PRG Theory

A PRG is defined as follows: PRG is a efficiently computable function which elongates a given seed of length n to $l(n)$ where $l(n) > n$ using a seed r.
This PRG is considered as secure if for all efficient adversaries A, there is a negligible function $negl(n)$ such that

$$Pr[A(G(s)) = 1 \,|\, s < -(0,1)n] - Pr[A(z) = 1 \,|\, z < -(0,1)l(n)] <= negl(n).$$

This equation suggests that any adversary A shouldn't be able to distinguish between a number chosen from a real random source and a PRG with a probability not greater than a negligible value.
The hard problem to assume was of discrete log, which has the following setup :
$y = g^x \ mod \ p$ where p is a large prime number of length n (In binary), and g is the generator of p. A generator of a cyclic group of the form Z_p^* is one which generates all the numbers in the group when performing modulo exponentiation with different powers. Thus in this question $p = 397597169$, $g = 3$ were chosen.
Then, a pseudorandom generator G can be formed with $G =< f(x_1), x_2, h_c(x) >$. In this, the $f(x)$ is the one way function chosen, which is the modular exponentiation and h_c is the hardcore predicate of the function f. Further, x_1 and x_2 are the first half and second half of the input seed x.
The hardcore predicate used in the assignment is the MSB (Most Significant Bit) of x. Thus the generator G returns a $N + 1$ length string. (The modular exponentiation is padded with zeros till it is the same length as the prime p). This generator can be easily proven to be pseudo random as discrete log is assumed to be a hard problem and MSB is the hardcore predicate of discrete log. Thus for it to break, the discrete log has to be solved, which is hard. Hence, breaking this pseudo random generator implies breaking the discrete log problem.
The above procedure is repeatedly performed to form strings of arbitrary length in the following way: $< h_c(x), h_c(f(x_1), x_2), h_c(f(f(x_1), x_2), x_2), \ldots >$
This can be proven to be one way by building on the previous construction. Since $f(x)$ is one way, $f(f(\ldots(x)))$ is also one way. The hardcore predicate is chosen in such a way that this one-way-ness is maintained. Thus the produced result is pseudo-random.
As the h_c for the discrete log is taken to be MSB, each computation of $f(x)$ results in a single output length increase and thus needs to be looped n times for the length

required.