

## Problem Set 1 Grading Codes

Handout written by Ian Goodfellow.

We used numeric codes to mark common mistakes while grading problem set 1. This saves us a lot of writing but also makes sure that the grades we assign are consistent, since each code is associated with a specific number of points to take off. To find out which comments are associated with any mistakes that we marked on your problem set, just look up the code in the table provided below.

3. (b)

Code	Points	Comment
3b1	-2.5	Fixes left recursion, but changes the language of the grammar
3b2	-3	Resulting grammar is still ambiguous
3b3	-0.5	Resulting grammar only generates infinite length strings
3b4	-3	Resulting grammar is not left factored

(c)

The parse table was graded based on your grammar from part b—i.e., even if your grammar was flawed it was usually possible to get full credit for the parse table if it correctly implemented your grammar (and identified any conflicts present in your grammar). We only took points off if the grammar was incorrectly altered in some way that made the parse table unreasonably easier to implement (e.g., if you never had to calculate any follow sets).

Code	Points	Comment
3c1	-1	Subtracted one point for each mistaken cell (except in case of entire column missing, see 3c2)
3c2	-3	Subtracted only 3 points for missing \$ column, instead of subtracting one point per missing cell. We did not subtract points if there was some other valid explanation of how to accept the string at the end of input.
3c3	-2	Parser can never terminate, for reason other than leaving out \$ column
3c4	-1	Wrote multiple productions in one cell (unless the grammar from part b actually does cause a conflict, in which case we did not take points off from part c)
3c5	-1	Grammar from part b should cause conflicts but only one production is marked in the table at those cells

(d) We also graded part d based on your previous answers. If your modified grammar and parse table implied that you should reject the string, we gave you full credit if you rejected the string, even though it is in the language.

---

Code	Points	Comment
3d1	-1	Subtracted one point for each inexplicable manipulation of stack or input
3d2	-0.5	Use of rule that is not in your parse table (yes, even if this is the correct rule to use at this point)
3d3	-0.5	Minor line-copying mistake that doesn't result in obvious parser misbehavior