Solution 3

Group A

1) Problem 6

   (a) Denote the drunk man's position at time t as P_t, then we know P_t is a martingale and $P_t^2 - t$ is also a martingale. Denote stopping time τ as the first time the drunk man at position -1 or 99. Then we know a martingale stops at a stopping time is also a martingale. Thus if we denote p as the probability that the drunk man will first visit -1 and q as the probability that the drunk man will first visit 99, we have

   $$p + q = 1$$

   $$p \times (-1) + q \times 99 = 0.$$

   Thus, we know $p = \frac{99}{100}$ and $q = \frac{1}{100}$. Since

   $$E[P_\tau - \tau] = (-1)^2 \times p + 99^2 \times q - E[\tau] = 0,$$

   thus we know $E[\tau] = 99$.

   (b) If the left door is locked, then when the drunk man hit -1, he still need to move right for 100 steps. Since left and right are symmetric, we know this is equivalent that we have two doors at -101 and 99. Thus, by the same analysis as part (a), we know $E[\tau] = 9999$.

2) Problem 8

   The LASSO, which is short for Least Absolute Shrinkage and Selection Operator, is a regression method that involves penalizing the absolute size of the regression coefficients. Its objective function is trying to minimize the following:

   $$\|Y - X\omega\|_2^2 + \alpha\|\omega\|_1.$$

   Compared to ordinary linear regression, the L1 penalization term can yield sparse models, and thus it can be used to do feature selection.

3) Problem 13

```
class Solution {
public:
    int singleNumber(int A[], int n) {
        int result = 0;
        for (int i = 0; i < n; i++) {
            result = result ^ A[i];
        }
        return result;
    }
};
```

4) Problem 14

The solution is found on this website: https://isocpp.org/wiki/faq/strange-inheritance#calling-virtuals-from-base. Yes. Its sometimes (not always!) a great idea. For example, suppose all Shape objects have a common algorithm for printing, but this algorithm depends on their area and they all have a potentially different way to compute their area. In this case Shapes area() method would necessarily have to be virtual (probably pure virtual) but Shape::print() could, if we were guaranteed no derived class wanted a different algorithm for printing, be a non-virtual defined in the base class Shape.

```
#include "Shape.h"
void Shape::print() const
{
    float a = this->area();  // area() is pure virtual
    // ...
}
```