基于产生式的动物识别专家系统

实验概览

实验目的

1. 理解产生式系统的基本概念。
2. 学习如何使用 JavaScript 实现一个简单的产生式系统。
3. 通过补充代码，巩固对产生式系统和 JavaScript 编程的理解。

实验要求

1. 根据实验代码，理解产生式系统的基本框架及逻辑
2. 按注释补全代码，完善代码逻辑
3. 新增至少[X]条规则和结论，确保逻辑严谨，丰富动物识别种类。
4. 测试2-3组能够输出正确结论的输入条件
5. 创造一组可能存在冲突的输入条件，描述冲突消除的解决思路。
6. 实现step 2中的冲突消解方案 ✓ `[加分项，非必要]`

提交地址

https://icloud.qd.sdu.edu.cn:7777/link/7071B2C9381A52E262085D501E68857C

截止日期

2025年3月16日

实验步骤

1：初始化项目

1. 创建一个新的 HTML 文件，命名为 `expert.html`。
2. 在文件中添加以下基本结构：

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Expert System</title>
    <style>
        .whole {
            width: 100%;
        }

        .whole td {
            width: calc(100% / 4);
            vertical-align: top;
        }

        .function {
            width: 80%;
            height: 100px;
            margin: 20px 0;
        }

        .fields {
            text-align: center;
            /* background-color:bisque; */
            padding: 30px 0;
        }

        .fields button {
            width: 70px;
        }

        .rule {
            font-weight: bold;
        }
    </style>
</head>
```

```
<body>
    <table class="whole">
        <tr id="display">
            <td id="rulesDisplay">
                <h1>Rules</h1>
                <ul id="rulesList">
                </ul>

            </td>
            <td id="factsDisplay">
                <h1>Facts</h1>
                <ul id="factsList">
                    <li>fact 1</li>
                    <li>fact 2</li>
                </ul>
            </td>
            <td id="resultsDisplay">
                <h1>Results</h1>
                <ul id="resultsList">
                    <li>result 1</li>
                    <li>result 2</li>
                </ul>
            </td>
            <td id="function">
                <button id="addRulesButton" class="function">
                    <b>Reset</b> and add other rules</button>
                <button id="addFactsButton" class="function">add facts</button>
                <button id="resetButton" class="function">reset</button>
                <div id="fields" class="fields">
                    <!-- 添加规则和结果 -->
                    <div id="rulesField" style="background-color:cornflowerblue;">
                        <div id="condition">
                            输入前提:
                            <div id="conditionInput">
                                <input type="text">
                            </div>
                            <button id="addConditionButton">AND</button><br>
                        </div>
```

```
                        <div id="result">
                            输入结论:<br>
                            <input type="text" id="conclusion">
                        </div>
                        <button id="addARuleButton">add</button><br>
                        <button id="addRulesDone">done</button>
                    </div>
                    <!-- 添加事实 -->
                    <div id="factsField" style="background-color: antiquewhite;">
                        输入事实:<br>
                        <input type="text" id="factInputField">
                        <button id="addAFactButton">add</button><br>
                        <button id="addFactsDone">done</button>
                    </div>
                </div>
            </td>
        </tr>
    </table>


    <script>

    </script>
</body>

</html>
```

2：核心数据结构定义

将下方代码段插入到<script> </script>代码段中（后续所有js代码都是）

```
// --- 数据定义部分 ---
function Rule(conditions, conclusion) {
    // TODO [1] 请根据产生式规则的需求完善Rule构造函数
    // 提示：考虑规则需要存储哪些信息？如何跟踪未满足的条件？
    // 预期功能：
    // - 存储前提条件集合
    // - 存储结论
    // - 初始化未满足条件的计数器
}
let rulesSet = new Set(); // 规则存储
let factsSet = new Set(); // 事实存储
```

3：DOM元素绑定

```
let rulesList = document.querySelector('#rulesList')
let factsList = document.querySelector('#factsList')
let resultsList = document.querySelector('#resultsList')
let functionField = document.querySelector('#function');

// rules
let addRulesButton = document.querySelector('#addRulesButton');
let addConditionButton = document.querySelector('#addConditionButton');
let conditionInputList = document.querySelector('#conditionInput');
let addRulesDone = document.querySelector('#addRulesDone');
let conclusion = document.querySelector('#conclusion');

