C. A. R. Hoare, "An Axiomatic Basis for Computer Programming" (1969)

Hoare describes a powerful logical framework for reasoning about the correctness of algorithms through rigorous deductive algorithms in this seminal paper. His focus is to rationalize programming with the intent of demonstrating a program works as intended, not debugging afterward. Assessment Axioms would have to further include assignment, composition, repetition, and properties of correctness. Such systematic approaches foundational for reasoning about properties of computer programs are called Hoare logics. This work inspired many other scientific and practical works in software engineering and formal methods.

Richard DeMillo, Richard Lipton, and Alan Perlis, "Social Processes and Proofs of Theorems and Programs" (1977)

A quite remarkable read, DeMillo, Lipton, and Perlis approaches the topic of software verification critically. In this document, they explain the core along with relationships between the mathematics side of software engineering and programming. This is done through their reliance on social processes for establishing confidence and acceptance given for correctness. Rationally explaining software verification, it is informal, social, iterative and not strictly logical and formal in nature. They focus on the impractical nature along with the verboseness of formally verifying software. They strongly argue for shifting the focus from absolute correctness to software reliability. As noted in the, this approach and attitude in formal verifications and arguments termed verification methods led to intense controversy within the region and also debate outside.

Discussion Questions:

>What can contemporary software engineering do to incorporate these formal verification methods despite their practical limitations?

>What approaches can organizations make use of in order to verify and ensure software reliability while dealing with the social and iterative nature of software verification?