom\_lesson.js

```

1 // brandom_lesson.js
2
3 document.addEventListener('DOMContentLoaded', () => {
4     // --- Module Navigation ---
5     const modules = document.querySelectorAll('.module');
6     const prevButton = document.getElementById('prevButton');
7     const nextButton = document.getElementById('nextButton');
8     const moduleIndicator = document.getElementById('moduleIndicator');
9     let currentModuleIndex = 0;
10    let moduleInitializers = {};// Store functions to initialize modules
11
12    // --- Helper function for SVG (assuming it's used elsewhere too, place outside initializers) ---
13    function getShapeSvg(shapeName) {
14        let svg = `<svg viewBox="0 0 100 100" width="70" height="70" class="shape-viz">`;
15        svg += `<title>${shapeName}</title>`;// Tooltip for SVG (shows on hover)
16        switch (shapeName) {
17            case "Square": svg += `<rect x="10" y="10" width="80" height="80" fill="#cfe2ff"
18                stroke="#0d6efd" stroke-width="2"/>`;break;
19            case "Rectangle": svg += `<rect x="10" y="20" width="80" height="60" fill="#d1e7dd"
20                stroke="#198754" stroke-width="2"/>`;break;
21            case "Rhombus": svg += `<polygon points="50,5 95,50 50,95 5,50" fill="#f8d7da"
22                stroke="#dc3545" stroke-width="2"/>`;break;
23            case "Parallelogram": svg += `<polygon points="25,5 95,5 75,95 5,95" fill="#fff3cd"
24                stroke="#ffc107" stroke-width="2"/>`;break;
25            case "Kite": svg += `<polygon points="50,10 85,50 50,90 15,50" fill="#f3d7f8"
26                stroke="#a30cff" stroke-width="2"/>`;break;
27            case "Trapezoid": svg += `<polygon points="30,10 70,10 100,90 0,90" fill="#e2e3e5"
28                stroke="#6c757d" stroke-width="2"/>`;break;
29            case "Quadrilateral": default: svg += `<polygon points="10,10 90,20 80,80 20,90"
30                fill="#f8f9fa" stroke="#adb5bd" stroke-width="2"/>`;break;
31        }
32        svg += `</svg>`;
33        return svg;
34    }
35
36    function showModule(index) {
37        const totalModules = 7;// **** UPDATE THIS ****
38        if (index < 0 || index >= totalModules) return;
39
40        modules.forEach((module, i) => {
41            module.style.display = 'none';
42            module.classList.remove('current-module', 'hidden-module');
43            if (i === index) {
44                module.style.display = 'block';
45                module.classList.add('current-module');
46            } else {
47                module.classList.add('hidden-module');
48            }
49        });
50
51        currentModuleIndex = index;
52        moduleIndicator.textContent = `Module ${index + 1} of ${totalModules}`;// **** UPDATE THIS ****
53        prevButton.disabled = index === 0;
54        nextButton.disabled = index === totalModules - 1;// **** UPDATE THIS ****
55
56        if (typeof moduleInitializers[index] === 'function') {
57            try {
58                moduleInitializers[index]();
59                moduleInitializers[index] = null;// Mark as run
60            } catch (error) {
61                console.error(`Error initializing module ${index + 1}:`, error);
62            }
63        }
64    }
65
66    // --- Initialization ---
67    document.addEventListener('click', (event) => {
68        if (event.target.classList.contains('current-module')) {
69            showModule(modules.indexOf(event.target));
70        }
71    });
72
73    // --- Navigation ---
74    prevButton.addEventListener('click', () => {
75        if (currentModuleIndex > 0) {
76            currentModuleIndex--;
77            showModule(currentModuleIndex);
78        }
79    });
80
81    nextButton.addEventListener('click', () => {
82        if (currentModuleIndex < totalModules - 1) {
83            currentModuleIndex++;
84            showModule(currentModuleIndex);
85        }
86    });
87
88    // --- Module Indicators ---
89    moduleIndicator.addEventListener('click', () => {
90        if (currentModuleIndex < totalModules - 1) {
91            currentModuleIndex++;
92            showModule(currentModuleIndex);
93        }
94    });
95
96    // --- Module Initializers ---
97    moduleInitializers.forEach(initializer => {
98        initializer();
99    });
100}

```

```

56     }
57
58     // Scroll to top
59     const targetModule = modules[index];
60     if (targetModule) {
61         const headerOffset = document.querySelector('header')?.offsetHeight || 60;
62         const elementPosition = targetModule.getBoundingClientRect().top;
63         const offsetPosition = elementPosition + window.pageYOffset - headerOffset - 20;
64         window.scrollTo({ top: offsetPosition, behavior: 'smooth' });
65     }
66 }
67
68 prevButton.addEventListener('click', () => showModule(currentModuleIndex - 1));
69 nextButton.addEventListener('click', () => showModule(currentModuleIndex + 1));
70
71 // --- Data Structures ---
72 const shapeHierarchy = {
73     "Square": { superclasses: ["Rectangle", "Rhombus"], properties: ["4 equal sides", "4 right
    ↪ angles", "opposite sides parallel", "diagonals bisect perpendicularly", "diagonals are
    ↪ equal"], incompatibilities: ["has obtuse angle", "unequal adjacent sides"] },
74     "Rectangle": { superclasses: ["Parallelogram", /*"Isosceles Trapezoid" - Removed for simpler
    ↪ linear hierarchy in slider */], properties: ["4 right angles", "opposite sides parallel",
    ↪ "opposite sides equal", "diagonals are equal", "diagonals bisect each other"],
    ↪ incompatibilities: ["has acute angle (internal)", "unequal diagonals"] },
75     "Rhombus": { superclasses: ["Parallelogram", "Kite"], properties: ["4 equal sides", "opposite
    ↪ sides parallel", "opposite angles equal", "diagonals bisect perpendicularly"],
    ↪ incompatibilities: ["has right angle but unequal adjacent sides"] },
76     "Parallelogram": { superclasses: ["Trapezoid"], properties: ["opposite sides parallel",
    ↪ "opposite sides equal", "opposite angles equal", "diagonals bisect each other"],
    ↪ incompatibilities: ["only one pair parallel sides"] },
77     "Kite": { superclasses: ["Quadrilateral"], properties: ["2 pairs adjacent equal sides", "one
    ↪ pair opposite angles equal", "diagonals perpendicular"], incompatibilities: ["opposite sides
    ↪ parallel", "all sides equal"] },
78     // "Isosceles Trapezoid": { superclasses: ["Trapezoid"], properties: ["one pair parallel sides",
    ↪ "base angles equal", "diagonals are equal"], incompatibilities: ["is equilateral", "has
    ↪ perpendicular diagonals"] }, // Removed for linear slider
79     "Trapezoid": { superclasses: ["Quadrilateral"], properties: ["at least one pair parallel
    ↪ sides"], incompatibilities: ["no parallel sides", "all sides equal"] },
80     "Quadrilateral": { superclasses: [], properties: ["4 sides"], incompatibilities: ["is a
    ↪ triangle", "has 5 sides"] }
81 };
82 // Define the chain for sliders - ensure it matches hierarchy logic where needed
83 const hierarchyChainForSlider = ["Quadrilateral", "Trapezoid", "Parallelogram", "Rectangle",
    ↪ "Square"]; // Weakest to Strongest
84
85
86 function isSubclass(shapeA, shapeB) {
87     if (shapeA === shapeB) return true;
88     if (!shapeHierarchy[shapeA] || !shapeHierarchy[shapeB]) {
89         // console.warn(`isSubclass: Shape definition missing for ${shapeA} or ${shapeB}`);
90         return false;
91     }
92     let queue = [...(shapeHierarchy[shapeA].superclasses || [])];
93     let visited = new Set([shapeA]);
94     while (queue.length > 0) {
95         const current = queue.shift();
96         if (current === shapeB) return true;
97         if (!visited.has(current) && shapeHierarchy[current]) {
98             visited.add(current);
99             if (shapeHierarchy[current].superclasses) {
100                 queue.push(...shapeHierarchy[current].superclasses);
101             }
102         }
103     }
}

```

```

104     return false;
105 }
106
107 // --- Module Initializers ---
108
109 // Module 1: Intro
110 moduleInitializers[0] = function() {
111   const vizSquare = document.getElementById('viz-square-m1');
112   const vizRectangle = document.getElementById('viz-rectangle-m1');
113   if (vizSquare) vizSquare.innerHTML = getShapeSvg("Square");
114   if (vizRectangle) vizRectangle.innerHTML = getShapeSvg("Rectangle");
115 };
116
117 // *** REVISED Module 2: Quadrilateral Checklist Initializer ***
118 moduleInitializers[1] = function setupQuadrilateralChecklist() {
119   // Define shape data using 'false' for "rejects property / incompatibility holds"
120   // Corresponds to Table 1 logic where X means incompatibility (rejects the "No..." property)
121   const shapesDataM2 = [
122     // Property keys match checkbox IDs r1-r6
123     { name: "Square", r1: false, r2: false, r3: false, r4: false, r5: false, r6: false
124       }, // Rejects all "No..." props
125     { name: "Rectangle", r1: false, r2: true, r3: false, r4: true, r5: false, r6: false
126       }, // Allows r2, r4
127     { name: "Rhombus", r1: false, r2: false, r3: false, r4: false, r5: false, r6: true
128       }, // Allows r6
129     { name: "Parallelogram", r1: false, r2: true, r3: false, r4: true, r5: false, r6: true
130       }, // Allows r2, r4, r6
131     { name: "Trapezoid", r1: true, r2: true, r3: true, r4: true, r5: false, r6: true
132       }, // Only rejects r5
133     { name: "Kite", r1: false, r2: false, r3: true, r4: false, r5: true, r6: true
134       }, // Allows r3, r5, r6
135     { name: "Quadrilateral", r1: true, r2: true, r3: true, r4: true, r5: true, r6: true
136       } // Allows all
137   ];
138
139   // Map names to data for easier lookup later (e.g., in Module 4)
140   window.shapeDataMap = shapesDataM2.reduce((acc, shape) => {
141     acc[shape.name] = shape;
142     return acc;
143   }, {});
144
145   const checkboxesContainer = document.getElementById('restrictionCheckboxesM2');
146   if (!checkboxesContainer) {
147     console.error("Module 2 checkboxes container not found!");
148     return; // Exit if container is missing
149   }
150   const checkboxes = checkboxesContainer.querySelectorAll('input[type="checkbox"]');
151   const shapesContainer = document.getElementById('shapesContainerM2'); // Use the new ID
152
153   // --- Calculate Strength ---
154   // Strength = number of properties the shape REJECTS (has 'false' for)
155   function calculateStrength(shape) {
156     let strength = 0;
157     for (let i = 1; i <= 6; i++) {
158       if (shape[`r${i}`] === false) {
159         strength++;
160       }
161     }
162     return strength;
163   }
164
165   // Add strength to the main data structure for easy access
166   shapesDataM2.forEach(shape => {
167     shape.strength = calculateStrength(shape);
168   });

```

```

162     });
163
164
165     function getActiveRestrictions() {
166         const active = {};
167         checkboxes.forEach(cb => {
168             active[cb.dataset.propertyKey] = cb.checked;
169         });
170         return active;
171     }
172
173
174     function filterShapes() {
175         const activeRestrictions = getActiveRestrictions();
176         const targetValue = false; // A shape survives if it REJECTS the property (has 'false')
177         ↳ when the restriction is active
178
179         return shapesDataM2.filter(shape => {
180             for (const restrictionKey in activeRestrictions) {
181                 // If the restriction checkbox IS CHECKED (activeRestrictions[restrictionKey] is
182                 ↳ true)...
183                 if (activeRestrictions[restrictionKey]) {
184                     // ...then the shape MUST have 'false' for this property to survive.
185                     if (shape[restrictionKey] !== targetValue) {
186                         return false; // Filter out this shape
187                     }
188                 }
189             }
190             return true; // Survived all active restrictions
191         });
192     }
193
194     function updateShapesDisplay() {
195         if (!shapesContainer) return; // Safety check
196         const filteredShapes = filterShapes();
197         shapesContainer.innerHTML = ''; // Clear previous shapes
198         if (filteredShapes.length === 0) {
199             shapesContainer.innerHTML = '<p>No quadrilaterals match the current
200             ↳ restrictions.</p>';
201         } else {
202             // Sort shapes maybe by strength? (Optional)
203             // filteredShapes.sort((a, b) => b.strength - a.strength);
204
205             filteredShapes.forEach(shape => {
206                 const div = document.createElement('div');
207                 div.className = 'shape-item';
208                 div.style.textAlign = 'center';
209                 div.style.margin = '10px';
210                 // Add SVG and strength label below it
211                 div.innerHTML = getShapeSvg(shape.name) +
212                     `<div style="margin-top: 8px; font-size: 0.9em; font-weight: 600; color:
213                     #667eea;">Strength: ${shape.strength}</div>`;
214                 shapesContainer.appendChild(div);
215             });
216         }
217     }
218
219     // Attach event listeners to checkboxes within this module
220     checkboxes.forEach(cb => {
221         cb.addEventListener('change', updateShapesDisplay);
222     });
223
224     // Initial display update for this module
225     updateShapesDisplay();
226 };
227 // End Module 2 Initializer

