HANGMAN AI ALGORITHMS

When the computer is the Guesser, it will maintain a list of all the words that are consistent with all the information known about the secret solution word thus far (including length, the letters it contains, the positions they are in, and the letters that it does NOT contain). Then for each letter in the alphabet, the AI will look at the percentage of those words that contain that letter, and decide randomly in the same proportion as those percentages. For example, suppose that the AI had the possible word list culled down to EEL, LOP, and PEL (realistically, this would be a much longer list). In this case, nearly all letters would be rated as 0% except for E=67%, L=100%, O=33%, and P=67% (note that the number of times that the letter appears in a the words are irrelevant, just whether it appears in it or not. This strategy is simply based on trying not to get strikes, as opposed to trying to minimize the amount of asterisks). Rather that going with the largest percentage chance of success (which the opponent might manipulate to their advantage) instead we will choose randomly in same proportion as the calculated percentages. In this case, if 0<=randnum<1, it would be divided up in alphabetic order as: 0.0 to 0.25 is E, 0.25 to 0.625 is L, 0.625 to 0.75 is O, and 0.75 to 1.0 is P. Make sure you fully understand how those numbers were computed (it works the same way as the likelyMove method from the previous MasterMind Project2)

When the computer is the Chooser, first it chooses a random word equally out of the entire word list. Then when the human makes a guess at a letter, it looks at all possible answers it might give for positions of that letter (the number of options could be as high as 2 to the power of the number letters in the solution. That is why I recommend setting a maximum word length of 30). It will then secretly change its solution word so that the position answer has the largest number of words that could match it (and still also be consistent with all previous answers it has given, of course). Tie breakers will be broken in alphabetical order, so there is NO RANDOMNESS involved with this strategy. The following example should illustrate the algorithm:

Suppose the Guesser asks about the letter 'b' and the AI's list of all possible remaining words was down to:

LOVE, OBOE, OBEY, BIBS, BOBS

Then the computer's analysis would go like this: there would only be one option (LOVE) if there were reported to be no b's, but there would be 2 options (OBOE/OBEY) if b was reported to have position [2] and also two options (BIBS/BOBS) if b had position [1 3]. Thus, the computer needs to do a tie-breaker. In order to avoid any randomness at all, the program should always choose the position by the one that appears first in "reverse binary order": in other words, imagine if the letters at the displayed positions were Z and every other letter in the words were an A. Then the computer will choose the one that comes first alphabetically. In this case pos=[2] corresponds to AZAA and pos=[1 3] corresponds to ZAZA and since AZAA comes first alphabetically the computer should declare that b appears in positions [2]. (You don't need to think of it this way, but in case it helps: the "reverse binary" comes in if you write a 0 for the A and a 1 for the Z, and then write the number in reverse order and choose the smaller one. In this case, pos[2] would be 0010 and pos[1,3] would be 0101. And 0010 (the number 2) is smaller than 0101 (the number 5) and that's another way to realize that pos=[2] gets chosen). Then once it knows the position, the secret word becomes the first word alphabetically (the normal dictionary alphabetically) that's in that group. So, in this case, the secret word would become "OBEY" (since that comes alphabetically before "OBOE").

Keep in mind, the tie-breaker is only needed in the very rare case of a tie. If the word "BUBS" had been on the list too, then pos=[1,3] would have had 3 words, which beats all the other possibilities, and the secret word would have become "BIBS" (the first alphabetically among that group).

This will be the last Project of the course, and must be submitted by exactly one of the partners (you will
b
o
t
h