

day21 factors

Due: 6th period, Tuesday 9/27

7th period, Wednesday 9/28

Today's program uses a bunch of skills and commands we have already learned to do some new things. Today's program will:

- Ask the user for a positive integer using the try/except structure. A positive integer is greater than 0. Give one error message if the user enters a non-positive number and a different error message if the user fails to enter a number of any kind.
- State if the number is odd or even. ( use this: `if num % 2 == 0:`  )
- Find the sum of all the integers from 1 to the number. For this you will use a for loop. You literally did this on our for loop day (part 2). Look it up.
- Find all the factors of the number, store them in a list, then print the list. For example, the factors of 6 are: 1, 2, 3, and 6. The factors of 7 are: 1 and 7. You can tell if one number is a factor of another by using the modulo operator. For example, this line checks if x is a factor of num:

```
if num % x == 0:
```

- Report if the number is prime or not. You'll use an if/else. A number is prime if it can only be divided evenly by itself and 1 (for example, 7 is prime, 6 is not). An easy way to tell if your number is prime is to see how long the list of factors is: if it is 2 numbers long, your number is prime, otherwise, it is not prime. You can tell how long a list is using the `len()` function.
- Ask if the user wants to run again.
- `"""Test your output with an odd number and with an even number. Also make sure you test with a prime number. Include output showing at least these two tests and proper output in triple quotes below your code."""`

Notes:

If you have forgotten how to do the try/except thing, go look it up. You did this last week multiple times. Read my tips up above.

To find the sum of the integers from 1 to the number, make a **for loop** using the range command where x starts at 1 and goes up to your number (`num+1`), and then add each x to a sum variable as you go up. To check your code, the sum of all integers up to and including 4 is 10 ( $1 + 2 + 3 + 4 = 10$ ). You should create a variable *before the for loop starts* and put zero into it so that you can add up all the numbers. ( `total = 0` outside of the loop, then inside the loop put `total = total + x`).

(continued on next page)

To find the factors of a number, create an empty list ( `factors = []` ) before your loop starts then do a for loop using a `range()` call from 1 to the number+1 using a variable `x`, and use the `%` operator with the number and each `x`. If there is no remainder (i.e. if `num % x == 0`), then `x` is a factor, and you should add it to a list variable using `append()`. Then you can print the list of factors out when the loop is done.

To find if a number is prime, check how many factors you have when you're done with the above loop. If you have two factors then the number is prime. For example, for 7 as `n`, you'd end up with a list containing `[ 1,7 ]`, because no other integers divide evenly into 7, so just by the length of the list (the "`len()`" function) you can tell if a number is prime. (If the list has only 2 entries, your number is prime.)

Sample output:

```
Please enter a positive integer: 7
7 is odd.
The sum of all positive integers from 1 to 7 is 28
The factors of 7 are: [1, 7]
7 is prime.
Do another? (y/n) y

Please enter a positive integer: hi
> I am asking for a number, please try again.

Please enter a positive integer: -7
> The number must be positive, please try again.

Please enter a positive integer: 240
240 is even.
The sum of all positive integers from 1 to 240 is 28920
The factors of 240 are: [1, 2, 3, 4, 5, 6, 8, 10, 12, 15, 16, 20, 24, 30, 40, 48,
60, 80, 120, 240]
240 is not prime.
Do another? (y/n) n
Too bad, I was having fun.
```

The above output satisfies my testing requirements. In your sample output I need to see the following:

- failing to enter a number
- entering a negative number,
- entering one odd number,
- entering one even number
- entering one prime number. In my example 7 is both odd and prime, which takes care of two of the tests.