Self-Evaluation: MMO Hash Function Implementation

Performance Criteria Self-Assessment

1. MMO Hash Algorithm Implementation

**Rating: Completely Meets Expectations**

**Justification:**

* The implementation correctly follows the MMO hash algorithm specifications:
* Properly implements 128-bit message digest output
* Correctly implements the modified g(s) function as specified
* Accurately handles key generation and encryption
* Uses correct padding and block splitting for arbitrary length inputs

**Code Organization:**

* Functions are well-documented with clear purpose
* Modular design with separate functions for g(s), encryption, and block processing
* Clear variable names (e.g., H0 for initial value)
* Proper use of helper functions from StringConversion.py
* Consistent code formatting and structure

2. Test Cases

**Rating: Completely Meets Expectations**

**Justification:**

Provided two distinct test cases that verify different aspects of the implementation:

1. Test Case 1: Basic functionality test with simple input

* Clear test input provided ("Hello, this is test case 1.")
* Uses fixed H0 value (12345) for reproducibility
* Tests basic hashing functionality

2. Test Case 2: Multi-block processing test

* Tests block splitting and chaining functionality
* Uses fixed H0 value (67890) for reproducibility
* Verifies handling of longer inputs

**Clear Instructions:**

- Step-by-step instructions for running each test case

- Clear expected outcomes and verification steps

- Directory structure and file locations clearly documented

- Troubleshooting guidance provided

- Prerequisites and setup requirements clearly stated

The implementation and documentation fully satisfy the requirements laid out in the project rubric, with both main components (algorithm implementation and test cases) meeting all specified criteria.