Information (PEAK)

November 19, 2024

Midterm project assignments

Due: Noon, December 3rd

Uchida Kaoru

Project / assignment outputs

1. Write a report and submit to UTOL
   - Main part in PDF; some screen shots should be helpful
   - For each problem, try to write a self-contained report consisting of:
     1. Problem setting with your interpretation
     2. Base idea of your approach and algorithm
     3. Implementation and test results
   - Optionally (for extra points): observations, further study, and your subjective comments
2. Additional points for a short presentation (as usual)
   - Brief summary of your idea and its originality
   - Effective and appealing demonstration

Midterm project – Due: Noon, December 3rd

Ethics – **No plagiarism**

- ✓ Plagiarism: “An act of fraud which involves both stealing someone else’s work and lying about it afterwards” (- www.plagiarism.org)
- ✓ Plagiarism is the use of another’s original works, words or ideas as though they were your own.
- ✓ If you turn in the work of someone else (including any generative AI) as your own, without understanding what is going on, you have committed plagiarism
- ✓ If you want to use generative AI as your studying assistant,
  do NOT just “copy and paste,” but be sure to understand the logic, algorithm, and the implementation of the proposed solution, so that you can explain the details and modify the code yourself.

Midterm project -- Problem 1

Multi-perfect number

A perfect number is what is equal to the sum of its divisors excluding the number itself. They are also referred to as P_2 numbers because the sum of the divisors, including the number itself, is twice the number.

Multiply perfect numbers are numbers such that the sum of their divisors are some multiple of the number; P_2 numbers have divisor-sums twice the number, triply perfect numbers P_3 have divisor-sums thrice the number, and so on.

1. Write a function divisorsum(n) that returns the sum of n's divisors.
2. Write a program that finds all P_2 perfect numbers below 10,000.
3. Write a program that finds all multiply perfect numbers below 50,000.
4. Divisors often come in pairs (e.g. 3 and 12 for 36); based on this, implement a faster algorithm to find the next P_x number (6 digits).

Midterm project -- Problem 2

Goldbach Conjecture

1. Write a function isprime(n) that returns whether n is a prime number (True or False).
2. Find a list of prime numbers up to *maxn* = 100.
3. Verify the Goldbach Conjecture for numbers up to *maxn* = 100 (i.e., that any even number greater than 2 can be represented as the sum of two primes).
4. Create a tool that, for any input number n up to *maxn* = 10000, finds all possible pairs of primes that sum to n.

Midterm project -- Problem 3

Find the loner

Implement a function findloner(List_of_int) to find the single number (loner) in a list of integers where every other number appears in pairs.

For example, if the input is [5, 67, 34, 67, 2, 5, 34], it should return 2.

Better to use bitwise XOR to accomplish this.

Midterm project -- Problem 4

Spiral drawing

Draw a colorful spiral.

Hint: polygon( ) is a good starting point.

Midterm project -- Problem 5

Happy list

A happy list HL is computed by the following process:

i. Starting with any positive integer n_0 and HL = [n_0],

ii. calculate the next number as the sum of the squares of the digits of the last number, and append the result to HL, and repeat this process.

iii. If the number equals 1, stop, with HL = [n_0, …, 1]. Then we call n_0 happy.

iv. If the number is the same as any number in HL, stop. Then we call n_0 unhappy.

1. Write a function to make a happy list for a given n_0, and check that 7 is happy, and 17 is unhappy.
2. How many happy numbers are there under 100, 1000, .., and 10**6 respectively, and what is the “happiness rate” for each case?
3. Plot the happy list among all (no matter happy or not) for under 1000.

Midterm project -- Problem 6

Circles around polygons

1. Draw six blue equilateral triangles with random positions, sizes, and orientations.
2. Then, for each triangle, draw a red circle centered at its centroid with a radius equal to the length of its sides. (Because our circle() is merely an approximation, small amount of error is acceptable)
3. Try the same with squares and regular hexagons. Give your observations, if any.