Homework 2 Answers

Glen Madsen

February 22, 2018

Problem 2.

1. Where did you include calls to checkRep (at the beginning of methods, the end of methods, the beginning of constructors, the end of constructors, some combination)? Why?

I included checkRep at the beginning of every function that I wrote with the noted exceptions of the constructor, which it is only included at the end, and scaleCoeff (where it is nowhere). Also, nearly every function had it at the end (including constructors, excluding ScaleCoeff) with the exception of degree. The list of functions is as follows: RatPoly(int c, int e), degree, getCoeff, isNaN, scaleCoeff, negate, add, sub, mul, eval, differentiate, antiDifferentiate, integrate (it is in others but not ones that I wrote). The reason why it is placed so often is because it is a very cheap check to ensure that the data representation remains in the specified form. Because it is cheap I placed it before any method tried to manipulate an array, and afterwards, even if the function was not intended to change anything, just in case something was indeed altered and the items passed are valid. However, since scaleCoeff is static, it is not needed in that method, and the constructors cannot have it at the beginning because the data representation does not exist yet, so it is absent in those locations.

2. Imagine that the representation invariant of RatPoly was weakened so that we did not require the last element of coeffs to be non-0. This means that the method implementations could no longer assume this invariant held on entry to the method, but they also no longer were required to enforce the invariant on exit. Which method(s) or constructor implementation(s) will change? Please list them. For each changed piece of code, describe the changes informally, and indicate how much more or less complex (in terms of code clarity and/or execution efficiency) the result would be. Note that the new implementations must still adhere to the given spec.

The constructors would have to change because, for example, the RatPoly(int c, int e) constructor is not supposed to construct a polynomial with a c of 0 of any size other than 0, but since trailing zeroes are allowed, this constructor would become simpler to implement. Depending on the return of degree (it is vague, would have to be changed 1 way or another) it might need to be changed too since the highest exponent could have a zero coefficient. Luckily most of the functions (at least my implementations) will be unaffected by

the change (an add operation with trailing zeroes with 0+0 is still 0 for example) outside of the cases listed earlier. It would make the code more difficult to read in some cases because the data implementation is weirder, and make some methods take longer/be redundant, but ultimately not affect much (checkrep would have to be altered to account for the trailing zeroes). That is pretty much all.