adding.sleep(): Race and Refusal in the Indian Tech Diaspora

When a program is running too fast; when other functions have to catch up; when the program might outpace the system on which it runs, a coder can add a function—called time.sleep()—to make the program wait. Programmers call this process ‘adding sleep’. Adding sleep usefully describes manifold tactics used to slow down the pace of work in neoliberal economies. This paper explores how short-term programmers in corporate offices add sleep to their projects. Based on fieldwork done with short-term programmers from India working in corporate offices in Germany and the United States, this paper follows how such ‘immaterial labor’ as programming relies on racializing Indian programmers as a particular kind of coder. These racializations devalue Indian programmers’ labor even while their presumed cultural knowledge becomes a source of future value for the firm. Into this organization of biocapital and coding, programmers insert slowdowns. They add sleep. I consider how adding sleep can mean both slowing down the speed of work and opening up spaces within programming economies for a politics of eros that imagines a different use of work and time itself.

Sareeta is Associate Professor of Anthropology at the University of Washington, Seattle. Her scholarship investigates personhood and labor within technological capital and throughout the South Asian diaspora. Her first book, Encoding Race, Encoding Class: Indian IT Workers in Berlin, was published by Duke University Press in Fall 2016.