// facts
let addFactsButton = document.querySelector('#addFactsButton');
let factInputField = document.querySelector('#factInputField');

// reset
let resetButton = document.querySelector('#resetButton');
let rulesField = document.querySelector('#rulesField');
let factsField = document.querySelector('#factsField');
```

4：规则添加功能

```
// add rules
addRulesButton.addEventListener('click', function () {
    // 初始化规则添加界面
    resetButton.click();
    addRulesButton.disabled = true;
    addFactsButton.disabled = true;
    resetButton.disabled = true;
    fields.append(rulesField);
    while (conditionInputList.hasChildNodes()) {
        conditionInputList.removeChild(conditionInputList.lastChild);
    }
    conditionInputList.append(document.createElement('input'));
    conclusion.value = '';
});
document.querySelector('#addARuleButton').addEventListener('click', function () {

    let newRuleDisplay = `
        <span class="rule">rule:</span><br>
        conditions:
        <ul>
    `
    let newConditions = new Set();
    for (let ruleInput of conditionInputList.childNodes) {
        if (ruleInput.nodeName !== 'INPUT') {
            continue;
        }
        newConditions.add(ruleInput.value);
        newRuleDisplay += `<li>${ruleInput.value}</li>`;
    }
    newRuleDisplay+=`
        </ul>
        result:
        <ul>
            <li>${conclusion.value}</li>
        </ul>
    `;
    let newRuleDisplayNode = document.createElement('li');
    newRuleDisplayNode.innerHTML = newRuleDisplay;
    rulesList.append(newRuleDisplayNode);
```

```
        rulesSet.add(new Rule(newConditions, conclusion.value));
        while (conditionInputList.hasChildNodes()) {
            conditionInputList.removeChild(conditionInputList.lastChild);
        }
        conditionInputList.append(document.createElement('input'));
        conclusion.value = '';
    });
    document.querySelector('#addConditionButton').addEventListener('click', function() {
       // 动态添加条件输入框
       conditionInputList.append(document.createElement('br'));
       conditionInputList.append(document.createElement('input'));
    });
    addRulesDone.addEventListener('click', function () {
        rulesField.remove();
        addRulesButton.disabled = false;
        addFactsButton.disabled = false;
        resetButton.disabled = false;
    });
    rulesField.remove();

    // add facts
    addFactsButton.addEventListener('click', function () {
        addRulesButton.disabled = true;
        addFactsButton.disabled = true;
        resetButton.disabled = true;
        factInputField.value = '';
        fields.append(factsField);
    })

    // --- 事实处理逻辑 ---
    // document.querySelector('#addAFactButton').addEventListener('click', function() {
    //     // TODO [2] 实现推理核心
    //     while (toProcessQ.length !== 0) {
    //         // 前向推理实现区
    //     }
    // });

    document.querySelector('#addFactsDone').addEventListener('click', function () {
        factsField.remove();
```

```
    addRulesButton.disabled = false;
    addFactsButton.disabled = false;
    resetButton.disabled = false;
});
factsField.remove();

function addRule(conditions, result) {

    let aFact = document.createElement('li');
    let something = document.createElement('span');
    something.setAttribute('class', 'rule');
    something.textContent = 'rule:';
    aFact.appendChild(something);
    aFact.appendChild(document.createElement('br'));
    aFact.append('conditions:');
    let conditionsList = document.createElement('ul');
    for (const conditionContent of conditions) {
        let aCondition = document.createElement('li');
        aCondition.textContent = conditionContent;
        conditionsList.appendChild();
    }
    aFact.appendChild(conditionsList);
    afact.append('result:');
    let resultList = document.createElement('ul');
    let resulttt = document.createElement('li');
    resulttt.textContent = result;
    resultList.append(resulttt);
    aFact.append(resultList);
}
```

5：事实处理逻辑

```
document.querySelector('#addAFactButton').addEventListener('click', function() {
    let aFactText = document.querySelector('#factInputField').value;
    if (!factsSet.has(aFactText)) {
        let toProcessQ = [aFactText];
        let newFact = document.createElement('li');
        newFact.textContent = aFactText;
        let newFactNode = document.createElement('li');
        newFactNode.innerHTML = (`
            <li>${aFactText}</li>
        `)
        factsList.appendChild(newFactNode);
        let newResultNode;
        while (toProcessQ.length !== 0) {
            let toProcessFact = toProcessQ.pop();
            newResultNode = document.createElement('li');
            newResultNode.innerHTML = (`
                <li>${toProcessFact}</li>
            `)
            resultsList.append(newResultNode);
            if (!factsSet.has(toProcessFact)) {
                factsSet.add(toProcessFact);

