



ภาคผนวก

ภาษาคอมพิวเตอร์ (Visual Basic 6.0) ที่ใช้ในการสร้างค่าตัวแปรสุ่ม 5 วิธีและนำไปสร้างข้อมูลที่มีการ

แจกแจง 4 แบบ คือ การแจกแจงแบบพัวซอง การแจกแจงแบบทวินาม

การแจกแจงแบบเอ็กซ์โปเนนเชียล และการแจกแจงแบบปกติ

Option Explicit

Public a, b, z, answer, answer5, i, j, k, x1, beta, x_expo, s, parameter, m_error,
loop_test As Variant

Public s_m, est_para, mse_error As Variant

Public x1_3, x1_8, x2_0, x2_4, x2_9 As Variant

Public s1_3, s1_8, s2_0, s2_4, s2_9 As Variant

Public r, p, F, x_poisson As Variant

Public x, a5, w, m, sample, g As Variant

Public m11_3, m11_8, m12_0, m12_4, m12_9, m21_3, m21_8, m22_0, m22_4,
m22_9 As Variant

Public m31_3, m31_8, m32_0, m32_4, m32_9, m41_3, m41_8, m42_0, m42_4,
m42_9 As Variant

Public m51_3, m51_8, m52_0, m52_4, m52_9 As Variant

Public m1_expo1, m1_expo2, m1_expo3, m1_expo4, m1_expo5 As Variant

Public m2_expo1, m2_expo2, m2_expo3, m2_expo4, m2_expo5 As Variant

Public m3_expo1, m3_expo2, m3_expo3, m3_expo4, m3_expo5 As Variant

Public m4_expo1, m4_expo2, m4_expo3, m4_expo4, m4_expo5 As Variant

Public m5_expo1, m5_expo2, m5_expo3, m5_expo4, m5_expo5 As Variant

Public m1s1_3, m1s1_8, m1s2_0, m1s2_4, m1s2_9 As Variant

Public m2s1_3, m2s1_8, m2s2_0, m2s2_4, m2s2_9 As Variant

Public m3s1_3, m3s1_8, m3s2_0, m3s2_4, m3s2_9 As Variant

Public m4s1_3, m4s1_8, m4s2_0, m4s2_4, m4s2_9 As Variant

Public m5s1_3, m5s1_8, m5s2_0, m5s2_4, m5s2_9 As Variant

Public ep11_3, ep11_8, ep12_0, ep12_4, ep12_9 As Variant

Public ep21_3, ep21_8, ep22_0, ep22_4, ep22_9 As Variant

Public ep31_3, ep31_8, ep32_0, ep32_4, ep32_9 As Variant

Public ep41_3, ep41_8, ep42_0, ep42_4, ep42_9 As Variant

Public ep51_3, ep51_8, ep52_0, ep52_4, ep52_9 As Variant

Public s_m11_3, s_m11_8, s_m12_0, s_m12_4, s_m12_9 As Variant

Public s_m21_3, s_m21_8, s_m22_0, s_m22_4, s_m22_9 As Variant

Public s_m31_3, s_m31_8, s_m32_0, s_m32_4, s_m32_9 As Variant

Public s_m41_3, s_m41_8, s_m42_0, s_m42_4, s_m42_9 As Variant

Public s_m51_3, s_m51_8, s_m52_0, s_m52_4, s_m52_9 As Variant

Public mlem11_3, mlem11_8, mlem12_0, mlem12_4, mlem12_9 As Variant

Public mlem21_3, mlem21_8, mlem22_0, mlem22_4, mlem22_9 As Variant

Public mlem31_3, mlem31_8, mlem32_0, mlem32_4, mlem32_9 As Variant

Public mlem41_3, mlem41_8, mlem42_0, mlem42_4, mlem42_9 As Variant

Public mlem51_3, mlem51_8, mlem52_0, mlem52_4, mlem52_9 As Variant

Public rmlem11_3, rmlem11_8, rmlem12_0, rmlem12_4, rmlem12_9 As Variant
Public rmlem21_3, rmlem21_8, rmlem22_0, rmlem22_4, rmlem22_9 As Variant
Public rmlem31_3, rmlem31_8, rmlem32_0, rmlem32_4, rmlem32_9 As Variant
Public rmlem41_3, rmlem41_8, rmlem42_0, rmlem42_4, rmlem42_9 As Variant
Public rmlem51_3, rmlem51_8, rmlem52_0, rmlem52_4, rmlem52_9 As Variant
Dim m1(0 To 5000), m2(0 To 5000), m3(0 To 5000), m4(0 To 5000), m5(0 To 5000)
As Variant

Dim normal1_01(0 To 5000), normal1_13(0 To 5000), normal1_25(0 To 5000),
normal1_36(0 To 5000), normal1_45(0 To 5000) As Variant

Dim normal2_01(0 To 5000), normal2_13(0 To 5000), normal2_25(0 To 5000),
normal2_36(0 To 5000), normal2_45(0 To 5000) As Variant

Dim normal3_01(0 To 5000), normal3_13(0 To 5000), normal3_25(0 To 5000),
normal3_36(0 To 5000), normal3_45(0 To 5000) As Variant

Dim normal4_01(0 To 5000), normal4_13(0 To 5000), normal4_25(0 To 5000),
normal4_36(0 To 5000), normal4_45(0 To 5000) As Variant

Dim normal5_01(0 To 5000), normal5_13(0 To 5000), normal5_25(0 To 5000),
normal5_36(0 To 5000), normal5_45(0 To 5000) As Variant

Public r11_3, r11_8, r12_0, r12_4, r12_9 As Variant

Public r21_3, r21_8, r22_0, r22_4, r22_9 As Variant

Public r31_3, r31_8, r32_0, r32_4, r32_9 As Variant

Public r41_3, r41_8, r42_0, r42_4, r42_9 As Variant

Public r51_3, r51_8, r52_0, r52_4, r52_9 As Variant

Public p11_3, p11_8, p12_0, p12_4, p12_9 As Variant

Public p21_3, p21_8, p22_0, p22_4, p22_9 As Variant

Public p31_3, p31_8, p32_0, p32_4, p32_9 As Variant

Public p41_3, p41_8, p42_0, p42_4, p42_9 As Variant

Public p51_3, p51_8, p52_0, p52_4, p52_9 As Variant

Public F11_3, F11_8, F12_0, F12_4, F12_9 As Variant

Public F21_3, F21_8, F22_0, F22_4, F22_9 As Variant

Public F31_3, F31_8, F32_0, F32_4, F32_9 As Variant

Public F41_3, F41_8, F42_0, F42_4, F42_9 As Variant

Public F51_3, F51_8, F52_0, F52_4, F52_9 As Variant

Public x11_3, x11_8, x12_0, x12_4, x12_9 As Variant

Public x21_3, x21_8, x22_0, x22_4, x22_9 As Variant

Public x31_3, x31_8, x32_0, x32_4, x32_9 As Variant

Public x41_3, x41_8, x42_0, x42_4, x42_9 As Variant

Public x51_3, x51_8, x52_0, x52_4, x52_9 As Variant

Public s11_3, s11_8, s12_0, s12_4, s12_9 As Variant

Public s21_3, s21_8, s22_0, s22_4, s22_9 As Variant

Public s31_3, s31_8, s32_0, s32_4, s32_9 As Variant

Public s41_3, s41_8, s42_0, s42_4, s42_9 As Variant

Public s51_3, s51_8, s52_0, s52_4, s52_9 As Variant

Public epp11_3, epp11_8, epp12_0, epp12_4, epp12_9 As Variant

Public epp21_3, epp21_8, epp22_0, epp22_4, epp22_9 As Variant
Public epp31_3, epp31_8, epp32_0, epp32_4, epp32_9 As Variant
Public epp41_3, epp41_8, epp42_0, epp42_4, epp42_9 As Variant
Public epp51_3, epp51_8, epp52_0, epp52_4, epp52_9 As Variant

Public m1p1_3, m1p1_8, m1p2_0, m1p2_4, m1p2_9 As Variant
Public m2p1_3, m2p1_8, m2p2_0, m2p2_4, m2p2_9 As Variant
Public m3p1_3, m3p1_8, m3p2_0, m3p2_4, m3p2_9 As Variant
Public m4p1_3, m4p1_8, m4p2_0, m4p2_4, m4p2_9 As Variant
Public m5p1_3, m5p1_8, m5p2_0, m5p2_4, m5p2_9 As Variant

Public sp_m11_3, sp_m11_8, sp_m12_0, sp_m12_4, sp_m12_9 As Variant
Public sp_m21_3, sp_m21_8, sp_m22_0, sp_m22_4, sp_m22_9 As Variant
Public sp_m31_3, sp_m31_8, sp_m32_0, sp_m32_4, sp_m32_9 As Variant
Public sp_m41_3, sp_m41_8, sp_m42_0, sp_m42_4, sp_m42_9 As Variant
Public sp_m51_3, sp_m51_8, sp_m52_0, sp_m52_4, sp_m52_9 As Variant

Public p1mle1_3, p1mle1_8, p1mle2_0, p1mle2_4, p1mle2_9 As Variant
Public p2mle1_3, p2mle1_8, p2mle2_0, p2mle2_4, p2mle2_9 As Variant
Public p3mle1_3, p3mle1_8, p3mle2_0, p3mle2_4, p3mle2_9 As Variant
Public p4mle1_3, p4mle1_8, p4mle2_0, p4mle2_4, p4mle2_9 As Variant
Public p5mle1_3, p5mle1_8, p5mle2_0, p5mle2_4, p5mle2_9 As Variant

Public rp1mle1_3, rp1mle1_8, rp1mle2_0, rp1mle2_4, rp1mle2_9 As Variant
Public rp2mle1_3, rp2mle1_8, rp2mle2_0, rp2mle2_4, rp2mle2_9 As Variant
Public rp3mle1_3, rp3mle1_8, rp3mle2_0, rp3mle2_4, rp3mle2_9 As Variant
Public rp4mle1_3, rp4mle1_8, rp4mle2_0, rp4mle2_4, rp4mle2_9 As Variant
Public rp5mle1_3, rp5mle1_8, rp5mle2_0, rp5mle2_4, rp5mle2_9 As Variant

Public M1c20_01, s_M1x20_01, M1i20_01, M1pr20_01, M1F20_01, M1x20_01 As Variant
Public M1c30_03, s_M1x30_03, M1i30_03, M1pr30_03, M1F30_03, M1x30_03 As Variant
Public M1c40_05, s_M1x40_05, M1i40_05, M1pr40_05, M1F40_05, M1x40_05 As Variant
Public M1c50_06, s_M1x50_06, M1i50_06, M1pr50_06, M1F50_06, M1x50_06 As Variant
Public M1c70_08, s_M1x70_08, M1i70_08, M1pr70_08, M1F70_08, M1x70_08 As Variant

Public M2c20_01, s_M2x20_01, M2i20_01, M2pr20_01, M2F20_01, M2x20_01 As Variant
Public M2c30_03, s_M2x30_03, M2i30_03, M2pr30_03, M2F30_03, M2x30_03 As Variant
Public M2c40_05, s_M2x40_05, M2i40_05, M2pr40_05, M2F40_05, M2x40_05 As Variant
Public M2c50_06, s_M2x50_06, M2i50_06, M2pr50_06, M2F50_06, M2x50_06 As Variant
Public M2c70_08, s_M2x70_08, M2i70_08, M2pr70_08, M2F70_08, M2x70_08 As Variant

