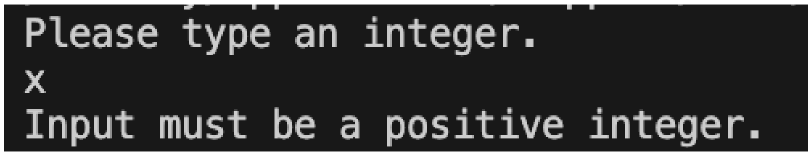
Adam Cartozian and Scott Wegley

9/3/23

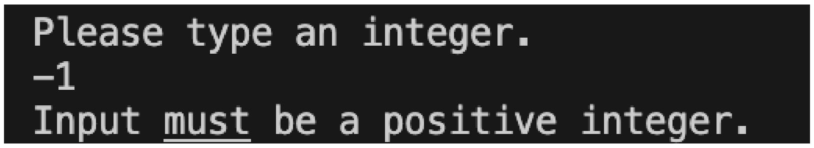
CSC-335

Happy Prime Numbers

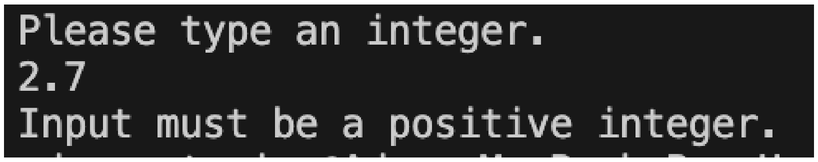
A. Testing:



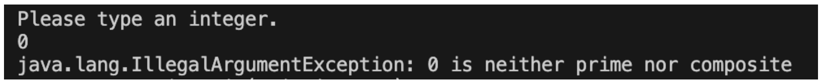
```
Please type an integer.
x
Input must be a positive integer.
```



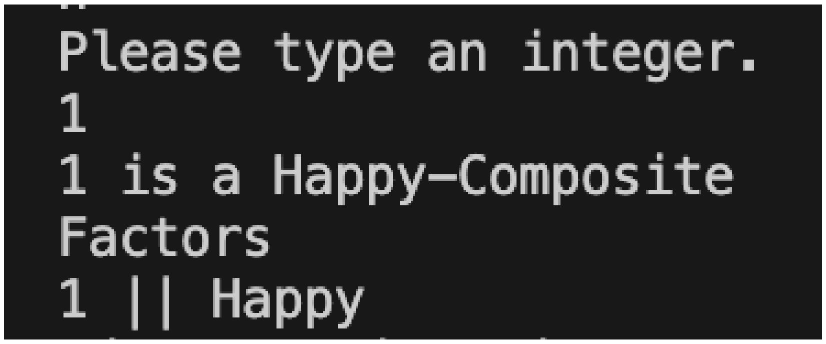
```
Please type an integer.
-1
Input must be a positive integer.
```



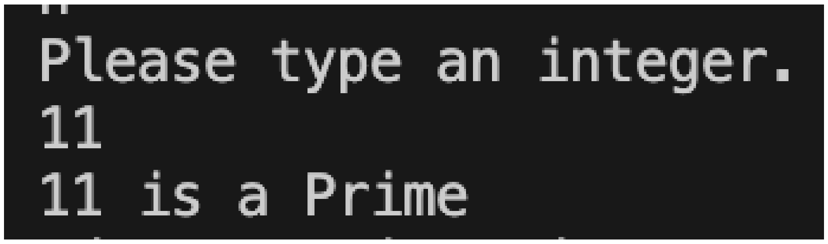
```
Please type an integer.
2.7
Input must be a positive integer.
```