                // TODO [2] 请实现推理核心逻辑-规则收集与展示
                // 预期功能：
                // 1. 遍历所有规则
                // 2. 对每个规则：
                //    a. 如果当前事实是该规则的前提之一
                //    b. 减少该规则的未满足条件计数器
                //    c. 当计数器归零时：
                //       - 将结论加入处理队列
                //       - 从规则集中移除该规则（避免重复触发）
            }
        }
    }
    document.querySelector('#factInputField').value = '';
})
```

6：系统初始化

```
// initiation
// TODO [3] 请补全初始化函数
function init() {

    // 预期功能：
    // 1. 清空规则集和事实集
    // 2. 添加以下预设规则：
    //    a. 条件：有毛发 → 结论：是哺乳动物 (下方示例)
    //    b. 条件：有羽毛 → 结论：是鸟
    //    c. 条件：会飞 AND 下蛋 → 结论：是鸟 (下方示例)
    //    d. 条件：吃肉 → 结论：是肉食动物
    //    e. 条件：犬齿 AND 有爪 AND 眼盯前方 → 结论：是肉食动物
    //    f. 条件：是哺乳动物 AND 有蹄 → 结论：是蹄类动物
    //    g. 添加你的自定义规则


    rulesSet.add(new Rule(new Set().add('有毛发'), '是哺乳动物'));
    rulesSet.add(new Rule(new Set().add('会飞').add('下蛋'), '是鸟'));
    // 在下面补充自定义规则
    // ......


    rulesList.innerHTML = `
            <li>
                <span class="rule">rule:</span><br>
                conditions:
                <ul>
                    <li>有毛发</li>
                </ul>
                result:
                <ul>
                    <li>是哺乳动物</li>
                </ul>
            </li>
            <li>
                <span class="rule">rule:</span><br>
                conditions:
                <ul>
                    <li>会飞</li>
```

```
                    <li>下蛋</li>
                </ul>
                result:
                <ul>
                    <li>是鸟</li>
                </ul>
            </li>
            <!-- 根据上面补充的自定义规则，在下面补充对应的html表格元素 -->
            <!-- ...... -->


    `;
}
```

7：重置功能

```
//reset
resetButton.addEventListener('click', function () {
    // DOM清理逻辑
    // reset rules
    while (rulesList.hasChildNodes()) {
        rulesList.removeChild(rulesList.lastChild);
    }
    // reset facts
    while (factsList.hasChildNodes()) {
        factsList.removeChild(factsList.lastChild);
    }
    // reset results
    while (resultsList.hasChildNodes()) {
        resultsList.removeChild(resultsList.lastChild);
    }
    init(); // 重新初始化
})
```

8：功能整合测试

// --- 启动系统 ---

```
<script>
    // 功能代码
    // ...

