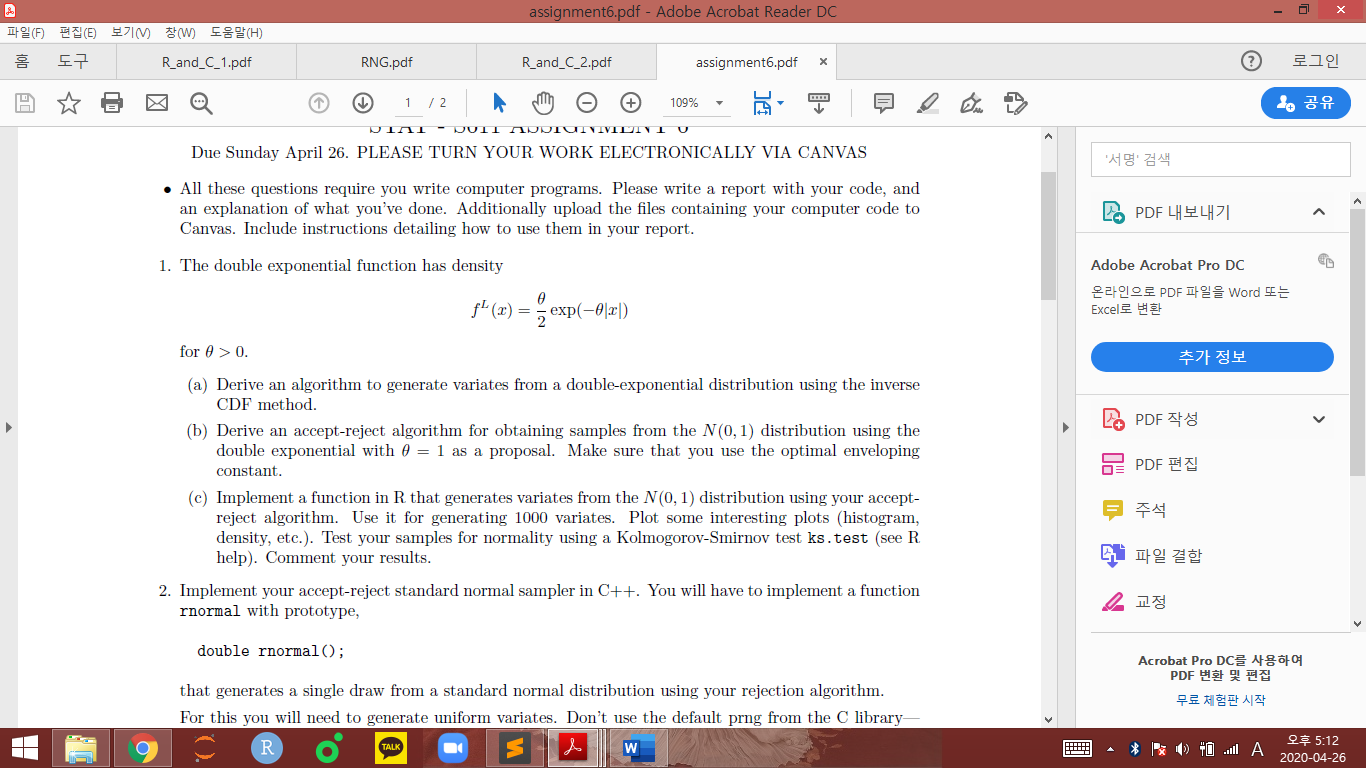
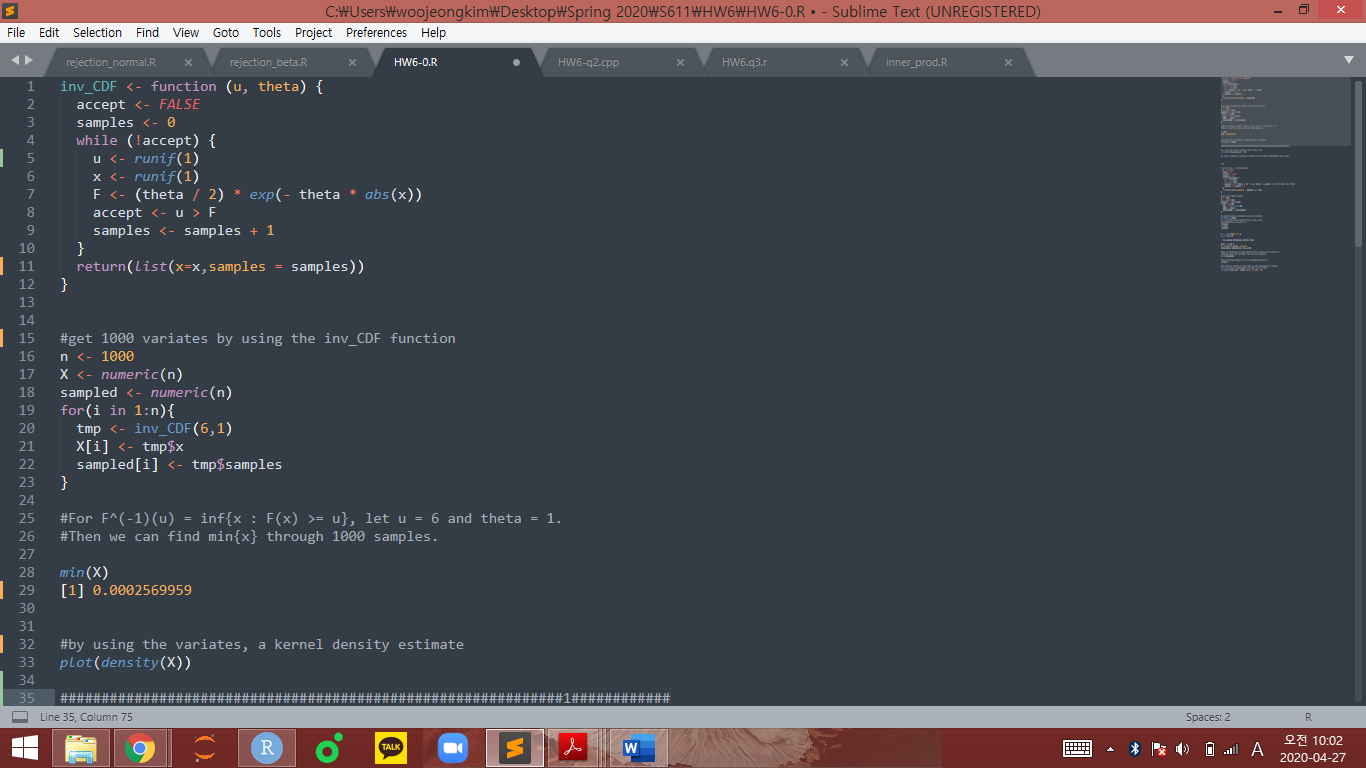
Q1.



