STATISTICAL RETHINKING 2022
WEEK 1 SOLUTIONS

1. Really all you need is to modify the grid approximation code in Chapter 2.

```
p_grid <- seq( from=0 , to=1 , length.out=1000 )
prior <- rep( 1 , 1000 )
prob_data <- dbinom( 4 , size=4+11 , prob=p_grid )
posterior <- prob_data * prior
posterior <- posterior / sum(posterior)
set.seed(100)
samples <- sample( p_grid , prob=posterior , size=1e4 , replace=TRUE )
```

When you plot the result, the posterior will have much more mass over values below 0.5, since that is what the sample indicates. The posterior mean is about 0.30.

2. Modifying the prior and changing the water and land counts:

```
p_grid <- seq( from=0 , to=1 , length.out=1000 )
prior <- c( rep( 0 , 500 ) , rep( 1 , 500 ) )
prob_data <- dbinom( 4 , size=4+2 , prob=p_grid )
posterior <- prob_data * prior
posterior <- posterior / sum(posterior)
set.seed(100)
samples2 <- sample( p_grid , prob=posterior , size=1e4 , replace=TRUE )
```

The posterior mean should be about 0.7. With the impossible values less than 0.5 ruled out, the second model piles up more plausibility on the higher values near the true value. The data are still misleading it to think that values just above 0.5 are the most plausible.

Informative priors, when based on real scientific information, help. Here, the informative prior helps because there isn't much data. That is common in a lot of fields, ranging from astronomy to paleontology.

3. The code is very simple, but the interpretation maybe less so:

```
set.seed(100)
PI( samples2 )
HPDI( samples2 )
```

```
       5%       94%
0.5245245 0.8798799
> HPDI( samples2 )
    |0.89     0.89|
0.5005005 0.8388388
```

The percentile interval (the top one) is wider. The lower bound is above 0.5 (a little bit) and the upper bound is 0.88. The HPDI (bottom interval) is narrower. The lower bound is right at 0.50, but the upper bound is 0.84 instead of 0.88. You should expect the HPDI to be narrower and the include the point with highest posterior probability. This is just like the example on page 57 in the book. But since the boundaries of these intervals aren't really informative—nothing special happens at the boundary—when these intervals are very different, the best thing is not to report intervals at all. Just draw the posterior distribution, so your colleagues can see what is going on.

4 - CHALLENGE. To simulate the biased sampling, you could do it in two steps or one step. I'll show both.

The two-step simulation first simulates a true count of water. Then it simulates the biased count. Both steps are binomial samples. But the true water count uses a probability of 0.7 for water. The biased step uses a probability of 0.8 for water and a size equal to the number of water samples in the true count. The idea is that each true water sample has only a 0.8 chance of being reported as water. Like this:

```
set.seed(100)
N <- 1e5
trueW <- rbinom(N,size=20,prob=0.7)
obsW <- rbinom(N,size=trueW,prob=0.8)
mean(trueW/20)
mean(obsW/20)
```

```
[1] 0.700241
[1] 0.5599695
```

So the true count has a mean of 0.7 for obvious reasons. But the biased one gives us a mean around 0.56 instead.

Now the one-step approach. If the are 0.7 out of 1 ways to sample water and 0.8 out of 1 ways for water to be reported as water, then there must be 0.7×0.8 ways to observe water. Remember, the garden multiplies independent

events. It's the produce rule of probability theory, but it is just a consequence of counting independent combinations. Here's the code:

```
set.seed(100)
W <- rbinom(N,size=20,prob=0.7*0.8)
mean(W/20)
```

```
[1] 0.560283
```

Again 0.56. That's the biased expectation.

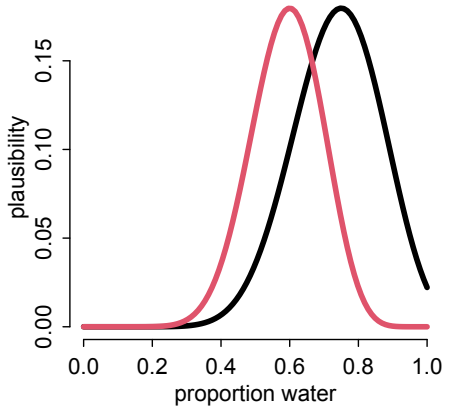
Now let's estimate the posterior distribution of water, accounting for sampling bias. To do this, we just use ordinary grid approximation. The only new bit is the probability of the data, which needs to use the biased sampling rate 0.8. Here is how I did it:

```
# sim some data
W <- rbinom(1,size=20,prob=0.7*0.8)

# compute posterior
grid_p <- seq(from=0,to=1,len=100)
pr_p <- dbeta(grid_p,1,1)
prW <- dbinom(W,20,grid_p*0.8)
post <- prW*pr_p
```

All I did was replace the 0.7 in the simulation with a parameter. Let's also compute the posterior, ignoring the bias, and then plot to compare:

```
post_bad <- dbinom(W,20,grid_p)
plot(grid_p,post,type="l",lwd=4,
    xlab="proportion water",ylab="plausibility")
lines(grid_p,post_bad,col=2,lwd=4)
```



Black is the posterior that accounts for sampling bias, so it has a higher mean than the red, which does not.

This kind of inference problem is a binary classification task with error. It is very commonplace. Much of medical testing has a similar structure. But often there is also error of the other type as well: Land can be misclassified as water. Can you figure out that simulation and model as well, in which both water and land have chances of being misclassified?