Public M3c20_01, s_M3x20_01, M3i20_01, M3pr20_01, M3F20_01, M3x20_01 As Variant

Public M3c30_03, s_M3x30_03, M3i30_03, M3pr30_03, M3F30_03, M3x30_03 As Variant

Public M3c40_05, s_M3x40_05, M3i40_05, M3pr40_05, M3F40_05, M3x40_05 As Variant

Public M3c50_06, s_M3x50_06, M3i50_06, M3pr50_06, M3F50_06, M3x50_06 As Variant

Public M3c70_08, s_M3x70_08, M3i70_08, M3pr70_08, M3F70_08, M3x70_08 As Variant

Public M4c20_01, s_M4x20_01, M4i20_01, M4pr20_01, M4F20_01, M4x20_01 As Variant

Public M4c30_03, s_M4x30_03, M4i30_03, M4pr30_03, M4F30_03, M4x30_03 As Variant

Public M4c40_05, s_M4x40_05, M4i40_05, M4pr40_05, M4F40_05, M4x40_05 As Variant

Public M4c50_06, s_M4x50_06, M4i50_06, M4pr50_06, M4F50_06, M4x50_06 As Variant

Public M4c70_08, s_M4x70_08, M4i70_08, M4pr70_08, M4F70_08, M4x70_08 As Variant

Public M5c20_01, s_M5x20_01, M5i20_01, M5pr20_01, M5F20_01, M5x20_01 As Variant

Public M5c30_03, s_M5x30_03, M5i30_03, M5pr30_03, M5F30_03, M5x30_03 As Variant

Public M5c40_05, s_M5x40_05, M5i40_05, M5pr40_05, M5F40_05, M5x40_05 As Variant

Public M5c50_06, s_M5x50_06, M5i50_06, M5pr50_06, M5F50_06, M5x50_06 As Variant

Public M5c70_08, s_M5x70_08, M5i70_08, M5pr70_08, M5F70_08, M5x70_08 As Variant

Public eBM120_01, M120_01, s_M120_01 As Variant

Public eBM130_03, M130_03, s_M130_03 As Variant

Public eBM140_05, M140_05, s_M140_05 As Variant

Public eBM150_06, M150_06, s_M150_06 As Variant

Public eBM170_08, M170_08, s_M170_08 As Variant

Public eBM220_01, M220_01, s_M220_01 As Variant

Public eBM230_03, M230_03, s_M230_03 As Variant

Public eBM240_05, M240_05, s_M240_05 As Variant

Public eBM250_06, M250_06, s_M250_06 As Variant

Public eBM270_08, M270_08, s_M270_08 As Variant

Public eBM320_01, M320_01, s_M320_01 As Variant

Public eBM330_03, M330_03, s_M330_03 As Variant

Public eBM340_05, M340_05, s_M340_05 As Variant

Public eBM350_06, M350_06, s_M350_06 As Variant

Public eBM370_08, M370_08, s_M370_08 As Variant

Public eBM420_01, M420_01, s_M420_01 As Variant

Public eBM430_03, M430_03, s_M430_03 As Variant

Public eBM440_05, M440_05, s_M440_05 As Variant

Public eBM450_06, M450_06, s_M450_06 As Variant

Public eBM470_08, M470_08, s_M470_08 As Variant

Public eBM520_01, M520_01, s_M520_01 As Variant

Public eBM530_03, M530_03, s_M530_03 As Variant

Public eBM540_05, M540_05, s_M540_05 As Variant

Public eBM550_06, M550_06, s_M550_06 As Variant

Public eBM570_08, M570_08, s_M570_08 As Variant

Public mlem120_01, rmlem120_01 As Variant

Public mlem130_03, rmlem130_03 As Variant

Public mlem140_05, rmlem140_05 As Variant

Public mlem150_06, rmlem150_06 As Variant

Public mlem170_08, rmlem170_08 As Variant

Public mlem220_01, rmlem220_01 As Variant

Public mlem230_03, rmlem230_03 As Variant

Public mlem240_05, rmlem240_05 As Variant

Public mlem250_06, rmlem250_06 As Variant

Public mlem270_08, rmlem270_08 As Variant

Public mlem320_01, rmlem320_01 As Variant

Public mlem330_03, rmlem330_03 As Variant

Public mlem340_05, rmlem340_05 As Variant

Public mlem350_06, rmlem350_06 As Variant

Public mlem370_08, rmlem370_08 As Variant

Public mlem420_01, rmlem420_01 As Variant

Public mlem430_03, rmlem430_03 As Variant

Public mlem440_05, rmlem440_05 As Variant

Public mlem450_06, rmlem450_06 As Variant

Public mlem470_08, rmlem470_08 As Variant

Public mlem520_01, rmlem520_01 As Variant

Public mlem530_03, rmlem530_03 As Variant

Public mlem540_05, rmlem540_05 As Variant

Public mlem550_06, rmlem550_06 As Variant

Public mlem570_08, rmlem570_08 As Variant

Public M1y0_1, M1y10_1, M1y20_1, M1z0_1, s_M1z0_1 As Variant

Public M1y1_3, M1y11_3, M1y21_3, M1z1_3, s_M1z1_3 As Variant

Public M1y2_5, M1y12_5, M1y22_5, M1z2_5, s_M1z2_5 As Variant

Public M1y3_6, M1y13_6, M1y23_6, M1z3_6, s_M1z3_6 As Variant

Public M1y4_5, M1y14_5, M1y24_5, M1z4_5, s_M1z4_5 As Variant

Public M2y0_1, M2y10_1, M2y20_1, M2z0_1, s_M2z0_1 As Variant
Public M2y1_3, M2y11_3, M2y21_3, M2z1_3, s_M2z1_3 As Variant
Public M2y2_5, M2y12_5, M2y22_5, M2z2_5, s_M2z2_5 As Variant
Public M2y3_6, M2y13_6, M2y23_6, M2z3_6, s_M2z3_6 As Variant
Public M2y4_5, M2y14_5, M2y24_5, M2z4_5, s_M2z4_5 As Variant

Public M3y0_1, M3y10_1, M3y20_1, M3z0_1, s_M3z0_1 As Variant
Public M3y1_3, M3y11_3, M3y21_3, M3z1_3, s_M3z1_3 As Variant
Public M3y2_5, M3y12_5, M3y22_5, M3z2_5, s_M3z2_5 As Variant
Public M3y3_6, M3y13_6, M3y23_6, M3z3_6, s_M3z3_6 As Variant
Public M3y4_5, M3y14_5, M3y24_5, M3z4_5, s_M3z4_5 As Variant

Public M4y0_1, M4y10_1, M4y20_1, M4z0_1, s_M4z0_1 As Variant
Public M4y1_3, M4y11_3, M4y21_3, M4z1_3, s_M4z1_3 As Variant
Public M4y2_5, M4y12_5, M4y22_5, M4z2_5, s_M4z2_5 As Variant
Public M4y3_6, M4y13_6, M4y23_6, M4z3_6, s_M4z3_6 As Variant
Public M4y4_5, M4y14_5, M4y24_5, M4z4_5, s_M4z4_5 As Variant

Public M5y0_1, M5y10_1, M5y20_1, M5z0_1, s_M5z0_1 As Variant
Public M5y1_3, M5y11_3, M5y21_3, M5z1_3, s_M5z1_3 As Variant
Public M5y2_5, M5y12_5, M5y22_5, M5z2_5, s_M5z2_5 As Variant
Public M5y3_6, M5y13_6, M5y23_6, M5z3_6, s_M5z3_6 As Variant
Public M5y4_5, M5y14_5, M5y24_5, M5z4_5, s_M5z4_5 As Variant

Public eN10_1, eN11_3, eN12_5, eN13_6, eN14_5 As Variant
Public eN20_1, eN21_3, eN22_5, eN23_6, eN24_5 As Variant
Public eN30_1, eN31_3, eN32_5, eN33_6, eN34_5 As Variant
Public eN40_1, eN41_3, eN42_5, eN43_6, eN44_5 As Variant
Public eN50_1, eN51_3, eN52_5, eN53_6, eN54_5 As Variant

Public m10_1, m1_1_3, m12_5, m13_6, m14_5 As Variant
Public m20_1, m2_1_3, m22_5, m23_6, m24_5 As Variant
Public m30_1, m3_1_3, m32_5, m33_6, m34_5 As Variant
Public m40_1, m4_1_3, m42_5, m43_6, m44_5 As Variant
Public m50_1, m5_1_3, m52_5, m53_6, m54_5 As Variant

Public s_m10_1, s_m1_1_3, s_m12_5, s_m13_6, s_m14_5 As Variant
Public s_m20_1, s_m2_1_3, s_m22_5, s_m23_6, s_m24_5 As Variant
Public s_m30_1, s_m3_1_3, s_m32_5, s_m33_6, s_m34_5 As Variant
Public s_m40_1, s_m4_1_3, s_m42_5, s_m43_6, s_m44_5 As Variant
Public s_m50_1, s_m5_1_3, s_m52_5, s_m53_6, s_m54_5 As Variant

Public mlem10_1, mlem1_1_3, mlem12_5, mlem13_6, mlem14_5 As Variant
Public mlem20_1, mlem2_1_3, mlem22_5, mlem23_6, mlem24_5 As Variant
Public mlem30_1, mlem3_1_3, mlem32_5, mlem33_6, mlem34_5 As Variant
Public mlem40_1, mlem4_1_3, mlem42_5, mlem43_6, mlem44_5 As Variant
Public mlem50_1, mlem5_1_3, mlem52_5, mlem53_6, mlem54_5 As Variant

