(1) `See separate solution`

(2) (total of 30 pts)

(a) (10 pts) After N trials we have 2^N possible outcomes, the strings from $\{H, T\}$ of length N. Each outcome has probability 2^{-N}. We need to find a_N, the number of outcomes which do not have a repeated T.
Let b_N be the number of such outcomes which end in T and c_N the number which end in H. So $a_N = b_N + c_N$. Clearly $b_1 = c_1 = 1$.
It is also easy to see that $b_{N+1} = c_N$ and $c_{N+1} = b_N + c_N$. So we can compute the b_N and c_N recursively. Actually we can give a recursive formula for a_N.
$a_{N+2} = b_{N+2} + c_{N+2} = b_{N+1} + 2c_{N+1} = (b_{N+1} + c_{N+1}) + (b_N + c_N) = a_{N+1} + a_N$. Since $a_1 = 2$ and $a_2 = 3$ we can compute $a_3 = 5$, $a_4 = 8$ etc. You may recognise the famous "Fibonacci sequence".
Let p_N be the probability of not seeing two tails after N trials, that is $p_N = a_N/2^N$. We get a recursion $p_{N+2} = 1/2p_{N+1} + 1/4p_N$ with starting values $p_1 = 1$, $p_2 = 3/4$.
We can solve this recursion to get an explicit formula for p_N, but we put this off as long as we can.

(b) (10 pts) We need to work out what happens when N becomes large. A little numerical experiment (easy with Maple) suggests that p_N is going to zero: this also looks reasonable from the recursive equation, p_{N+2} is just a little more than half of p_{N+1}.
One approach (out of many): show by induction that $p_M \leq (9/10)^{M-1}$. This is true for $M \leq 2$. For the general step we suppose we have it for $M < N+2$, and then $p_{N+2} \leq 1/2(9/10)^N + 1/4(9/10)^{N-1} = (9/10)^{N-1}(1/4 + 9/20) < (9/10)^{N-1}(9/10)^2 = (9/10)^{N+1}$.

(c) (10 pts) (The tricky bit) If we have an outcome which involves going exactly N steps, it must have the form xTT where x is a TT-free sequence ending in an H or the empty sequence. The number of these sequences is a_{N-2}, where we let $a_0 = 1$. So the probability of going exactly n steps is $q_N = a_{N-2}/2^N$, and easily we can compute q_N by the recursion $q_2 = 1/4$, $q_3 = 1/8$, $q_{N=2} = 1/2q_{N+1} + 1/4q_N$.
Now we bite the bullet and get an explicit formula. Recall from the calculus sequence (the bit about finding series solutions of differential equations to be precise) that the formula for q_N is of the form $X\alpha^N + Y\beta^N$ where α and β are the roots of $\lambda^2 = 1/2\lambda + 1/4$, X and Y are unknown constants. We let $\alpha = (1+\sqrt{5})/4$, $\beta = (1-\sqrt{5})/4$. Solving for X and Y we get

$$Y = \frac{\alpha/4 - 1/8}{\beta^2(\alpha - \beta)}, X = \frac{\beta/4 - 1/8}{\alpha^2(\beta - \alpha)}.$$

The expectation value is $\sum_{N=2}^{\infty} Nq_N$.
Again from calculus we know

$$\frac{1}{(1-x)^2} = 1 + 2x + 3x^2 + \dots$$

so

$$2x^2 + 3x^3 + \dots = \frac{x}{(1-x)^2} - x.$$

Putting it all together the expectation value is

$$X(\frac{\alpha}{(1-\alpha)^2} - \alpha) + Y(\frac{\beta}{(1-\beta)^2} - \beta).$$

Evaluating by hand or with Maple, we find that the answer is 6. [1]

(3) (total of 20 pts) I solve the second part and then set $k = 1$ to solve the first part. Notice that the problem is equivalent to choosing two disjoint sets each of size $N - k$ from the M element set, and then choosing k of the remaining $M - 2N + 2k$ elements.

As we saw in an old homework, the number of ways of choosing the two disjoint sets is

$$\frac{1}{2} \times \binom{M}{N-k} \times \binom{M-N+k}{N-k}$$

So the answer is

$$\frac{1}{2} \times \binom{M}{N-k} \times \binom{M-N+k}{N-k} \times \binom{M-2N+2k}{k}$$

(4) (total of 40 pts)

(a) (10 pts) First suppose that a is a primitive root. Clearly p does not divide a, so in particular $a^i + p\mathbb{Z} \neq 0 + p\mathbb{Z}$ for all $i \geq 0$. By definition the classes $a^i + p\mathbb{Z}$ for $0 \leq i < p-1$ are all distinct.