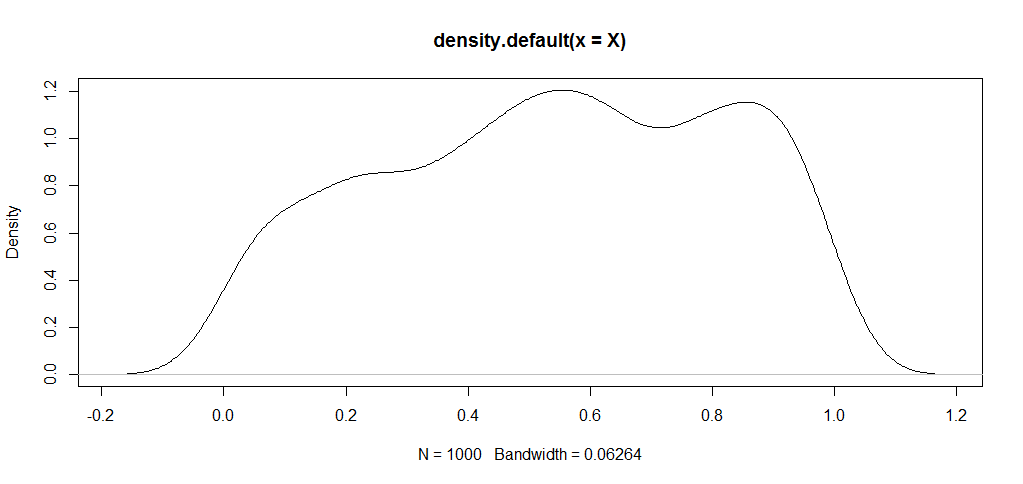
(a) Derive an algorithm to generate variates from a double-exponential distribution using the inverse CDF method.

