In each problem, write some Sage code to help find the answer. Some problems explicitly ask for a Sage function, others just ask a math question and you will need to develop your own functions. As part of your solutions, I would like to see some test cases showing that the code is producing the right answers in small examples. Sometimes I include a nontrivial test case so that you can check your code, but you should make up your own test cases too. Finally, make sure to include the answers to any questions asked in the problems.

Do enough problems to get to 20 points. The points for each problem are in the left margin. If you are new to coding, it is okay to just do the 4-point problems. If you are experienced, choose problems that are more appropriate for your level.

As always, it is fine to get help from other people or from me. Just make sure you understand things and are not just blindly copying someone else’s code.

Tip: Although we used our hand-crafted function `primetest` during the first couple weeks of class, the built-in function `is_prime` is much faster. Your code will run more quickly if you use that from now on.

Tip #2: If you really want to get good at coding, I highly recommend that you do all the problems at some point. But not necessarily all in one week!

(4) 1. Given a positive integer \( n \), \( \text{sum}\div(n) \) is the sum of all the divisors of \( n \). What is the sum when \( n = 10,000 \)? [Test case: \( \text{sum}\div(10)=18 \)]

(4) 2. Given a positive integer \( n \), \( \phi(n) \) is the number of integers from 1 to \( n \) that are relatively prime to \( n \). This is usually called the Euler \( \varphi \)-function (the name of this Greek letter is “phi”). [Test case: \( \phi(10)=4 \)]. What is \( \phi(1000) \)?

(4) 3. How many primes are there between \( 10^5 \) and \( 10^8 \)?

(4) 4. What are the first 20 primes above \( 10^5 \)? Above \( 10^6 \)?

(4) 5. Write some Sage code to produce the following picture (or something close to it), but change each color to something else:

![Picture](image)

(4) 6. The first prime number is 2, the second is 3, and so forth. Find the 10,001st prime number.
7. If you roll three dice, what is the probability that the sum of the numbers you get is at least 15? Write some Sage code to experimentally estimate this. [Test cases: The probability of getting at least 3 is 1. The probability of getting at least 18 is about 0.0046.]

8. Find all the primes between 1 and 10,000 whose last three digits are 111.

9. Consider the sum
\[\frac{1}{2} + \frac{1}{3} + \frac{1}{5} + \frac{1}{7} + \frac{1}{11} + \frac{1}{13} + \cdots\]
where the denominators are the primes.

(a) Write a function \texttt{series(n)} that adds up the first \(n\) terms and returns a decimal answer. [Test cases: \texttt{series(1)}=0.33333, \texttt{series(2)}=???, \texttt{series(10)}\approx1.533]. Find the sum of the first \(10^5\) terms.

(b) What is the smallest denominator we can stop at in order for the sum to be larger than 2? [Note: This does not ask for the number of terms, it asks for the last denominator.]

10. Given a positive integer \(n\), write a Sage function that finds the smallest integer that is divisible by all the numbers 1, 2, 3, \ldots, \(n\).

11. You have a bin with 10 red balls, 5 yellow balls, 4 blue balls, and 8 green balls. You one-by-one randomly pick out four balls and place them on a table near the bin. Write some Sage code to experimentally estimate the probability of getting at least three red balls.

12. A lottery randomly chooses five numbered balls from a big container will balls labelled 00–99 (one ball with each label). What is the probability that the resulting lottery number has at least two primes? Write Sage code to experimentally approximate this.

13. Write Sage code to draw a pentagram (5-pointed star) in a circle. Then do a similar 7-pointed star, where every vertex is connected to the “second vertex down” from it. (If the vertices are labelled in order as \(abcdefg\), then \(a\) is connected to \(c\), \(c\) is connected to \(e\), and so forth.) Generalize to the \(n\)-pointed star.

14. Write Sage code that will approximate \(\pi\) by computing the perimeter of a regular \(n\)-gon inscribed in a circle of radius 1. What does \(n\) have to be for the approximation to be correct to 3 decimal places (after the decimal point)?

15. A standard deck of cards has four suits and 13 “ranks” (ace through king). We can represent a card in Sage as a list \([a, b]\) where \(a\) is a number in \(\{1, 2, 3, 4\}\) and \(b\) is a number in \(\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13\}\). If we randomly pick five cards from a standard deck, what is the probability that all five have the same suit? Approximate this experimentally using Sage. [All five having the same suit is called a ‘flush’. The probability can be easily looked up with Google, to confirm that your code is functioning correctly.]
16. (The Buffon needle problem) Draw the vertical lines $x = n$ on the plane, for every integer $n$. (So this is an infinite series of vertical lines one unit apart). Take a needle of length 1 and randomly toss it onto the plane. What is the probability that the needle crosses at least one of the lines? It turns out the answer is $\frac{2}{\pi}$. Write some Sage code that tries to demonstrate this by repeating the needle-throwing experiment a large number of times and experimentally approximating the probability. Have the code output both the probability and a graphics picture showing the lines and all the needles that were thrown. Also, have the code output an approximation of $\pi$ by printing $2/p$ where $p$ is your computed probability. [Tip: You might need to only look at the plane between (for example) $x = -10$ and $x = 10$.]

17. In the following grid, a bug starts at the origin $(0, 0)$. At each move, he randomly chooses one of the edges to move along. (So at $(0, 0)$ he has two choices, but if he ever gets to $(1, 1)$ he has four choices). If the bug ever makes it to $(3, 2)$, he stops. Use Sage to experimentally approximate the probability that the bug makes it to $(3, 2)$ in a most 10 steps.