Version	Core approach	Digit access	Carry handling	Unequal length handling	Number size limit	Key notes / known issues	Status
v1a	Recursion for aligned digits + leftover branch	substring + Integer.valueOf in leftover	carry digit tracked	leftover logic	**int-limited**	Bug noted: **999 + 1 → 10100**; recursive calls not returned (addition(...) vs return addition(...))	Mixed
v2a	v1a + explicit “carry through 9s” logic	same as v1a	carry + 9-chain handling	improved leftover 9-chain handling	**int-limited**	Still has recursion-not-returned behavior	Better than v1a for 9s
v3a	long peel + explicit consecutiveNines state	%10 and /10 on long	carryForward + backups	leftover number + 9-chain state	**long-limited** (Long.valueOf)	Strong 9-chain handling; relies on flags	Good within long
v3bV1	simpler long peel (no separate consecutiveNines)	%10 and /10 on long	carryForward + backups	leftover branch handles lastDigit==9 inline	**long-limited** (Long.valueOf)	More edge-case risk than v3a on 9 transitions	Works on your tests (<= long)
v3d	**string-based** recursion (peel last char each step)	substring(len-1) → Long.valueOf of *single digit*	carryForward + backups	leftover handled as string remainingPortion with 9-carry walk	**unbounded by value**, recursion-depth limited	**StackOverflow** on thousands of digits (recursion depth ≈ digits); one spot uses Integer.valueOf(...)==9 (inconsistent)	Correct on moderate lengths
v4a	recursion refactor using long math	long math	carry/backups	remainder mode via remainingPortion/number	**long-limited**	“return doesn’t stop” symptom remains due to non-returned recursive calls	Good for <= long
v4c	v4a cleanup / clearer remainder variables	long math	same	same	**long-limited**	Tidier but still long-bounded	Best of long set