This paper investigates the idea of proving certain properties about computer programs. It can logically be divided into two parts: a presentation of the mathematical axioms and rules of inference used in proofs, and a discussion of the importance and scope of such proofs.

The presentation of the mathematical basis begins with a discussion of the limitations of computer arithmetic and how to avoid the problems associated with a finite set of integers. After this, a number of mathematical axioms are listed and the axiom of assignment is introduced. The notation P {Q} R is suggested to mean "If the assertion P is true before initiation of a program Q, the assertion R will be true on its completion." Using this notation, rules of consequence, composition, and iteration are presented and an example proof is cited. This first section is brief and interesting.

The second part of the paper deals with general reservations about the ideas presented and then goes on to talk about the future of these types of proof and the areas of applicability and usefulness. In this section he tends to get carried away with the whole idea, suggesting the idea of proving whole systems correct (from the hardware up), but since it is basicly an introductory paper this is excusable. He leaves open to research a large number of questions that have since been addressed in detail.

Thus even though the paper is out of date, it is interesting to read. In any case, it is one of the early papers in the area of proving correctness of programs and is therefore of historical importance.