Statistical Computational Methods - Exam

A computer virus can damage a file with probability 0.2, independently of other files. A computer manager checks the condition of important files. Conduct a Monte Carlo study to estimate:

a) the probability that the manager has to check 8 files in order to find 3 damaged ones;

b) the expected number of clean files found before finding the third damaged one.

Compare your results with the exact values.

```
format longg;

p = 0.2;
function N=computeSimulationsCount()
    err = input('error = '); % maximum error
    alpha = input('alpha (level of significance) = '); % significance level

    % Compute MC size to ensure that the error is < err, with confidence
level 1 - alpha
    N = ceil(0.25 * (norminv(alpha / 2, 0, 1) / err)^2);
    fprintf('Nr. of simulations N = %d\n', N);
end
N = computeSimulationsCount();
```

```
Nr. of simulations N = 41059361
```

```
% a) The manager has to check 8 files in order to find 3 damaged one.
% Events:
% X - clean files files checked until finding the third damaged one
% - X failures until the nth (third) success, probability of success p
% -> Negative Binomial Distribution
% - success -> damaged file -> probability p
% We need to find P(X == 5)

n = 3;
X = zeros(1, N);
for i = 1:N
    Y = ceil(log(1 - rand(n, 1))/log(1 - p) - 1); % Geo variables
    X(i) = sum(Y);
end
X
```

```
X = 1×41059361
    12    18    11    11    15     5     9    34     7     1    24     2    14···
```

```
fprintf('simulated probab. P(X = 5) = %g', mean(X==5));
```

```
simulated probab. P(X = 5) = 0.0550431
```

```
fprintf('true probab. P(X = 5) = %g', nbinpdf(5, n, p));
```

```
true probab. P(X = 5) = 0.0550502
```

```
fprintf('error= %e', abs(nbinpdf(5, n, p)) - mean(X==5));
```

```
error= 7.176860e-06
```

```
% b) The number of clean files found before finding the third damaged one
% Events:
% X - clean files found before the nth damaged file
% - X failures before nth success
% -> Negative Binomial Distribution
% - success -> damaged file -> probability p
% We need to find E(X)
fprintf('simulated probab. E(X) = %g', mean(X));
```

```
simulated probab. E(X) = 11.9994
```

```
fprintf('true probab. E(X) = %g', (n * (1 - p) / p));
```

```
true probab. E(X) = 12
```

```
fprintf('error= %e', abs(n * (1 - p) / p - mean(X)));
```

```
error= 5.765555e-04
```