Public rmlem10_1, rmlem1_1_3, rmlem12_5, rmlem13_6, rmlem14_5 As Variant
Public rmlem20_1, rmlem2_1_3, rmlem22_5, rmlem23_6, rmlem24_5 As Variant

```
Public rmlem30_1, rmlem3_1_3, rmlem32_5, rmlem33_6, rmlem34_5 As Variant
Public rmlem40_1, rmlem4_1_3, rmlem42_5, rmlem43_6, rmlem44_5 As Variant
Public rmlem50_1, rmlem5_1_3, rmlem52_5, rmlem53_6, rmlem54_5 As Variant
```

```
Public v10_1, s_v10_1, var1_01, mle_v10_1, rmle_v10_1 As Variant
Public v11_3, s_v11_3, var1_13, mle_v11_3, rmle_v11_3 As Variant
Public v12_5, s_v12_5, var1_25, mle_v12_5, rmle_v12_5 As Variant
Public v13_6, s_v13_6, var1_36, mle_v13_6, rmle_v13_6 As Variant
Public v14_5, s_v14_5, var1_45, mle_v14_5, rmle_v14_5 As Variant
```

```
Public v20_1, s_v20_1, var2_01, mle_v20_1, rmle_v20_1 As Variant
Public v21_3, s_v21_3, var2_13, mle_v21_3, rmle_v21_3 As Variant
Public v22_5, s_v22_5, var2_25, mle_v22_5, rmle_v22_5 As Variant
Public v23_6, s_v23_6, var2_36, mle_v23_6, rmle_v23_6 As Variant
Public v24_5, s_v24_5, var2_45, mle_v24_5, rmle_v24_5 As Variant
```

```
Public v30_1, s_v30_1, var3_01, mle_v30_1, rmle_v30_1 As Variant
Public v31_3, s_v31_3, var3_13, mle_v31_3, rmle_v31_3 As Variant
Public v32_5, s_v32_5, var3_25, mle_v32_5, rmle_v32_5 As Variant
Public v33_6, s_v33_6, var3_36, mle_v33_6, rmle_v33_6 As Variant
Public v34_5, s_v34_5, var3_45, mle_v34_5, rmle_v34_5 As Variant
```

```
Public v40_1, s_v40_1, var4_01, mle_v40_1, rmle_v40_1 As Variant
Public v41_3, s_v41_3, var4_13, mle_v41_3, rmle_v41_3 As Variant
Public v42_5, s_v42_5, var4_25, mle_v42_5, rmle_v42_5 As Variant
Public v43_6, s_v43_6, var4_36, mle_v43_6, rmle_v43_6 As Variant
Public v44_5, s_v44_5, var4_45, mle_v44_5, rmle_v44_5 As Variant
```

```
Public v50_1, s_v50_1, var5_01, mle_v50_1, rmle_v50_1 As Variant
Public v51_3, s_v51_3, var5_13, mle_v51_3, rmle_v51_3 As Variant
Public v52_5, s_v52_5, var5_25, mle_v52_5, rmle_v52_5 As Variant
Public v53_6, s_v53_6, var5_36, mle_v53_6, rmle_v53_6 As Variant
Public v54_5, s_v54_5, var5_45, mle_v54_5, rmle_v54_5 As Variant
```

```
Private Sub Form_Activate()
```

```
    s = 0
    s1_3 = 0
    s1_8 = 0
    s2_0 = 0
    s2_4 = 0
    s2_9 = 0
    x_expo = 0
    s_m = 0
    m_error = 0
    mse_error = 0
    r = 0
```

```
    Randomize Timer
    m1(0) = Round(Rnd, 4) * 10000
```



```
m2(0) = Round(Rnd, 4) * 10000
m2(1) = Round(Rnd, 4) * 10000
```

```
m3(0) = Round(Rnd, 4) * 10000
k = Round(Rnd, 4) * 10000
```

```
m4(0) = Rnd * 10000
m4(1) = Rnd * 10000
m4(2) = Rnd * 10000
m4(3) = Rnd * 10000
m4(4) = Rnd * 10000
m = Rnd * 10000
```

```
If m4(0) Or m4(1) Or m4(2) Or m4(3) Or m4(4) Or m = 0 Then
    Do Until m4(0) And m4(1) And m4(2) And m4(3) And m4(4) And m <> 0
```

```
        Randomize Timer
        m4(0) = Rnd * 10000
        m4(1) = Rnd * 10000
        m4(2) = Rnd * 10000
        m4(3) = Rnd * 10000
        m4(4) = Rnd * 10000
        m = Rnd * 10000
```

```
    Loop
```

```
End If
```

```
m5(0) = Rnd * 10000
a5 = Rnd * 10000
k = Rnd * 10000
w = Rnd * 10000
```

```
If m5(0) Or a5 Or k Or w = 0 Then
    Do Until m5(0) And a5 And k And w <> 0
```

```
        Randomize Timer
        m5(0) = Rnd * 10000
        a5 = Rnd * 10000
        k = Rnd * 10000
        w = Rnd * 10000
```

```
    Loop
```

```
End If
```

```
sample = InputBox("SAMPLE TEST ")
loop_test = InputBox("LOOP TEST ")
```

For j = 1 To loop_test

For i = 1 To sample

Call crn_m1
Call expo1
Call poisson1
Call binomial1
Call normal1

Call crn_m2
Call expo2
Call poisson2
Call binomial2
Call normal2

Call crn_m3
Call expo3
Call poisson3
Call binomial3
Call normal3

Call crn_m4
Call expo4
Call poisson4
Call binomial4
Call normal4

Call crn_m5
Call expo5
Call poisson5
Call binomial5
Call normal5

Next i

Call cal_est1
Call cal_est2
Call cal_est3
Call cal_est4
Call cal_est5

Call cal_est_poi1
Call cal_est_poi2
Call cal_est_poi3
Call cal_est_poi4
Call cal_est_poi5

Call cal_est_binomial1
Call cal_est_binomial2
Call cal_est_binomial3
Call cal_est_binomial4
Call cal_est_binomial5

Call cal_est_normal1
Call cal_est_normal2
Call cal_est_normal3
Call cal_est_normal4
Call cal_est_normal5

Call est_std
Call cal_variance

Next j

Call cal_mse_err1
Call cal_mse_err2
Call cal_mse_err3
Call cal_mse_err4
Call cal_mse_err5

Call mse_poisson1
Call mse_poisson2
Call mse_poisson3
Call mse_poisson4
Call mse_poisson5

Call mse_binomial1
Call mse_binomial2
Call mse_binomial3
Call mse_binomial4
Call mse_binomial5

Call mse_normal1
Call mse_normal2
Call mse_normal3
Call mse_normal4
Call mse_normal5

Call mse_variance

End Sub

Function crn_m3()

z = k * m3(i - 1)

```
If z < 100000000 Then
  Randomize Timer
  m3(i - 1) = Round(Rnd, 4) * 10000
  z = k * m3(i - 1)
  Call Cal
  If b = 0 Then
    Do Until b <> 0
      Randomize Timer
      m3(i - 1) = Round(Rnd, 4) * 10000
      z = k * m3(i - 1)
      Call Cal
    Loop
  End If
Else
  Call Cal
  If b = 0 Then
    Do Until b <> 0
      Randomize Timer
      m3(i - 1) = Round(Rnd, 4) * 10000
      z = k * m3(i - 1)
      Call Cal
    Loop
  End If
End If
m3(i) = b
```

End Function

Function crn_m1()

```
z = m1(i - 1) * m1(i - 1)
If z < 100000000 Then
  Randomize Timer
  m1(i - 1) = Round(Rnd, 4) * 10000
  z = m1(i - 1) * m1(i - 1)
  Call Cal
  If b = 0 Then
    Do Until b <> 0
      Randomize Timer
      m1(i - 1) = Round(Rnd, 4) * 10000
      z = m1(i - 1) * m1(i - 1)
      Call Cal
    Loop
  Else
    Call Cal
  End If
Else
```

```
Call Cal
If b = 0 Then
  Do Until b <> 0
    Randomize Timer
    m1(i - 1) = Round(Rnd, 4) * 10000
    z = m1(i - 1) * m1(i - 1)
    Call Cal
  Loop
Else
  Call Cal
End If
End If
m1(i) = b
End Function

Function crn_m2()
  z = m2(i - 1) * m2(i)
  If z < 100000000 Then
    Randomize Timer
    m2(i - 1) = Round(Rnd, 4) * 10000
    m2(i) = Round(Rnd, 4) * 10000
    z = m2(i - 1) * m2(i)
    Call Cal
    If b = 0 Then
      Do Until b <> 0
        Randomize Timer
        m2(i - 1) = Round(Rnd, 4) * 10000
        m2(i) = Round(Rnd, 4) * 10000
        z = m2(i - 1) * m2(i)
        Call Cal
      Loop
    Else
      Call Cal
    End If
  Else
    Call Cal
    If b = 0 Then
      Do Until b <> 0
        Randomize Timer
        m2(i - 1) = Round(Rnd, 4) * 10000
        m2(i) = Round(Rnd, 4) * 10000
        z = m2(i - 1) * m2(i)
        Call Cal
      Loop
    Else
      Call Cal
    End If
  End If
End If
```

```
End If
m2(i) = b
End Function
```

```
Function crn_m4()
```

```
    m4(i + 4) = (m4(i + 3) + m4(i - 1)) Mod m
    answer = Round(m4(i + 4) / m, 4)
    If answer = 0 Then
        Do Until answer <> 0
            Randomize Timer
            m4(0) = Rnd * 10000
            m4(1) = Rnd * 10000
            m4(2) = Rnd * 10000
            m4(3) = Rnd * 10000
            m4(4) = Rnd * 10000
            m = Rnd * 10000
            m4(i + 4) = (m4(i + 3) + m4(i - 1)) Mod m
            answer = m4(i + 4) / m
        Loop
    End If
```

```
End Function
```

```
Function crn_m5()
```

```
    m5(i) = ((a5 * m5(i - 1)) + k) Mod w
    answer = m5(i) / w
    If answer = 0 Then
        Do Until answer <> 0
            Randomize Timer
            m5(0) = Rnd * 10000
            a5 = Rnd * 10000
            k = Rnd * 10000
            w = Rnd * 10000
            m5(i) = ((a5 * m5(i - 1)) + k) Mod w
            answer = m5(i) / w
        Loop
    End If
```

```
End Function
```

```
Function cal_mse_err1()
```

```
    mlem11_3 = (s_m11_3 / loop_test)
    rmlm11_3 = Round(Sqr(mlem11_3), 4)
```

```
    mlem11_8 = (s_m11_8 / loop_test)
    rmlm11_8 = Round(Sqr(mlem11_8), 4)
```

```
mlem12_0 = (s_m12_0 / loop_test)
rmlem12_0 = Round(Sqr(mlem12_0), 4)
```

```
mlem12_4 = (s_m12_4 / loop_test)
rmlem12_4 = Round(Sqr(mlem12_4), 4)
```

```
mlem12_9 = (s_m12_9 / loop_test)
rmlem12_9 = Round(Sqr(mlem12_9), 4)
Print "n=" & sample, "loop test = " & loop_test
Print "M1: RMSE EXPO 1.3=" & rmlem11_3, "1.8=" & rmlem11_8, "2.0=" &
rmlem12_0, "2.4=" & rmlem12_4, "2.9=" & rmlem12_9
```

End Function

Function cal_mse_err2()

```
mlem21_3 = (s_m21_3 / loop_test)
rmlem21_3 = Round(Sqr(mlem21_3), 4)
```

```
mlem21_8 = (s_m21_8 / loop_test)
rmlem21_8 = Round(Sqr(mlem21_8), 4)
```

```
mlem22_0 = (s_m22_0 / loop_test)
rmlem22_0 = Round(Sqr(mlem22_0), 4)
```

```
mlem22_4 = (s_m22_4 / loop_test)
rmlem22_4 = Round(Sqr(mlem22_4), 4)
```

```
mlem22_9 = (s_m22_9 / loop_test)
rmlem22_9 = Round(Sqr(mlem22_9), 4)
```

```
Print "M2: RMSE EXPO 1.3=" & rmlem21_3, "1.8=" & rmlem21_8, "2.0=" &
rmlem22_0, "2.4=" & rmlem22_4, "2.9=" & rmlem22_9
```