First, The inverse CDF Method is the formula to find minimal x among the variables that satisfy F(x) >=u where u follows uniform distribution on interval [0, 1]. To derive this algorithm, we give the uniform distribution to the x and u and write the function of exponential function as F. Since the character “accept” is assign as FALSE, we can take loop for getting the values of x that satisfies until x gets to the value satisfying u > F(x). Also, by giving the sample size, we can adjust sample size as well. As the results, we can get values of list that consists of the x satisfying F(x) >= u and sample size.(1:12 line in the figure below)

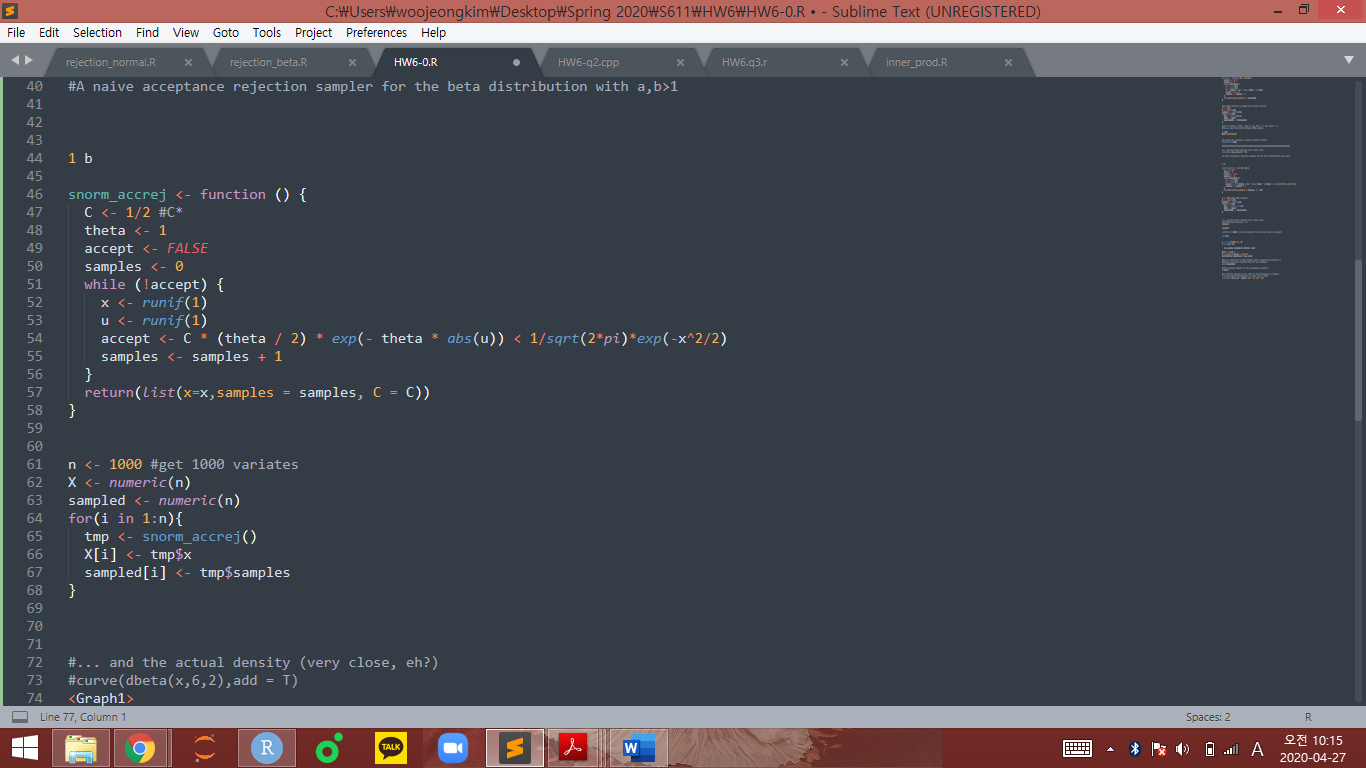
Second, We can use the function to get the variables in determinated value. By setting n as sample size we want and X as the recording vector of a value tmp from each operating of loop, we can get final vector X consisting of the x satisgying F(x) >= u.(15:23 line in the figure below)



Third, we can get the minimal value among the values in each index of X. Finally, we can get the minimal x satisfying our formula of inverse CDF method. By using the values of X, we also draw the density of X. as follows.