```

```

223
224 // Module 3: Substitution Conceptual Intro
225 moduleInitializers[2] = function() {
226     // This module is now primarily text and static examples in the HTML.
227     // No dynamic JS needed unless you add interactive highlighting later.
228 };
229
230 // Module 4: Polarity Demo
231 // *** REVISED Module 4: Polarity Demo Initializer ***
232 moduleInitializers[3] = function setupPolarityDemo() {
233     // Ensure shape data is available from Module 2
234     if (typeof window.shapeDataMap === 'undefined') {
235         console.error("Shape data not initialized from Module 2. Run Module 2 first.");
236         // Attempt to run Module 2 initializer if it hasn't run
237         if (typeof moduleInitializers[1] === 'function') {
238             console.warn("Attempting to initialize Module 2 now...");
239             try {
240                 moduleInitializers[1]();
241                 moduleInitializers[1] = null; // Mark as run
242                 if (typeof window.shapeDataMap === 'undefined') {
243                     // Still failed
244                     alert("Error: Module 2 data needed for Module 4. Please reload and navigate
245                         → sequentially.");
246                     return;
247                 }
248             } catch(e) {
249                 alert("Error initializing Module 2 data. Please reload and navigate
250                         → sequentially.");
251                 return;
252             }
253         } else {
254             alert("Module 2 already initialized but data missing. Please reload.");
255             return;
256         }
257     }
258
259     const fixedSelect = document.getElementById('fixedConceptSelectM4');
260     const strengthSlider = document.getElementById('strengthSliderM4');
261     const varConceptLabel = document.getElementById('variableConceptLabelM4');
262     const varConceptLabelCond = document.getElementById('variableConceptLabelCondM4');
263     const vizP_El = document.getElementById('vizPM4');
264     const vizQ_R_El = document.getElementById('vizQM4'); // Renamed for clarity, shows Q
265     const relationArrowEl = document.getElementById('relationArrowM4');
266     const strengthPEl = document.getElementById('strengthPM4'); // Span for P strength
267     const strengthQEEl = document.getElementById('strengthQM4'); // Span for Q strength
268
269     const baseInferEl = document.getElementById('baseInferM4');
270     const converseInferEl = document.getElementById('converseInferM4');
271     const contraInferEl = document.getElementById('contraInferM4');
272     const inverseInferEl = document.getElementById('inverseInferM4');
273
274     const propertyRSelect = document.getElementById('propertyRSelectM4');
275     const propertyRLabel = document.getElementById('propertyRLabelM4');
276     const condAntecedentEl = document.getElementById('condAntecedentM4');
277     const condAntecedentStatusEl = document.getElementById('condAntecedentStatusM4');
278     const condConsequentEl = document.getElementById('condConsequentM4');
279     const condConsequentStatusEl = document.getElementById('condConsequentStatusM4');
280
281     // Use the defined slider chain
282     const sliderChain = hierarchyChainForSlider; // Uses the global constant
283
284     // Populate fixed concept select
285     fixedSelect.innerHTML = sliderChain.map(t => `<option value="${t}">${t}</option>`).join('');

```

```

286     fixedSelect.value = "Rectangle"; // Default Q
287
288     // Populate Property R select (remains the same)
289     const availableProperties = {
290         "HAS_4_SIDES": { value: "has 4 sides", description: "has 4 sides" },
291         "HAS_PARALLEL_SIDES": { value: "at least one pair parallel sides", description: "at least
292             ↪ one pair parallel sides" },
293         "OPPOSITE_SIDES_PARALLEL": { value: "opposite sides parallel", description: "opposite sides
294             ↪ parallel" },
295         "HAS_4_RIGHTANGLES": { value: "4 right angles", description: "has 4 right angles" },
296         "HAS_4_EQUALSIDES": { value: "4 equal sides", description: "has 4 equal sides" }
297     };
298     propertyRSelect.innerHTML = Object.values(availableProperties)
299         .map(p => `<option value="${p.value}">${p.description}</option>`)
300         .join('');
301     propertyRSelect.value = "4 right angles"; // Default R
302
303     // Set Slider Labels and Range
304     document.getElementById('sliderMinLabelM4').textContent = sliderChain[0];
305     document.getElementById('sliderMaxLabelM4').textContent = sliderChain[sliderChain.length - 1];
306     strengthSlider.max = sliderChain.length - 1;
307     strengthSlider.value = sliderChain.findIndex(s => s === "Square"); // Default P = Strongest
308
309     // --- Validity Helper Functions (remain the same) ---
310     function conceptEntailsProperty(conceptP, propertyR_value) {
311         if (!shapeHierarchy[conceptP]) return false;
312         let queue = [conceptP];
313         let visited = new Set();
314         while (queue.length > 0) {
315             let node = queue.shift();
316             if (visited.has(node)) continue;
317             visited.add(node);
318             if (shapeHierarchy[node]?.properties?.includes(propertyR_value)) {
319                 return true;
320             }
321             if (shapeHierarchy[node]?.superclasses) {
322                 shapeHierarchy[node].superclasses.forEach(sc => { if (!visited.has(sc))
323                     ↪ queue.push(sc); });
324             }
325         }
326         return false;
327     }
328
329     function propertyEntailsConcept(propertyR_value, conceptP) {
330         if (!shapeHierarchy[conceptP]) return false;
331         let shapesThatGuaranteeR = [];
332         for (const shape in shapeHierarchy) {
333             if (conceptEntailsProperty(shape, propertyR_value)) {
334                 shapesThatGuaranteeR.push(shape);
335             }
336         }
337         if (shapesThatGuaranteeR.length === 0) return false;
338         // Check if *all* shapes guaranteeing R are subclasses of conceptP
339         return shapesThatGuaranteeR.every(shape => isSubclass(shape, conceptP));
340     }
341
342     function updatePolarityDemo() {
343         const fixedConceptQ = fixedSelect.value;
344         const sliderIndex = parseInt(strengthSlider.value);
345         const variableConceptP = sliderChain[sliderIndex];
346         const selectedPropertyR_value = propertyRSelect.value;
347         const selectedPropertyR_desc = propertyRSelect.options[propertyRSelect.selectedIndex].text;
348
349         // --- Get Strengths using Module 2 data ---
350         const strengthP = window.shapeDataMap[variableConceptP]?.strength ?? '?';

```

```

348     const strengthQ = window.shapeDataMap[fixedConceptQ]?.strength ?? '?';
349
350     // Update labels and strengths
351     varConceptLabel.textContent = variableConceptP;
352     strengthPEl.textContent = strengthP;
353     strengthQEl.textContent = strengthQ;
354     varConceptLabelCond.textContent = variableConceptP;
355     propertyRLabel.textContent = selectedPropertyR_desc;
356
357     vizP_El.innerHTML = getShapeSvg(variableConceptP);
358     vizQ_R_El.innerHTML = getShapeSvg(fixedConceptQ); // Display Q here
359
360     const pImpliesQ = isSubclass(variableConceptP, fixedConceptQ);
361     const qImpliesP = isSubclass(fixedConceptQ, variableConceptP);
362     let relationSymbol = '□';
363     if (pImpliesQ && qImpliesP) { relationSymbol = '↔ Equivalent'; }
364     else if (pImpliesQ) { relationSymbol = '⇒ Stronger (S=${strengthP}) ⇒'; } // Show P is
365     → stronger
366     else if (qImpliesP) { relationSymbol = '⇐ Weaker (S=${strengthP}) ⇐'; } // Show P is weaker
367     else { relationSymbol = 'unrelated'; } // Added case for unrelated
368     relationArrowEl.textContent = relationSymbol;
369
370     // Update Base/Converse/Contra/Inverse validity displays (logic remains the same)
371     baseInferEl.innerHTML = `Base (P ⇒ Q): If X is <span class="term">${variableConceptP}</span>
372     → then X is <span class="term">${fixedConceptQ}</span>? <span class="status-indicator
373     → ${pImpliesQ ? 'valid' : 'invalid'}>${pImpliesQ ? 'Valid' : 'Invalid'}</span>`;
374     converseInferEl.innerHTML = `Converse (Q ⇒ P): If X is <span
375     → class="term">${fixedConceptQ}</span> then X is <span
376     → class="term">${variableConceptP}</span>? <span class="status-indicator ${qImpliesP ?
377     → 'valid' : 'invalid'}>${qImpliesP ? 'Valid' : 'Invalid'}</span>`;
378     const notQImpliesNotP = pImpliesQ; // Validity matches base
379     const notPImpliesNotQ = qImpliesP; // Validity matches converse
380     contraInferEl.innerHTML = `Contrapositive (¬Q ⇒ ¬P): If X is not <span
381     → class="term">${fixedConceptQ}</span> then X is not <span
382     → class="term">${variableConceptP}</span>? <span class="status-indicator ${notQImpliesNotP ?
383     → ? 'valid' : 'invalid'}>${notQImpliesNotP ? 'Valid' : 'Invalid'}</span>`;
384     inverseInferEl.innerHTML = `Inverse (¬P ⇒ ¬Q): If X is not <span
385     → class="term">${variableConceptP}</span> then X is not <span
386     → class="term">${fixedConceptQ}</span>? <span class="status-indicator ${notPImpliesNotQ ?
387     → 'valid' : 'invalid'}>${notPImpliesNotQ ? 'Valid' : 'Invalid'}</span>`;
388
389     // --- Calculate Validity ONCE at the beginning of this section ---
390     const ifPThenR_Valid = conceptEntailsProperty(variableConceptP, selectedPropertyR_value);
391     const ifRThenP_Valid = propertyEntailsConcept(selectedPropertyR_value, variableConceptP);
392
393     // --- Update Conditional Embedding displays (using the calculated values) ---
394
395     // Rebuild Antecedent Element Content
396     const antecedentSentenceHTML = `Antecedent Position: "If X is <span
397     → class="term">${variableConceptP}</span>, then X ${selectedPropertyR_desc}``;
398     condAntecedentEl.innerHTML = antecedentSentenceHTML; // Set the sentence HTML
399     // Use the already calculated 'ifPThenR_Valid'
400     condAntecedentStatusEl.className = `status-indicator ${ifPThenR_Valid ? 'valid' :
401     → 'invalid'}`;
402     condAntecedentStatusEl.textContent = ifPThenR_Valid ? 'Valid' : 'Invalid';
403     condAntecedentEl.appendChild(condAntecedentStatusEl); // Append the status indicator
404
405     // Rebuild Consequent Element Content
406     const consequentSentenceHTML = `Consequent Position: "If X ${selectedPropertyR_desc}, then X
407     → is <span class="term">${variableConceptP}</span>"`;
408     condConsequentEl.innerHTML = consequentSentenceHTML; // Set the sentence HTML
409     // Use the already calculated 'ifRThenP_Valid'
410     condConsequentStatusEl.className = `status-indicator ${ifRThenP_Valid ? 'valid' :
411     → 'invalid'}`;