End Function

Function cal_mse_err3()

```
mlem31_3 = (s_m31_3 / loop_test)
rmlem31_3 = Round(Sqr(mlem31_3), 4)
```

```
mlem31_8 = (s_m31_8 / loop_test)
rmlem31_8 = Round(Sqr(mlem31_8), 4)
```

```
mlem32_0 = (s_m32_0 / loop_test)
rmlem32_0 = Round(Sqr(mlem32_0), 4)
```

```
mlem32_4 = (s_m32_4 / loop_test)
rmlem32_4 = Round(Sqr(mlem32_4), 4)
```

```
mlem32_9 = (s_m32_9 / loop_test)
rmlem32_9 = Round(Sqr(mlem32_9), 4)
```

```
Print "M3: RMSE EXPO 1.3=" & rmlem31_3, "1.8 =" & rmlem31_8, "2.0 =" &
rmlem32_0, "2.4 =" & rmlem32_4, "2.9 =" & rmlem32_9
```

End Function

Function cal_mse_err4()

```
mlem41_3 = (s_m41_3 / loop_test)
rmlem41_3 = Round(Sqr(mlem41_3), 4)
```

```
mlem41_8 = (s_m41_8 / loop_test)
rmlem41_8 = Round(Sqr(mlem41_8), 4)
```

```
mlem42_0 = (s_m42_0 / loop_test)
rmlem42_0 = Round(Sqr(mlem42_0), 4)
```

```
mlem42_4 = (s_m42_4 / loop_test)
rmlem42_4 = Round(Sqr(mlem42_4), 4)
```

```
mlem42_9 = (s_m42_9 / loop_test)
rmlem42_9 = Round(Sqr(mlem42_9), 4)
```

```
Print "M4: RMSE EXPO 1.3=" & rmlem41_3, "1.8 =" & rmlem41_8, "2.0 =" &
rmlem42_0, "2.4 =" & rmlem42_4, "2.9 =" & rmlem42_9
```

End Function

Function cal_mse_err5()

```
mlem51_3 = (s_m51_3 / loop_test)
rmlem51_3 = Round(Sqr(mlem51_3), 4)
```

```
mlem51_8 = (s_m51_8 / loop_test)
rmlem51_8 = Round(Sqr(mlem51_8), 4)
```

```
mlem52_0 = (s_m52_0 / loop_test)
rmlem52_0 = Round(Sqr(mlem52_0), 4)
```



```
mlem52_4 = (s_m52_4 / loop_test)
rmlem52_4 = Round(Sqr(mlem52_4), 4)
```

```
mlem52_9 = (s_m52_9 / loop_test)
rmlem52_9 = Round(Sqr(mlem52_9), 4)
```

```
Print "M5: RMSE EXPO 1.3=" & rmlem51_3, "1.8=" & rmlem51_8, "2.0=" &
rmlem52_0, "2.4=" & rmlem52_4, "2.9=" & rmlem52_9
```

```
End Function
```

```
Function Cal()
```

```
a = z Mod 1000000
b = Int(a / 100)
answer = b / 10000
```

```
End Function
```

```
Function expo1()
```

```
m11_3 = -(1.3) * Log(answer)
m1_expo1 = m11_3
m1s1_3 = m1s1_3 + m1_expo1
```

```
m11_8 = -(1.8) * Log(answer)
m1_expo2 = m11_8
m1s1_8 = m1s1_8 + m1_expo2
```

```
m12_0 = -(2) * Log(answer)
m1_expo3 = m12_0
m1s2_0 = m1s2_0 + m1_expo3
```

```
m12_4 = -(2.4) * Log(answer)
m1_expo4 = m12_4
m1s2_4 = m1s2_4 + m1_expo4
```

```
m12_9 = -(2.9) * Log(answer)
m1_expo5 = m12_9
m1s2_9 = m1s2_9 + m1_expo5
```

```
End Function
```

Function expo2()

```
m21_3 = -(1.3) * Log(answer)
m2_expo1 = Round(m21_3, 4)
m2s1_3 = m2s1_3 + m2_expo1
```

```
m21_8 = -(1.8) * Log(answer)
m2_expo2 = Round(m21_8, 4)
m2s1_8 = m2s1_8 + m2_expo2
```

```
m22_0 = -(2) * Log(answer)
m2_expo3 = Round(m22_0, 4)
m2s2_0 = m2s2_0 + m2_expo3
```

```
m22_4 = -(2.4) * Log(answer)
m2_expo4 = Round(m22_4, 4)
m2s2_4 = m2s2_4 + m2_expo4
```

```
m22_9 = -(2.9) * Log(answer)
m2_expo5 = Round(m22_9, 4)
m2s2_9 = m2s2_9 + m2_expo5
```

End Function

Function expo3()

```
m31_3 = -(1.3) * Log(answer)
m3_expo1 = Round(m31_3, 4)
m3s1_3 = m3s1_3 + m3_expo1
```

```
m31_8 = -(1.8) * Log(answer)
m3_expo2 = Round(m31_8, 4)
m3s1_8 = m3s1_8 + m3_expo2
```

```
m32_0 = -(2) * Log(answer)
m3_expo3 = Round(m32_0, 4)
m3s2_0 = m3s2_0 + m3_expo3
```

```
m32_4 = -(2.4) * Log(answer)
m3_expo4 = Round(m32_4, 4)
m3s2_4 = m3s2_4 + m3_expo4
```

```
m32_9 = -(2.9) * Log(answer)
m3_expo5 = Round(m32_9, 4)
m3s2_9 = m3s2_9 + m3_expo5
```

End Function

Function expo4()

```
m41_3 = -(1.3) * Log(answer)
m4_expo1 = Round(m41_3, 4)
m4s1_3 = m4s1_3 + m4_expo1
```

```
m41_8 = -(1.8) * Log(answer)
m4_expo2 = Round(m41_8, 4)
m4s1_8 = m4s1_8 + m4_expo2
```

```
m42_0 = -(2) * Log(answer)
m4_expo3 = Round(m42_0, 4)
m4s2_0 = m4s2_0 + m4_expo3
```

```
m42_4 = -(2.4) * Log(answer)
m4_expo4 = Round(m42_4, 4)
m4s2_4 = m4s2_4 + m4_expo4
```

```
m42_9 = -(2.9) * Log(answer)
m4_expo5 = Round(m42_9, 4)
m4s2_9 = m4s2_9 + m4_expo5
```

End Function

Function expo5()

```
m51_3 = -(1.3) * Log(answer)
m5_expo1 = Round(m51_3, 4)
m5s1_3 = m5s1_3 + m5_expo1
```

```
m51_8 = -(1.8) * Log(answer)
m5_expo2 = Round(m51_8, 4)
m5s1_8 = m5s1_8 + m5_expo2
```

```
m52_0 = -(2) * Log(answer)
```

```
m5_expo3 = Round(m52_0, 4)
m5s2_0 = m5s2_0 + m5_expo3
```

```
m52_4 = -(2.4) * Log(answer)
m5_expo4 = Round(m52_4, 4)
m5s2_4 = m5s2_4 + m5_expo4
```

```
m52_9 = -(2.9) * Log(answer)
m5_expo5 = Round(m52_9, 4)
m5s2_9 = m5s2_9 + m5_expo5
```

End Function

Function cal_est1()

```
ep11_3 = m1s1_3 / sample
m11_3 = (ep11_3 - 1.3) ^ 2
s_m11_3 = s_m11_3 + m11_3
```

```
ep11_8 = m1s1_8 / sample
m11_8 = (ep11_8 - 1.8) ^ 2
s_m11_8 = s_m11_8 + m11_8
```

```
ep12_0 = m1s2_0 / sample
m12_0 = (ep12_0 - 2) ^ 2
s_m12_0 = s_m12_0 + m12_0
```

```
ep12_4 = m1s2_4 / sample
m12_4 = (ep12_4 - 2.4) ^ 2
s_m12_4 = s_m12_4 + m12_4
```

```
ep12_9 = m1s2_9 / sample
m12_9 = (ep12_9 - 2.9) ^ 2
s_m12_9 = s_m12_9 + m12_9
```

End Function

Function cal_est2()

```
ep21_3 = m2s1_3 / sample
m21_3 = (ep21_3 - 1.3) ^ 2
s_m21_3 = s_m21_3 + m21_3
```

```
ep21_8 = m2s1_8 / sample
m21_8 = (ep21_8 - 1.8) ^ 2
s_m21_8 = s_m21_8 + m21_8
```

```
ep22_0 = m2s2_0 / sample  
m22_0 = (ep22_0 - 2) ^ 2  
s_m22_0 = s_m22_0 + m22_0
```

```
ep22_4 = m2s2_4 / sample  
m22_4 = (ep22_4 - 2.4) ^ 2  
s_m22_4 = s_m22_4 + m22_4
```

```
ep22_9 = m2s2_9 / sample  
m22_9 = (ep22_9 - 2.9) ^ 2  
s_m22_9 = s_m22_9 + m22_9
```

End Function

Function cal_est3()

```
ep31_3 = m3s1_3 / sample  
m31_3 = (ep31_3 - 1.3) ^ 2  
s_m31_3 = s_m31_3 + m31_3
```

```
ep31_8 = m3s1_8 / sample  
m31_8 = (ep31_8 - 1.8) ^ 2  
s_m31_8 = s_m31_8 + m31_8
```

```
ep32_0 = m3s2_0 / sample  
m32_0 = (ep32_0 - 2) ^ 2  
s_m32_0 = s_m32_0 + m32_0
```

```
ep32_4 = m3s2_4 / sample  
m32_4 = (ep32_4 - 2.4) ^ 2  
s_m32_4 = s_m32_4 + m32_4
```

```
ep32_9 = m3s2_9 / sample  
m32_9 = (ep32_9 - 2.9) ^ 2  
s_m32_9 = s_m32_9 + m32_9
```

End Function

Function cal_est4()

```
ep41_3 = m4s1_3 / sample  
m41_3 = (ep41_3 - 1.3) ^ 2  
s_m41_3 = s_m41_3 + m41_3
```

```
ep41_8 = m4s1_8 / sample  
m41_8 = (ep41_8 - 1.8) ^ 2
```

$s_m41_8 = s_m41_8 + m41_8$

$ep42_0 = m4s2_0 / \text{sample}$
 $m42_0 = (ep42_0 - 2) ^ 2$
 $s_m42_0 = s_m42_0 + m42_0$

$ep42_4 = m4s2_4 / \text{sample}$
 $m42_4 = (ep42_4 - 2.4) ^ 2$
 $s_m42_4 = s_m42_4 + m42_4$

$ep42_9 = m4s2_9 / \text{sample}$
 $m42_9 = (ep42_9 - 2.9) ^ 2$
 $s_m42_9 = s_m42_9 + m42_9$

End Function

Function cal_est5()

$ep51_3 = m5s1_3 / \text{sample}$
 $m51_3 = (ep51_3 - 1.3) ^ 2$
 $s_m51_3 = s_m51_3 + m51_3$

$ep51_8 = m5s1_8 / \text{sample}$
 $m51_8 = (ep51_8 - 1.8) ^ 2$
 $s_m51_8 = s_m51_8 + m51_8$

$ep52_0 = m5s2_0 / \text{sample}$
 $m52_0 = (ep52_0 - 2) ^ 2$
 $s_m52_0 = s_m52_0 + m52_0$

$ep52_4 = m5s2_4 / \text{sample}$
 $m52_4 = (ep52_4 - 2.4) ^ 2$
 $s_m52_4 = s_m52_4 + m52_4$