(b) Derive an accept-reject algorithm for obtaining samples from the N(0; 1) distribution using the double exponential with thesta = 1 as a proposal. Make sure that you use the optimal enveloping constant.

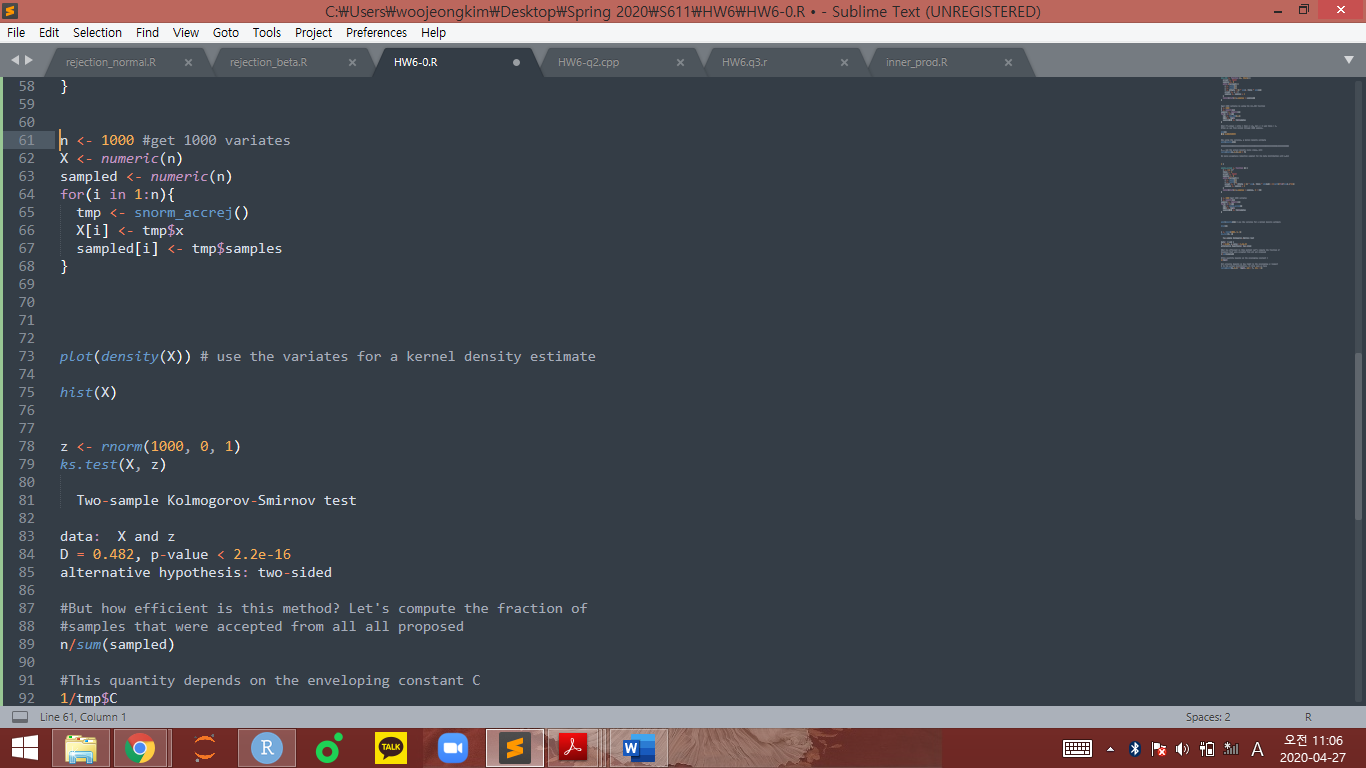


First, for constructing function “snorm\_accrej”, the logic is same as in (a). This is because we should find variable x satisfying cU>= f(x) where U is variable following Uniform distribution and f(x) double-exponential distribution with theta = 1. Therefore, the key idea for constructing function is set while statement to get variable x satisfies cU>=f(x) until x gets to the variable satisfying cU<f(x), which is FALSE statement by setting “accept”. Therefore, we can get variables x by comparing f(x) and cU where U is well known function as proposal function U.