```

```

396     condConsequentStatusEl.textContent = ifRThenP_Valid ? 'Valid' : 'Invalid';
397     condConsequentEl.appendChild(condConsequentStatusEl); // Append the status indicator
398 }
399
400 // Event Listeners
401 fixedSelect.addEventListener('change', updatePolarityDemo);
402 strengthSlider.addEventListener('input', updatePolarityDemo);
403 propertyRSelect.addEventListener('change', updatePolarityDemo);
404
405 updatePolarityDemo(); // Initial call
406 }; // End Module 4 Initializer
407
408 // Module 5: Substitution Argument + Animation
409 moduleInitializers[4] = function setupSubstitutionDemoViz() {
410     const exampleSelect = document.getElementById('substExampleSelectViz');
411     const frameSelect = document.getElementById('frameSelectViz');
412     const frameVizEl = document.getElementById('frameViz');
413     const exprA_VizEl = document.getElementById('exprA_Viz');
414     const exprB_VizEl = document.getElementById('exprB_Viz');
415     const animateBtn = document.getElementById('animateSubstButtonViz');
416     const resetBtn = document.getElementById('resetSubstButtonViz');
417     const baseSentenceEl = document.getElementById('baseSentenceViz');
418     const resultSentenceEl = document.getElementById('resultSentenceViz');
419     const infer1StatusEl = document.getElementById('infer1StatusViz');
420     const infer2StatusEl = document.getElementById('infer2StatusViz');
421     const analysisEl = document.getElementById('substAnalysisViz'); // The explanation area
422
423 // Define frames with polarity info
424 const frames = {
425     simple_assertion: {
426         template: "Shape S {expr}.",
427         display: "Shape S _.",
428         isInverting: false,
429         property: null
430     },
431     conditional_antecedent: { // UPDATED
432         template: "If Shape S {expr}, then S is a Rhombus.", // New Consequent
433         display: "If Shape S _, then S is a Rhombus.", // New Display
434         isInverting: true,
435         property: "is a Rhombus" // Store the consequent property
436     },
437     negation: {
438         template: "It is NOT the case that Shape S {expr}.",
439         display: "It is NOT the case that Shape S _.",
440         isInverting: true,
441         property: null
442     }
443 };
444
445 // Define examples with underlying rules
446 const examples = {
447     singularTerms: {
448         exprA: "Mark Twain",
449         exprB: "Samuel Clemens",
450         isSymmetric: true,
451         underlyingRule: "A ↔ B (Assumed Co-referential)",
452         explanationTemplate: "Substituting co-referential terms ('${exprA}' ↔ '${exprB}').\n→ Significance is SYMMETRIC."
453     },
454     predicatesStrongerToWeaker: {
455         exprA: "is a Square",
456         exprB: "is a Rectangle",
457         isSymmetric: false,
458         underlyingRule: "A ⇒ B (Square ⇒ Rectangle)",

```

```

459         explanationTemplate: "Substituting a stronger predicate ('${exprA}') with a weaker one  

460         ↪ ('${exprB}'). Significance is ASYMMETRIC."  

461     },  

462     predicatesWeakerToStronger: {  

463         exprA: "is a Rectangle",  

464         exprB: "is a Square",  

465         isSymmetric: false,  

466         underlyingRule: "A ⇒ B (Rectangle ⇒ Square), but B ⇒ A", // Note the base  

467         ↪ directionality  

468         explanationTemplate: "Substituting a weaker predicate ('${exprA}') with a stronger one  

469         ↪ ('${exprB}'). Significance is ASYMMETRIC."  

470     }  

471 };  

472  

473     let animationTimeout = null; // To clear existing timeouts on reset/change  

474  

475     // --- Helper to build sentences with spans for animation ---  

476     function buildSentence(template, expression) {  

477         // Simple replacement for now, assuming one placeholder '{expr}'  

478         // Escape expression to prevent HTML issues if needed, though unlikely here  

479         const escapedExpr = expression; // Simplification  

480         const placeholder = "{expr}";  

481         const parts = template.split(placeholder);  

482         if (parts.length === 2) {  

483             // Wrap the expression part in a span for animation targeting  

484             return parts[0] + `<span class="substituted-part">${escapedExpr}</span>` + parts[1];  

485         }  

486         return template.replace	placeholder, `<span  

487             ↪ class="substituted-part">${escapedExpr}</span>`); // Fallback  

488     }  

489  

490     // --- Setup the initial view ---  

491     function setupInitialView() {  

492         clearTimeout(animationTimeout); // Clear any pending animation  

493         animateBtn.disabled = false; // Re-enable button  

494  

495         const selectedExampleKey = exampleSelect.value;  

496         const selectedFrameKey = frameSelect.value;  

497         const example = examples[selectedExampleKey];  

498         const frame = frames[selectedFrameKey];  

499  

500         exprA_VizEl.textContent = example.exprA;  

501         exprB_VizEl.textContent = example.exprB;  

502         frameVizEl.textContent = frame.display;  

503  

504         // Reset styling on expression boxes  

505         exprA_VizEl.classList.remove('highlight-replace', 'lift-out-anim', 'fade-out-anim');  

506         exprB_VizEl.classList.remove('highlight-incoming');  

507         exprA_VizEl.style.opacity = '1'; exprA_VizEl.style.transform = '';  

508         exprB_VizEl.style.opacity = '1'; exprB_VizEl.style.transform = '';  

509  

510         // Build and display initial sentences  

511         baseSentenceEl.innerHTML = buildSentence(frame.template, example.exprA);  

512         // Reset result sentence visually, perhaps hide it initially or show placeholder  

513         resultSentenceEl.innerHTML = buildSentence(frame.template, '...'); // Placeholder  

514         resultSentenceEl.style.opacity = 0.5; // Dim it initially  

515  

516         // Clear status indicators  

517         infer1StatusEl.textContent = ''; infer1StatusEl.className = 'status-indicator';  

518         infer2StatusEl.textContent = ''; infer2StatusEl.className = 'status-indicator';  

519  

520         // Reset analysis text  

521         analysisEl.innerHTML = `<h4>4. Analysis: What Happens?</h4><p>Select an example and context,  

522         ↪ then click □ Animate to see the substitution and evaluate the inferences.</p>`;  

523     }

```

```

519
520    // Ensure spans inside sentences are reset
521    baseSentenceEl.querySelectorAll('.substituted-part, .substituting-part').forEach(span => {
522        span.style.opacity = '';
523        span.style.transform = '';
524        span.className = 'substituted-part'; // Ensure it starts as the base part
525    });
526    resultSentenceEl.querySelectorAll('.substituted-part, .substituting-part').forEach(span =>
527    {
528        span.textContent = '...';
529        span.style.opacity = '0';
530        span.style.transform = '';
531        span.className = 'substituted-part';
532    });
533
534
535    // --- Run the animation ---
536    function runSubstitutionAnimation() {
537        clearTimeout(animationTimeout); // Clear previous timeouts
538        animateBtn.disabled = true; // Disable button during animation
539
540        const selectedExampleKey = exampleSelect.value;
541        const selectedFrameKey = frameSelect.value;
542        const example = examples[selectedExampleKey];
543        const frame = frames[selectedFrameKey];
544
545        const baseSpan = baseSentenceEl.querySelector('.substituted-part');
546        const resultSpanTemplate = `<span class="substituting-part">${example.exprB}</span>`;
547        // Prepare the incoming part
548
549        // 1. Highlight the term to be substituted
550        exprA_VizEl.classList.add('highlight-replace');
551        if(baseSpan) baseSpan.style.backgroundColor = '#a8d5ff'; // Highlight in sentence
552
553        animationTimeout = setTimeout(() => {
554            // 2. Lift out the original term visually
555            exprA_VizEl.classList.add('lift-out-anim');
556            if(baseSpan) baseSpan.classList.add('lift-out-anim');
557
558            animationTimeout = setTimeout(() => {
559                // 3. Fade out the original term
560                exprA_VizEl.classList.add('fade-out-anim');
561                if (baseSpan) {
562                    baseSpan.classList.add('fade-out-anim');
563                }
564
565                // Prepare result sentence content *while* base fades
566                resultSentenceEl.innerHTML = baseSentenceEl.innerHTML.replace(/<span
567                class="substituted-part lift-out-anim fade-out-anim" [^>]*>.*?</span>/,
568                resultSpanTemplate);
569                resultSentenceEl.style.opacity = 1; // Make result sentence visible
570
571                animationTimeout = setTimeout(() => {
572                    // 4. Highlight the incoming term
573                    exprB_VizEl.classList.add('highlight-incoming');
574                    const resultSpan = resultSentenceEl.querySelector('.substituting-part');
575                    // Find the newly added span
576
577                    animationTimeout = setTimeout(() => {
578                        // 5. Move in the new term
579                        exprB_VizEl.classList.remove('highlight-incoming'); // Remove highlight
580                        if (resultSpan) {
581                            resultSpan.classList.add('move-in-anim'); // Trigger move-in style
582                        }
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2049
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2079
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2089
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2119
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2139
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2149
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2179
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2189
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2199
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2209
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2219
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2229
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2239
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2249
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2259
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2269
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2279
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2289
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2299
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2309
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2319
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2329
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2339
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2349
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2359
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2369
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2379
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2389
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2399
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2409
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2419
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2429
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2439
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2449
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2459
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2469
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2479
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2489
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2499
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2509
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2519
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2529
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2539
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2549
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2559
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2569
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2579
2579
2580
25
```

```

579         // 6. Evaluate and display inferences AFTER animation settles
580         animationTimeout = setTimeout(() => {
581             evaluateAndDisplayInferences(); // Calculate validity and update text
582             animateBtn.disabled = false; // Re-enable button
583             // Optional: Clean up animation classes on spans if needed, though
584             // → reset handles it
585             if(baseSpan) { baseSpan.className = 'substituted-part'; baseSpan.style
586             // → = ''; }
587             const finalResultSpan =
588             // → resultSentenceEl.querySelector('.substituting-part');
589             if(finalResultSpan) { finalResultSpan.className = 'substituted-part';
590             // → finalResultSpan.style = ''; } // Treat it as base now
591             }, 400); // Wait for move-in animation
592
593             }, 300); // Duration of incoming highlight
594
595             }, 300); // Duration of fade-out
596
597             }, 300); // Duration of lift-out
598
599             }, 200); // Initial highlight duration
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2198
2199
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2298
2299
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2398
2399
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2478
2479
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2489
2490
2491
2492
2493
2494
2495
2496
2497
2497
2498
2499
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2698
2699
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2729
2730
2731
2732
2733
2734
2735
2736

```

```

637     // --- Determine Validity of the Reverse Substitution Inference (Result => Base) ---
638     let infer2_valid = false;
639     if (isSymmetricExample) {
640         infer2_valid = true; // Symmetric
641     } else {
642         if (frame.isInverting) {
643             // Polarity inverted: Inference Result => Base is valid IF A materially implies B
644             infer2_valid = aMateriallyImpliesB;
645         } else {
646             // Normal polarity: Inference Result => Base is valid IF B materially implies A
647             infer2_valid = bMateriallyImpliesA;
648         }
649     }
650
651     // --- Update DOM elements ---
652     infer1StatusEl.textContent = infer1_valid ? 'Valid' : 'Invalid';
653     infer1StatusEl.className = `status-indicator ${infer1_valid ? 'valid' : 'invalid'}`;
654     infer2StatusEl.textContent = infer2_valid ? 'Valid' : 'Invalid';
655     infer2StatusEl.className = `status-indicator ${infer2_valid ? 'valid' : 'invalid'}`;
656
657     // Update analysis text based on results
658     let analysisText = `

#### 4. Analysis: What Happened?

`;
659     analysisText += `<p>Underlying Rule: <strong>${example.underlyingRule}</strong>. Context
660     ↪ Polarity: <strong>${frame.isInverting ? 'Inverting' : 'Non-Inverting'}</strong>.</p>`;
661     if (isSymmetricExample) {
662         analysisText += `<p>[] With <strong>Symmetric Terms</strong>, the substitution is
663         ↪ valid in <strong>both directions</strong> (${infer1_valid ? '✓' : '✗'} Base=>Result,
664         ↪ ${infer2_valid ? '✓' : '✗'} Result=>Base), regardless of the context's polarity.
665         ↪ This stability is key for terms referring to objects.</p>`;
666     } else {
667         analysisText += `<p>[] With <strong>Asymmetric Predicates</strong>:</p><ul>`;
668         analysisText += `<li>Base => Result validity (${infer1_valid ? '✓ Valid' : '✗ Invalid'})  

669         ↪ ${frame.isInverting ? 'depends on the REVERSE material rule (B=>A)' : 'depends on  

670         ↪ the FORWARD material rule (A=>B)'} because the context is ${frame.isInverting ?  

671         ↪ 'INVERTING' : 'Non-Inverting'}.</li>`;
672         analysisText += `<li>Result => Base validity (${infer2_valid ? '✓ Valid' : '✗ Invalid'})  

673         ↪ ${frame.isInverting ? 'depends on the FORWARD material rule (A=>B)' : 'depends on  

674         ↪ the REVERSE material rule (B=>A)'}.</li>`;
675         if (frame.isInverting && aMateriallyImpliesB !== bMateriallyImpliesA) {
676             analysisText += `<li style="color: purple; font-weight: bold;">[] Notice the flip!
677             ↪ The valid inference direction changed compared to a simple context because this
678             ↪ context is inverting.</li>`;
679         } else if (!frame.isInverting && aMateriallyImpliesB !== bMateriallyImpliesA) {
680             analysisText += `<li>This follows the basic asymmetric pattern, as the context is
681             ↪ not inverting.</li>`;
682         }
683         analysisText += `</ul>`;
684         analysisText += `<p><strong>The Breakdown:</strong> If '${termA}'/'${termB}' were
685         ↪ playing the basic 'Substituted-For' role but *had* this asymmetric rule, the rule
686         ↪ wouldn't work consistently across all contexts. One rule would demand A=>B in
687         ↪ simple contexts but B=>A in inverting ones!</p>`;
688         analysisText += `<p><strong>Brandom's Conclusion:</strong> Therefore, the basic
689         ↪ 'Substituted-For' role *must* have SYMMETRIC significance (like terms). The
690         ↪ asymmetric rules (like Square=>Rectangle) belong to the 'Frame' role (predicates),
691         ↪ and logic handles the polarity flips for those frames correctly.</p>`;
692     }
693     analysisEl.innerHTML = analysisText;
694 }
695
696
697 // Event Listeners
698 exampleSelect.addEventListener('change', setupInitialView);
699 frameSelect.addEventListener('change', setupInitialView);
700 animateBtn.addEventListener('click', runSubstitutionAnimation);