$ep52_9 = m5s2_9 / \text{sample}$
 $m52_9 = (ep52_9 - 2.9) ^ 2$
 $s_m52_9 = s_m52_9 + m52_9$

End Function

Function poisson1()

$r11_3 = 0$
 $p11_3 = \text{Exp}(-1.3)$
 $F11_3 = p11_3$
If answer < F11_3 Then
 $x11_3 = r11_3$

```
Else
  Do Until answer < F11_3
    p11_3 = (1.3 * p11_3) / (r11_3 + 1)
    F11_3 = F11_3 + p11_3
    r11_3 = r11_3 + 1
    x11_3 = r11_3
  Loop
End If
s11_3 = s11_3 + x11_3
```

```
r11_8 = 0
p11_8 = Exp(-1.8)
F11_8 = p11_8
If answer < F11_8 Then
  x11_8 = r11_8
Else
  Do Until answer < F11_8
    p11_8 = (1.8 * p11_8) / (r11_8 + 1)
    F11_8 = F11_8 + p11_8
    r11_8 = r11_8 + 1
    x11_8 = r11_8
  Loop
End If
s11_8 = s11_8 + x11_8
```

```
r12_0 = 0
p12_0 = Exp(-2)
F12_0 = p12_0
If answer < F12_0 Then
  x12_0 = r12_0
Else
  Do Until answer < F12_0
    p12_0 = (2 * p12_0) / (r12_0 + 1)
    F12_0 = F12_0 + p12_0
    r12_0 = r12_0 + 1
    x12_0 = r12_0
  Loop
End If
s12_0 = s12_0 + x12_0
```

```
r12_4 = 0
p12_4 = Exp(-2.4)
F12_4 = p12_4
If answer < F12_4 Then
  x12_4 = r12_4
Else
  Do Until answer < F12_4
```

```
        p12_4 = (2.4 * p12_4) / (r12_4 + 1)
        F12_4 = F12_4 + p12_4
        r12_4 = r12_4 + 1
        x12_4 = r12_4
    Loop
End If
s12_4 = s12_4 + x12_4

r12_9 = 0
p12_9 = Exp(-2.9)
F12_9 = p12_9
If answer < F12_9 Then
    x12_9 = r12_9
Else
    Do Until answer < F12_9
        p12_9 = (2.9 * p12_9) / (r12_9 + 1)
        F12_9 = F12_9 + p12_9
        r12_9 = r12_9 + 1
        x12_9 = r12_9
    Loop
End If
s12_9 = s12_9 + x12_9
```

End Function

Function poisson2()

```
r21_3 = 0
p21_3 = Exp(-1.3)
F21_3 = p21_3
If answer < F21_3 Then
    x21_3 = r21_3
Else
    Do Until answer < F21_3
        p21_3 = (1.3 * p21_3) / (r21_3 + 1)
        F21_3 = F21_3 + p21_3
        r21_3 = r21_3 + 1
        x21_3 = r21_3
    Loop
End If
s21_3 = s21_3 + x21_3
```

```
r21_8 = 0
p21_8 = Exp(-1.8)
F21_8 = p21_8
If answer < F21_8 Then
    x21_8 = r21_8
```



```
Else
  Do Until answer < F21_8
    p21_8 = (1.8 * p21_8) / (r21_8 + 1)
    F21_8 = F21_8 + p21_8
    r21_8 = r21_8 + 1
    x21_8 = r21_8
  Loop
End If
s21_8 = s21_8 + x21_8
```

```
r22_0 = 0
p22_0 = Exp(-2)
F22_0 = p22_0
If answer < F22_0 Then
  x22_0 = r22_0
Else
  Do Until answer < F22_0
    p22_0 = (2 * p22_0) / (r22_0 + 1)
    F22_0 = F22_0 + p22_0
    r22_0 = r22_0 + 1
    x22_0 = r22_0
  Loop
End If
s22_0 = s22_0 + x22_0
```

```
r22_4 = 0
p22_4 = Exp(-2.4)
F22_4 = p22_4
If answer < F22_4 Then
  x22_4 = r22_4
Else
  Do Until answer < F22_4
    p22_4 = (2.4 * p22_4) / (r22_4 + 1)
    F22_4 = F22_4 + p22_4
    r22_4 = r22_4 + 1
    x22_4 = r22_4
  Loop
End If
s22_4 = s22_4 + x22_4
```

```
r22_9 = 0
p22_9 = Exp(-2.9)
F22_9 = p22_9
If answer < F22_9 Then
  x22_9 = r22_9
Else
  Do Until answer < F22_9
    p22_9 = (2.9 * p22_9) / (r22_9 + 1)
```

```
F22_9 = F22_9 + p22_9
r22_9 = r22_9 + 1
x22_9 = r22_9
Loop
End If
s22_9 = s22_9 + x22_9
```

End Function

Function poisson3()

```
r31_3 = 0
p31_3 = Exp(-1.3)
F31_3 = p31_3
If answer < F31_3 Then
    x31_3 = r31_3
Else
    Do Until answer < F31_3
        p31_3 = (1.3 * p31_3) / (r31_3 + 1)
        F31_3 = F31_3 + p31_3
        r31_3 = r31_3 + 1
        x31_3 = r31_3
    Loop
End If
s31_3 = s31_3 + x31_3
```

```
r31_8 = 0
p31_8 = Exp(-1.8)
F31_8 = p31_8
If answer < F31_8 Then
    x31_8 = r31_8
Else
    Do Until answer < F31_8
        p31_8 = (1.8 * p31_8) / (r31_8 + 1)
        F31_8 = F31_8 + p31_8
        r31_8 = r31_8 + 1
        x31_8 = r31_8
    Loop
End If
s31_8 = s31_8 + x31_8
```

```
r32_0 = 0
p32_0 = Exp(-2)
F32_0 = p32_0
If answer < F32_0 Then
    x32_0 = r32_0
Else
```

```
Do Until answer < F32_0
    p32_0 = (2 * p32_0) / (r32_0 + 1)
    F32_0 = F32_0 + p32_0
    r32_0 = r32_0 + 1
    x32_0 = r32_0
Loop
End If
s32_0 = s32_0 + x32_0
```

```
r32_4 = 0
p32_4 = Exp(-2.4)
F32_4 = p32_4
If answer < F32_4 Then
    x32_4 = r32_4
Else
    Do Until answer < F32_4
        p32_4 = (2.4 * p32_4) / (r32_4 + 1)
        F32_4 = F32_4 + p32_4
        r32_4 = r32_4 + 1
        x32_4 = r32_4
    Loop
End If
s32_4 = s32_4 + x32_4
```

```
r32_9 = 0
p32_9 = Exp(-2.9)
F32_9 = p32_9
If answer < F32_9 Then
    x32_9 = r32_9
Else
    Do Until answer < F32_9
        p32_9 = (2.9 * p32_9) / (r32_9 + 1)
        F32_9 = F32_9 + p32_9
        r32_9 = r32_9 + 1
        x32_9 = r32_9
    Loop
End If
s32_9 = s32_9 + x32_9
```

End Function

Function poisson4()

```
r41_3 = 0
p41_3 = Exp(-1.3)
F41_3 = p41_3
```

```
If answer < F41_3 Then
  x41_3 = r41_3
Else
  Do Until answer < F41_3
    p41_3 = (1.3 * p41_3) / (r41_3 + 1)
    F41_3 = F41_3 + p41_3
    r41_3 = r41_3 + 1
    x41_3 = r41_3
  Loop
End If
s41_3 = s41_3 + x41_3
```

```
r41_8 = 0
p41_8 = Exp(-1.8)
F41_8 = p41_8
If answer < F41_8 Then
  x41_8 = r41_8
Else
  Do Until answer < F41_8
    p41_8 = (1.8 * p41_8) / (r41_8 + 1)
    F41_8 = F41_8 + p41_8
    r41_8 = r41_8 + 1
    x41_8 = r41_8
  Loop
End If
s41_8 = s41_8 + x41_8
```

```
r42_0 = 0
p42_0 = Exp(-2)
F42_0 = p42_0
If answer < F42_0 Then
  x42_0 = r42_0
Else
  Do Until answer < F42_0
    p42_0 = (2 * p42_0) / (r42_0 + 1)
    F42_0 = F42_0 + p42_0
    r42_0 = r42_0 + 1
    x42_0 = r42_0
  Loop
End If
s42_0 = s42_0 + x42_0
```

```
r42_4 = 0
p42_4 = Exp(-2.4)
F42_4 = p42_4
If answer < F42_4 Then
  x42_4 = r42_4
```

```
Else
  Do Until answer < F42_4
    p42_4 = (2.4 * p42_4) / (r42_4 + 1)
    F42_4 = F42_4 + p42_4
    r42_4 = r42_4 + 1
    x42_4 = r42_4
  Loop
End If
s42_4 = s42_4 + x42_4

r42_9 = 0
p42_9 = Exp(-2.9)
F42_9 = p42_9
If answer < F42_9 Then
  x42_9 = r42_9
Else
  Do Until answer < F42_9
    p42_9 = (2.9 * p42_9) / (r42_9 + 1)
    F42_9 = F42_9 + p42_9
    r42_9 = r42_9 + 1
    x42_9 = r42_9
  Loop
End If
s42_9 = s42_9 + x42_9
```

End Function

Function poisson5()

```
r51_3 = 0
p51_3 = Exp(-1.3)
F51_3 = p51_3
If answer < F51_3 Then
  x51_3 = r51_3
Else
  Do Until answer < F51_3
    p51_3 = (1.3 * p51_3) / (r51_3 + 1)
    F51_3 = F51_3 + p51_3
    r51_3 = r51_3 + 1
    x51_3 = r51_3
  Loop
End If
s51_3 = s51_3 + x51_3

r51_8 = 0
p51_8 = Exp(-1.8)
F51_8 = p51_8
```

```
If answer < F51_8 Then
  x51_8 = r51_8
Else
  Do Until answer < F51_8
    p51_8 = (1.8 * p51_8) / (r51_8 + 1)
    F51_8 = F51_8 + p51_8
    r51_8 = r51_8 + 1
    x51_8 = r51_8
  Loop
End If
s51_8 = s51_8 + x51_8
```

```
r52_0 = 0
p52_0 = Exp(-2)
F52_0 = p52_0
If answer < F52_0 Then
  x52_0 = r52_0
Else
  Do Until answer < F52_0
    p52_0 = (2 * p52_0) / (r52_0 + 1)
    F52_0 = F52_0 + p52_0
    r52_0 = r52_0 + 1
    x52_0 = r52_0
  Loop
End If
s52_0 = s52_0 + x52_0
```

```
r52_4 = 0
p52_4 = Exp(-2.4)
F52_4 = p52_4
If answer < F52_4 Then
  x52_4 = r52_4
Else
  Do Until answer < F52_4
    p52_4 = (2.4 * p52_4) / (r52_4 + 1)
    F52_4 = F52_4 + p52_4
    r52_4 = r52_4 + 1
    x52_4 = r52_4
  Loop
End If
s52_4 = s52_4 + x52_4
```

```
r52_9 = 0
p52_9 = Exp(-2.9)
F52_9 = p52_9
If answer < F52_9 Then
  x52_9 = r52_9
Else
```

```
Do Until answer < F52_9
    p52_9 = (2.9 * p52_9) / (r52_9 + 1)
    F52_9 = F52_9 + p52_9
    r52_9 = r52_9 + 1
    x52_9 = r52_9
Loop
End If
s52_9 = s52_9 + x52_9
```

