Case Study:

Write a program in Python to,

(a) generate 3 streams (A, B, C) of 1,000,000 independent random number simulations following standard normal distribution,

(b) calculate 3 streams of correlated random numbers by multiplying these 3 streams of independent correlated random numbers with a 3*3 correlation matrix given below.

	A	B	C
A	100%	75%	50%
B	75%	100%	75%
C	50%	75%	100%

(c) Define and set correlation of A (R_A) = 10%, R_B = 25%, R_C = 40%.

(d) For each simulation, define value of A (V_A) and calculate V_A = SQRT(1 - R_A**2) * Independent random number from (a) + R_A * Correlated random number from (b) for the entire 1,000,000 simulations. Do the same for B and C. [Check: Resultant array would be of the size of [3 columns x 1,000,000 rows]

(e) Define Probability of Default of A (PD_A) = 1%, PD_B = 10% and PD_C = 50%

(f) Calculate a default threshold using the formula for A, B and C. Default threshold of A (DT_A) = Inverse of standard normal cumulative distribution of (PD_A) [Hint: Excel formula is NORMSINV(PD)]. Do the same for B and C.

(g) Define and set loss of A (L_A) = 0. In each simulation, if Value(V_A) < default threshold (DT_A), add a counter to loss. Calculate the loss for B(L_B) and C(L_C) separately. Calculate the total loss (T) = (L_A + L_B + L_C).

Outputs required:

1. Number of simulations generated for A, B, C
2. Loss under A, B and C (L_A, L_B and L_C) and Total Loss (T) with time taken to execute (with single threading)
3. Loss under A, B and C (L_A, L_B and L_C) and Total Loss (T) with time taken to execute (with multi-threading that can split the 1,000,000 calculations into multiple chunks to speed up).