Assignment 1.

Given September 6, due September 20.

Objective: To explore dynamic programming.

This is a qualitative model. It is unrealistic in several ways in order to be as simple as possible. We suppose that a floating interest rate fluctuates between 1% and 10% (annualized). Now it is 5%. Each week it may move up or down by .2% (e.g. 4.4%, 4.6%, 4.8%, 5%, etc.), except that it is blocked from moving outside its range. It moves in either direction with probability 1/3 and remains unchanged with this probability. All movements are independent. This interest rate is $R(t)$. A loan of \$1 is to be repaid in $T = 250$ weeks. Each week, we add some interest and make a payment. If our interest rate at week t is $r(t)$, and the outstanding balance is $B(t)$, then, at week t we first add interest, replacing $B(t)$ by $B'(t) = (1 + r(t)/52) * B(t)$. Next we make a prorated payment, $P(t) = B'(t)/(T - t)$. This leaves $B(t + 1) = B'(t) - P(t)$. Our interest rate is determined as follows. At week 0, $r(t) = R(t) = 5\%$. However, $r(t)$ remains fixed until we "refinance". Whenever we refinance, $r(t)$ is replaced by $R(t)$. We are allowed to refinance up to four times during this 250 week period. We want to do so so to minimize our total expected payments, $\mathbf{E}[P(0)+P(1)+\cdots+P(249)]$. Set this up as a dynamic programming problem. Identify the state space. Write a computer program to solve it by dynamic programming.
Hint: The "cost to go" function, $f(r, R, t, T, k)$, with k being the number of refinancings so far, should be the total payment starting at time t *supposing that* $B(t) = 1$. That is, the expected total payment starting at time t with balance $B(t)$ and the optimal refinancing strategy is $B(t) \cdot f$, rather than just f. Otherwise f would have to depend on B also, which would make the computation more complicated.
Programming tip of the day: Don't hard code numbers such as 52, 4, and 250. It is not more efficient and makes the code hard to test.
Guidelines for good homeworks:

- Explain what steps you took to insure that the answers produced by the program are correct. Did you run the program in a case for which you know the answer? Did you check afterwards that the answer is correct? Are there consistency checks (sums being one or zero, ...).'?

- Hand in a good reader friendly program. It should be modular, clearly documented, and have logic that is easy to follow.

- Choose output wisely. Don't hand in a tree worth of paper, or just one number. Think of informative graphics.

- Make a brief writeup of results. Hand in more than just the program and the results.