End Function

Function cal_est_poi1()

```
epp11_3 = s11_3 / sample
m1p1_3 = (epp11_3 - 1.3) ^ 2
sp_m11_3 = sp_m11_3 + m1p1_3

epp11_8 = s11_8 / sample
m1p1_8 = (epp11_8 - 1.8) ^ 2
sp_m11_8 = sp_m11_8 + m1p1_8

epp12_0 = s12_0 / sample
m1p2_0 = (epp12_0 - 2) ^ 2
sp_m12_0 = sp_m12_0 + m1p2_0

epp12_4 = s12_4 / sample
m1p2_4 = (epp12_4 - 2.4) ^ 2
sp_m12_4 = sp_m12_4 + m1p2_4

epp12_9 = s12_9 / sample
m1p2_9 = (epp12_9 - 2.9) ^ 2
sp_m12_9 = sp_m12_9 + m1p2_9
```

End Function

Function cal_est_poi2()

```
epp21_3 = s21_3 / sample
m2p1_3 = (epp21_3 - 1.3) ^ 2
sp_m21_3 = sp_m21_3 + m2p1_3

epp21_8 = s21_8 / sample
m2p1_8 = (epp21_8 - 1.8) ^ 2
sp_m21_8 = sp_m21_8 + m2p1_8

epp22_0 = s22_0 / sample
m2p2_0 = (epp22_0 - 2) ^ 2
```

$sp_m22_0 = sp_m22_0 + m2p2_0$

$epp22_4 = s22_4 / sample$
 $m2p2_4 = (epp22_4 - 2.4) ^ 2$
 $sp_m22_4 = sp_m22_4 + m2p2_4$

$epp22_9 = s22_9 / sample$
 $m2p2_9 = (epp22_9 - 2.9) ^ 2$
 $sp_m22_9 = sp_m22_9 + m2p2_9$

End Function

Function cal_est_poi3()

$epp31_3 = s31_3 / sample$
 $m3p1_3 = (epp31_3 - 1.3) ^ 2$
 $sp_m31_3 = sp_m31_3 + m3p1_3$

$epp31_8 = s31_8 / sample$
 $m3p1_8 = (epp31_8 - 1.8) ^ 2$
 $sp_m31_8 = sp_m31_8 + m3p1_8$

$epp32_0 = s32_0 / sample$
 $m3p2_0 = (epp32_0 - 2) ^ 2$
 $sp_m32_0 = sp_m32_0 + m3p2_0$

$epp32_4 = s32_4 / sample$
 $m3p2_4 = (epp32_4 - 2.4) ^ 2$
 $sp_m32_4 = sp_m32_4 + m3p2_4$

$epp32_9 = s32_9 / sample$
 $m3p2_9 = (epp32_9 - 2.9) ^ 2$
 $sp_m32_9 = sp_m32_9 + m3p2_9$

End Function

Function cal_est_poi4()

$epp41_3 = s41_3 / sample$
 $m4p1_3 = (epp41_3 - 1.3) ^ 2$
 $sp_m41_3 = sp_m41_3 + m4p1_3$

$epp41_8 = s41_8 / sample$
 $m4p1_8 = (epp41_8 - 1.8) ^ 2$
 $sp_m41_8 = sp_m41_8 + m4p1_8$

$epp42_0 = s42_0 / sample$


```
m4p2_0 = (epp42_0 - 2) ^ 2  
sp_m42_0 = sp_m42_0 + m4p2_0
```

```
epp42_4 = s42_4 / sample  
m4p2_4 = (epp42_4 - 2.4) ^ 2  
sp_m42_4 = sp_m42_4 + m4p2_4
```

```
epp42_9 = s42_9 / sample  
m4p2_9 = (epp42_9 - 2.9) ^ 2  
sp_m42_9 = sp_m42_9 + m4p2_9
```

End Function

Function cal_est_poi5()

```
epp51_3 = s51_3 / sample  
m5p1_3 = (epp51_3 - 1.3) ^ 2  
sp_m51_3 = sp_m51_3 + m5p1_3
```

```
epp51_8 = s51_8 / sample  
m5p1_8 = (epp51_8 - 1.8) ^ 2  
sp_m51_8 = sp_m51_8 + m5p1_8
```

```
epp52_0 = s52_0 / sample  
m5p2_0 = (epp52_0 - 2) ^ 2  
sp_m52_0 = sp_m52_0 + m5p2_0
```

```
epp52_4 = s52_4 / sample  
m5p2_4 = (epp52_4 - 2.4) ^ 2  
sp_m52_4 = sp_m52_4 + m5p2_4
```

```
epp52_9 = s52_9 / sample  
m5p2_9 = (epp52_9 - 2.9) ^ 2  
sp_m52_9 = sp_m52_9 + m5p2_9
```

End Function

Function mse_poisson1()

```
p1mle1_3 = (sp_m11_3 / loop_test)  
rp1mle1_3 = Round(Sqr(p1mle1_3), 4)
```

```
p1mle1_8 = (sp_m11_8 / loop_test)  
rp1mle1_8 = Round(Sqr(p1mle1_8), 4)
```

```
p1mle2_0 = (sp_m12_0 / loop_test)  
rp1mle2_0 = Round(Sqr(p1mle2_0), 4)
```

```
p1mle2_4 = (sp_m12_4 / loop_test)
rp1mle2_4 = Round(Sqr(p1mle2_4), 4)
```

```
p1mle2_9 = (sp_m12_9 / loop_test)
rp1mle2_9 = Round(Sqr(p1mle2_9), 4)
```

```
Print "
Print "M1: RMSE POI 1.3 = " & rp1mle1_3, "1.8 = " & rp1mle1_8, "2.0= " &
rp1mle2_0, "2.4 = " & rp1mle2_4, "2.9 = " & rp1mle2_9
```

End Function

Function mse_poisson2()

```
p2mle1_3 = (sp_m21_3 / loop_test)
rp2mle1_3 = Round(Sqr(p2mle1_3), 4)
```

```
p2mle1_8 = (sp_m21_8 / loop_test)
rp2mle1_8 = Round(Sqr(p2mle1_8), 4)
```

```
p2mle2_0 = (sp_m22_0 / loop_test)
rp2mle2_0 = Round(Sqr(p2mle2_0), 4)
```

```
p2mle2_4 = (sp_m22_4 / loop_test)
rp2mle2_4 = Round(Sqr(p2mle2_4), 4)
```

```
p2mle2_9 = (sp_m22_9 / loop_test)
rp2mle2_9 = Round(Sqr(p2mle2_9), 4)
```

```
Print "M2 : RMSE POI 1.3 = " & rp2mle1_3, "1.8 = " & rp2mle1_8, "2.0= " &
rp2mle2_0, "2.4 = " & rp2mle2_4, "2.9 = " & rp2mle2_9
```

End Function

Function mse_poisson3()

```
p3mle1_3 = (sp_m31_3 / loop_test)
rp3mle1_3 = Round(Sqr(p3mle1_3), 4)
```

```
p3mle1_8 = (sp_m31_8 / loop_test)
rp3mle1_8 = Round(Sqr(p3mle1_8), 4)
```

```
p3mle2_0 = (sp_m32_0 / loop_test)
rp3mle2_0 = Round(Sqr(p3mle2_0), 4)
```

```
p3mle2_4 = (sp_m32_4 / loop_test)
```

```
rp3mle2_4 = Round(Sqr(p3mle2_4), 4)
p3mle2_9 = (sp_m32_9 / loop_test)
rp3mle2_9 = Round(Sqr(p3mle2_9), 4)
```

```
Print "M3 : RMSE POI 1.3 = " & rp3mle1_3, "1.8 = " & rp3mle1_8, "2.0= " &
rp3mle2_0, "2.4 = " & rp3mle2_4, "2.9 = " & rp3mle2_9
```

End Function

Function mse_poisson4()

```
p4mle1_3 = (sp_m41_3 / loop_test)
rp4mle1_3 = Round(Sqr(p4mle1_3), 4)

p4mle1_8 = (sp_m41_8 / loop_test)
rp4mle1_8 = Round(Sqr(p4mle1_8), 4)

p4mle2_0 = (sp_m42_0 / loop_test)
rp4mle2_0 = Round(Sqr(p4mle2_0), 4)

p4mle2_4 = (sp_m42_4 / loop_test)
rp4mle2_4 = Round(Sqr(p4mle2_4), 4)

p4mle2_9 = (sp_m42_9 / loop_test)
rp4mle2_9 = Round(Sqr(p4mle2_9), 4)
```

```
Print "M4 : RMSE POI 1.3 = " & rp4mle1_3, "1.8 = " & rp4mle1_8, "2.0= " &
rp4mle2_0, "2.4 = " & rp4mle2_4, "2.9 = " & rp4mle2_9
```

End Function

Function mse_poisson5()

```
p5mle1_3 = (sp_m51_3 / loop_test)
rp5mle1_3 = Round(Sqr(p5mle1_3), 4)

p5mle1_8 = (sp_m51_8 / loop_test)
rp5mle1_8 = Round(Sqr(p5mle1_8), 4)

p5mle2_0 = (sp_m52_0 / loop_test)
rp5mle2_0 = Round(Sqr(p5mle2_0), 4)

p5mle2_4 = (sp_m52_4 / loop_test)
rp5mle2_4 = Round(Sqr(p5mle2_4), 4)

p5mle2_9 = (sp_m52_9 / loop_test)
rp5mle2_9 = Round(Sqr(p5mle2_9), 4)
```

Print "M5 : RMSE POI 1.3 = " & rp5mle1_3, "1.8 =" & rp5mle1_8, " 2.0=" &
rp5mle2_0, "2.4 =" & rp5mle2_4, "2.9 =" & rp5mle2_9

