1. Why do we use predictors with 2-bit saturating counters (i.e., bimodal)? Contrast this with the 1-bit approach.

   1-bit counters change its mind too quickly. 2-bit counters allow two mis-predicts before changing its mind. Consider branch outcome pattern: TNNNNNN* (This is a common pattern seen in a nested for loop). Having a 2-bit saturating counters allows the prediction to only mis-predict once. Having a 1-bit counter, there will be 2 mis-predictions.

2. You are given the following code snippet, with two conditional branches clearly marked:

```
int a;
for (int i = 0; i < 100000; i++) { // conditional branch B1
        if ((i % 4) == 0) { // conditional branch B2
                a = 10;
        }
        a = 15;
}
```

   (a) Write down the taken/not-taken sequence for the second conditional branch B2. List any assumptions you make about how this if-statement translates in assembly.

   Assume: the if statement evaluated to true == TAKEN
   the if statement evaluated to false == NOT_TAKEN
   Let: T = TAKEN, N=NOT_TAKEN
   The taken/not-taken sequence for B2 is: TNNNTNNNTNNN…

   (b) Assume you are keeping track of prior branch history for conditional branch B2. What is the smallest number of history bits (h) you need to correctly predict all B2's branch outcomes?

   If the branch history registers are not shared. Then will need 3 bits.
   Assume you only have two bits, the possible 2-bit sequences for B2 and their possible outcomes are:
   TN: N
   NN: N or T
   NT: N
   For the sequence NN, there are two possible outcomes. By looking at NN, the predictor is not able to make the correct prediction.
   However, if you use 3-bit sequences, the possible outcomes are:
   TNN: N

NNN: T
NNT: N
NTN: N
The possible outcomes for all possible sequences are fixed.

(c) Assume you have a PAg predictor (Figure 3.20) with an eight-entries BHT and 3 history bits per entry (h is the previously computed number). Why would it be a bad idea to index the BHT with the 3 most significant PC bits?

If the BHT is indexed using the 3 most significant bits, aliasing will happen for the close branches. In this case, the outer for loop is not correlated with the inner for loop. To make the correct prediction, more history bits might need to be used.