```

```
683     resetBtn.addEventListener('click', setupInitialView);
684
685     // Initial setup
686     setupInitialView(); // Make sure the view is correct on load
687 };
688
689 // Module 6: Matrix (Original Module 5)
690 moduleInitializers[5] = function() { /* Module 6 (Original 5) - Likely static */ };
691
692 // Module 7: Conclusion (Original Module 6)
693 moduleInitializers[6] = function() { /* Module 7 (Original 6) - Likely static */ };
694
695 // --- Initialize first module ---
696 showModule(0);
697
698}); // End DOMContentLoaded
```

## 2 Quadrilateral\_Substitution/brandom\_styles.css

```

1  /* brandom_styles.css */
2
3  /* --- General Page Styling --- */
4  body {
5      font-family: system-ui, -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, Oxygen, Ubuntu,
6      ↪ Cantarell, "Open Sans", "Helvetica Neue", sans-serif;
7      line-height: 1.6;
8      margin: 0;
9      padding: 0;
10     background-color: #f8f9fa; /* Lighter background */
11     color: #212529; /* Darker text for contrast */
12 }
13
14 header {
15     background-color: #343a40; /* Dark header */
16     color: #fff;
17     padding: 1.2em 0;
18     text-align: center;
19     margin-bottom: 2em;
20 }
21
22 main {
23     max-width: 950px;
24     margin: 2em auto;
25     padding: 2em;
26     background-color: #fff;
27     box-shadow: 0 2px 15px rgba(0, 0, 0, 0.1); /* Softer shadow */
28     border-radius: 8px;
29 }
30
31 footer {
32     text-align: center;
33     margin-top: 3em;
34     padding: 1em;
35     font-size: 0.85em;
36     color: #6c757d; /* Muted footer text */
37 }
38
39 /* --- Module Styling --- */
40 .module {
41     border: 1px solid #dee2e6; /* Light border */
42     padding: 1.5em 2em;
43     margin-bottom: 2.5em;
44     background-color: #fff;
45     transition: opacity 0.4s ease-in-out;
46     border-radius: 5px;
47 }
48
49 .current-module {
50     display: block;
51     opacity: 1;
52 }
53
54 .hidden-module {
55     display: none;
56     opacity: 0;
57 }
58
59 /* --- Typography and Structure --- */
60 h1, h2, h3, h4 {
61     color: #343a40; /* Consistent dark headings */
62     margin-bottom: 0.8em;
63 }
```

```
62 }
63
64 h2 {
65   border-bottom: 2px solid #ced4da; /* Subtler heading separator */
66   padding-bottom: 0.4em;
67   margin-top: 0;
68 }
69
70 h3 {
71   border-bottom: 1px solid #e9ecef; /* Very light separator for H3 */
72   padding-bottom: 0.3em;
73   margin-top: 1.5em;
74 }
75
76 h4 {
77   margin-top: 1.2em;
78   color: #495057; /* Slightly lighter heading */
79 }
80
81 p {
82   margin-bottom: 1em;
83 }
84
85 ul, ol {
86   margin-bottom: 1em;
87   padding-left: 1.8em; /* Slightly more indent */
88 }
89
90 li {
91   margin-bottom: 0.6em;
92 }
93
94 code, .term {
95   font-family: "SF Mono-Regular", Consolas, "Liberation Mono", Menlo, Courier, monospace;
96   background-color: #e9ecef; /* Light grey background */
97   padding: 0.2em 0.4em;
98   border-radius: 3px;
99   font-size: 0.9em;
100  border: 1px solid #dee2e6; /* Subtle border for terms */
101 }
102
103 .concept {
104   font-weight: 600; /* Slightly bolder */
105   color: #0056b3; /* Adjusted blue */
106 }
107
108 /* --- Explanations and Examples --- */
109 .explanation {
110   background: #f0f4f8; /* Very light blue-grey */
111   padding: 1em;
112   border: 1px solid #c9d6e5;
113   border-left: 5px solid #6c757d; /* Grey left border */
114   margin-top: 1.2em;
115   font-size: 0.95em;
116   border-radius: 0 4px 4px 0;
117 }
118
119 .example {
120   border-left: 4px solid #17a2b8; /* Teal border */
121   padding-left: 1em;
122   margin: 1.2em 0;
123   font-style: italic;
124   background-color: #f8f9fa;
125   padding: 0.8em;
126   border-radius: 4px;
```

```
127 }
128
129 /* --- Interactive Area --- */
130 .interactive-area {
131   margin: 2em 0;
132   padding: 1.5em;
133   border: 1px dashed #adb5bd; /* Dashed border */
134   background-color: #fdfdfe;
135   border-radius: 5px;
136 }
137
138 label {
139   margin-right: 0.5em;
140   display: inline-block;
141   min-width: 150px;
142   font-weight: 500;
143   margin-bottom: 0.3em; /* Add space below labels */
144 }
145
146 select, input[type=range], button {
147   font-size: 1em;
148   padding: 0.5em; /* Slightly more padding */
149   margin: 0.5em 0;
150   vertical-align: middle;
151   border-radius: 4px;
152   border: 1px solid #ced4da;
153 }
154
155 input[type=range] {
156   width: 50%;
157   margin: 0 0.5em;
158 }
159
160 button {
161   cursor: pointer;
162   background-color: #6c757d; /* Bootstrap secondary grey */
163   color: white;
164   border: none;
165   padding: 0.5em 1em;
166   transition: background-color 0.2s ease;
167 }
168
169 button:hover:not(:disabled) {
170   background-color: #5a687d; /* Darker grey on hover */
171 }
172
173 button:disabled {
174   cursor: not-allowed;
175   opacity: 0.6;
176   background-color: #adb5bd; /* Lighter grey when disabled */
177 }
178
179 /* --- Navigation --- */
180 #navigation {
181   text-align: center;
182   margin-top: 2.5em;
183   padding-bottom: 2em;
184 }
185
186 #navigation button {
187   padding: 0.7em 1.5em;
188   margin: 0 1em;
189   background-color: #007bff; /* Bootstrap primary blue */
190   font-size: 1em;
191   color: white;
```

```
192     border: none;
193     border-radius: 4px;
194 }
195
196 #navigation button:hover:not(:disabled) {
197   background-color: #0056b3; /* Darker blue on hover */
198 }
199
200 #navigation button:disabled {
201   background-color: #6c757d; /* Use secondary grey for disabled nav */
202 }
203
204 #moduleIndicator {
205   font-weight: bold;
206   color: #495057; /* Dark grey */
207   margin: 0 1.5em;
208   vertical-align: middle;
209 }
210
211 /* --- Visualization --- */
212 .viz-container {
213   display: flex;
214   align-items: center;
215   justify-content: center;
216   gap: 1em;
217   margin: 1.5em 0;
218   flex-wrap: wrap;
219   padding: 15px;
220   background-color: #e9ecef; /* Light background for viz */
221   border-radius: 5px;
222 }
223
224 .shape-viz svg {
225   border: 1px solid #ced4da; /* Slightly darker border */
226   background-color: white;
227   border-radius: 3px;
228   overflow: visible; /* Ensure text below is visible */
229 }
230
231 .arrow {
232   font-size: 1.8em;
233   margin: 0 0.8em;
234   color: #495057;
235 }
236
237 /* --- Status Indicators --- */
238 .status-indicator {
239   font-weight: bold;
240   padding: 0.3em 0.6em;
241   border-radius: 4px;
242   display: inline-block;
243   margin-left: 0.7em; /* More space */
244   font-size: 0.9em;
245   vertical-align: baseline;
246 }
247
248 .valid {
249   color: #155724; /* Darker green text */
250   background-color: #d4edda; /* Light green background */
251   border: 1px solid #c3e6cb; /* Green border */
252 }
253
254 .invalid {
255   color: #721c24; /* Darker red text */
256   background-color: #f8d7da; /* Light red background */
```

```

257     border: 1px solid #f5c6cb; /* Red border */
258 }
259
260 /* --- Substitution Styles --- */
261 .frame-style {
262   color: #495057;
263   background-color: #e9ecef;
264   padding: 0.2em 0.4em;
265   border: 1px solid #ced4da;
266   border-radius: 3px;
267   font-family: monospace;
268 }
269
270 .substituted-for-style {
271   color: #0d6efd; /* Primary blue */
272   font-weight: bold;
273   border-bottom: 2px solid #0d6efd;
274 }
275
276 .substituting-style {
277   color: #198754; /* Success green */
278   font-weight: bold;
279   border-bottom: 2px solid #198754;
280 }
281
282 .expression-box {
283   display: inline-block;
284   padding: 0.4em 0.8em; /* Slightly larger */
285   border: 1px solid #dee2e6;
286   background-color: #fff;
287   border-radius: 4px;
288   margin: 0 0.6em;
289   min-width: 90px;
290   text-align: center;
291   transition: transform 0.3s ease-out, opacity 0.3s ease-out;
292 }
293
294 .sentence-display .substituted-part {
295   display: inline-block;
296   background-color: #cfe2ff; /* Lighter primary blue */
297   padding: 1px 4px;
298   border-radius: 3px;
299   border: 1px dashed #9ec5fe;
300   transition: background-color 0.3s, opacity 0.3s, transform 0.3s; /* Add transition */
301 }
302
303 .sentence-display .substituting-part {
304   display: inline-block;
305   background-color: #d1e7dd; /* Lighter success green */
306   padding: 1px 4px;
307   border-radius: 3px;
308   border: 1px dashed #a3cfbb;
309   opacity: 0; /* Start invisible */
310   transition: opacity 0.3s ease-in, transform 0.3s ease-in; /* Add transition */
311 }
312
313 /* --- Animation Classes --- */
314 .highlight-replace {
315   box-shadow: 0 0 10px 3px rgba(0, 123, 255, 0.5); /* Brighter blue glow */
316 }
317
318 .lift-out {
319   transform: translateY(-10px) scale(1.05);
320   opacity: 0.7;
321 }

```

```

322     .fade-out {
323         opacity: 0 !important; /* Ensure it stays faded out */
324         transform: translateY(-20px) scale(0.8);
325     }
326
327     .move-in {
328         opacity: 1 !important; /* Ensure it becomes fully visible */
329         transform: translateY(0) scale(1);
330     }
331
332
333     /* --- Matrix Table --- */
334     .matrix-table {
335         border-collapse: collapse;
336         width: 100%;
337         margin: 1.5em 0;
338         font-size: 0.9em;
339         box-shadow: 0 1px 3px rgba(0,0,0,0.1);
340     }
341     .matrix-table th, .matrix-table td {
342         border: 1px solid #dee2e6;
343         padding: 12px; /* More padding */
344         text-align: left;
345         vertical-align: top;
346     }
347     .matrix-table th {
348         background-color: #e9ecf;
349         color: #495057;
350         font-weight: 600;
351     }
352     .matrix-table td ul {
353         padding-left: 1.5em;
354         margin-top: 0.5em;
355         margin-bottom: 0; /* Reduce extra space */
356     }
357     .matrix-table td li {
358         margin-bottom: 0.3em;
359     }
360
361     /* --- Info Icons & Tooltips --- */
362     .info-icon {
363         position: relative; /* Needed for absolute positioning of tooltip */
364         display: inline-block;
365         width: 18px; height: 18px;
366         background-color: #0d6efd;
367         color: white;
368         border-radius: 50%;
369         text-align: center;
370         font-size: 12px; line-height: 18px;
371         cursor: help;
372         margin-left: 8px; /* More spacing */
373         font-weight: bold;
374         vertical-align: middle;
375         position: relative; /* Needed for tooltip positioning */
376     }
377
378     .tooltip-text {
379         visibility: hidden;
380         width: 250px; /* Wider tooltips */
381         background-color: #343a40; /* Dark background */
382         color: #fff;
383         text-align: left;
384         border-radius: 6px;
385         padding: 10px; /* More padding */
386         position: absolute;

