C. Monte Carlo 101

a) After compiling the program, I observed that TestMC.cpp correctly implemented the idea of Monte Carlo simulation, using NormalGenerator to mock the Wiener process. And after using the number of subintervals = 100 and the number of simulations =50000 by default, we obtain the price after discounting = 5.87749 for the batch 1 put option, which is similar to the exact answer 5.84628.

b) After running the MC program with data from batches 1 and 2 with different time steps NT and simulations NSIM, I found that as the time steps increase and as the number of simulations increases, the approximation becomes generally more accurate, but not always, because our simulation is random. And we can achieve 3-digit accuracy when NT=100 and NSIM=200000 for the Batch 2 Put Option.

Batch 1 Call:

NT	NSIM	Approximation	Exact Value	Absolute Error	Relative Error
100	50000	2.11675	2.13337	-0.01662	-0.007790491101
200	50000	2.14956	2.13337	0.01619	0.007588932065
300	50000	2.16432	2.13337	0.03095	0.01450756315
400	50000	2.15482	2.13337	0.02145	0.01005451469
500	50000	2.16105	2.13337	0.02768	0.012974777
100	100000	2.13043	2.13337	-0.00294	-0.001378101314
100	150000	2.13831	2.13337	0.00494	0.002315585201
100	200000	2.13851	2.13337	0.00514	0.00240933359
100	250000	2.13689	2.13337	0.00352	0.001649971641

Batch 1 Put:

NT	NSIM	Approximation	Exact Value	Absolute Error	Relative Error
100	50000	5.87749	5.84628	0.03121	0.005338437434
200	50000	5.89724	5.84628	0.05096	0.008716654009
300	50000	5.87223	5.84628	0.02595	0.004438720007
400	50000	5.85702	5.84628	0.01074	0.001837065621
500	50000	5.84315	5.84628	-0.00313	-0.0005353831838
100	100000	5.87321	5.84628	0.02693	0.004606347968
100	150000	5.85201	5.84628	0.00573	0.0009801104292
100	200000	5.8475	5.84628	0.00122	0.0002086797074
100	250000	5.83973	5.84628	-0.00655	-0.00112037056

Batch 2 Call:

NT	NSIM	Approximation	Exact Value	Absolute Error	Relative Error
100	50000	7.90889	7.96557	-0.05668	-0.007115623866
200	50000	7.99439	7.96557	0.02882	0.003618071274
300	50000	8.0428	7.96557	0.07723	0.009695476909
400	50000	8.00341	7.96557	0.03784	0.004750444726
500	50000	8.04316	7.96557	0.07759	0.009740671415
100	100000	7.94362	7.96557	-0.02195	-0.002755609454
100	150000	7.97081	7.96557	0.00524	0.0006578311408
100	200000	7.97462	7.96557	0.00905	0.001136139661
100	250000	7.97386	7.96557	0.00829	0.001040729038

Batch 2 Put:

NT	NSIM	Approximation	Exact Value	Absolute Error	Relative Error
100	50000	8.0132	7.96557	0.04763	0.005979484205
200	50000	8.06316	7.96557	0.09759	0.0122514773
300	50000	8.02559	7.96557	0.06002	0.007534928448
400	50000	7.97708	7.96557	0.01151	0.001444968784
500	50000	7.97047	7.96557	0.0049	0.0006151474408
100	100000	8.0079	7.96557	0.04233	0.005314120647
100	150000	7.97154	7.96557	0.00597	0.0007494755554
100	200000	7.96548	7.96557	-0.00009	-0.00001129862646
100	250000	7.95213	7.96557	-0.01344	-0.001687261552

c) Doing the stress test on batch 4, I found that if the NT are small, for example, 100, increasing the NSIM does not improve the accuracy since the T =30 in Batch 4 is large. But keep NSIM constant, increasing NT generally improves the accuracy of the approximation. When I set NT=800, and NSIM=500000, we achieve 2-digit accuracy in put option

Batch 4 Call:

NT	NSIM	Approximation	Exact Value	Absolute Error	Relative Error
100	50000	89.6513	92.1757	-2.5244	-0.02738682755
200	50000	92.9557	92.1757	0.78	0.008462100098
300	50000	91.8207	92.1757	-0.355	-0.003851340429
400	50000	92.0019	92.1757	-0.1738	-0.001885529483
500	50000	94.3209	92.1757	2.1452	0.02327294504
100	100000	89.4248	92.1757	-2.7509	-0.02984409123
100	150000	89.5599	92.1757	-2.6158	-0.0283784121
100	200000	89.4562	92.1757	-2.7195	-0.02950343746
100	250000	89.534	92.1757	-2.6417	-0.02865939722

Batch 4 Put:

NT	NSIM	Approximation	Exact Value	Absolute Error	Relative Error
100	50000	1.29981	1.2475	0.05231	0.04193186373
200	50000	1.29202	1.2475	0.04452	0.03568737475
300	50000	1.27906	1.2475	0.03156	0.02529859719
400	50000	1.2683	1.2475	0.0208	0.01667334669
500	50000	1.25449	1.2475	0.00699	0.005603206413
100	100000	1.29604	1.2475	0.04854	0.03890981964
100	150000	1.28795	1.2475	0.04045	0.0324248497
100	200000	1.28854	1.2475	0.04104	0.03289779559
100	250000	1.28496	1.2475	0.03746	0.03002805611
800	500000	1.2493	1.2475	0.0018	0.001442885772

D. Advanced Monte Carlo

a) Implemented in main.cpp

b) Testing on batch 1 and 2, I find that as NT increases, the SD and SE basically remain at the same level, and as NSIM increases, SD does not change a lot, but SE decreases, which makes sense from the formula SE =SD/ sqrt(NSIM).

Batch 1 call:

NT	NSIM	Approximation	Exact Value	SD	SE
100	50000	2.11675	2.13337	4.97727	0.022259
200	50000	2.14956	2.13337	5.04605	0.0225666
300	50000	2.16432	2.13337	5.04173	0.0225473
100	100000	2.13043	2.13337	4.99057	0.0157816
100	150000	2.13831	2.13337	5.00194	0.0129149

Batch 1 put:

NT	NSIM	Approximation	Exact Value	SD	SE
100	50000	5.87749	5.84628	8.4433	0.0377596
200	50000	5.89724	5.84628	8.4728	0.0378915
300	50000	5.87223	5.84628	8.44169	0.0377524
100	100000	5.87321	5.84628	8.43749	0.0266817
100	150000	5.85201	5.84628	8.41344	0.0217234

Batch 2 call:

NT	NSIM	Approximation	Exact Value	SD	SE
100	50000	7.90889	7.96557	11.4143	0.036095
200	50000	7.99439	7.96557	11.5414	0.0364971
300	50000	8.0428	7.96557	11.545	0.0365084
100	100000	7.94362	7.96557	11.443	0.0255873
100	150000	7.97081	7.96557	11.4668	0.0209353

Batch 2 put:

NT	NSIM	Approximation	Exact Value	SD	SE
100	50000	8.0132	7.96557	11.1256	0.0351822
200	50000	8.06316	7.96557	11.1713	0.0353269
300	50000	8.02559	7.96557	11.1223	0.0351719
100	100000	8.0079	7.96557	11.116	0.0248561
100	150000	7.97154	7.96557	11.0798	0.0202288