End Function

Function binomial1()

```
M1i20_01 = 0
M1c20_01 = 0.1 / (1 - 0.1)
M1pr20_01 = (1 - 0.1) ^ 20
M1F20_01 = M1pr20_01
If answer < M1F20_01 Then
    M1x20_01 = M1i20_01
Else
    Do Until answer < M1F20_01
        M1pr20_01 = (M1c20_01 * (20 - M1i20_01) / (M1i20_01 + 1)) *
M1pr20_01
        M1F20_01 = M1F20_01 + M1pr20_01
        M1i20_01 = M1i20_01 + 1
        M1x20_01 = M1i20_01
    Loop
End If
s_M1x20_01 = s_M1x20_01 + M1x20_01

M1i30_03 = 0
M1c30_03 = 0.3 / (1 - 0.3)
M1pr30_03 = (1 - 0.3) ^ 30
M1F30_03 = M1pr30_03
If answer < M1F30_03 Then
    M1x30_03 = M1i30_03
Else
    Do Until answer < M1F30_03
        M1pr30_03 = (M1c30_03 * (30 - M1i30_03) / (M1i30_03 + 1)) *
M1pr30_03
        M1F30_03 = M1F30_03 + M1pr30_03
        M1i30_03 = M1i30_03 + 1
        M1x30_03 = M1i30_03
    Loop
End If
s_M1x30_03 = s_M1x30_03 + M1x30_03

M1i40_05 = 0
M1c40_05 = 0.5 / (1 - 0.5)
M1pr40_05 = (1 - 0.5) ^ 40
M1F40_05 = M1pr40_05
If answer < M1F40_05 Then
    M1x40_05 = M1i40_05
```

```
Else
  Do Until answer < M1F40_05
    M1pr40_05 = (M1c40_05 * (40 - M1i40_05) / (M1i40_05 + 1)) *
M1pr40_05
    M1F40_05 = M1F40_05 + M1pr40_05
    M1i40_05 = M1i40_05 + 1
    M1x40_05 = M1i40_05
  Loop
End If
s_M1x40_05 = s_M1x40_05 + M1x40_05

M1i50_06 = 0
M1c50_06 = 0.6 / (1 - 0.6)
M1pr50_06 = (1 - 0.6) ^ 50
M1F50_06 = M1pr50_06
If answer < M1F50_06 Then
  M1x50_06 = M1i50_06
Else
  Do Until answer < M1F50_06
    M1pr50_06 = (M1c50_06 * (50 - M1i50_06) / (M1i50_06 + 1)) *
M1pr50_06
    M1F50_06 = M1F50_06 + M1pr50_06
    M1i50_06 = M1i50_06 + 1
    M1x50_06 = M1i50_06
  Loop
End If
s_M1x50_06 = s_M1x50_06 + M1x50_06

M1i70_08 = 0
M1c70_08 = 0.8 / (1 - 0.8)
M1pr70_08 = (1 - 0.8) ^ 70
M1F70_08 = M1pr70_08
If answer < M1F70_08 Then
  M1x70_08 = M1i70_08
Else
  Do Until answer < M1F70_08
    M1pr70_08 = (M1c70_08 * (70 - M1i70_08) / (M1i70_08 + 1)) *
M1pr70_08
    M1F70_08 = M1F70_08 + M1pr70_08
    M1i70_08 = M1i70_08 + 1
    M1x70_08 = M1i70_08
  Loop
End If
s_M1x70_08 = s_M1x70_08 + M1x70_08

End Function
```

Function binomial2()

```
M2i20_01 = 0
M2c20_01 = 0.1 / (1 - 0.1)
M2pr20_01 = (1 - 0.1) ^ 20
M2F20_01 = M2pr20_01
If answer < M2F20_01 Then
    M2x20_01 = M2i20_01
Else
    Do Until answer < M2F20_01
        M2pr20_01 = (M2c20_01 * (20 - M2i20_01) / (M2i20_01 + 1)) *
M2pr20_01
        M2F20_01 = M2F20_01 + M2pr20_01
        M2i20_01 = M2i20_01 + 1
        M2x20_01 = M2i20_01
    Loop
End If
s_M2x20_01 = s_M2x20_01 + M2x20_01

M2i30_03 = 0
M2c30_03 = 0.3 / (1 - 0.3)
M2pr30_03 = (1 - 0.3) ^ 30
M2F30_03 = M2pr30_03
If answer < M2F30_03 Then
    M2x30_03 = M2i30_03
Else
    Do Until answer < M2F30_03
        M2pr30_03 = (M2c30_03 * (30 - M2i30_03) / (M2i30_03 + 1)) *
M2pr30_03
        M2F30_03 = M2F30_03 + M2pr30_03
        M2i30_03 = M2i30_03 + 1
        M2x30_03 = M2i30_03
    Loop
End If
s_M2x30_03 = s_M2x30_03 + M2x30_03

M2i40_05 = 0
M2c40_05 = 0.5 / (1 - 0.5)
M2pr40_05 = (1 - 0.5) ^ 40
M2F40_05 = M2pr40_05
If answer < M2F40_05 Then
    M2x40_05 = M2i40_05
Else
    Do Until answer < M2F40_05
        M2pr40_05 = (M2c40_05 * (40 - M2i40_05) / (M2i40_05 + 1)) *
M2pr40_05
        M2F40_05 = M2F40_05 + M2pr40_05
```

```
M2i40_05 = M2i40_05 + 1
M2x40_05 = M2i40_05
Loop
End If
s_M2x40_05 = s_M2x40_05 + M2x40_05

M2i50_06 = 0
M2c50_06 = 0.6 / (1 - 0.6)
M2pr50_06 = (1 - 0.6) ^ 50
M2F50_06 = M2pr50_06
If answer < M2F50_06 Then
    M2x50_06 = M2i50_06
Else
    Do Until answer < M2F50_06
        M2pr50_06 = (M2c50_06 * (50 - M2i50_06) / (M2i50_06 + 1)) *
M2pr50_06
        M2F50_06 = M2F50_06 + M2pr50_06
        M2i50_06 = M2i50_06 + 1
        M2x50_06 = M2i50_06
    Loop
End If
s_M2x50_06 = s_M2x50_06 + M2x50_06

M2i70_08 = 0
M2c70_08 = 0.8 / (1 - 0.8)
M2pr70_08 = (1 - 0.8) ^ 70
M2F70_08 = M2pr70_08
If answer < M2F70_08 Then
    M2x70_08 = M2i70_08
Else
    Do Until answer < M2F70_08
        M2pr70_08 = (M2c70_08 * (70 - M2i70_08) / (M2i70_08 + 1)) *
M2pr70_08
        M2F70_08 = M2F70_08 + M2pr70_08
        M2i70_08 = M2i70_08 + 1
        M2x70_08 = M2i70_08
    Loop
End If
s_M2x70_08 = s_M2x70_08 + M2x70_08
```

End Function

Function binomial3()

```
M3i20_01 = 0
M3c20_01 = 0.1 / (1 - 0.1)
M3pr20_01 = (1 - 0.1) ^ 20
M3F20_01 = M3pr20_01
```

```
If answer < M3F20_01 Then
    M3x20_01 = M3i20_01
Else
    Do Until answer < M3F20_01
        M3pr20_01 = (M3c20_01 * (20 - M3i20_01) / (M3i20_01 + 1)) *
M3pr20_01
        M3F20_01 = M3F20_01 + M3pr20_01
        M3i20_01 = M3i20_01 + 1
        M3x20_01 = M3i20_01
    Loop
End If
s_M3x20_01 = s_M3x20_01 + M3x20_01

M3i30_03 = 0
M3c30_03 = 0.3 / (1 - 0.3)
M3pr30_03 = (1 - 0.3) ^ 30
M3F30_03 = M3pr30_03
If answer < M3F30_03 Then
    M3x30_03 = M3i30_03
Else
    Do Until answer < M3F30_03
        M3pr30_03 = (M3c30_03 * (30 - M3i30_03) / (M3i30_03 + 1)) *
M3pr30_03
        M3F30_03 = M3F30_03 + M3pr30_03
        M3i30_03 = M3i30_03 + 1
        M3x30_03 = M3i30_03
    Loop
End If
s_M3x30_03 = s_M3x30_03 + M3x30_03

M3i40_05 = 0
M3c40_05 = 0.5 / (1 - 0.5)
M3pr40_05 = (1 - 0.5) ^ 40
M3F40_05 = M3pr40_05
If answer < M3F40_05 Then
    M3x40_05 = M3i40_05
Else
    Do Until answer < M3F40_05
        M3pr40_05 = (M3c40_05 * (40 - M3i40_05) / (M3i40_05 + 1)) *
M3pr40_05
        M3F40_05 = M3F40_05 + M3pr40_05
        M3i40_05 = M3i40_05 + 1
        M3x40_05 = M3i40_05
    Loop
End If
s_M3x40_05 = s_M3x40_05 + M3x40_05
```



```
M3i50_06 = 0
M3c50_06 = 0.6 / (1 - 0.6)
M3pr50_06 = (1 - 0.6) ^ 50
M3F50_06 = M3pr50_06
If answer < M3F50_06 Then
    M3x50_06 = M3i50_06
Else
    Do Until answer < M3F50_06
        M3pr50_06 = (M3c50_06 * (50 - M3i50_06) / (M3i50_06 + 1)) *
M3pr50_06
        M3F50_06 = M3F50_06 + M3pr50_06
        M3i50_06 = M3i50_06 + 1
        M3x50_06 = M3i50_06
    Loop
End If
s_M3x50_06 = s_M3x50_06 + M3x50_06

M3i70_08 = 0
M3c70_08 = 0.8 / (1 - 0.8)
M3pr70_08 = (1 - 0.8) ^ 70
M3F70_08 = M3pr70_08
If answer < M3F70_08 Then
    M3x70_08 = M2i70_08
Else
    Do Until answer < M3F70_08
        M3pr70_08 = (M3c70_08 * (70 - M3i70_08) / (M3i70_08 + 1)) *
M3pr70_08
        M3F70_08 = M3F70_08 + M3pr70_08
        M3i70_08 = M3i70_08 + 1
        M3x70_08 = M3i70_08
    Loop
End If
s_M3x70_08 = s_M3x70_08 + M3x70_08
```

End Function

Function binomial4()

```
M4i20_01 = 0
M4c20_01 = 0.1 / (1 - 0.1)
M4pr20_01 = (1 - 0.1) ^ 20
M4F20_01 = M4pr20_01
If answer < M4F20_01 Then
    M4x20_01 = M4i20_01
Else
    Do Until answer < M4F20_01
```

```
M4pr20_01 = (M4c20_01 * (20 - M4i20_01) / (M4i20_01 + 1)) *
M4pr20_01
M4F20_01 = M4F20_01 + M4pr20_01
M4i20_01 = M4i20_01 + 1
M4x20_01 = M4i20_01
Loop
End If
s_M4x20_01 = s_M4x20_01 + M4x20_01

M4i30_03 = 0
M4c30_03 = 0.3 / (1 - 0.3)
M4pr30_03 = (1 - 0.3) ^ 30
M4F30_03 = M4pr30_03
If answer < M4F30_03 Then
    M4x30_03 = M4i30_03
Else
    Do Until answer < M4F30_03
        M4pr30_03 = (M4c30_03 * (30 - M4i30_03) / (M4i30_03 + 1)) *
M4pr30_03
        M4F30_03 = M4F30_03 + M4pr30_03
        M4i30_03 = M4i30_03 + 1
        M4x30_03 = M4i30_03
    Loop
End If
s_M4x30_03 = s_M4x30_03 + M4x30_03

M4i40_05 = 0
M4c40_05 = 0.5 / (1 - 0.5)
M4pr40_05 = (1 - 0.5) ^ 40
M4F40_05 = M4pr40_05
If answer < M4F40_05 Then
    M4x40_05 = M4i40_05
Else
    Do Until answer < M4F40_05
        M4pr40_05 = (M4c40_05 * (40 - M4i40_05) / (M4i40_05 + 1)) *
M4pr40_05
        M4F40_05 = M4F40_05 + M4pr40_05
        M4i40_05 = M4i40_05 + 1
        M4x40_05 = M4i40_05
    Loop
End If
s_M4x40_05 = s_M4x40_05 + M4x40_05

M4i50_06 = 0
M4c50_06 = 0.6 / (1 - 0.6)
M4pr50_06 = (1 - 0.6) ^ 50
M4F50_06 = M4pr50_06
```

```
If answer < M4F50_06 Then
    M4x50_06 = M4i50_06
Else
    Do Until answer < M4F50_06
        M4pr50_06 = (M4c50_06 * (50 - M4i50_06) / (M4i50_06 + 1)) *
M4pr50_06
        M4F50_06 = M4F50_06 + M4pr50_06
        M4i50_06 = M4i50_06 + 1
        M4x50_06 = M4i50_06
    Loop
End If
s_M4x50_06 = s_M4x50_06 + M4x50_06
```

```
M4i70_08 = 0
M4c70_08 = 0.8 / (1 - 0.8)
M4pr70_08 = (1 - 0.8) ^ 70
M4F70_08 = M4pr70_08
If answer < M4F70_08 Then
    M4x70_08 = M4i70_08
Else
    Do Until answer < M4F70_08
        M4pr70_08 = (M4c70_08 * (70 - M4i70_08) / (M4i70_08 + 1)) *
M4pr70_08
        M4F70_08 = M4F70_08 + M4pr70_08
        M4i70_08 = M4i70_08 + 1
        M4x70_08 = M4i70_08
    Loop
End If
s_M4x70_08 = s_M4x70_08 + M4x70_08
```