The different thing from (a) is that we should find optimal enveloping constant c derived from the formula c= sup\_x{f(x)/g(x)} where f(x) is standard normal distribution and g(x) is double exponential distribution with theta = 1. By computing the ratio of distribution functions c = sup\_x({sqrt{2}/pi}exp(-x^2/2 + abs(x))). For abs(x)=1, the optimal value c^\* should be 1/2. Therefore, we can plug c =1/2 as line 47 in the figure above.

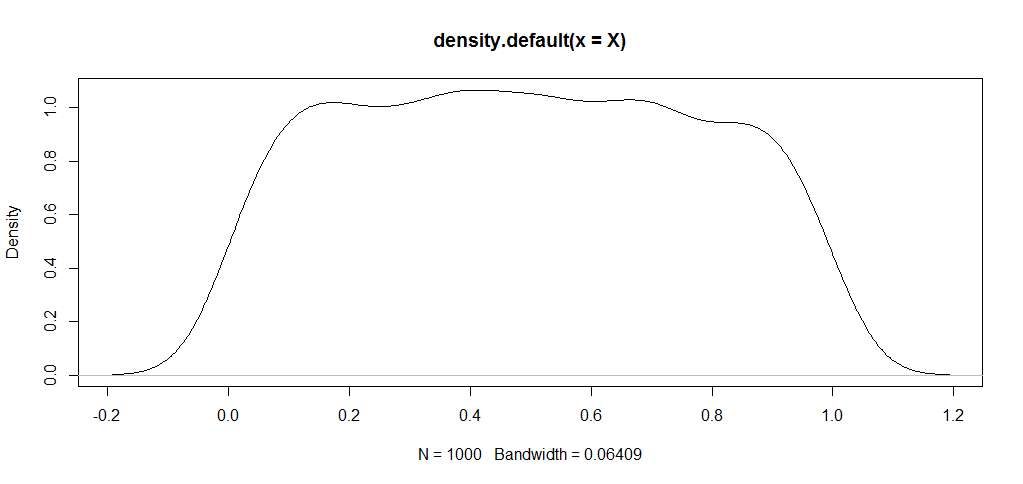
(c) Implement a function in R that generates variates from the N(0; 1) distribution using your accept-reject algorithm. Use it for generating 1000 variates. Plot some interesting plots (histogram, density, etc.). Test your samples for normality using a Kolmogorov-Smirnov test ks.test (see R help). Comment your results.

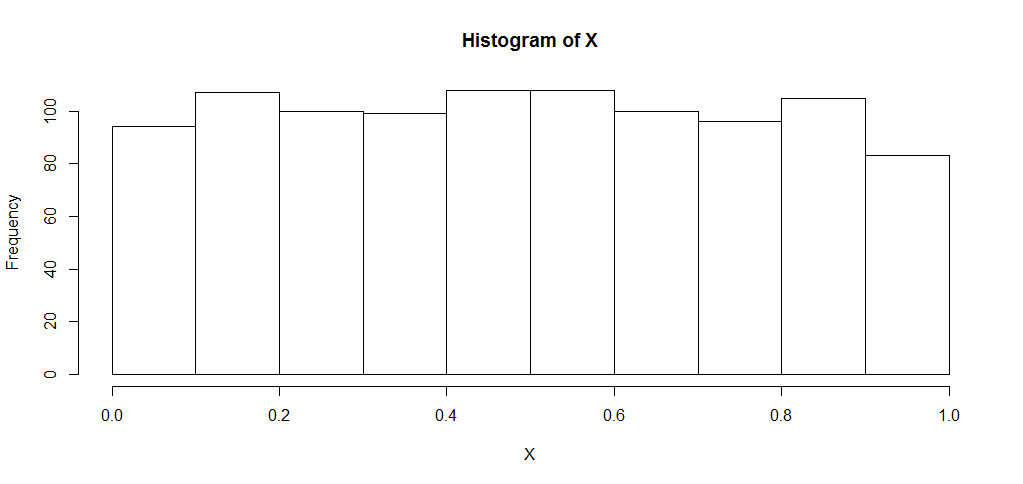
As shown in the (b), we can get sample size n=1000 by using the function “snorm\_accrej”. For size n sampled vector, operate “snorm\_accrej” function and record it as “tmp” list of variables and tmp$x is recorded on vector X in each index i while i is in {1, 2, …, n=1000}. Also, for each operating of the for statement, we can give the counting number of operating to sampled vector from the samples lavel of tmp vector for each operating.(line 61:68 in the figure below)



As a result, we can get x satisfying cU>=f(x) for each 1000 times of operating “for” statement as I mentioned. Therefore, we can plot the variable x as line 73 and the figure is shown in the below. Also, by setting vector z consisting of 1000 variables following normal distribution, we can compare the values in vector X and z by ks. test function.(line 78:85 in the figure above) As we can see in the figure, the alternative hypothesis is chosen from the test. This is because kt.test function indicates the alternative hypothesis and must be one of "two.sided" (default), "less", or "greater" in its comment in result. Since p-value is smaller than 2.2e-16, the means of X and z are tested to be different from kt.test.

