Description: Typical complexity analysis of computer programs proceeds by informal considerations of quantities like “the number of times loop L is executed”; it also often relies on tricky combinatorics techniques and ad-hoc reasoning. We propose to use and apply techniques from Programming Languages like Hoare logic, formal operational semantics, and proof assistants to the problem of resource analysis. By doing so, we provide a solid ground for sound reasoning on the resource consumption of programs. Moreover, by noticing how classic reasoning patterns like amortized reasoning fit in this new presentation we are able to develop automatic tools to compute automatically high-assurance worst-case resource bounds.