    // 在script片段最下方，插入init()函数
    init(); // 初始加载
</script>
```

9：完善代码

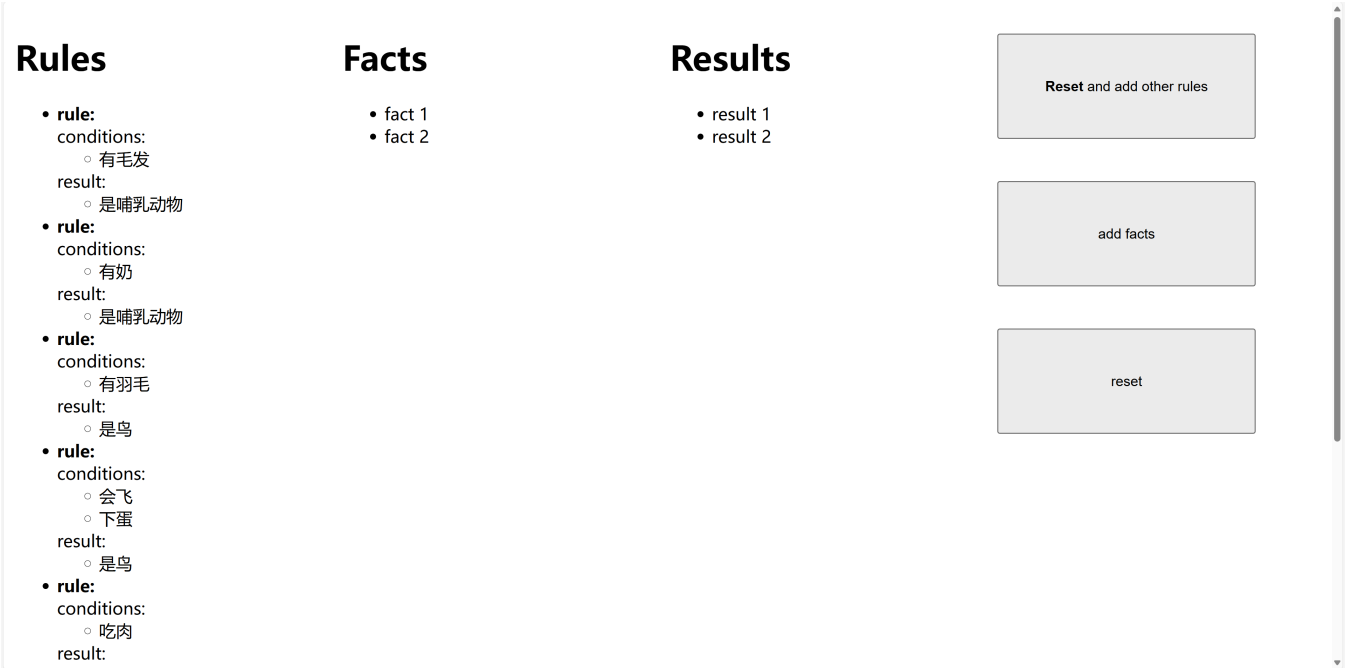
1. 根据代码段内`//TODO`注释的提示，将代码逻辑功能补充完整
2. 增加新的规则和结论，是系统功能更加丰富

10：测试系统

1. 打开 `expert.html` 文件，测试添加规则和事实的功能。
2. 确认系统能够根据输入的条件和事实，得出正确的结论。
3. 创造一组可能存在冲突的输入条件，描述冲突消除的解决思路。

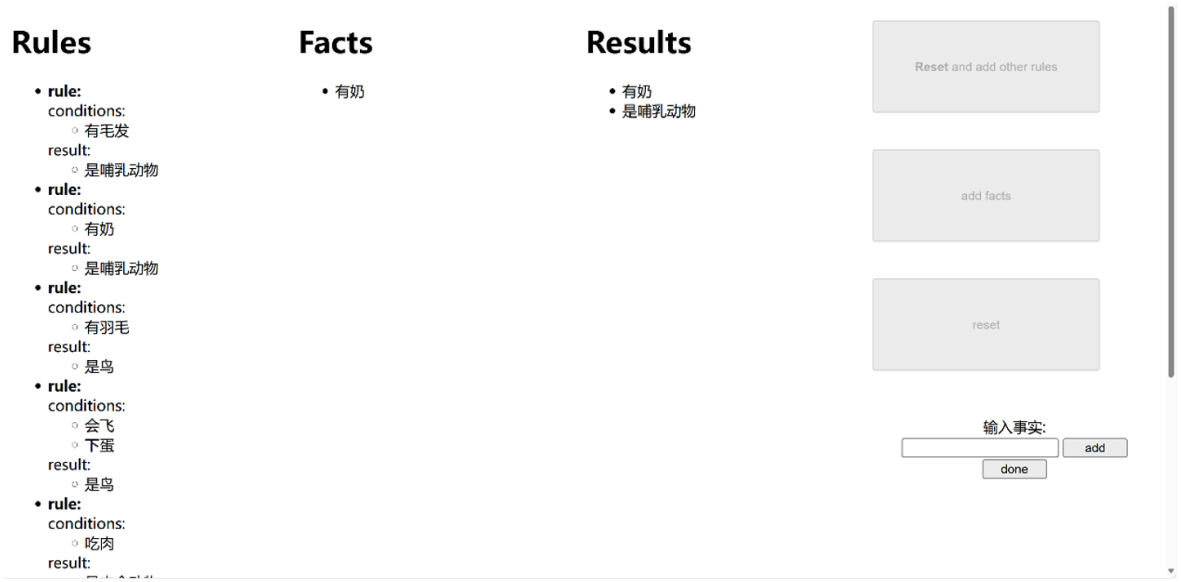
实验预期效果

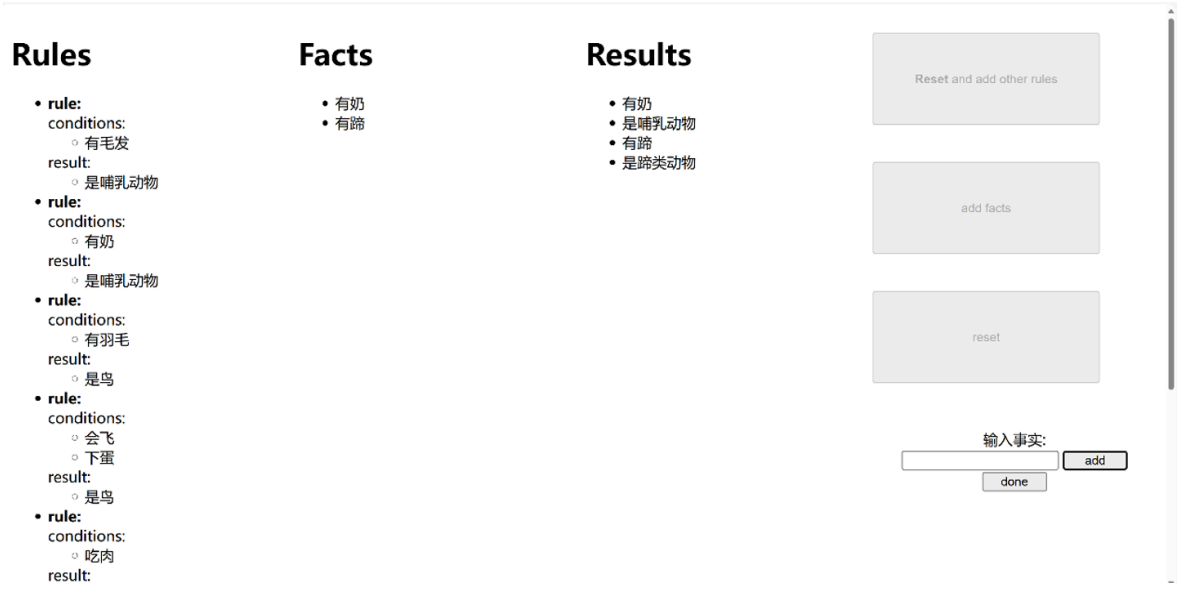
初始界面



- 点击`add facts`，出现输入框，输入`有奶`，点击`add`，**Facts**处更新有奶，**Results**处更新有奶、是哺乳动物

- 再次输入`有蹄`，点击`add`，**Facts**处更新有蹄，**Results**处更新有蹄、是蹄类动物
- 每次点击add后，输入框会清空





11: 冲突消解方案[非必要]

根据提出的可能存在的冲突输入条件，设计相应的代码实现

附录：完整代码