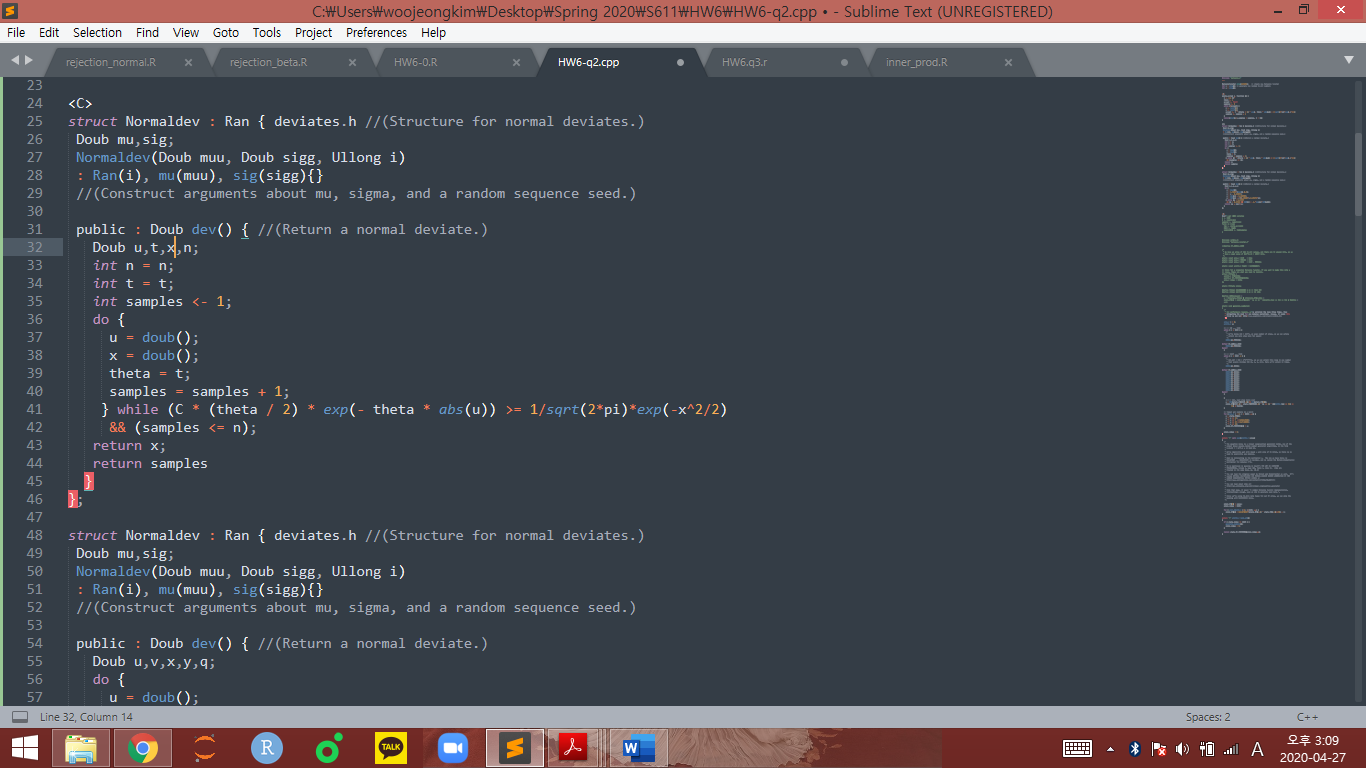
Finally, the plot and histogram of values of X is as follows.





