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:

```
# All the possible prime numbers
prime.numbers <- 2:100

# Outer loop to find all numbers from 2 to 100
for (i in 1:100) {
  # Inner loop to check if it can go evenly into each number by numbers smaller than itself
  for (x in 2:(i-1)) {#Loop through everything smaller than i
    # If it can be divided by things that are smaller than it than it is not prime so we get rid of everything that
    #can do that
    prime.numbers = prime.numbers[prime.numbers != i | i %% x != 0]
    #Only doesn't add 2
  }
}
#Would've just done c(...) but thought we had to code it


#Adding 2
prime.numbers = c(prime.numbers, 2)
prime.numbers = sort(prime.numbers)

#Making the products into a vector

num.to.check=c()

for(i in 1:length(prime.numbers)){
  for(x in 1:length(prime.numbers)){
    num.to.check = c(num.to.check, prime.numbers[i]*prime.numbers[x])

  }
}

num.to.check = sort(num.to.check)


#Removing the squares and duplicates
num.to.check = unique(num.to.check)

x = 1:100
y = x^2

num.to.check = num.to.check[-which(num.to.check %in% y)]
num.to.check = sort(num.to.check)

#Adding the original problem

original.numbers = 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)

#Finding the one that does not belong

original.numbers[-which(original.numbers %in% num.to.check)]


## [1] 75


#Finding the one to add back in

num.to.check = num.to.check[which(num.to.check < 100)] #Sorting below 100

num.to.check[-which(num.to.check %in% original.numbers)]


## [1] 74
```

Reasoning: 75 should **not** be in this set of numbers because it can be broken down into 25 and 3 and 25 is not prime. 74 **should** replace it because it can be broken down into 37 and 2 which are both prime numbers.