Math 430 – SageMath Extra Credit Project

Due December 8, 2017

If you choose to work on this project, you should do the following:

1. Create a free account at `www.cocalc.com` (you can spend $7 per month for more computational resources, but the free account should be sufficient for this project; you can also download Sage to your computer from `www.sagemath.org`).

2. Create a project with name "Math 430: YOUR NAME"

3. Within the project, click on the Project Settings tab and add me as a collaborator (`roed.math@gmail.com`)

4. For each of the sections below, click on the New tab, enter `PartN` in the box labeled `Name your file, folder or paste in a link` (where N is the number of the section), then click on the `Sage Worksheet` button. This will create a file for you to work on that section. Also create a `Scratch.sagews` file.

5. Download the version of the textbook that includes Sage sections and problems from the following url (the Sage material is also included in the online version).

   `http://abstract.ups.edu/download/aata-20170805-sage-8.0.pdf`

6. Read through the Sage sections in the chapters we have covered (1.5, 2.6, 3.7, 4.7, 5.4, 6.5, 9.4, 10.4, 11.5, 13.6, 16.9, 17.6, 18.5) and experiment with what you're learning in your scratch file.

7. If you don't know Python (the programming language used by Sage), you can learn some through the Non-programmer's tutorial for Python or Dive into Python. Learning to write functions and use `for` loops and list comprehension will be particularly useful. In addition to our textbook, you can find more help on using Sage on the Sage website. For help on CoCalc, try `https://github.com/sagemathinc/cocalc/wiki`.

8. Do the exercises described below. Since you've added me as a collaborator, I will be able to see your work. Make sure that the files are named appropriately so I know where to look.

1 Part 1: Number Theory

1. Prime GCD. This is a combination of problems 1 and 4 from section 2.7

   Write a function that takes as input a positive integer k, and returns a 4-tuple (p, q, a, b), where p and q are k-digit primes (Hint: use Sage's `next_prime` function) and a and b are integers so that $ap + bq = 1$.

2. Prime sieve. This is problem 1 from section 2.4.

   Write a function that takes as input an integer n, and returns a list of all primes less than n. Do not use Sage's built in functions related to primes, such as `prime_range`, `is_prime`, `next_prime` or `factor` in your final answer.

3. Fermat's little theorem. This is a modification of problem 2 of Section 6.6.

   Write a function that takes as input an integer n, and returns the list all nonzero integers a so that $a^{n-1} \not\equiv 1 \pmod{n}$ and $0 < a < n$.

   As a test, make sure that your function returns `[]` when n is prime.

4. Carmichael numbers.

   A *Carmichael number* is a composite integer n so that, for every a that is relatively prime to n, $a^{n-1} \equiv 1 \pmod{n}$.

   Write a function that takes as input an integer N, and returns the list of all Carmichael numbers less than N.

   As a test, the first Carmichael number is 561. What are the other Carmichael numbers less than 3000?

2 Part 2: Groups

1. Subgroups of dihedral groups. This is problem 5 from section 3.8.

   Write a function that takes as input a positive integer n and returns two subgroups A and B as described below. Create a cyclic group C and dihedral group D, both of order $4n$ using `CyclicPermutationGroup(4*n)` and `DihedralGroup(2*n)`. Then A should be a subgroup of C and B a subgroup of D, both of order $2n$, but not isomorphic to each other.

   Hint: You can use `C.subgroups()` and `D.subgroups()` to get a list of all subgroups.

   The result of your function for $n = 3$ should look like `(Subgroup of (Cyclic group of order 12 as a permutation group) generated by [(1,3,5,7,9,11)(2,4,6,8,10,12)], Subgroup of (Dihedral group of order 12 as a permutation group) generated by [(2,6)(3,5), (1,3,5)(2,4,6)])`

2. The converse of Lagrange's theorem is false. This is a modification of problem 1 of Section 6.6.

   Write a function that takes as input a positive integer n, and returns the list of groups G of order n with the following property: for some m dividing n there is no subgroup of G of order m. You may find the function `gap.SmallGroup` useful (see http://www.gap-system.org/Manuals/doc/ref/chap50.html#X814D329A7B59F0EB for more details on this Gap function).

   As a test, make sure your function returns `[]` for $n < 12$, and includes A_4 when $n = 12$.

3. Counting subgroups of D_n. Do problem 6 from Section 10.5.

4. Isomorphism between permutation groups. Do problem 3 from Section 11.6.

5. Normal subgroups of D_{20}. Do problem 5 from Section 11.6.

3 Part 3: Rings

1. The finite field $\mathbb{F}_{81}$. Do problem 3 from Section 16.10.

2. Factoring $x^3 - 3x + 4$ over different rings. Do problem 1 from Section 17.7.

3. Ramification of polynomials.

   Write a function that takes as input a polynomial f with integral coefficients and an integer N, and returns the list of primes p less than N with the following property:

   > If you factor f modulo p, at least one of the terms appears with degree larger than 1.

   For example, $x^2 + 1$ is ramified at 2 since $x^2 + 1 \equiv (x + 1)^2 \pmod 2$.

   Don't use the method `f.discriminant()` when writing your function, but after you're done, find a relationship between the primes you return and the result of `f.discriminant()`.

4. Splitting of polynomials.

   Write a function that takes as input an irreducible quadratic polynomial f with integer coefficients and an integer N, and returns the list of primes p less than N so that f factors into two linear polynomials modulo p.

   What patterns do you notice in these lists for different f? Try $f = x^2 + 1$ first.