# CS 305 Module Five Coding Assignment Checksum Verification Mike Brown

## Instructions

Using the instructions from theModule Five Coding Assignment Checksum Verification Guidelines and Rubric, replace the bracketed text with the relevant information in your own words.

## Algorithm Cipher

For this assignment, I recommend using the **SHA-256** algorithm from the SHA-2 family. It is supported in Java through the MessageDigest class and is widely recognized as a secure, collision-resistant hash function.

## Justification

Collision resistance means it should be computationally infeasible for two different inputs to produce the same hash value. Older algorithms such as MD5 and SHA-1 have been broken, with real-world examples of collisions being demonstrated. Because of this, they are no longer considered secure for checksums. SHA-256, however, remains a trusted choice. It produces a 256-bit (64-character hex) hash, which makes accidental collisions extremely unlikely and deliberate attacks impractical with current technology.

Using SHA-256 aligns with security recommendations for modern cryptography in Java. Sources note that strong primitives like the SHA-2 family should be used, while outdated or weaker ciphers should be avoided (Walker et al., 2009; *Iron-Clad Java*, n.d.).

## Generate Checksum

You’ll submit your refactored code to your instructor. Your instructor will review it and this document.

## Verification

Insert a screenshot below of the web browser with your unique information.

A screenshot of a computer

AI-generated content may be incorrect.

**References**

Walker, J., Kounavis, M., Gueron, S., & Graunke, G. (2009). Recent contributions to cryptographic hash functions. *Intel Technology Journal, 13*(2), 80–95.

*Iron-Clad Java: Protecting Sensitive Data* (Ch. 6). (n.d.). Guidance on secure Java crypto/TLS usage and strong algorithms.