**Why we use math in Software Engineering? P.1**

Mathematics teaches a way of wondering and trouble solving. It is uncommon for a software engineer to apply a partial differential equation in a program, however everyone who has discovered a way to solve partial differential equations may have skilled their thoughts to solve a big range of problems. A software engineer must solve problems, and the look at of mathematics can be the best way to prepare the thoughts to create new solutions to new and vintage problems.

What does mathematics play in software engineering? Consider the following statements: Software engineers do not use mathematics, and Software engineers want to think logically and precisely. They represent an obvious contradiction in light of the similarity of the reasoning underlying software engineering and mathematics. Perhaps software engineers who say, I don’t use mathematics, actually mean, I don’t use mathematics explicitly or formally. Many software engineers don’t explicitly use calculus on a daily basis, however do implicitly use mathematical reasoning all of the time. Similarly, software engineers need to learn how to use discrete mathematics concepts and logical reasoning at all times.

“Ask conventional engineers if calculus need to be removed from undergraduate engineering curricula, the answer could be no.” “In contrast, software engineers have argued that mathematics isn't that important in software engineering education since engineers don’t use it explicitly.” Was it continuous or discrete mathematics (or both) that software engineers considered much less important? The answer was unclear. The role of discrete mathematics and logic in software engineering these days is not well understood by either academic researchers or commercial engineers. This lack of understanding will change as the discipline matures and academics and engineers work together to develop that role, making it much like the role of continuous mathematics in conventional engineering disciplines. Key reasons for wanting to learn and use mathematical reasoning include: Abstract software. Constructing abstract artifacts requires abstract reasoning. Which human endeavor was developed to deal with abstraction? Mathematics. Hence one view of a software system is as a mathematically precise model of some desired process or computation. Mathematics is one tool for reasoning about software systems, as well as for software engineers rigorous reasoning and analysis. Symbols, notations, abstractions, precision. The expression y = ax + b is familiar from algebra, and count == 0 is familiar from programming. Each uses notations and symbols and is precise, given the types of data and semantics of the operations, specified mathematically. Learning a formal notation is no more difficult than learning a programming language. Indeed, it is often easier, as the syntax and semantics are cleaner. Programming appeals to our innately process/imperative-oriented minds, and programming tools breathe life into programs. Mathematics tends to be declarative and static, though such tools as Mathematica, and Maple help mitigate this perception.

STD : Abdulrahman Alghamdi Words : 461

ID : 442051025 Quote rate : %9