But there are only $p-1$ classes of the form $b + p\mathbb{Z}$ for b not a multiple of p, namely the classes $k + p\mathbb{Z}$ for $0 < k < p$. So every such class is of the form $a^i + p\mathbb{Z}$ for a suitable i. It follows that for every n not divisible by p there is i such that $n + p\mathbb{Z} = a^i + p\mathbb{Z}$, or equivalently $a^i \equiv n \mod p$.

Now conversely suppose that every n not divisible by p is congruent to some power of a. Note that p does not divide a, so $a^{p-1} \equiv 1 \mod p$. Suppose (for a contradiction) that we have $i < j < p-1$ such that $a^i \equiv a^j \mod p$. Multiplying by a^{p+1-i} we get $a^{j-i} \equiv 1$. An easy induction shows that if $N = j - i$ then $a^k \equiv a^{k+N}$ so the sequence of classes $a^k + p\mathbb{Z}$ is periodic with period N; since $N < p-1$ this contradicts our assumption on a, since there are $p-1$ possibilities for the class $n + p\mathbb{Z}$. So a must be a primitive root.

(b) (10 pts) If $j \equiv i \mod p-1$ then let $j - i = N(p-1)$. So $a^{j-i} = (a^{p-1})^N \equiv 1 \mod p$ (since $a^{p-1} \equiv 1 \mod p$ by Fermat) and thus $a^j \equiv a^i \mod p$.

Conversely let $a^j \equiv a^i \mod p$ and let $j - i = N(p-1) + r$ for $0 \leq r < p-1$. As above we can argue that $a^{j-i} \equiv a^r \equiv 1$. Now r must be equal to zero, for otherwise the sequence of classes $a^k + p\mathbb{Z}$ would be periodic with period $r < p-1$ contradicting the assumption that a is a primitive root.

(c) (10 pts) Argument one: notice that $a^2 \equiv b^2 \mod p$ if and only if p divides $a - b$ or $a + b$, that is if and only if $a \equiv b \mod p$ or $a \equiv -b \mod p$. So if we consider the sequence $1^2 \mod p$, $2^2 \mod p$, $\ldots (p-1)^2 \mod p$ then the first $(p-1)/2$ values are distinct and the

[1] There ought to be a slick proof of this but I have been unable to find it.

second $(p-1)/2$ values repeat the first $(p-1)/2$ values in the opposite order.
Example: when $p = 7$ we get $1, 4, 9 \equiv 2, 16 \equiv 2, 25 \equiv 4, 36 \equiv 1$.

Argument two: let a be a primitive root. Then every even power of a gives a quadratic residue. Conversely if a^i is a quadratic residue then $a^i \equiv a^{2j} \mod p$ for some j, so $i \equiv 2j \mod p-1$ and i is even. The powers $a^0, a^2, \ldots a^{p-3}$ give us $(p-1)/2$ quadratic residues.
Example: 3 is a primitive root mod 7. Now $3^0 = 1$, $3^2 \equiv 2$, $3^4 \equiv 4$ gives us the three residues.

(d) (10 pts) A little computation shows that 3 is a primitive root. So the residues are $3^0 = 1$, $3^2 = 9$, $3^4 \equiv 13$, $3^6 \equiv 15$, $3^8 \equiv 16$, $3^{10} \equiv 8$, $3^{12} \equiv 4$, $3^{14} \equiv 2$.
As a double check we do it the other way: the residues should be $1^2 = 1$, $2^2 = 4$, $3^2 = 9$, $4^2 \equiv 16$, $5^2 \equiv 8$, $6^2 \equiv 2$, $7^2 \equiv 15$, $8^2 \equiv 13$.

(5) (20 pts total)

(a) (10 pts) The 50 digit numbers are those with $10^{49} \le n < 10^{50}$. So the estimate from the PNT is $10^{50}/ln(10^{50}) - 10^{49}/ln(10^{49})$. Evaluating with Maple we see that there are roughly 7.8×10^{47} 50 digit primes.

