Exercise 4.47.

Louis Reasoner suggests that, since a verb phrase is either a verb or a verb phrase followed by a prepositional phrase, it would be much more straightforward to define the procedure `parse-verb-phrase` as follows (and similarly for noun phrases):

```
(define (parse-verb-phrase)
  (amb (parse-word verbs)
       (list 'verb-phrase
             (parse-verb-phrase)
             (parse-prepositional-phrase))))
```

Does this work? Does the program's behavior change if we interchange the order of expressions in the `amb`?

Answer.

We can verify Louis's proposal by typing a simple sentence "The student with the cat sleeps in the class." the to the `amb` evaluator driver loop:

```
;;; Amb-Eval input:
(parse '(the student with the cat sleeps in the class))

;;; Starting a new problem
;Aborting!: out of memory
;GC #1446: took:   0.10   (4%) CPU time,   0.40  (13%) real time; free: 16769214
;GC #1447: took:   0.10  (50%) CPU time,   0.40  (98%) real time; free: 16769245
```

Obviously, the modification Louis proposed doesn't work at all, even if we interchange the order of expressions in the `amb`. The reason is lies in that `parse-verb-phrase` calls on itself directly in `amb`, which therefore arises infinite recursion. The original `parse-verb-phrase` procedure in the text guarantees that arguments passed to its internal procedure `maybe-extend` have been reduced compared to the preceding call, thereby avoiding infinite recursion.