```
<!DOCTYPE html>
<html>

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Expert System</title>
    <style>
        .whole {
            width: 100%;
        }

        .whole td {
            width: calc(100% / 4);
            vertical-align: top;
        }

        .function {
            width: 80%;
            height: 100px;
            margin: 20px 0;
        }

        .fields {
            text-align: center;
            /* background-color:bisque; */
            padding: 30px 0;
        }

        .fields button {
            width: 70px;
        }

        .rule {
            font-weight: bold;
        }
    </style>
</head>
```

```
<body>
    <table class="whole">
        <tr id="display">
            <td id="rulesDisplay">
                <h1>Rules</h1>
                <ul id="rulesList">
                </ul>

            </td>
            <td id="factsDisplay">
                <h1>Facts</h1>
                <ul id="factsList">
                    <li>fact 1</li>
                    <li>fact 2</li>
                </ul>
            </td>
            <td id="resultsDisplay">
                <h1>Results</h1>
                <ul id="resultsList">
                    <li>result 1</li>
                    <li>result 2</li>
                </ul>
            </td>
            <td id="function">
                <button id="addRulesButton" class="function">
                    <b>Reset</b> and add other rules</button>
                <button id="addFactsButton" class="function">add facts</button>
                <button id="resetButton" class="function">reset</button>
                <div id="fields" class="fields">
                    <!-- 添加规则和结果 -->
                    <div id="rulesField" style="background-color:cornflowerblue;">
                        <div id="condition">
                            输入前提:
                            <div id="conditionInput">
                                <input type="text">
                            </div>
                            <button id="addConditionButton">AND</button><br>
                        </div>
```

