Homework Assignment #2 (100 points, weight 8%)
Due: Saturday Nov 14, 11:59PM

Guidelines for programming parts: Write your program in some high level programming language such as C, C++, Java. Hand in pseudocode, program and output results (note if too many tests are done, submit only a sample of output results and summarize results in tables). Please, specify the platform you run your tests on (machine speed, machine RAM and operating system).

1. (25 points) **Backtracking for self avoiding walks** (written question)
   A self-avoiding walk is described by a sequence of edges in the Euclidean plane, beginning at the origin, such that each of the edges is a horizontal or vertical segment of length 1, and such that no point in the plane is visited more than once. There are precisely 4 such walks of length 1, 12 walks of length 2, and 36 walks of length 3. Define choice sets and describe a backtracking algorithm for the problem of finding all self-avoiding walks of length \( n \).

2. (25 points) **Estimating backtracking tree size** (written question)
   Write an algorithm in pseudocode that uses the method of estimating the size of a backtrack tree described in Section 4.4, in order to estimate the total number of cliques of a given graph. The input for your algorithm consists of a graph \( G \) and the number \( P \) of probes, and the output is the estimated number of cliques of the graph based on \( P \) probes.

3. (50 points) **Backtracking program for maximum constant weight codes.**
   If \( x, y \in \{0, 1\}^n \), then recall that \( \text{Dist}(x, y) \) denotes the Hamming distance between \( x \) and \( y \); the weight of \( x \) is the number of non-zero components of \( x \) (since \( x \) is binary this is the number of 1s). A non-linear code of word length \( n \), minimum distance \( d \) and constant weight \( w \) is a subset \( \mathcal{C} \subseteq \{x \in \{0, 1\}^n : \text{weight}(x) = w\} \) such that \( \text{Dist}(x, y) \geq d \) for all \( x, y \in \mathcal{C} \). Denote by \( A(n, d, w) \) the maximum number of \( n \)-tuples in a length-\( n \) binary code of minimum distance \( d \) and weight \( w \).
   
   (a) Describe a backtracking algorithm to compute \( A(n, d, w) \) (give pseudocode and any other pertinent explanation).
   
   (b) Implement your algorithm and compute \( A(n, 4, w) \) for \( w = 3, 4, 5, \) and as many values as possible of \( n \geq 2w \). The known values for \( A(n, 4, w) \) for small values of \( n \) and \( d \) can be found in the following web page:

   \[
   \text{http://www.win.tue.nl/~aeb/codes/Andw.html}
   \]
   
   For each of your tests, report the input values, the final answer, the number of backtracking nodes visited and CPU time. Show a sample of results where you also show the binary codes produced, in addition to their size.
   
   You can use bounding and/or any problem characteristics to find an optimal solution as quickly as possible. Efficiency and clarity count.