```

```

387     z-index: 10; /* Ensure tooltip is on top */
388     bottom: 140%; /* Position above the icon */
389     left: 50%;
390     margin-left: -125px; /* Center the tooltip */
391     opacity: 0;
392     transition: opacity 0.3s ease;
393     font-size: 0.9em;
394     font-weight: normal;
395     box-shadow: 0 3px 8px rgba(0,0,0,0.4);
396     pointer-events: none; /* Prevent tooltip from interfering with hover */
397   }
398
399   /* Arrow for tooltip */
400   .tooltip-text::after {
401     content: "";
402     position: absolute;
403     top: 100%; /* At the bottom of the tooltip */
404     left: 50%;
405     margin-left: -5px;
406     border-width: 5px;
407     border-style: solid;
408     border-color: #343a40 transparent transparent transparent;
409   }
410   .info-icon:hover .tooltip-text {
411     visibility: visible;
412     opacity: 1;
413   }
414
415   /* Styles for substitution animation elements */
416   .sentence-display .substituted-part {
417     display: inline-block; /* Crucial for transform */
418     background-color: #cfe2ff; /* Light blue */
419     padding: 1px 4px;
420     border-radius: 3px;
421     border: 1px dashed #9ec5fe;
422     transition: background-color 0.3s, opacity 0.3s, transform 0.3s ease-out;
423     /* Start with normal styling */
424   }
425   .sentence-display .substituting-part {
426     display: inline-block; /* Crucial for transform */
427     background-color: #d1e7dd; /* Light green */
428     padding: 1px 4px;
429     border-radius: 3px;
430     border: 1px dashed #a3cfbb;
431     opacity: 0; /* Start invisible */
432     transform: translateY(10px) scale(0.9); /* Start slightly below and small */
433     transition: opacity 0.3s 0.2s ease-in, transform 0.3s 0.2s ease-in; /* Delayed transition */
434   }
435
436   /* Animation classes */
437   .highlight-replace {
438     box-shadow: 0 0 10px 3px rgba(0, 123, 255, 0.5); /* Blue glow */
439     transition: box-shadow 0.2s ease-in-out;
440   }
441   .highlight-incoming {
442     box-shadow: 0 0 10px 3px rgba(25, 135, 84, 0.5); /* Green glow */
443     transition: box-shadow 0.2s ease-in-out;
444   }
445
446   .lift-out-anim {
447     transform: translateY(-10px) scale(1.05);
448     opacity: 0.5;
449   }
450
451   .fade-out-anim {

```

```
452     opacity: 0 !important;
453     transform: translateY(-20px) scale(0.8);
454 }
455
456 .move-in-anim {
457     opacity: 1 !important;
458     transform: translateY(0) scale(1) !important; /* Ensure final state */
459 }
460
461 /* Adjust Reset Button color */
462 #resetSubstButtonViz {
463     background-color: #ffc107; /* Warning yellow */
464     color: #343a40; /* Dark text for contrast */
465 }
466 #resetSubstButtonViz:hover {
467     background-color: #e0a800;
468 }
```

### 3 Quadrilateral\_Substitution/inferential\_strength.html

```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>Brandom Ch 4: Why Singular Terms?</title>
6      <link rel="stylesheet" href="inferential_strength_styles.css">
7      <!-- MathJax for mathematical notation -->
8      <script src="https://polyfill.io/v3/polyfill.min.js?features=es6"></script>
9      <script id="MathJax-script" async
   →   src="https://cdn.jsdelivr.net/npm/mathjax@3/es5/tex-mml-chtml.js"></script>
10 </head>
11 <body>
12     <header>
13         <h1>Why Singular Terms? An Interactive Guide to Brandom (AR Ch 4)</h1>
14         <p style="font-size: 0.9em; color: #ccc;">Exploring Substitution, Polarity, and the Structure of
   →   Concepts</p>
15     </header>
16
17     <main id="lessonContainer">
18         <button class="back-button" onclick="window.location.href='../../Calculator/index.html'">← Back to
   →   Calculator</button>
19
20         <!-- MODULE 1: Intro to Inferential Roles -->
21         <section id="module1" class="module current-module">
22             <h2>Module 1: Meaning as Inferential Role</h2>
23             <p>Welcome! This guide explores Robert Brandom's idea that the meaning of concepts lies in
   →   how they are used in reasoning (<span class="concept">inference</span>), rather than
   →   just what they point to (<span class="concept">reference</span>).</p>
24             <p>We'll use geometric shapes like squares and rectangles. Their relationships provide clear
   →   examples of <span class="concept">material inferences</span> – inferences valid because
   →   of the *contents* of the concepts, not just their logical form.</p>
25             <div class="example">
26                 Consider the inference: "If X is a <span class="term">Square</span>, then X is a <span
   →   class="term">Rectangle</span>."
27                 This is a good inference because the concept 'Square' includes all the properties of
   →   'Rectangle' (and more). Understanding 'Square' involves knowing this connection.
28             </div>
29             <div class="viz-container">
30                 <div class="shape-viz" id="viz-square-m1"></div>
31                 <div class="arrow">⇒</div>
32                 <div class="shape-viz" id="viz-rectangle-m1"></div>
33             </div>
34             <div class="explanation">
35                 Brandom calls 'Square' <span class="concept">inferentially stronger</span> than
   →   'Rectangle'. The stronger concept entails the weaker one. In the next modules, we'll
   →   see how this simple idea, combined with substitution, helps explain the structure of
   →   language.
36             </div>
37         </section>
38
39
40
41         <!-- MODULE 2: Quadrilateral Checklist & Strength -->
42         <section id="module2" class="module hidden-module">
43             <h2>Module 2: Content Example – Incompatibility & Strength</h2>
44             <p>In Module 1, we saw that meaning involves inferential connections (like "Square ⇒ Rectangle").
   →   Brandom suggests another way to grasp content is through <span
   →   class="concept">incompatibility</span> – what a concept *rules out*. A more specific concept
   →   rules out more, making it <span class="concept">inferentially stronger</span>.</p>
45             <p>Let's quantify this strength using quadrilaterals. We'll define strength as the <span
   →   class="concept">number of incompatibility restrictions</span> a shape rejects (its "hard
   →   no's").</p>

```

46 <p>Use the checkboxes below. Each checked box enforces a "hard no". See which shapes remain possible  
 47 → and note their calculated strength.</p>

48 <div class="interactive-area">  
 49   <h3>Quadrilateral Incompatibility Restrictions</h3>  
 50   <div class="checkbox-group" id="restrictionCheckboxesM2"> <!-- Added ID for easier JS targeting  
 51   → -->  
 52     <label title="Shape cannot have all sides of different lengths.">  
 53       <input type="checkbox" id="r1" data-property-key="r1" checked> Reject: "No sides are  
 54       → equal"  
 55     </label>  
 56     <label title="Shape cannot have every pair of adjacent sides be of different lengths.">  
 57       <input type="checkbox" id="r2" data-property-key="r2" checked> Reject: "No pair of  
 58       → adjacent sides are equal"  
 59     </label>  
 60     <label title="Shape cannot have every pair of opposite sides be of different lengths.">  
 61       <input type="checkbox" id="r3" data-property-key="r3" checked> Reject: "No pair of  
 62       → opposite sides are equal"  
 63     </label>  
 64     <label title="Shape cannot have unequal non-parallel sides (if applicable).">  
 65       <input type="checkbox" id="r4" data-property-key="r4" checked> Reject: "Non-parallel  
 66       → sides are not congruent"  
 67     </label>  
 68     <label title="Shape cannot have zero pairs of parallel sides.">  
 69       <input type="checkbox" id="r5" data-property-key="r5" checked> Reject: "No pair of  
 70       → opposite sides are parallel"  
 71     </label>  
 72     <label title="Shape cannot lack right angles.">  
 73       <input type="checkbox" id="r6" data-property-key="r6" checked> Reject: "No angles are  
 74       → right angles"  
 75     </label>  
 76 </div>  
 77 <div class="explanation">  
 78   <p><strong>How It Works:</strong> Checking a box means the shape \*must be incompatible\* with  
 79   → that property (it must have a 'No' for it in Table 1, represented as `false` in the  
 80   → data). Unchecking a box \*eases\* that restriction.</p>  
 81   <p>A shape's <span class="concept">Strength</span> is the count of checked restrictions it  
 82   → satisfies (i.e., the number of properties it rejects).</p>  
 83 </div>  
 84 <h4>Possible Quadrilaterals (and their Strength):</h4>  
 85 <div id="shapesContainerM2" class="viz-container" style="min-height: 100px; justify-content:  
 86   flex-start;"> <!-- Added ID, changed display slightly -->  
 87   <!-- The list of matching shapes with SVG graphics and strength will appear here -->  
 88   <p>Initializing...</p>  
 89 </div>  
 90 <div class="explanation" style="margin-top: 2em;">  
 91   <h4>Connecting to Inferential Roles</h4>  
 92   <p>This demonstrates how the specific incompatibilities defining a concept determine its <span  
 93   → class="concept">inferential strength</span>. A Square (Strength 6) rejects all restrictions,  
 94   → while a general Quadrilateral (Strength 0) rejects none. Crucially, a Square rejects \*all\*  
 95   → the restrictions a Rectangle rejects, which reflects the inferential entailment: <span  
 96   → class="term">Square ⇒ Rectangle</span>. This strength metric will help us understand  
 97   → polarity inversion in Module 4.</p>  
 98 </div>  
 99 </section>  
100 <!-- MODULE 3: Substitution Roles & Significance -->

```

93
94
95     <section id="module3" class="module hidden-module">
96         <h2>Module 3: Substitution Roles & Significance</h2>
97         <p>To understand Brandom's argument, we first need to grasp how he analyzes sentences using
98             ↳ <span class="concept">substitution</span>. This involves seeing sentences as built from
99                 ↳ parts that can be swapped out.</p>
100
101     <h3>Substitution Roles (Syntax)</h3>
102     <p>When we substitute one expression for another within a sentence, we can distinguish two
103         ↳ main roles:</p>
104     <ul>
105         <li><span class="concept substituted-for-style">Substituted-For:</span> The expression
106             ↳ being replaced (e.g., '<span class="term">Mark Twain</span>' in "Mark Twain wrote
107                 ↳ HF"). These are typically the basic building blocks.</li>
108         <li><span class="concept frame-style">Substitutional Frame:</span> The part of the
109             ↳ sentence that remains constant when substitution occurs (e.g., '<span
110                 ↳ class="term">... wrote Huckleberry Finn</span>'). These frames are derived from
111                 ↳ sentences by seeing them as patterns.</li>
112     </ul>
113     <div class="example">
114         Sentence: "<span class="term">Fido is a dog</span>"<br>
115         If we substitute '<span class="term">Fido</span>' with '<span class="term">Rex</span>',  

116             ↳ the frame is "<span class="term">... is a dog</span>". '<span
117                 ↳ class="term">Fido</span>' is <span
118                     ↳ class="substituted-for-style">substituted-for</span>.<br>
119         If we replace '<span class="term">... is a dog</span>' with '<span class="term">... is a
120             ↳ mammal</span>', the frame itself is replaced. '<span class="term">... is a
121                 ↳ dog</span>' acts as the <span class="frame-style">frame</span>.
122     </div>
123
124     <h3>Substitution Significance (Semantics)</h3>
125     <p>The *meaning* or content contributed by an expression is revealed by how substitution
126         ↳ affects inferences:</p>
127     <ul>
128         <li><span class="concept">Symmetric Significance:</span> Substituting A for B (or B for
129             ↳ A) always preserves the correctness of inferences the sentence is involved in.
130             Typical of <span class="concept">singular terms</span> that co-refer. The inference
131                 ↳ is reversible ( $A \leftrightarrow B$ ).</li>
132         <li><span class="concept">Asymmetric Significance:</span> Substituting A for B preserves
133             ↳ correctness only one way. Typical of <span class="concept">predicates</span> related
134                 ↳ by strength (e.g., 'Square'  $\Rightarrow$  'Rectangle', but not vice-versa). The inference is
135                     ↳ one-way ( $A \Rightarrow B$ ).</li>
136     </ul>
137     <div class="example">
138         Symmetric: "Mark Twain wrote HF"  $\Leftrightarrow$  "Samuel Clemens wrote HF".<br>
139         Asymmetric: "Shape S is a Square"  $\Rightarrow$  "Shape S is a Rectangle", but "Shape S is a
140             ↳ Rectangle" <span style="text-decoration: line-through;"> $\Rightarrow$ </span> "Shape S is a
141                 ↳ Square".
142     </div>
143     <div class="explanation">
144         Brandom argues that expressions playing the <span
145             ↳ class="substituted-for-style">substituted-for</span> syntactic role (like terms)
146             ↳ have <span class="concept">symmetric</span> semantic significance, while expressions
147                 ↳ playing the <span class="frame-style">frame</span> role (like predicates) typically
148                     ↳ have <span class="concept">asymmetric</span> significance. Module 4 explores *why*
149                         ↳ this specific combination is necessary in a logically expressive language.
150     </div>
151 </section>
152
153     <!-- MODULE 4: Polarity & Logical Operators -->
154 <section id="module4" class="module hidden-module">
155     <h2>Module 4: Inferential Polarity – Strength & Logic</h2> <!-- Updated Title -->