```
                    <div id="result">
                        输入结论:<br>
                        <input type="text" id="conclusion">
                    </div>
                    <button id="addARuleButton">add</button><br>
                    <button id="addRulesDone">done</button>
                </div>
                <!-- 添加事实 -->
                <div id="factsField" style="background-color: antiquewhite;">
                    输入事实:<br>
                    <input type="text" id="factInputField">
                    <button id="addAFactButton">add</button><br>
                    <button id="addFactsDone">done</button>
                </div>
            </div>
        </td>
    </tr>
</table>


<script>
    function Rule(conditions, conclusion) {
        // TODO [1] 请根据产生式规则的需求完善Rule构造函数
        // 提示：考虑规则需要存储哪些信息？如何跟踪未满足的条件？
        // 预期功能：
        // - 存储前提条件集合
        // - 存储结论
        // - 初始化未满足条件的计数器
    }
    let rulesSet = new Set();
    let factsSet = new Set();


    let rulesList = document.querySelector('#rulesList')
    let factsList = document.querySelector('#factsList')
    let resultsList = document.querySelector('#resultsList')

    let functionField = document.querySelector('#function');
```

```
// rules
let addRulesButton = document.querySelector('#addRulesButton');
let addConditionButton = document.querySelector('#addConditionButton');
let conditionInputList = document.querySelector('#conditionInput');
let addRulesDone = document.querySelector('#addRulesDone');
let conclusion = document.querySelector('#conclusion');

// facts
let addFactsButton = document.querySelector('#addFactsButton');
let factInputField = document.querySelector('#factInputField');

// reset
let resetButton = document.querySelector('#resetButton');

let rulesField = document.querySelector('#rulesField');
let factsField = document.querySelector('#factsField');

// add rules
addRulesButton.addEventListener('click', function () {
    resetButton.click();
    addRulesButton.disabled = true;
    addFactsButton.disabled = true;
    resetButton.disabled = true;
    fields.append(rulesField);
    while (conditionInputList.hasChildNodes()) {
        conditionInputList.removeChild(conditionInputList.lastChild);
    }
    conditionInputList.append(document.createElement('input'));
    conclusion.value = '';
});
document.querySelector('#addARuleButton').addEventListener('click', function () {

    let newRuleDisplay = `
        <span class="rule">rule:</span><br>
        conditions:
        <ul>
    `
    let newConditions = new Set();
```

```
        for (let ruleInput of conditionInputList.childNodes) {
            if (ruleInput.nodeName !== 'INPUT') {
                continue;
            }
            newConditions.add(ruleInput.value);
            newRuleDisplay += `<li>${ruleInput.value}</li>`;
        }
        newRuleDisplay+=`
            </ul>
            result:
            <ul>
                <li>${conclusion.value}</li>
            </ul>
        `;
        let newRuleDisplayNode = document.createElement('li');
        newRuleDisplayNode.innerHTML = newRuleDisplay;
        rulesList.append(newRuleDisplayNode);
        rulesSet.add(new Rule(newConditions, conclusion.value));
        while (conditionInputList.hasChildNodes()) {
            conditionInputList.removeChild(conditionInputList.lastChild);
        }
        conditionInputList.append(document.createElement('input'));
        conclusion.value = '';
    });
    document.querySelector('#addConditionButton').addEventListener('click', function() {
        conditionInputList.append(document.createElement('br'));
        conditionInputList.append(document.createElement('input'));
    });
    addRulesDone.addEventListener('click', function () {
        rulesField.remove();
        addRulesButton.disabled = false;
        addFactsButton.disabled = false;
        resetButton.disabled = false;
    });
    rulesField.remove();

    // add facts
    addFactsButton.addEventListener('click', function () {
        addRulesButton.disabled = true;
```

