Final Purdom
5-8-97 B501

1. Approximate $\sum_{0 \le i \le n} i^2 \lg i$.

2. Suppose an algorithm has a running time given by

$$T_n = 2T_{n-2}.$$

   Solve the recurrence.

3. Suppose an algorithm takes time.

$$S_n = S_{n-1} + S_{n-3}.$$

   Solve the recurrence.

4. Determine which of the two previous algorithms need the most time for large n. (If your answer is that it depends on the boundary conditions, then give enough additional information so that some one who knew the boundary conditions could figure out which algorithm to use.) Give your reasoning.

5. Suppose you want to multiply a number with $h2^k$ digits by a number with 2^k digits. One way to do it is to do h multiplication problems with 2^k digits for each number and to add the results together (after sifting each result by the right amount). Write an equation to describe the time needed by this approach. One of the items on the right side of the equation should be the time needed to multiply 2^k digit numbers. Be clear about the meaning of each item in your equation.

6. Three ways to multiply a number with $h2^k$ digits by one with 2^k digits are:
   a. use the approach of Question 5 and then multiply the h pairs of numbers using the recursive algorithm from the project,
   b. treat each number as though it has $h2^k$ digits using the recursive algorithm from the project, or
   c. improve method b by keeping track of how many digits each number actually had and skipping computations involving multiplication by numbers known to be zero.

   Determine as best you can the relative merits of these three methods using analysis. Make clear any assumptions that go into your analysis. If some English description above suggest several possible algorithms, make clear precisely which one you are analyzing. Save time by not going into any more detail than is needed to determine the relative quality of the three algorithms (which is best, which is second best, etc.). Don't try to do more than is possible with analysis. Perhaps measurements are needed to distinguish some of the algorithms, perhaps not.