```

```

129   <p>Logical operators like <span class="term">if...then...</span> and <span class="term">not</span>
→ interact with the <span class="concept">inferential strength</span> of concepts (measured by
→ rejected restrictions from Module 2). Some logical contexts <span class="concept">invert</span>
→ the relationship between concept strength and the strength/validity of the overall claim.</p>
130
131 <div class="interactive-area">
132   <h3>Polarity Demonstration</h3>
133
134   <div style="margin-bottom: 1em;">
135     <label for="fixedConceptSelectM4">Fixed Concept (Q):</label> <!-- Added M4 to ID -->
136     <select id="fixedConceptSelectM4"></select>
137     <strong style="margin-left:10px;">Strength: <span id="strengthQM4">-</span></strong> <!--
→ Display Q Strength -->
138     <span class="info-icon" title="This concept remains constant in the
→ Base/Converse/Contrapositive/Inverse inferences.">?</span>
139   </div>
140
141   <div style="margin-bottom: 1em;">
142     <label for="strengthSliderM4">Variable Concept (P):</label> <!-- Added M4 to ID -->
143     <span id="sliderMinLabelM4" style="font-size: 0.8em;">Weaker</span>
144     <input type="range" id="strengthSliderM4" min="0" max="4" value="4" step="1"> <!-- Adjusted
→ max based on sliderChain -->
145     <span id="sliderMaxLabelM4" style="font-size: 0.8em;">Stronger</span>
146     <br>
147     <span style="margin-left: 155px; font-weight: bold;">
→ id="variableConceptLabelM4">Square</span> <!-- Updated Default -->
148     <strong style="margin-left:10px;">Strength: <span id="strengthPM4">-</span></strong> <!--
→ Display P Strength -->
149     <span class="info-icon" title="Slide to change Concept P along the hierarchy: Quadrilateral
→ ⇔ Trapezoid ⇔ Parallelogram ⇔ Rectangle ⇔ Square. Note its strength changes.">?</span>
150   </div>
151
152   <div class="viz-container">
153     <div class="shape-viz" id="vizPM4"></div> <!-- Added M4 -->
154     <div class="arrow" id="relationArrowM4">?</div> <!-- Added M4 -->
155     <div class="shape-viz" id="vizQM4"></div> <!-- Added M4 -->
156   </div>
157   <hr>
158
159   <h4>1. Base Material Inference (P vs Q)</h4>
160   <div id="baseInferM4" class="example">P = Q: ...</div> <!-- Added M4 -->
161   <div id="converseInferM4" class="example">Q  $\Rightarrow$  P: ...</div> <!-- Added M4 -->
162   <div class="explanation">Observe how validity changes as you alter P's strength relative to Q's
→ strength. <span class="concept">P  $\Rightarrow$  Q is valid if P is stronger than or equivalent to
→ Q</span> (meaning P rejects at least all the properties Q rejects, i.e., `isSubclass(P,
→ Q)`).</div>
163
164   <h4>2. Negation & Polarity (Contrapositive & Inverse)</h4>
165   <div id="contraInferM4" class="example"> $\neg$ Q  $\Rightarrow$   $\neg$ P: ...</div> <!-- Added M4 -->
166   <div id="inverseInferM4" class="example"> $\neg$ P  $\Rightarrow$   $\neg$ Q: ...</div> <!-- Added M4 -->
167   <div class="explanation">
168     The <span class="concept">Contrapositive</span> ( $\neg$ Q  $\Rightarrow$   $\neg$ P) validity matches the Base
→ Inference (P  $\Rightarrow$  Q). Why? Negation reverses the direction of inference: If having P's
→ properties *forces* having Q's properties (P  $\Rightarrow$  Q), then *lacking* Q's properties must
→ *force* lacking P's properties ( $\neg$ Q  $\Rightarrow$   $\neg$ P). The <span class="concept">Inverse</span> ( $\neg$ P
→  $\Rightarrow$   $\neg$ Q) validity matches the Converse (Q  $\Rightarrow$  P) for the same reason.
169   </div>
170
171   <h4>3. Conditional Embedding Polarity</h4>
172   <div style="margin-bottom: 1em;">
173     <label for="propertyRSelectM4">Fixed Property (R):</label> <!-- Added M4 -->
174     <select id="propertyRSelectM4"></select>
175     <span class="info-icon" title="Select the property used in the 'if' or 'then' clause of the
→ conditionals below.">?</span>

```

```

176     </div>
177     <p style="font-size: 0.9em;">Comparing conditionals using the variable concept P (<span
178     ↵ style="font-weight:bold;" id="variableConceptLabelCondM4">...</span>) and the fixed
179     ↵ property R (<span style="font-weight:bold;" id="propertyRLabelM4">...</span>):</p>
180
181     <div id="condAntecedentM4" class="example"> <!-- Added M4 -->
182         Antecedent Position: "If X is <span class="term">[P]</span>, then X <span
183         ↵ class="term">[R]</span>"<br/>
184         <span id="condAntecedentStatusM4" class="status-indicator"></span>
185     </div>
186     <div id="condConsequentM4" class="example"> <!-- Added M4 -->
187         Consequent Position: "If X <span class="term">[R]</span>, then X is <span
188         ↵ class="term">[P]</span>"<br/>
189         <span id="condConsequentStatusM4" class="status-indicator"></span>
190     </div>
191     <div class="explanation">
192         The antecedent ('if...') position <span class="concept">inverts polarity</span>. Making P
193         ↵ <span style="color:red;">weaker</span> (sliding left, <span
194         ↵ style="color:red;">decreasing Strength P</span>) makes the conditional statement 'If P
195         ↵ then R' <span style="color:green;">stronger overall</span> as a claim. It demands R
196         ↵ hold true under *more* conditions (e.g., for all Rectangles, not just Squares).
197         Stronger claims are *less* likely to be valid. Conversely, making P <span
198         ↵ style="color:green;">stronger</span> (sliding right, <span
199         ↵ style="color:green;">increasing Strength P</span>) makes 'If P then R' <span
200         ↵ style="color:red;">weaker overall</span> (applies to fewer cases), making it *more*
201         ↵ likely to be valid. <br>
202         The consequent ('then...') position does <span class="concept">not invert polarity</span>.
203         ↵ Making P <span style="color:red;">weaker</span> (<span style="color:red;">decreasing
204         ↵ Strength P</span>) makes the conditional 'If R then P' <span style="color:red;">weaker
205         ↵ overall</span> (easier to satisfy, *more* likely valid).
206     </div>
207     </div>
208 </section>
209
210     <!-- MODULE 5: The Argument – Why Terms Must Be Symmetric -->
211     <!-- MODULE 5: The Argument – Why Terms Must Be Symmetric -->
212     <section id="module5" class="module hidden-module">
213         <h2>Module 5: The Argument – Why Terms Must Be Symmetric</h2>
214         <p>We've seen that expressions have <span class="concept">syntactic roles</span> (<span
215         ↵ class="substituted-for-style">Substituted-For</span> vs. <span class="frame-style">Frame</span>)
216         ↵ and <span class="concept">semantic significance</span> (Symmetric vs. Asymmetric inferential
217         ↵ behavior, reflected in <span class="concept">Strength</span>). We also saw how logical contexts
218         (<code class="term">if</code>, <code class="term">not</code>) can <span class="concept">invert
219         ↵ polarity</span>, flipping inferential relationships (Module 4).</p>
220         <p><strong>Brandom's Core Question:</strong> Can expressions playing the <span
221         ↵ class="substituted-for-style">Substituted-For</span> role (the role singular terms play)
222         ↵ consistently have *Asymmetric* significance (like predicates, e.g., 'Square' ⇒ 'Rectangle') in a
223         ↵ language that also uses logic?</p>
224         <p>Let's test this. Use the visualization with the improved "If S {expr}, then S is a Rhombus"
225         ↵ context. Pay close attention to how the validity of the sentences and the inferences
226         ↵ *changes*.</p>
227
228         <div class="interactive-area">
229             <h3>Substitution Visualization & Argument</h3>
230
231             <label for="substExampleSelectViz">1. Choose Substitution Type:</label>
232             <select id="substExampleSelectViz">
233                 <option value="singularTerms">Symmetric: Terms ('Mark Twain' ⇔ 'Samuel Clemens')</option>
234                 <option value="predicatesStrongerToWeaker" selected>Asymmetric: Predicates ('Square' ⇒
235                 ↵ 'Rectangle')</option>
236                 <option value="predicatesWeakerToStronger">Asymmetric: Predicates ('Rectangle' ⇒
237                 ↵ 'Square')</option>
238             </select>

```

```

211    <span class="info-icon">? <span class="tooltip-text">Symmetric: Inference A⇒B and B⇒A are both
212    ↳ valid (like co-referring terms). Asymmetric: Only one direction holds (A⇒B or B⇒A, like
213    ↳ stronger/weaker predicates).</span></span>
214    <br>
215
216    <label for="frameSelectViz">2. Choose Sentence Context (Frame):</label>
217    <select id="frameSelectViz">
218        <option value="simple_assertion">Simple: "Shape S {expr}." (Non-Inverting)</option>
219        <!-- UPDATED OPTION TEXT -->
220        <option value="conditional_antecedent">Conditional Antecedent: "If Shape S {expr}, then S is
221        ↳ a Rhombus." (Inverting)</option>
222        <option value="negation">Negation: "It is NOT the case that Shape S {expr}."</option>
223        <br><br>
224        <!-- Animation Area remains the same -->
225        <div id="substitutionAnimationArea" style="margin-top: 1.5em; text-align: center; font-size:
226        ↳ 1.1em;">
227            <p style="margin-bottom: 0.5em;"><strong>Context/Frame:</strong> <span id="frameViz"
228            ↳ class="frame-style">Frame Text...</span></p>
229            <div style="margin: 1em 0; display: flex; justify-content: center; align-items: center; gap:
230            ↳ 1em; flex-wrap: wrap;">
231                <span style="text-align: center;">Original Expression (A):<br><span id="exprA_Viz"
232                ↳ class="expression-box substituted-for-style">Expr A</span></span>
233                <span class="arrow" style="flex-shrink: 0;"> Substituting with --</span>
234                <span style="text-align: center;">New Expression (B):<br><span id="exprB_Viz"
235                ↳ class="expression-box substituting-style">Expr B</span></span>
236            </div>
237            <button id="animateSubstButtonViz">[] Animate Substitution</button>
238            <button id="resetSubstButtonViz">[] Reset View</button>
239        </div>
240        <hr style="margin: 1.5em 0;">
241
242        <!-- Results Area remains the same structure -->
243        <div id="substitutionResultAreaViz" style="margin-top: 1em;">
244            <h4>3. Resulting Sentences & Inferences:</h4>
245            <p><strong>Base Sentence:</strong> <span id="baseSentenceViz" class="term"
246            ↳ sentence-display">Initial sentence structure.</span></p>
247            <p><strong>Result Sentence:</strong> <span id="resultSentenceViz" class="term"
248            ↳ sentence-display">Sentence after substitution.</span></p>
249
250            <div id="subInfer1Viz" class="example">
251                Inference 1: <span class="term">[Base]</span> implies <span
252                ↳ class="term">[Result]</span>?
253                <span id="infer1StatusViz" class="status-indicator"></span>
254                <span class="info-icon">?<span class="tooltip-text">Is inferring the Result sentence
255                ↳ FROM the Base sentence valid? Depends on the expressions' rule (A⇒B or A↔B) AND the
256                ↳ context's polarity.</span></span>
257            </div>
258            <div id="subInfer2Viz" class="example">
259                Inference 2: <span class="term">[Result]</span> implies <span
260                ↳ class="term">[Base]</span>?
261                <span id="infer2StatusViz" class="status-indicator"></span>
262                <span class="info-icon">?<span class="tooltip-text">Is inferring the Base sentence FROM
263                ↳ the Result sentence valid? Depends on the expressions' rule (B⇒A or A⇒B) AND the
264                ↳ context's polarity.</span></span>
265            </div>
266        </div>
267        <hr style="margin: 1.5em 0;">