```
        addFactsButton.disabled = true;
        resetButton.disabled = true;
        factInputField.value = '';
        fields.append(factsField);
    })
    document.querySelector('#addAFactButton').addEventListener('click', function() {
        let aFactText = document.querySelector('#factInputField').value;
        if (!factsSet.has(aFactText)) {
            let toProcessQ = [aFactText];
            let newFact = document.createElement('li');
            newFact.textContent = aFactText;
            let newFactNode = document.createElement('li');
            newFactNode.innerHTML = (`
                <li>${aFactText}</li>
            `)
            factsList.appendChild(newFactNode);
            let newResultNode;
            while (toProcessQ.length !== 0) {
                let toProcessFact = toProcessQ.pop();
                newResultNode = document.createElement('li');
                newResultNode.innerHTML = (`
                    <li>${toProcessFact}</li>
                `)
                resultsList.append(newResultNode);
                if (!factsSet.has(toProcessFact)) {
                    factsSet.add(toProcessFact);

                    // TODO [2] 请实现推理核心逻辑
                    // 预期功能：
                    // 1. 遍历所有规则
                    // 2. 对每个规则：
                    //    a. 如果当前事实是该规则的前提之一
                    //    b. 减少该规则的未满足条件计数器
                    //    c. 当计数器归零时：
                    //       - 将结论加入处理队列
                    //       - 从规则集中移除该规则（避免重复触发）
                }
            }
        }
```

```
        document.querySelector('#factInputField').value = '';
    })
    document.querySelector('#addFactsDone').addEventListener('click', function () {
        factsField.remove();
        addRulesButton.disabled = false;
        addFactsButton.disabled = false;
        resetButton.disabled = false;
    });
    factsField.remove();

    //reset
    resetButton.addEventListener('click', function () {
        // reset rules
        while (rulesList.hasChildNodes()) {
            rulesList.removeChild(rulesList.lastChild);
        }
        // reset facts
        while (factsList.hasChildNodes()) {
            factsList.removeChild(factsList.lastChild);
        }
        // reset results
        while (resultsList.hasChildNodes()) {
            resultsList.removeChild(resultsList.lastChild);
        }
        init();
    })

    // initiation
    // TODO [3] 请补全初始化函数
    function init() {

        // 预期功能：
        // 1. 清空规则集和事实集
        // 2. 添加以下预设规则：
        //    a. 条件：有毛发 → 结论：是哺乳动物 (下方示例)
        //    b. 条件：有羽毛 → 结论：是鸟
        //    c. 条件：会飞 AND 下蛋 → 结论：是鸟 (下方示例)
        //    d. 条件：吃肉 → 结论：是肉食动物
        //    e. 条件：犬齿 AND 有爪 AND 眼盯前方 → 结论：是肉食动物
```

```
//     f. 条件：是哺乳动物 AND 有蹄 → 结论：是蹄类动物
//     g. 添加你的自定义规则


rulesSet.add(new Rule(new Set().add('有毛发'), '是哺乳动物'));
rulesSet.add(new Rule(new Set().add('会飞').add('下蛋'), '是鸟'));
// 在下面补充自定义规则
// ......


rulesList.innerHTML = `
        <li>
            <span class="rule">rule:</span><br>
            conditions:
            <ul>
                <li>有毛发</li>
            </ul>
            result:
            <ul>
                <li>是哺乳动物</li>
            </ul>
        </li>
        <li>
            <span class="rule">rule:</span><br>
            conditions:
            <ul>
                <li>会飞</li>
                <li>下蛋</li>
            </ul>
            result:
            <ul>
                <li>是鸟</li>
            </ul>
        </li>
        <!-- 根据上面补充的自定义规则，在下面补充对应的html表格元素 -->
        <!-- ...... -->


`;
```

```
        }

        function addRule(conditions, result) {

            let aFact = document.createElement('li');
            let something = document.createElement('span');
            something.setAttribute('class', 'rule');
            something.textContent = 'rule:';
            aFact.appendChild(something);
            aFact.appendChild(document.createElement('br'));
            aFact.append('conditions:');
            let conditionsList = document.createElement('ul');
            for (const conditionContent of conditions) {
                let aCondition = document.createElement('li');
                aCondition.textContent = conditionContent;
                conditionsList.appendChild();
            }
            aFact.appendChild(conditionsList);
            afact.append('result:');
            let resultList = document.createElement('ul');
            let resulttt = document.createElement('li');
            resulttt.textContent = result;
            resultList.append(resulttt);
            aFact.append(resultList);
        }

        init();
    </script>
</body>
</html>
```