MATH299M/CMSC389W

Spring 2019 – Ajeet Gary, Devan Tamot, Vlad Dobrin

Model H11: Create Bases Package

Assigned: Friday April 12th

Due: Sunday May 12th, 11:59PM

Note: Remember that out of the H models from Part 3, that is, anything numbered 10 or above, you must only complete 2 assignments. Any extra you complete can replace low scores from Part 1 and Part 2 of the course.

This week we learned how to make packages and other developer-type things in Mathematica. For this assignment I want you to practice by making a package for something that could actually be pretty useful!

Writing numbers in different bases is common in computer science, specifically you jump a lot between decimal (base 10), binary (base 2) and hexadecimal (base 16). Just in case you haven't seen this in a while, here's an example (the subscript on the number denotes the base it's being written in; if a subscript is not present then it is assumed to be base 10 – this notation isn't used all of the time though, often it's clear from context what base we're in):

$$123_{10} = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0$$
$$= 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 1111011_2$$
$$= 7 \times 16^1 + 11 \times 16^0 = 7B_{16}$$

When you're working in, say, binary, and you try to add two numbers in Mathematica, it's going to add them in base 10, which screws everything up. Instead of converting back and forth between decimal and whatever base you're working in, wouldn't it be nice to have a package with operations designed for that base? That is, operations for addition, multiplication, subtraction and division that take in numbers in binary and return numbers in binary. What I mean by this is that they'll still *look* like numbers in decimal to Mathematica, like Mathematica won't know that 1111011 actually means 123, but the user will know, and 1111011 "+" 110 will return 10000001 rather than 1111121.

Extensions:

Arbitrary Base

Can you make the package accommodate an arbitrary base, that is, any base from 1 up through 36 (using the full alphabet) or 62 (upper and lower case letters) or even something as general as 127 (ASCII)?

Notation

Perhaps building in the subscript notation is a good way to handle computation in different bases, is there a nice way to do this?

Complicated Arithmetic

Can we do more complicated functions, like exponentiation, modular arithmetic (sounds like a headache, but super interesting), or trig functions?

Fractions
What do negative numbers and fractions look like in different bases? What does π_2 look like?

Complex
What do complex numbers in different bases look like?

I'm not sure what the best way to implement this is, or how far we can go with making a nice system without overriding functions that Mathematica already has defined (I wouldn't suggest using Unprotect on anything, as any Mathematica function dependent on a function that you redefine will be messed up) – see what you can make! I'm excited to see if you all can create a bases package that's useful and clean enough to use for real projects B)