```

```

256    <!-- SIGNIFICANTLY UPDATED Explanation -->
257    <div id="substAnalysisViz" class="explanation" aria-live="polite">
258        <h4>4. Analysis: Why Terms MUST Be Symmetric</h4>
259        <p>Select "Asymmetric: Predicates ('Square'  $\Rightarrow$  'Rectangle')) and the "Conditional Antecedent
260             $\hookrightarrow$  (...then S is a Rhombus)" context, then Animate.</p>
261        <ul>
262            <li><strong>Base Sentence:</strong> "If S is <span class='term'>Square</span>, then S
263                 $\hookrightarrow$  is a Rhombus" is <span class='valid'>Valid</span> (geometrically true).</li>
264            <li><strong>Result Sentence:</strong> "If S is <span class='term'>Rectangle</span>,
265                 $\hookrightarrow$  then S is a Rhombus" is <span class='invalid'>Invalid</span> (geometrically
266                 $\hookrightarrow$  false).</li>
267            <li><strong>Inference 1 (Base  $\Rightarrow$  Result):</strong> Valid Sentence  $\Rightarrow$  Invalid Sentence.
268                 $\hookrightarrow$  This inference is <span class='invalid'>Invalid</span>. </li>
269            <li><strong>Inference 2 (Result  $\Rightarrow$  Base):</strong> Invalid Sentence  $\Rightarrow$  Valid Sentence.
270                 $\hookrightarrow$  This inference is <span class='valid'>Valid</span> (a falsehood implies
271                 $\hookrightarrow$  anything).</li>
272            <li style="margin-top:1em;">
273                <strong style="color:purple;">The Flip in Action:</strong> The underlying material
274                     $\hookrightarrow$  rule is <span class="term">Square</span>  $\Rightarrow$  <span class="term">Rectangle</span>
275                     $\hookrightarrow$  ( $A \Rightarrow B$ ).
276            <ul>
277                <li>In a Simple Context (Module 3), substituting A with weaker B makes the
278                     $\hookrightarrow$  inference  $A \Rightarrow B$  <span class='valid'>Valid</span>. </li>
279                <li>But here, in the <span class="concept">Inverting Context</span> (Conditional
280                     $\hookrightarrow$  Antecedent), the valid substitution inference was Inference 2 (Result =
281                     $\Rightarrow$  Base). This direction corresponds to needing the material rule  $B \Rightarrow A$  (<span
282                    class="term">Rectangle</span>  $\Rightarrow$  <span class="term">Square</span>), which is
283                     $\hookrightarrow$  false! The direction required for a valid substitution has <span
284                    style="font-weight:bold;">flipped</span> compared to the simple
285                     $\hookrightarrow$  context.</li>
286            </ul>
287        </li>
288        <li style="margin-top:1em;">
289            <strong style="color:red;">The Contradiction for Hypothetical "Asymmetric
290                 $\hookrightarrow$  Terms":</strong>
291            Imagine '<span class="substituted-for-style">SquareTerm</span>' was a
292                 $\hookrightarrow$  Substituted-For term with the rule "<span class='term'>SquareTerm  $\Rightarrow$ 
293                 $\hookrightarrow$  RectangleTerm</span>".
294            <ul>
295                <li>This single rule ( $A \Rightarrow B$ ) would need to make the substitution <span
296                    class="term">[Base]</span> $\Rightarrow$ <span class="term">[Result]</span> valid in
297                    simple contexts.</li>
298                <li>But it would also need to make the substitution <span
299                    class="term">[Result]</span> $\Rightarrow$ <span class="term">[Base]</span> valid in
300                    inverting contexts (like the one above).</li>
301                <li>One fixed asymmetric rule ( $A \Rightarrow B$ ) cannot simultaneously satisfy both
302                    requirements! It leads to a contradiction depending on the logical
303                    context.</li>
304            </ul>
305        </li>
306        <li style="margin-top:1em;">
307            <strong style="color:green;">□ Brandom's Conclusion:</strong>
308            To allow consistent substitution across *all* logical contexts, the <span
309                 $\hookrightarrow$  class="substituted-for-style">Substituted-For</span> role *must* have <span
310                 $\hookrightarrow$  class="concept">Symmetric</span> significance ( $A \Leftrightarrow B$ ). This is why we have
311                 $\hookrightarrow$  singular terms that behave like 'Mark Twain'  $\Leftrightarrow$  'Samuel Clemens', providing
312                 $\hookrightarrow$  stable reference. The asymmetries essential to meaning (like Square =
313                 $\Rightarrow$  Rectangle) belong to the <span class="frame-style">Frame</span> (predicate)
314                 $\hookrightarrow$  role, where logical operators correctly handle the necessary polarity flips.
315            </li>
316        </ul>
317        <p><em>Try substituting Symmetric Terms ('Twain'/'Clemens') – notice both Inference 1 and 2
318             $\hookrightarrow$  remain Valid in *all* contexts. Symmetry provides the required stability.</em></p>
319    </div>

```

```

288     </div>
289 </section>
290
291     <!-- MODULE 6: The Matrix of Possibilities -->
292     <section id="module6" class="module hidden-module">
293         <h2>Module 6: The Matrix of Substitutional Possibilities</h2>
294         <p>Brandom considers four theoretical ways a language could structure substitutional roles
295             ↳ based on the syntactic role (<span class="concept"
296                 ↳ substituted-for-style">Substituted-For</span> vs. <span class="concept
297                 ↳ frame-style">Frame</span>) and the semantic significance (<span
298                 ↳ class="concept">Symmetric</span> vs. <span class="concept">Asymmetric</span>).</p>
299         <table class="matrix-table">
300             <thead>
301                 <tr>
302                     <th></th>
303                     <th>Substitutional Frame has SYMMETRIC Significance</th>
304                     <th>Substitutional Frame has ASYMMETRIC Significance</th>
305                 </tr>
306             </thead>
307             <tbody>
308                 <tr>
309                     <th>Substituted-For expression has SYMMETRIC Significance</th>
310                     <td><strong>(i) Both Symmetric</strong><br>
311                         <ul><li><em>Problem:</em> Fails to capture asymmetric predicate inferences
312                             ↳ (e.g., genus/species). Too expressively weak.</li></ul>
313                     </td>
314                     <td><strong>(iv) Terms Symmetric, Predicates Asymmetric</strong><br>
315                         <ul><li><em>Result:</em> This is Brandom's proposed structure, corresponding
316                             ↳ to Singular Terms (Substituted-For, Symmetric) and Predicates (Frames,
317                             ↳ Asymmetric). It allows both stable object reference and hierarchical
318                             ↳ conceptual relations, and is compatible with logic. □</li></ul>
319                     </td>
320                 </tr>
321                 <tr>
322                     <th>Substituted-For expression has ASYMMETRIC Significance</th>
323                     <td><strong>(ii) Terms Asymmetric, Predicates Symmetric</strong><br>
324                         <ul><li><em>Problem:</em> Fails to capture asymmetric predicate inferences
325                             ↳ (as roles are swapped vs. (iv)). Also too weak.</li></ul>
326                     </td>
327                     <td><strong>(iii) Both Asymmetric</strong><br>
328                         <ul><li><em>Problem:</em> Leads to incoherence when combined with logical
329                             ↳ operators (conditionals/negation) that invert polarity, as demonstrated
330                             ↳ in Module 4. Cannot consistently project inferences.</li></ul>
331                     </td>
332                 </tr>
333             </tbody>
334         </table>
335         <div class="explanation">
336             The argument presented in Module 4 aims to rule out options (i), (ii), and especially
337             ↳ (iii) for any language rich enough to contain basic sentential logic. This leaves
338             ↳ only option (iv), the familiar structure distinguishing singular terms from
339             ↳ predicates based on their different syntactic roles and semantic
340             ↳ (substitution-inferential) significances.
341         </div>
342     </section>
343
344     <!-- MODULE 7: Conclusion -->
345     <section id="module7" class="module hidden-module">
346         <h2>Module 7: Conclusion – The Expressive Deduction</h2>
347         <p>Brandom's argument provides an <span class="concept">expressive deduction of the
348             ↳ necessity of singular terms and predicates</span>. It's 'expressive' because it relies
349             ↳ on what's needed for a language to explicitly express its own inferential structure
350             ↳ using logic. It's a 'deduction' because it argues this structure is a necessary
351             ↳ consequence of combining basic substitution with logical operators.</p>

```

```

333     <p>The argument shows that:</p>
334     <ul>
335         <li>Languages use <span class="concept">substitution</span> to create novel sentences
336             → from existing parts (projectibility).</li>
337         <li>Understanding meaning <span class="concept">inferentially</span> involves tracking
338             → symmetric vs. asymmetric substitution patterns.</li>
339         <li><span class="concept">Logical vocabulary</span> (conditionals, negation) makes these
340             → inferential patterns explicit but also introduces <span
341                 → class="concept">polarity-inverting</span> contexts.</li>
342         <li>Compatibility with these inverting contexts <span class="concept">requires</span>
343             → that the basic substituted-for expressions have <span
344                 → class="concept">symmetric</span> significance (functioning as singular terms) while
345             → the substitutional frames have <span class="concept">asymmetric</span> significance
346             → (functioning as predicates).</li>
347     </ul>
348     <div class="explanation">
349         Ultimately, the reason we structure our talk around <span class="concept">objects</span>
350             → (named by singular terms) having <span class="concept">properties</span> (ascribed
351             → by predicates) is deeply connected to our capacity for logical, inferential
352             → reasoning. It's the structure demanded by a language that can not only make claims
353             → but reflect on the inferential connections between them.
354     </div>
355     </section>
356 </main>
357
358     <nav id="navigation">
359         <button id="prevButton" disabled>Previous Module</button>
360         <span id="moduleIndicator">Module 1 of 7</span>
361         <button id="nextButton">Next Module</button>
362     </nav>
363
364     <footer>
365         <p>Based on Robert Brandom's "Articulating Reasons", Chapter 4.</p>
366     </footer>
367
368     <script src="brandom_lesson.js"></script> <!-- Link to your JS file -->
369 </body>
370 </html>

```

## 4 Quadrilateral\_Substitution/inferential\_strength\_styles.css

```

1  /* Inferential Strength Interactive Lesson - Improved Styles */
2
3  @import
4    →  url('https://fonts.googleapis.com/css2?family=Orbitron:wght@400;700&family=Inter:wght@400;500;600;700&display=swap');
5
6  /* Basic Layout */
7  body {
8    font-family: 'Inter', -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, sans-serif;
9    line-height: 1.6;
10   margin: 0;
11   padding: 0;
12   background: linear-gradient(135deg, #e0e0e0 0%, #c0c0c0 100%);
13   color: #212529;
14 }
15
16 header {
17   background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
18   color: #fff;
19   padding: 2em 0;
20   text-align: center;
21   margin-bottom: 2em;
22   box-shadow: 0 4px 15px rgba(0, 0, 0, 0.2);
23 }
24
25 header h1 {
26   font-family: 'Orbitron', sans-serif;
27   font-size: 2em;
28   margin: 0 0 0.3em 0;
29   font-weight: 700;
30 }
31
32 header p {
33   font-size: 1em;
34   color: rgba(255, 255, 255, 0.85);
35   margin: 0;
36 }
37
38 main {
39   max-width: 950px;
40   margin: 2em auto;
41   padding: 2em;
42   background-color: #fff;
43   box-shadow: 0 10px 30px rgba(0, 0, 0, 0.15);
44   border-radius: 15px;
45 }
46
47 .module {
48   border: 1px solid #dee2e6;
49   padding: 2em;
50   margin-bottom: 2.5em;
51   background-color: #fff;
52   transition: opacity 0.4s ease-in-out;
53   border-radius: 10px;
54 }
55
56 .current-module {
57   display: block;
58   opacity: 1;
59 }
60
61 .hidden-module {
62   display: none;

