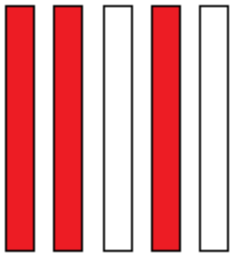
3) (10 pts) DSN (Bitwise Operators)

Imagine the task of painting a picket fence with 20 pickets. Let each picket be numbered from 0 to 19, from left to right and initially each is painted white. A pattern is 5 pickets long and can be placed with the pattern's left end aligned with any picket in between number 0 and number 15, inclusive. (If you line the pattern up with any of the pickets 16 through 19, the right end of the pattern goes past the right end of the fence and this isn't allowed.) Below is a picture of an example pattern, with the 3 of the 5 possible pickets painted:



If this pattern was lined up with picket number 4, then pickets 4, 5 and 7 would get painted. Think of the process as a placing a stamp on a portion of the whole fence. We can represent this pattern with the integer 11 ($2^0 + 2^1 + 2^3$), the integer where bits 0, 1 and 3 are set to 1. The bit positions represent, relative to the left end of the pattern, which positions have paint on them.

One way to paint a fence with a pattern is to line up the pattern with a few different picket numbers and apply the pattern. For example, if we lined up this pattern with the pickets at positions 1 and 4, then the pickets that would be painted would be at positions 1, 2, 4, 5 and 7, which corresponds to a bitmask value of 182. (Notice that painting an individual picket more than once leaves it still painted.)

Complete the function below so that it takes in an integer, ***pattern***, in between 0 and 31, inclusive, representing the pattern, an integer array, ***paintLoc***, which stores the locations to line up the pattern with for painting, and ***paintLen***, representing the length of the array ***paintLoc*** and returns a single integer storing the state of the painted fence (for each picket number, k, that is painted, bit k in the returned integer should be set to 1). Each of the values in ***paintLoc*** will be distinct integers in between 0 and 15, inclusive.

```
int paintFence(int pattern, int paintLoc[], int paintLen) {

    int res = 0;                                    // 1 pt
    for (int i=0; i<paintLen; i++)                  // 2 pts
        res = res | (pattern<<(paintLoc[i]));       // 6 pts
    return res;                                     // 1 pt
}
```

Grading Breakdown 3rd line: 1 pt for update res, 2 pts for |, 2 pts for shifting pattern, 1 pt for shifting it by paintLoc[i].