E-402-STFO Problems for Module N

Updated November 23, 2024

This module has 8 parts, each giving a variable amount of points. 100 points give you full marks. In each time complexity we assume a basic operation on integers like addition or multiplication is constant time, even though it may grow with arbitrary size integers. To get full marks the code needs to be correct, perform in the desired time complexity.

The parts walk you through making an algorithm that finds all ways to write $n = a^2 + b^2$ for a given n.

Due to static validation functionality not quite yet being part of problemtools (the system I am using to autograde), you will receive full marks from the automated system even if you solve the problems not in the manner asked of you. I will manually check accepted submissions once the submission window closes, and will then reject any submission that fails to do what is asked. This applies to things like importing gcd in the first problem.

Problems marked with a `b` at the end are bonus problems taking the total points above 100. They need to be submitted on canvas and are graded manually.

`mAp1`, 10 points Implement code that reads two inputs $a, b \leq 10^{18}$ and outputs their greatest common divisor. Do not use a built-in function to do this. It should have time complexity $\mathcal{O}(\log(a + b))$, which is achievable using Euclid's algorithm.

`mAp1b`, 10 points Prove/explain why Euclid's algorithm has this time complexity.

`mAp2`, 10 points Implement a modular power function, taking as input the base b, exponent e and modulus m, outputting $b^e \pmod{m}$. Do not use a built-in function to do this. It should have time complexity $\mathcal{O}(\log(e))$, so binary exponentiation is needed. The inputs will satisfy $b, e, m \leq 10^{18}$, $b, e \geq 0$ and $m > 1$. We will consider 0^0 to be 1.

`mAp3`, 10 points Implement a function that given a prime p such that $p = 1 \pmod{4}$ it finds an r such that $r^2 = -1 \pmod{p}$. This is done by first finding a value c such that $c^{(p-1)/2} = -1 \pmod{p}$ and letting $r = c^{(p-1)/4}$. It can be shown that exactly half the non-zero values mod p satisfy this property, so c can be chosen randomly until it satisfies this property $\mathcal{O}(1)$ expected picks. The code written in `mAp2` may come in handy here. There can be several correct answers, any one of them will be accepted. The input will satisfy $p \leq 10^{18}$.

`mAp4`, 10 points Given a, b, k as input use Euclid's algorithm but stop at $1 \leq a, b \leq \sqrt{k}$. This code will be similar to `mAp1` but you have to quit as soon as $a, b \leq \sqrt{k}$. The code should return the values a, b. The inputs will satisfy $a, b, k \leq 10^{18}$.

`mAp5`, 20 points Implement a function which when given a prime p such that $p = 1 \pmod 4$ it finds $a, b > 0$ such that $a^2 + b^2 = p$. Start by getting the value r from the code in `mAp3`. Then get x, y from the code in `mAp4` for the inputs p, r, p. Now $x^2 + y^2 = p$. p will be at most 10^{18}.

`mAp5b`, 20 points Prove/explain why $x^2 + y^2 = p$ in the code above.

`mAp6`, 10 points Implement code which when given a, b, c, d such that $a^2 + b^2 = p$ and $c^2 + d^2 = q$ outputs two solutions to writing pq as a sum of two squares. This is best done using the formulas $(a^2 + b^2)(c^2 + d^2) = (ac - bd)^2 + (ad + bc)^2 = (ac + bd)^2 + (ad - bc)^2$. This should be printed as `x^2 + y^2 = pq` where x, y and pq are replaced by the corresponding numbers, one solution to the formula per line. If x or y are negative, put parentheses around them. The input will satisfy $a, b, c, d \leq 10^9$.

`mAp7`, 30 points Finally, use all the code above to find all ways to write n as a sum of squares. Dividing out a single prime factor of n and recursing makes the implementation simpler and is thus recommended. If this factor is 2, use $2 = 1^2 + 1^2$. If this factor satisfies $p = 1 \pmod 4$, use the earlier code. Otherwise check if n is divisible by p^2, if not return an empty list. Otherwise we can write $p^2 = p^2 + 0^2$ by taking out p^2 as a factor instead of p. If $n = 1$ there are four ways $(0, 1), (0, -1), (1, 0), (-1, 0)$. This method may produce duplicates, so filter duplicates before printing. First print the number of solutions on a single line, then output each solution in the same format as in `mAp6` on its own line. The input will satisfy $n \leq 10^{18}$.

`mAp8`, 10 points Use the code in `mAp7` to find the number of solutions to the equation $a^2 + b^2 = c^2 + c$ for $0 \leq a \leq b \leq c \leq n$. This will be tested up to $n = 10^5$.

Example:

Let's consider $N = 52706$. First we factor as $52706 = 2 \cdot 19^2 \cdot 73$. Then we have to write each of these as a sum of squares. So we have $2 = 1^2 + 1^2$ and $19^2 = 19^2 + 0^2$. Now for 73 we can find $27^2 = -1 \pmod{73}$. Then we do the

Euclidean algorithm:

$$\begin{aligned}73 &= 2\cdot 27+19\\ 27 &= 1\cdot 19+8\\ 19 &= 2\cdot 8+3\\ 8 &= 2\cdot 3+2\\ 3 &= 1\cdot 2+1\\ 2 &= 2\cdot 1+0\end{aligned}$$

The first time $a, b \leq \sqrt{73}$ is at $8 = 2\cdot 3+2$, so our result is $8, 3$. And indeed $73 = 8^2+3^2$. Now we start with the solutions for $N = 1$ which are

$$(1,0),(-1,0),(0,1),(0,-1)$$

If we use the given formula to combine if with our pair for 2 we get

$$(-1,1),(1,1),(1,-1),(-1,-1)$$

Next we combine with the pair $(0,19)$ to get

$$(-19,19),(-19,-19),(19,-19),(19,19)$$

Finally we combine with $(3,8)$ to get our final list of

$$\begin{gathered}(209,95),(209,-95),(-209,95),(-209,-95),\\ (95,209),(95,-209),(-95,209),(-95,-209)\end{gathered}$$