```

```
62     opacity: 0;
63 }
64
65 /* Typography */
66 h1, h2, h3, h4 {
67   font-family: 'Orbitron', sans-serif;
68   color: #343a40;
69 }
70
71 h2 {
72   border-bottom: 3px solid #667eea;
73   padding-bottom: 0.5em;
74   margin-top: 0;
75   margin-bottom: 1.2em;
76   color: #667eea;
77   font-weight: 700;
78 }
79
80 h3 {
81   border-bottom: 2px solid #e9ecef;
82   padding-bottom: 0.4em;
83   margin-top: 1.5em;
84   font-weight: 600;
85 }
86
87 h4 {
88   margin-top: 1.2em;
89   color: #495057;
90   font-weight: 600;
91 }
92
93 p {
94   margin-bottom: 1em;
95   font-family: 'Inter', sans-serif;
96 }
97
98 ul, ol {
99   margin-bottom: 1em;
100  padding-left: 1.5em;
101 }
102
103 li {
104   margin-bottom: 0.5em;
105 }
106
107 code, .term {
108   font-family: "SFMono-Regular", Consolas, "Liberation Mono", Menlo, Courier, monospace;
109   background-color: #e9ecef;
110   padding: 0.2em 0.4em;
111   border-radius: 4px;
112   font-size: 0.9em;
113 }
114
115 .concept {
116   font-weight: 600;
117   color: #667eea;
118 }
119
120 /* Explanations and Examples */
121 .explanation {
122   background: #f8f9fa;
123   padding: 1.2em;
124   border: 1px solid #dee2e6;
125   border-left: 5px solid #667eea;
126   margin-top: 1.2em;
```

```
127     font-size: 0.95em;
128     border-radius: 5px;
129 }
130
131 .example {
132     border-left: 4px solid #17a2b8;
133     padding: 1em 1em 1em 1.2em;
134     margin: 1.2em 0;
135     font-style: italic;
136     background-color: #f1f8fb;
137     border-radius: 5px;
138 }
139
140 /* Interactive Elements */
141 .interactive-area {
142     margin: 1.5em 0;
143     padding: 1.8em;
144     border: 2px dashed #adb5bd;
145     background-color: #fdfdfd;
146     border-radius: 10px;
147 }
148
149 .checkbox-group {
150     display: flex;
151     flex-direction: column;
152     gap: 0.8em;
153 }
154
155 .checkbox-group label {
156     display: flex;
157     align-items: center;
158     font-weight: 500;
159     padding: 0.6em;
160     background: #f8f9fa;
161     border-radius: 5px;
162     transition: background-color 0.2s;
163     cursor: pointer;
164 }
165
166 .checkbox-group label:hover {
167     background: #e9ecef;
168 }
169
170 .checkbox-group input[type="checkbox"] {
171     margin-right: 0.8em;
172     width: 18px;
173     height: 18px;
174     cursor: pointer;
175 }
176
177 label {
178     margin-right: 0.5em;
179     display: inline-block;
180     min-width: 150px;
181     font-weight: 500;
182 }
183
184 select, input[type=range], button {
185     font-family: 'Inter', sans-serif;
186     font-size: 1em;
187     padding: 0.5em;
188     margin: 0.5em 0;
189     vertical-align: middle;
190     border-radius: 6px;
191     border: 1px solid #ced4da;
```

```
192 }
193
194 input[type=range] {
195   width: 50%;
196   margin: 0 0.5em;
197   cursor: pointer;
198 }
199
200 button {
201   font-family: 'Orbitron', sans-serif;
202   cursor: pointer;
203   background-color: #667eea;
204   color: white;
205   border: none;
206   padding: 0.6em 1.2em;
207   transition: background-color 0.2s, transform 0.1s;
208   font-weight: 500;
209 }
210
211 button:hover:not(:disabled) {
212   background-color: #5568d3;
213   transform: translateY(-1px);
214 }
215
216 button:active:not(:disabled) {
217   transform: translateY(0);
218 }
219
220 button:disabled {
221   cursor: not-allowed;
222   opacity: 0.5;
223   background-color: #6c757d;
224 }
225
226 /* Navigation */
227 #navigation {
228   text-align: center;
229   margin-top: 2.5em;
230   padding: 2em 0;
231 }
232
233 #navigation button {
234   padding: 0.8em 1.8em;
235   margin: 0 1em;
236   background-color: #667eea;
237   font-size: 1em;
238 }
239
240 #navigation button:hover:not(:disabled) {
241   background-color: #5568d3;
242 }
243
244 #navigation button:disabled {
245   background-color: #6c757d;
246 }
247
248 #moduleIndicator {
249   font-family: 'Orbitron', sans-serif;
250   font-weight: 600;
251   color: #667eea;
252   margin: 0 1em;
253   font-size: 1.1em;
254 }
255
256 .back-button {
```

```
257     background-color: #90ee90;
258     color: #2d5016;
259     margin-bottom: 20px;
260   }
261
262   .back-button:hover {
263     background-color: #a0ffa0;
264   }
265
266   /* Visualization */
267   .viz-container {
268     display: flex;
269     align-items: center;
270     justify-content: center;
271     gap: 1.5em;
272     margin: 1.5em 0;
273     flex-wrap: wrap;
274     padding: 20px;
275     background-color: #f8f9fa;
276     border-radius: 10px;
277     border: 1px solid #dee2e6;
278   }
279
280   .shape-viz svg {
281     border: 2px solid #dee2e6;
282     background-color: white;
283     border-radius: 5px;
284   }
285
286   .arrow {
287     font-size: 2em;
288     margin: 0 0.8em;
289     color: #667eea;
290     font-weight: bold;
291   }
292
293   /* Status Indicators */
294   .status-indicator {
295     font-weight: 600;
296     padding: 0.4em 0.8em;
297     border-radius: 5px;
298     display: inline-block;
299     margin-left: 0.5em;
300     font-size: 0.9em;
301   }
302
303   .valid {
304     color: #155724;
305     background-color: #d4edda;
306     border: 1px solid #c3e6cb;
307   }
308
309   .invalid {
310     color: #721c24;
311     background-color: #f8d7da;
312     border: 1px solid #f5c6cb;
313   }
314
315   /* Substitution Styles */
316   .frame-style {
317     color: #495057;
318     background-color: #e9ecef;
319     padding: 0.2em 0.5em;
320     border: 1px solid #ced4da;
321     border-radius: 4px;
```

```
322     font-family: monospace;
323 }
324
325 .substituted-for-style {
326   color: #667eea;
327   font-weight: 600;
328   border-bottom: 2px solid #667eea;
329 }
330
331 .substituting-style {
332   color: #28a745;
333   font-weight: 600;
334   border-bottom: 2px solid #28a745;
335 }
336
337 .expression-box {
338   display: inline-block;
339   padding: 0.4em 0.8em;
340   border: 2px solid #dee2e6;
341   background-color: #fff;
342   border-radius: 6px;
343   margin: 0 0.5em;
344   min-width: 100px;
345   text-align: center;
346   transition: transform 0.3s ease-out, opacity 0.3s ease-out, box-shadow 0.3s ease-out;
347 }
348
349 .sentence-display .substituted-part {
350   display: inline-block;
351   background-color: #e7f5ff;
352   padding: 0 5px;
353   border-radius: 3px;
354   border: 1px dashed #99cff;
355 }
356
357 .sentence-display .substituting-part {
358   display: inline-block;
359   background-color: #e6ffe6;
360   padding: 0 5px;
361   border-radius: 3px;
362   border: 1px dashed #a3e9a4;
363   opacity: 0;
364   transition: opacity 0.3s ease-out;
365 }
366
367 /* Animation Classes - FIXED */
368 .highlight-replace {
369   box-shadow: 0 0 12px 3px rgba(102, 126, 234, 0.6);
370 }
371
372 .lift-out {
373   transform: translateY(-10px) scale(1.05);
374   opacity: 0.7;
375 }
376
377 .lift-out-anim {
378   animation: liftOut 0.4s ease-out;
379 }
380
381 .fade-out {
382   opacity: 0 !important;
383   transform: translateY(-20px) scale(0.9);
384 }
385
386 .fade-out-anim {
```

```
387     animation: fadeOut 0.4s ease-out forwards;
388 }
389
390 .move-in {
391   opacity: 1 !important;
392   transform: translateY(0) scale(1);
393 }
394
395 .move-in-anim {
396   animation: moveIn 0.5s ease-out forwards;
397 }
398
399 .highlight-incoming {
400   animation: highlightPulse 0.6s ease-in-out;
401 }
402
403 /* Keyframe Animations */
404 @keyframes liftOut {
405   0% {
406     transform: translateY(0) scale(1);
407     opacity: 1;
408   }
409   100% {
410     transform: translateY(-10px) scale(1.05);
411     opacity: 0.7;
412   }
413 }
414
415 @keyframes fadeOut {
416   0% {
417     opacity: 1;
418     transform: translateY(0) scale(1);
419   }
420   100% {
421     opacity: 0;
422     transform: translateY(-20px) scale(0.9);
423   }
424 }
425
426 @keyframes moveIn {
427   0% {
428     opacity: 0;
429     transform: translateY(10px) scale(0.95);
430   }
431   100% {
432     opacity: 1;
433     transform: translateY(0) scale(1);
434   }
435 }
436
437 @keyframes highlightPulse {
438   0%, 100% {
439     box-shadow: 0 0 0 rgba(40, 167, 69, 0);
440   }
441   50% {
442     box-shadow: 0 0 15px 5px rgba(40, 167, 69, 0.6);
443   }
444 }
445
446 /* Matrix Table */
447 .matrix-table {
448   border-collapse: collapse;
449   width: 100%;
450   margin: 1.5em 0;
451   font-size: 0.9em;
```

```
452 }
453
454 .matrix-table th, .matrix-table td {
455   border: 1px solid #dee2e6;
456   padding: 12px;
457   text-align: left;
458   vertical-align: top;
459 }
460
461 .matrix-table th {
462   background-color: #e9ecef;
463   color: #495057;
464   font-weight: 600;
465 }
466
467 .matrix-table td ul {
468   padding-left: 1.2em;
469   margin-top: 0.5em;
470 }
471
472 /* Info Icons & Tooltips */
473 .info-icon {
474   display: inline-block;
475   width: 20px;
476   height: 20px;
477   background-color: #667eea;
478   color: white;
479   border-radius: 50%;
480   text-align: center;
481   font-size: 13px;
482   line-height: 20px;
483   cursor: help;
484   margin-left: 5px;
485   font-weight: bold;
486   vertical-align: middle;
487   position: relative;
488 }
489
490 .tooltip-text {
491   visibility: hidden;
492   width: 240px;
493   background-color: #343a40;
494   color: #fff;
495   text-align: left;
496   border-radius: 8px;
497   padding: 10px;
498   position: absolute;
499   z-index: 1;
500   bottom: 130%;
501   left: 50%;
502   margin-left: -120px;
503   opacity: 0;
504   transition: opacity 0.3s;
505   font-size: 0.9em;
506   font-weight: normal;
507   box-shadow: 0 4px 10px rgba(0,0,0,0.3);
508 }
509
510 .info-icon:hover .tooltip-text {
511   visibility: visible;
512   opacity: 1;
513 }
514
515 /* Footer */
516 footer {
```

```
517     text-align: center;
518     padding: 2em;
519     background-color: #f8f9fa;
520     color: #6c757d;
521     margin-top: 3em;
522     border-top: 2px solid #dee2e6;
523 }
524
525 footer p {
526     margin: 0;
527     font-size: 0.9em;
528 }
529
530 /* Fix for SVG label cutoff */
531 .shape-viz svg {
532     overflow: visible !important;
533 }
534
535 .shape-item {
536     display: inline-block;
537     margin: 10px;
538     text-align: center;
539 }
540
```