6. (10 points) The following loop contains 2 branches.

```c
done = 0;
i = 0;
while (!done) {
i++;
    if (i == 10) { ;location of branch B1
        done = 1;
    }
    // some other code that does not touch i
} ;location of branch B2
```

Consider 2 different possible branch predictor structures. The first is a 2-level local predictor, with an n-bit wide pattern history table used to index a \(2^n\)-entry table of 2-bit predictors (saturating counters). The second is a correlating predictor, simply using an n-bit global history register (GHR) to index a \(2^n\)-entry table of 2-bit predictors (also saturating counters).

Complete the following table, assuming the loop is entered many times, without any aliasing from other branches in the predictors:

<table>
<thead>
<tr>
<th>Predictor</th>
<th>n</th>
<th>B1 predict accuracy</th>
<th>B2 predict accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-level local</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-level local</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correlating</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correlating</td>
<td></td>
<td>100%</td>
<td>100%</td>
</tr>
</tbody>
</table>

Don’t forget this one!

Note that the only reasonable choices for the prediction accuracy are numbers like 80%, 90%, or 100% (i.e., multiples of 10). That is, we’re looking for steady-state behavior, not startup behavior.