MATHEMATICS 152, SPRING 2020 DISCRETE MATHEMATICS
Computational Homework Problems

Last modified: January 24, 2020

There are 63 points worth of problems here; so you can meet the entire exploartory problem requirement.

Problems 1-3 each count as two exploratory problems (4 points each).

You may do them on codepad.org in C, Lua, Python, or PHP and email the Web link to Paul. Alternatively, you can bring your laptop to Paul's office hours and demonstrate the program in operation.

No interactive user input is required, but it must be made obvious how to change the input data.
Lines like `a = "(145)(26)"`, along with appropriate comments, will suffice.

If there is sufficient interest, there will be a few extra classes on how to write these in R Shiny, with a 21st century user interface.

1. Write a program that takes two permutations a and b of 1...9, represented in cycle notation as character strings, and computes and displays ab, ba, aba^{-1} and bab^{-1}. You may represent the identity permutation eithe as the null string or as the string "I".

2. Write a program that takes an integer $q < 50$, determines the set S of integers $< q$ that are coprime to q, and displays the group multiplication table for S along with a table showing the orders of elements that occur, and the number of elements of that order, e.g."There are 6 elements of order 4." To get full credit you must format the multiplication table nicely, not just output one multiplication fact per line.

3. A well-known prime number (a so-called "Fermat prime") is
   $p = 2^{16} + 1 = 65537$.
   For this value of p, write a program that takes an integer a between 2 and $p - 1$, uses the Euclidean algorithm to find m and n such that $mp + na = 1$, expresses the multiplicative inverse of $[a]_p$ as $[b]_p$ with $1 < b < p$, and checks the answer.

4. Finite Fields (up to 15 points)

   Write a program that does arithmetic in finite fields.The user interface can be quite primitive. The user specifies the order of the field from your list of available choices), types in two field elements as strings or as coefficients, and the program then outputs the sum, product, and quotient of the elements (or perhaps a "divide by zero" error message).

   It is OK to build logarithm tables into your program, which makes multiplication and division much easier. However, hard-coding a function that multiplies polynomials of degree 3 or less that are represented by an array of coefficients is not hard.

   Here are the point values:

   Orders 4 and 5: 4 points
   Orders 7, 8, 9: 3 point for any two, 4 points for all 3
   If you get carried away:
   Field of order 25: 1 point
   Field of order 27: 1 point
   Field of order 16: 1 point (much easier if you use a log table)
   Field of order 32: 1 point (much easier if you use a log table, but it's a pain to do one by hand)
   Field of order 49: 1 point (easy if you have a multiply-and-reduce function for degree-1 polynomials)
   2 point bonus if you define a finite-field base class and then define classes for specific orders by inheritance. This is great practice in object-oriented programming. I have done it in C++ and PHP but have not tried in Python or Java.

Here are some good choices for irreducible polynomials. According to the book in which I found these polynomials (Lidl and Pilz, Applied Abstract Algebra), for these choices x is a "primitive element:" that is, every nonzero element of the field is a power of x.

- If prime = 2

  power = 2: $x^2 + x + 1 = 0$ so replace x^2 by $-x - 1$ (which is the same as x + 1)

  power = 3: $x^3 + x + 1 = 0$ so replace x^3 by $-x - 1$ (which is the same as x + 1)

  power = 4: $x^4 + x + 1 = 0$ so replace x^4 by $-x - 1$(which is the same as x + 1)

  power = 5: $x^5 + x^2 + 1 = 0$ so replace x^5 by $-x^2 - 1$ (which is the same as $x^2 + 1$)

- If prime = 3

  power = 2: $x^2 + 2x + 2 = 0$ so replace x^2 by $-2x - 2$ (which is the same as $x + 1$)

  power = 3: $x^3 + 2x + 1 = 0$ so replace $x^3by - 2x - 1$(which is the same as $x + 2$)

- If prime = 5

  power = 2: $x^2 + 3x + 3 = 0$ so replace $x^2by - 3x - 3$ (which is the same as $2x + 2$)

- If prime = 7

  power = 2: $x^2 + x + 3 = 0$ so replace $x^2by - x - 3$ (which is the same as $6x + 4$)

You can either put the project on codepad.org (which makes it almost impossible to use libraries) or leave it on your own laptop and do a demo at Paul's office hours.

Since this is a math class, you can get full credit even if you use single-letter variable names, have nothing but global variables, and include no comments! Functionality is all that matters. However, you wiil find it easier to debug, maintain, and extend the program if you follow good software practices.

5. Euler Walks (6 points)

   - (4 points) Write a program that constructs an Euler cycle for a graph where every vertex has even degree and show that it works correctly for the graph shown in outline 9, topic 8.
   - (2 points) Write a program that constructs an Euler walk for a graph where two vertices have odd degree and show that it works correctly for the graph shown in outline 9, topic 7b. You may reuse code from the preceding problem.

6. Group Isomorphisms (18 points) You can build some or all of these features into a single program.

   - Multiply two matrices in $SL(2, \mathbb{F}_4)$ – 3 points
   - Multiply two permutations in A_5 – 3 points
   - Construct the permutation in A_5 that is isomorphic to a given matrix in $SL(2, \mathbb{F}_4)$ – 3 points
   - Demonstrate the isomorphism between $SL(2, \mathbb{F}_4)$ and A_5 – 3 points
   - Construct the matrix in $SL(2, \mathbb{F}_4)$ that is isomorphic to a given permutation in A_5 – 3 points
   - Modify the program to use $SL(2, \mathbb{Z}_5)$ in place of $SL(2, \mathbb{F}_4)$ – 3 points

7. Generators and relations (12 points)

   - (4 points) Write a program that represents the six elements of D_3 as strings of zero, one or two r's followed by zero or one f's and does group multiplication by concatenating strings and using defining relations to rewrite the result. A language that supports regular expressions will make this easy to do.
   - (4 points) Show that each element of A_4 can be represented either as r^α or as $r^\alpha f r^\beta$, where α and β may be 0, 1, or 2.

     Then write a program that represents the twelve elements of A_4 as specified in the preceding problem and does group multiplication by concatenating strings and using defining relations to rewrite the result. A language that supports regular expressions will make this easy to do.
   - (4 points) Write a program that represents the 24 elements of S_4 in a manner similar to the preceding two problems, but also using $r^\alpha f r^2 f$, and does group multiplication by concatenating strings and using defining relations to rewrite the result. A language that supports regular expressions will make this easy to do.