(b) (10 pts) The Maple code

```
count:=0;
for n from 10^60 to 10^60 + 10^ 5 do
if( is prime(n) ) then count:=count+1; end if; end do;
count;
```

counts the primes. There are 720 of them.
If we attempt to evaluate $(10^{60} + 10^5)/ln(10^{60} + 10^5) - 10^{60}/ln(10^{60})$ with Maple we get zero. This is not surprising: each of the numbers whose difference we are computing is in the neighbourhood of 10^{58}, and we are looking for a difference in the neighbourhood of 10^3, the problem is that we are not working to a high enough precision.
To be on the safe side we compute to 70 digit precision.

```
evalf( (10^{60} + 10^5)/ln(10^{60} + 10^5) - 10^{60}/ln(10^{60}), 70 );
```

gives an answer about 718.585. The PNT seems to do pretty well!

(6) (35 pts total)

(a) (10 pts) The Maple code

```
t0:=time();
myrand:=rand(10^9..10^10-1);
for i from 1 to 100 do
n:=myrand():
ifactor(n):
end do:
time()-t0;
```

run on my desktop machine (800 MHz Pentium III, 512M RAM) uses about 0.2 seconds of CPU time.

(b) (10 pts) Here are the results of some trials I ran using similar code to that above with more digits. n is the number of digits, t is the total CPU time used to factor 100 random n digit numbers. I have also recorded $ln(t)$.
$n = 10$, $t = 0.163$, $ln(t) = -1.81$

$n = 12$, $t = 0.489$, $ln(t) = -0.71$
$n = 14$, $t = 0.767$, $ln(t) = -0.27$
$n = 16$, $t = 2.35$, $ln(t) = 0.85$
$n = 18$, $t = 3.39$, $ln(t) = 1.22$
$n = 20$, $t = 4.4$, $ln(t) = 1.48$
$n = 22$, $t = 11.43$, $ln(t) = 2.43$
$n = 24$, $t = 13.89$, $ln(t) = 2.63$
. $n = 26$, $t = 27.9$, $ln(t) = 3.32$
$n = 28$, $t = 54.7$, $ln(t) = 4.00$
$n = 30$, $t = 77.7$, $ln(t) = 4.35$
$n = 32$, $t = 178.3$, $ln(t) = 5.18$
$n = 34$, $t = 287$, $ln(t) = 5.65$.
$n = 36$, $t = 456$, $ln(t) = 6.12$.
$n = 38$, $t = 1240$, $ln(t) = 7.12$.

Sketching a graph we see that we can fit the data quite well by $ln(t) = n/4 - 3$.

Using this prediction we expect that when $n = 100$, $ln(t)$ is about 22 and it will take about 4×10^7 seconds to factorise a random 100 digit number. There are about 3×10^7 seconds in a year.

This calculation is (more than) a bit misleading. As you may have seen from watching your calculations proceed, some numbers take a lot longer than others. The worst 100 digit numbers (products of tow 50 digit primes) will probably take centuries using the Maple factoring algorithm. However the probability of finding a really bad number is quite low.

(c) (5 pts) If we take the code above and replace colons by semicolons, Maple will print each number being generated and its factorisation. A little observation shows that the slowest numbers are the ones with a small number of prime factors, at least two of which are large. As one might expect, the very slowest numbers are the products of two primes of roughly equal size.

(7) (20 pts total)

(a) (10 pts) Let f have degree m and g have degree n, say $f = \sum_{i=0}^{m} a_i x^i$ and $g = \sum_{j=0}^{n} b_j x^j$ where a_m and b_n are nonzero. We can write

$$\frac{g}{x^{n-m} f} = \frac{\sum_{j=0}^{n} b_j x^{j-n}}{\sum_{i=0}^{m} a_i x^{i-m}}$$

As x increases, this expression approaches the value b_n/a_m. Since f and g take values in $\mathbb{N}$ for $x \in \mathbb{N}$, $b_n/a_m > 0$. For sufficiently large x we see that $g/(x^{n-m} f) > b_n/(2a_m)$, that is $g/f > (b_n x^{n-m})/(2a_m)$. So when x is large enough $g/f > 1$, that is $g(x) > f(x)$.

(b) (10 pts) Let $g(n) = \sum_{i<n} f_i(n) + 1$. Then for $n > i$ we have $g(n) \geq f_i(n) + 1 > f_i(n)$, so $g >^* f_i$.

(8) (20 pts total) The program

```
1:  R_1=1
2:  IF(R_2 = 0) THEN STOP
3:  R_2 = R_2 -1
4:  R_1 = R_1 + R_1
```

```
5    GOTO 2
```

will, if started with n in R_2, stop with 2^n in R_1. It is is easy to see that it runs for $4n+2$ steps on argument n.