Short Problems:

1. We take equal-sized samples from two plant species: a Native species which has had constant population size, and an Invasive species which has expanded rapidly. Their mutation rates and current population sizes are the same. Briefly explain whether you expect Native or Invasive to have more of the following:

   (a) (2 pts) Sites that vary within the sample? *Native. Invasive has had lower effective population size on average (as it expanded rapidly to its current size) and this will lead to a shallow coalescent and fewer variable sites.*

   (b) (2 pgs) Variants found in just one individual in the sample, as a fraction of total variants in the sample? *Invasive. In a rapidly growing population, most of the branch length in the coalescent is in branches leading to the tips, so most mutations are found in just one individual ("private").*

   (c) (2 pts) Effective population size? *Native. Invasive is the same size now and was smaller in the past, so it must have a lower effective size, probably much lower. As one student said, its genetic history dominated by a few invasive ancestors.*

2. Consider one of your two copies of a particular genomic region. For this problem, assume only genetic drift, no selection; ignore population subdivision; and assume that reproduction is at random. The current human population size is roughly 7.7 billion ($7.7x10^9$) and the human generation time is roughly 25 years.

   (a) (2 pts) Eventually all copies in the human species will trace back to just one copy that exists today. What is the approximate chance that your specific gene copy will be the winner? *For your specific copy only,* $\frac{1}{2*7.7x10^9}$*; every copy has an equal chance. If we instead wanted to know the chance that a copy identical to yours would win, we would need to know the population frequency of your allele.*

   (b) (2 pts) Does future population growth or shrinkage affect this estimate? *No. Your copy continues to have a fair chance among all the others; it is no more or less likely to be increased by growth or decreased by shrinkage than the others.*

   (c) (2 pts) How many generations is the lucky winner expected to take to reach fixation? $4N_e$ *generations. If we assume* $N_e = N$ *this would be* $4x7.7x10^9$*. This number is so ridiculously large that the* N *versus* N_e *issue hardly matters. The Sun is predicted to turn into a red giant in only* $5x10^9$ *years.*

   (d) (2 pts) If the population instead continues to grow, will the expected time to fixation increase or decrease? *Increase.*

3. (3 pts) Does genetic drift even matter in humans anymore now that there are so many of us? Briefly defend your answer. *Because of population subdivision, many humans exist in populations which functionally have much lower sizes. Furthermore, drift is always important for a newly arisen allele. Also, drift fluctuations in alleles with a fitness effect can matter even if they never come close to fixing it, as with CFTR–if its high frequency in Europeans is due to drift, that's a big deal for a lot of people.*

4. (3 pts) Give a formula for the expected number of variable sites in a sequence of length L for a sample of size k from a haploid population of size N and mutation rate μ per base per generation. *This is only tractable if we assume infinite sites–that is, mutations don't overwrite each other. If we assume that, then this is the total branch length of the coalescent tree for k samples with scaling factor* $4N\mu$*. The time length of each interval is* $2N/[k(k-1)/2]$ *for that interval's k. There are k branches in the interval, so the total branch length in the interval is* $4N/(k-1)$*. The desired formula is therefore:*

$$4N \times \mu \times L \times \sum_{k=2}^{n-1} \frac{1}{k-1}$$

   This was a lot to ask (it would have been easier a week later!) so I gave credit for answers that varied in the right direction. For example, μ *should not be in the denominator or we're predicting that higher mutation rate means fewer variable sites.*

Long Problem (10 pts, but see below):

This problem was ill posed. I graded it, but will probably not use the grades.

When a region of the genome contains far more variable positions than average, researchers often propose "interesting" explanations: a mutational hotspot, selection for diversity, or introgression (interbreeding) with another species. The null hypothesis to which these must be compared is stochastic variation in coalescent depth or mutation accumulation. This problem will focus on coalescent depth.

A fictional organism has a genome containing 10 segments. Assume that there is no linkage between segments and no recombination within segments, for simplicity.

- What is the probability that one or more of the 10 genomic segments will have a common ancestor at least twice as old as expected?

I needed to specify whether I meant the common ancestor of the samples or the population common ancestor. I meant to ask for the common ancestor of the samples, but that problem is unsolvable without the number of samples. I also apparently was not clear that the segments are different genomic loci, not different samples of the same locus.

If I were solving for the sample common ancestor of a sample of size 10, I would use simulation as follows:

(0) (A trick I should have made clearer) An exponential distribution is entirely described by its mean; they are all the same shape. It therefore doesn't matter what N is. We're just interested in how often a coalescent tree is twice as long as average, and the answer is the same for any (large) N. (With small N the coalescent approximation breaks down.)

Another way of saying this is that a group of random coalescent trees could represent any (large) population equally well: just rescale the branch lengths, like enlarging or reducing them on a photocopier. The shapes and proportions will stay the same.

(1) Calculate the expected length of a tree of k samples as the sum of its expected times in each of the intervals.

(2) Write a function which draws times from the exponential with mean 1.

(3) To simulate one coalescent tree, draw a time for each of its time intervals between coalescents, which will be an exponential draw divided by k(k-1)/2, and sum them all. Do thousands of these, keeping track of the total time for each one.

(4) The proportion of these which are more than twice as long as the expectation in step 1 is the chance that one locus will be twice as long as expected. But we want the chance that at least 1 in 10 is twice as long. We can calculate that or simulate it, but the calculation is easy. If the chance that one locus is twice as long as expected is x, the chance that it is not is $1 - x$, and the chance that none of 10 loci are twice as long is $(1 - x)^10$.

This problem can be solved by calculation, computer simulation, or a mix of the two. If you use calculation, please show your reasoning. If you use computer simulation, please attach at least the key calculation routine of your code.

Helpful hints:

In general. You do not need to track which lineages coalesce for this exercise, only the time it takes them to do so. Also, under the coalescent the answer doesn't depend on N or μ, so you can make life simpler by setting $2N\mu = 1$. (The actual answer does depend on these parameters, but if N is reasonably large and μ is reasonably small, the approximation is extremely good.)

Solving by calculation. Directly calculating the chance that one or more events will happen is often painful due to combinatorics– there are a lot of different ways that, say, 3 out of 10 segments could be high. Instead, note that the chance that something will happen is 1 minus the chance that nothing will happen. In this and many similar problems, the chance that nothing will happen is relatively straightforward to calculate.

Solving by simulation. If your programming language doesn't have a routine to draw randomly from an exponential, but only from a uniform between 0 and 1, the following trick is useful:

exponential draw = -1.0 * log(uniform random draw)

These simulations should be fast, so don't skimp on how many you do. Also, please resist the temptation to use PopG as your simulation engine. (You could use PopG to confirm your results, though: verifying that two independent algorithms agree is a powerful test of software correctness.)