a. If g = log N, using binary search method will give us the speed log N time and it minimizes the number of guesses we have to use.
b. If g = 1, we can guess 1+ $n^{1/2}$ + 1 times and we do this until we guess the prize value. Since the range reduces by sqrt of N at each step, you can find the correct value in approximately sqrt of N steps. Therefore we can always guess the value without guessing too high.
c. In order for the algorithm to win in fewer than $\Omega(N^{1/2})$ guesses, it must guess the correct value before guessing the value at position sqrt of N + 1. Or it will take at least sqrt of N + 1 guesses to reach the correct value. However, the probability of guessing the correct value before position sqrt of N + 1 is 1/sqrt of N. Therefore, the probability of the algorithm winning in fewer than Ω (N^0.5) guesses is at most 1/sqrt of N.
d. We can use linear search as it starts from 1 and goes until the prize value. If we just start with the beginning and increase it by 1, we will never use our "g" guesses so it will satisfy the worst condition when constant g = 0.