3) (10 pts) DSN (Bitwise Operators)

User IDs are stored in a system as single integers in between 0 and $2^{25} - 1$, inclusive. Thus, each User ID can be viewed as a bitstring of length 25 (using an integer variables 25 least significant bits). When a new user ID is added, in order not to cause confusion, it's required that it differs in **at least three bits** compared to all other user IDs in the system. Write a function that takes in an array of current user IDs (**curIDs**), the length of that array (**n**), and a potential ID to be added (**pid**) and returns 1 if the potential ID can be added to the current list based on the previously given criteria, OR 0 if it can't be added if there exists a current ID with which the new ID differs in 2 or fewer bits. (For example, **10010000** differs with **00110000** in exactly 2 positions: 2^5 and 2^7. Thus, if the former was in the current ID list, the latter would NOT be an allowable password to add. **PLEASE DO NOT WRITE ANY AUXILIARY FUNCTIONS.**

```
int canBeAdded(int* curIDs, int n, int pid) {


    for (int i=0; i<n; i++) {        // Grading: 1 pt

        int XOR = curIDs[i]^pid;     // Grading: 1 pt

        int diff = 0;                // Grading: 1 pt
        for (int j=0; j<25; j++)     // Grading: 1 pt
            if (XOR & (1<<j))        // Grading: 2 pts
                diff++;              // Grading: 1 pt

        if (diff < 3) return 0;      // Grading: 2 pts
    }

    return 1;                        // Grading: 1 pt



}
```

Grading Note: If a line of code is not in the proper nesting of loops, take off 1 or 2 points as you see fit.