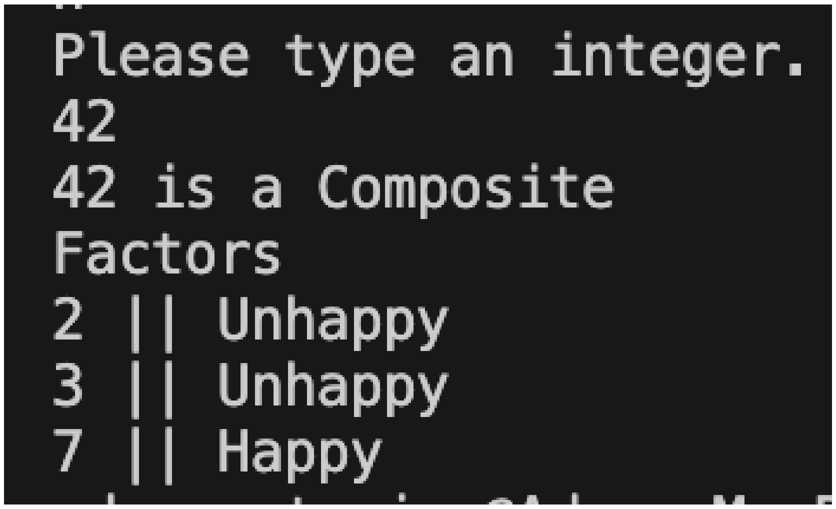
```
Please type an integer.
0
java.lang.IllegalArgumentException: 0 is neither prime nor composite
```



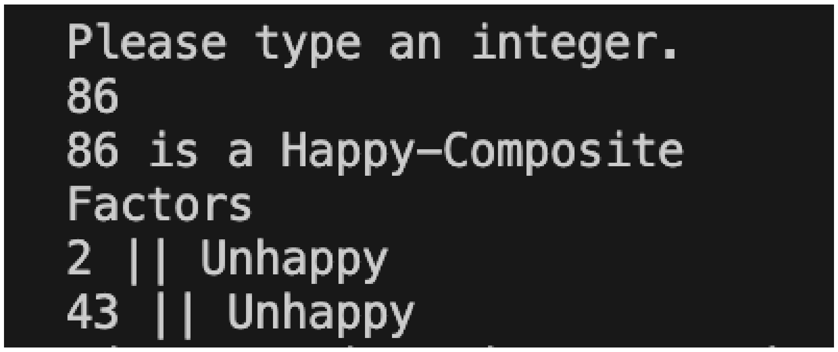
```
Please type an integer.
1
1 is a Happy-Composite
Factors
1 || Happy
```



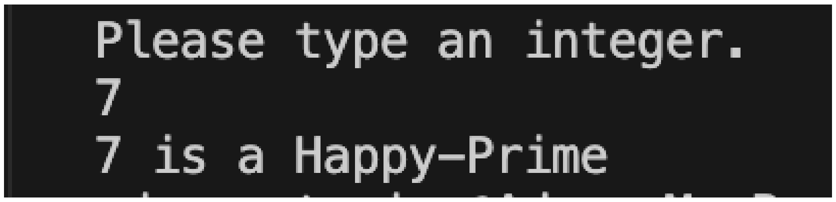
```
Please type an integer.
11
11 is a Prime
```



```
Please type an integer.
42
42 is a Composite
Factors
2 || Unhappy
3 || Unhappy
7 || Happy
```



```
Please type an integer.
86
86 is a Happy-Composite
Factors
2 || Unhappy
43 || Unhappy
```



```
Please type an integer.
7
7 is a Happy-Prime
```

B. Before we began coding, we first researched what exactly a happy number was and how a happy number is calculated. We then looked at the various ways that we could implement the equation. We settled on reading the integers from the number, then squaring those integers and adding them together. This repeats until there is a single digit result. We then turned to the prime numbers in which we took our code that we wrote in class and began adding to it to fulfill the requirements of the assignment. Once the happy number and prime numbers classes were built, we added temporary lines of code to test the output of the respective classes. Once tested, we turned to creating our main app class that would allow the user to input an integer which would call the other classes.

C. In order to test our program for correctness, we took the inputs that were given to us and calculated by hand if they were prime, happy, happy prime, or composite number. Once we had those results, we imputed the numbers into the interface and tested the outputs against the outputs that we hand calculated.