1. This week's Problem of the Week in Math is described as follows:

   > *There are thirty positive integers less than 100 that share a certain property. Your friend, Blake, wrote them down in the table to the left. But Blake made a mistake! One of the numbers listed is wrong and should be replaced with another. Which number is incorrect, what should it be replaced with, and why?*

   The numbers are listed below.

6	10	14	15	21
22	26	33	34	35
38	39	46	51	55
57	58	62	65	69
75	77	82	85	86
87	91	93	94	95

Use the fact that the "certain" property is that these numbers are all supposed to be the product of *unique* prime numbers to find and fix the mistake that Blake made.
Reminder: Code your solution in an `R` script and copy it over to this `.Rnw` file.
Hint: You may find the `%in%` operator and the `setdiff()` function to be helpful.

Solution:

75 is the incorrect number, and it should be replaced with the number 74. 75 is the only number in the provided list that contains a prime in its factorization twice (3x5x5). The one number missing from this list, and therefore the replacement for 75, should be 74, since it is the only missing number within the provided list's range that is the product of two unique primes (2x37).

For my solution, I created a list of primes up to a specified max using a makeshift Sieve of Eratosthenes, then multiplied unique pairs of these primes to create a list of all numbers that are the products of two unique primes within the provided list's range. I then took the asymmetric difference of the provided list and my new list to determine the incorrect number, and the asymmetric difference in the opposite order to determine its replacement.

```
# Copy your solution code here.

nums.in.list <- c(6,10,14,15,21,22,26,33,34,35,38,39,46,51,55,57,58,62,65,69,75,77,82,85,86,87,91,93,94,95)
max.of.list = max(nums.in.list)

# tried a sieve of eratosthenes
max.to.check <- floor(max.of.list/2) # since 2 is the smallest prime, if a number is the product of two unique primes,
                                     # the largest possible candidate prime is at most half the list max rounded down
primes <- 2:max.to.check
for(i in 2:max.to.check){ # not maximally efficient, but rather not use primes here since it gets dynamically modified
  primes <- primes[primes == i | primes %% i != 0] # first part keeps the original candidate,
                                                   # second part removes multiples of it
}

good.nums <- rep(0,times = max.to.check) # initializing a numeric vector to hold all numbers in range
                                         # that are products of two unique primes
for(j in 1:(length(primes)-1)){ # ending at index length(primes)-1 avoids out of bounds issues
  for(k in (j+1):length(primes)){ # starting at index j+1 excludes all squares of primes
    n <- primes[j]*primes[k] # product of two unique primes, no duplicates since
                             # j<k is an invariant + fundamental theorem of arithmetic
    if(n <= max.of.list){ # not interested in numbers outside our given list range, also avoids index issues
      good.nums[n] <- n; # avoids having to count instances of "good nums"
    }
  }
}
good.nums <- good.nums[which(good.nums != 0)] # filter out the zeros at indices of "bad nums"

(incorrect.num <- setdiff(nums.in.list, good.nums)) # s-e


## [1] 75


(replacement.num <- setdiff(good.nums, nums.in.list))


## [1] 74
```