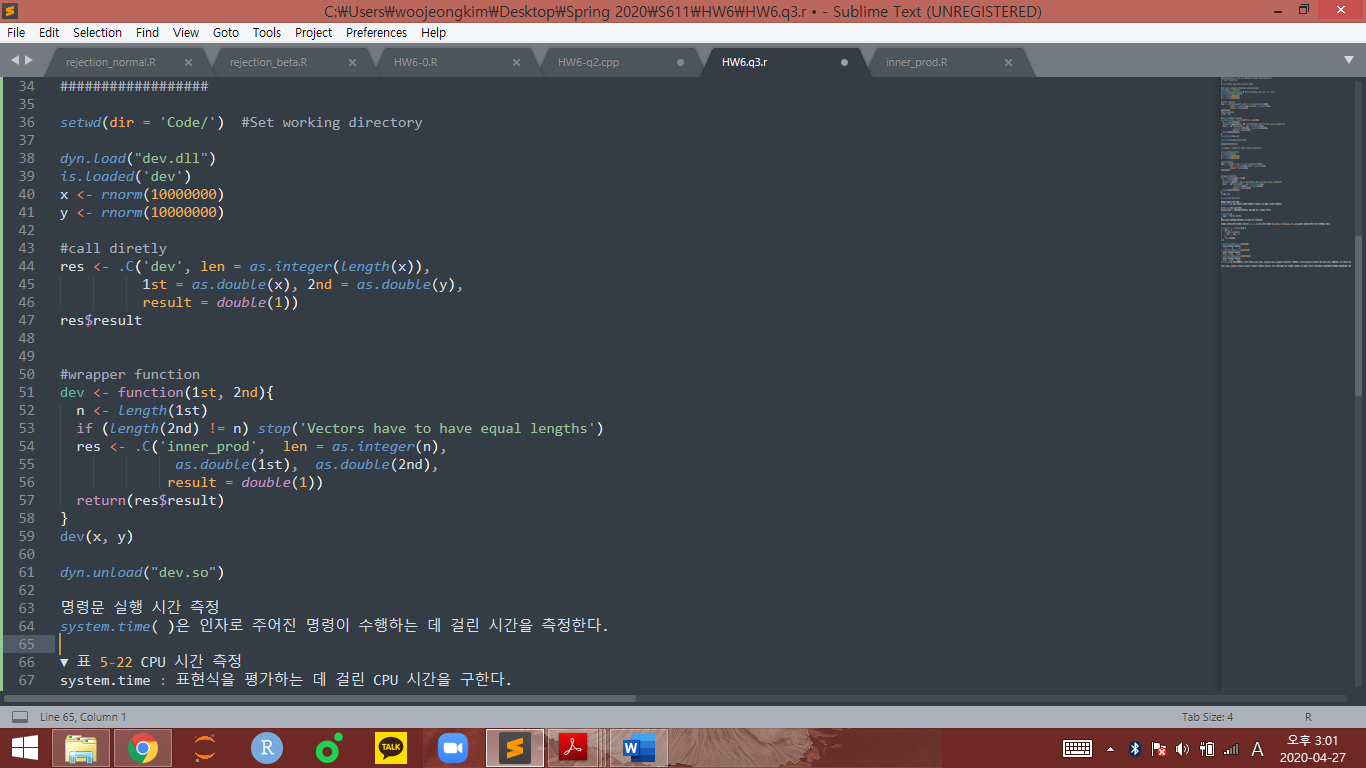
Q2. Implement your accept-reject standard normal sampler in C++. You will have to implement a function rnormal with prototype,

The concept to make algorithm is same as Q1. By set the Normaldev structure, we can get the normal variates.(line25:29) After that, in the same structure, we set the public function by using “public”. The variables “u” and “x” can be made by doub() as normal variables. By using them, we can make while statement. In the while statement, the condition “f(x)<cg(x)” where f(x) is normal distribution and c=1/2 and g(x) is proposal function as exponential distribution as Q1. Also, sample size condition “samples<=n” is in the while statement.



Q3. Now you will write an R interface to your C++ code.

As shown in the figure below, we set the directory in R to use the C code in Q2. This will create the shared library \main\_le.so" (linux/mac), or \main\_le.dll" (windows).(line 36:39) After the setting, we call the function “dev” in C directly. Also, by using the wrapper function as the line 50:59, we can apply the function



Also, for measuring the time by the system.time function, wrap up the function in Q1 as “wrap\_snorm\_accrej” with sample size n=1000000 as follows.



Finally, we can use the system.time function as follows to measure the operating time of same works in R and in C.

