JSON Search Example

Let's create a more complex example for JSON.Search function in MBS FileMaker Plugin by combining various JMESPath features. Consider the following JSON data representing a list of students and their courses:


```
{
    "students": [
        {
            "name":     "Alice",
            "age":      21,
            "courses":  [
                {
                    "title":    "Math",
                    "grade":    90
                },
                {
                    "title":    "Physics",
                    "grade":    85
                }
            ]
        },
        {
            "name":     "Bob",
            "age":      22,
            "courses":  [
                {
                    "title":    "Math",
                    "grade":    78
                },
                {
                    "title":    "History",
                    "grade":    92
                }
            ]
        },
        {
            "name":     "Charlie",
            "age":      20,
            "courses":  [
                {
                    "title":    "Physics",
                    "grade":    88
                },
                {
```

```
                        "title":        "History",
                        "grade":      85
                    }
                ]
            }
        ]
}
```


Now, let's perform a few JMESPath queries:

1. Get the names of students who are taking the "Math" course:

students[?courses[?title == 'Math']].name

["Alice", "Bob"]

2. Find the average age of students:

avg(students[].age)

21.0

3. List the courses with grades greater than 80 for each student:

students[].{Name: name, Courses: courses[?grade > `80`].title}


```
[
    {
        "Name":     "Alice",
        "Courses":  [
            "Math",
            "Physics"
        ]
    },
    {
        "Name":     "Bob",
        "Courses":  [
            "History"
        ]
    },
    {
        "Name":     "Charlie",
        "Courses":  [
            "Physics",
            "History"
```

```
            ]
        }
]
```


4. Calculate the total grade points for each student:

```
students[].{Name: name, TotalGradePoints: (courses[*].grade | sum(@)) }
```


```
[
        {
                "Name":     "Alice",
                "TotalGradePoints":     175.0
        },
        {
                "Name":     "Bob",
                "TotalGradePoints":     170.0
        },
        {
                "Name":     "Charlie",
                "TotalGradePoints":     173.0
        }
]
```


5. Find students who have a grade higher than 85 in at least one course:

```
students[?courses[?grade > `85`]].name
```


```
["Alice", "Bob"]
```


6. Retrieve the names of students who take both "Math" and "History":

```
students[?courses[?title == 'Math']] | [?courses[?title == 'History']].name
```


```
[
        "Bob"
]
```


These examples demonstrate how you can combine JMESPath expressions to perform more complex queries on nested JSON data. Feel free to modify and experiment with the expressions to suit your specific needs.