For each of the algorithm problems, design as asymptotically efficient an algorithm as possible. Give a correctness argument (explanation, if it is relatively simple, or proof if not) and time analysis. You may use any well-known algorithm or data structure, or algorithm from the text or from class, as a sub-routine without needing to provide details.

**Dance partners** You are pairing couples for a very conservative formal ball. There are \( n \) men and \( m \) women, and you know the height and gender of each person there. Each dancing couple must be a man and a woman, and the man must be at least as tall as, but no more than 3 inches taller than, his partner. You wish to maximize the number of dancing couples given this constraint. (4 pts correct poly-time algorithm, 4 pts correctness proof, 2 pts efficiency. My best time is \( O(n \log n + m \log m) \)).

**Homework grade maximization** In a class, there are \( n \) assignments. You have \( H \) hours to spend on all assignments, and you cannot divide an hour between assignments, but must spend each hour entirely on a single assignment. The \( I \)’th hour you spend on assignment \( J \) will improve your grade on assignment \( J \) by \( B[I, J] \), where for each \( J \), \( B[1, J] \geq B[2, J] \geq \ldots \geq B[H, J] \geq 0 \). In other words, if you spend \( h \) hours on assignment \( J \), your grade will be \( \sum_{i=1}^{h} B[i, J] \) and time spent on each project has diminishing returns, the next hour being worth less than the previous one. You want to divide your \( H \) hours between the assignments to maximize your total grade on all the assignments.

Give an efficient algorithm for this problem. (3 pts. correct poly-time algorithm, 3 points correctness proof, 4 points efficiency. My best time is \( O(n + H \log n) \)).

**Approximation for bin filling** In the bin filling problem, you have \( n \) items of positive integral sizes \( a_1, \ldots, a_n \) and \( m < n \) bins, where bin \( j \) is of size \( B_j \). You need to assign each item \( i \) to a bin \( A[i] \), in a way to fill the maximum number of bins, where a bin \( j \) is full if \( \sum_{i \mid A[i]=j} a_i \geq B_j \).

Give an efficient approximation algorithm for this problem. Most of the points will be based on your approximation ratio and the proof that it achieves this ratio. The best possible ratio is to fill at least 1/2 as many bins as the optimal solution.

**Base Conversion** Give an algorithm that inputs an array of \( n \) base base 10 digits representing a positive integer and outputs an array of bits representing the same integer in base 2. Your algorithm should be \( o(n^2) \), strictly better than the time asked for on the calibration homework. You
will probably need to use a divide-and-conquer strategy, and use a fast integer multiplication as a subroutine (from class). [3 points correct algorithm and correctness proof, 7 points efficiency]

**Implementation: Integer Multiplication** Implement the $O(n \log^3 n)$ divide-and-conquer algorithm for integer multiplication from class, but with a threshold, below which naive “gradeschool” multiplication is used. Use an array of digits to represent inputs and outputs. Experimentally determine the optimal threshold. For what values of $n$ do you see an improvement in the time using divide-and-conquer, both using no threshold and using the optimal threshold?

When describing an algorithm, don’t write out an entire pseudo-code; just describe it at a high level. Be sure to specify completely all data structures used in the algorithm. Include correctness proofs and time analysis for all algorithms, except for the implementation problem.