End Function

Function binomial5()

```
M5i20_01 = 0
M5c20_01 = 0.1 / (1 - 0.1)
M5pr20_01 = (1 - 0.1) ^ 20
M5F20_01 = M5pr20_01
If answer < M5F20_01 Then
    M5x20_01 = M5i20_01
Else
    Do Until answer < M5F20_01
        M5pr20_01 = (M5c20_01 * (20 - M5i20_01) / (M5i20_01 + 1)) *
M5pr20_01
        M5F20_01 = M5F20_01 + M5pr20_01
        M5i20_01 = M5i20_01 + 1
        M5x20_01 = M5i20_01
```

```

    Loop
End If
s_M5x20_01 = s_M5x20_01 + M5x20_01

M5i30_03 = 0
M5c30_03 = 0.3 / (1 - 0.3)
M5pr30_03 = (1 - 0.3) ^ 30
M5F30_03 = M5pr30_03
If answer < M5F30_03 Then
    M5x30_03 = M5i30_03
Else
    Do Until answer < M5F30_03
        M5pr30_03 = (M5c30_03 * (30 - M5i30_03) / (M5i30_03 + 1)) *
M5pr30_03
        M5F30_03 = M5F30_03 + M5pr30_03
        M5i30_03 = M5i30_03 + 1
        M5x30_03 = M5i30_03
    Loop
End If
s_M5x30_03 = s_M5x30_03 + M5x30_03

M5i40_05 = 0
M5c40_05 = 0.5 / (1 - 0.5)
M5pr40_05 = (1 - 0.5) ^ 40
M5F40_05 = M5pr40_05
If answer < M5F40_05 Then
    M5x40_05 = M5i40_05
Else
    Do Until answer < M5F40_05
        M5pr40_05 = (M5c40_05 * (40 - M5i40_05) / (M5i40_05 + 1)) *
M5pr40_05
        M5F40_05 = M5F40_05 + M5pr40_05
        M5i40_05 = M5i40_05 + 1
        M5x40_05 = M5i40_05
    Loop
End If
s_M5x40_05 = s_M5x40_05 + M5x40_05

M5i50_06 = 0
M5c50_06 = 0.6 / (1 - 0.6)
M5pr50_06 = (1 - 0.6) ^ 50
M5F50_06 = M5pr50_06
If answer < M5F50_06 Then
    M5x50_06 = M5i50_06
Else
    Do Until answer < M5F50_06
```

```

M5pr50_06 = (M5c50_06 * (50 - M5i50_06) / (M5i50_06 + 1)) *
M5pr50_06
M5F50_06 = M5F50_06 + M5pr50_06
M5i50_06 = M5i50_06 + 1
M5x50_06 = M5i50_06
Loop
End If
s_M5x50_06 = s_M5x50_06 + M5x50_06
```

```

M5i70_08 = 0
M5c70_08 = 0.8 / (1 - 0.8)
M5pr70_08 = (1 - 0.8) ^ 70
M5F70_08 = M5pr70_08
If answer < M5F70_08 Then
    M5x70_08 = M5i70_08
Else
    Do Until answer < M5F70_08
        M5pr70_08 = (M5c70_08 * (70 - M5i70_08) / (M5i70_08 + 1)) *
M5pr70_08
        M5F70_08 = M5F70_08 + M5pr70_08
        M5i70_08 = M5i70_08 + 1
        M5x70_08 = M5i70_08
    Loop
End If
s_M5x70_08 = s_M5x70_08 + M5x70_08
```

End Function

Function cal_est_binomial1()

```

eBM120_01 = s_M1x20_01 / sample
M120_01 = (eBM120_01 - (20 * 0.1)) ^ 2
s_M120_01 = s_M120_01 + M120_01
```

```

eBM130_03 = s_M1x30_03 / sample
M130_03 = (eBM130_03 - (30 * 0.3)) ^ 2
s_M130_03 = s_M130_03 + M130_03
```

```

eBM140_05 = s_M1x40_05 / sample
M140_05 = (eBM140_05 - (40 * 0.5)) ^ 2
s_M140_05 = s_M140_05 + M140_05
```

```

eBM150_06 = s_M1x50_06 / sample
M150_06 = (eBM150_06 - (50 * 0.6)) ^ 2
s_M150_06 = s_M150_06 + M150_06
```

```

eBM170_08 = s_M1x70_08 / sample
```

```
M170_08 = (eBM170_08 - (70 * 0.8)) ^ 2  
s_M170_08 = s_M170_08 + M170_08
```

End Function

Function cal_est_binomial2()

```
eBM220_01 = s_M2x20_01 / sample  
M220_01 = (eBM220_01 - (20 * 0.1)) ^ 2  
s_M220_01 = s_M220_01 + M220_01
```

```
eBM230_03 = s_M2x30_03 / sample  
M230_03 = (eBM230_03 - (30 * 0.3)) ^ 2  
s_M230_03 = s_M230_03 + M230_03
```

```
eBM240_05 = s_M2x40_05 / sample  
M240_05 = (eBM240_05 - (40 * 0.5)) ^ 2  
s_M240_05 = s_M240_05 + M240_05
```

```
eBM250_06 = s_M2x50_06 / sample  
M250_06 = (eBM250_06 - (50 * 0.6)) ^ 2  
s_M250_06 = s_M250_06 + M250_06
```

```
eBM270_08 = s_M2x70_08 / sample  
M270_08 = (eBM270_08 - (70 * 0.8)) ^ 2  
s_M270_08 = s_M270_08 + M270_08
```

End Function

Function cal_est_binomial3()

```
eBM320_01 = s_M3x20_01 / sample  
M320_01 = (eBM320_01 - (20 * 0.1)) ^ 2  
s_M320_01 = s_M320_01 + M320_01
```

```
eBM330_03 = s_M3x30_03 / sample  
M330_03 = (eBM330_03 - (30 * 0.3)) ^ 2  
s_M330_03 = s_M330_03 + M330_03
```

```
eBM340_05 = s_M3x40_05 / sample  
M340_05 = (eBM340_05 - (40 * 0.5)) ^ 2  
s_M340_05 = s_M340_05 + M340_05
```

```
eBM350_06 = s_M3x50_06 / sample  
M350_06 = (eBM350_06 - (50 * 0.6)) ^ 2  
s_M350_06 = s_M350_06 + M350_06
```

```
eBM370_08 = s_M3x70_08 / sample  
M370_08 = (eBM370_08 - (70 * 0.8)) ^ 2  
s_M370_08 = s_M370_08 + M370_08
```

End Function

Function cal_est_binomial4()

eBM420_01 = s_M4x20_01 / sample
M420_01 = (eBM420_01 - (20 * 0.1)) ^ 2
s_M420_01 = s_M420_01 + M420_01

eBM430_03 = s_M4x30_03 / sample
M430_03 = (eBM430_03 - (30 * 0.3)) ^ 2
s_M430_03 = s_M430_03 + M430_03

eBM440_05 = s_M4x40_05 / sample
M440_05 = (eBM440_05 - (40 * 0.5)) ^ 2
s_M440_05 = s_M440_05 + M440_05

eBM450_06 = s_M4x50_06 / sample
M450_06 = (eBM450_06 - (50 * 0.6)) ^ 2
s_M450_06 = s_M450_06 + M450_06

eBM470_08 = s_M4x70_08 / sample
M470_08 = (eBM470_08 - (70 * 0.8)) ^ 2
s_M470_08 = s_M470_08 + M470_08

End Function

Function cal_est_binomial5()

eBM520_01 = s_M5x20_01 / sample
M520_01 = (eBM520_01 - (20 * 0.1)) ^ 2
s_M520_01 = s_M520_01 + M520_01

eBM530_03 = s_M5x30_03 / sample
M530_03 = (eBM530_03 - (30 * 0.3)) ^ 2
s_M530_03 = s_M530_03 + M530_03

eBM540_05 = s_M5x40_05 / sample
M540_05 = (eBM540_05 - (40 * 0.5)) ^ 2
s_M540_05 = s_M540_05 + M540_05

eBM550_06 = s_M5x50_06 / sample
M550_06 = (eBM550_06 - (50 * 0.6)) ^ 2
s_M550_06 = s_M550_06 + M550_06

eBM570_08 = s_M5x70_08 / sample
M570_08 = (eBM570_08 - (70 * 0.8)) ^ 2
s_M570_08 = s_M570_08 + M570_08

End Function

Function mse_binomial1()

```
mlem120_01 = (s_M120_01 / loop_test)
rmlem120_01 = Round(Sqr(mlem120_01), 4)
```

```
mlem130_03 = (s_M130_03 / loop_test)
rmlem130_03 = Round(Sqr(mlem130_03), 4)
```

```
mlem140_05 = (s_M140_05 / loop_test)
rmlem140_05 = Round(Sqr(mlem140_05), 4)
```

```
mlem150_06 = (s_M150_06 / loop_test)
rmlem150_06 = Round(Sqr(mlem150_06), 4)
```

```
mlem170_08 = (s_M170_08 / loop_test)
rmlem170_08 = Round(Sqr(mlem170_08), 4)
```

```
Print "
Print "M1: RMSE BI 20,0.1=" & rmlem120_01, "30,0.3=" & rmlem130_03, "40,0.5="
& rmlem140_05, "50,0.6=" & rmlem150_06, "70,0.8=" & rmlem170_08
```

End Function

Function mse_binomial2()

```
mlem220_01 = (s_M220_01 / loop_test)
rmlem220_01 = Round(Sqr(mlem220_01), 4)
```

```
mlem230_03 = (s_M230_03 / loop_test)
rmlem230_03 = Round(Sqr(mlem230_03), 4)
```

```
mlem240_05 = (s_M240_05 / loop_test)
rmlem240_05 = Round(Sqr(mlem240_05), 4)
```

```
mlem250_06 = (s_M250_06 / loop_test)
rmlem250_06 = Round(Sqr(mlem250_06), 4)
```

```
mlem270_08 = (s_M270_08 / loop_test)
rmlem270_08 = Round(Sqr(mlem270_08), 4)
```

```
Print "M2: RMSE BI 20,0.1=" & rmlem220_01, "30,0.3=" & rmlem230_03, "40,0.5="
& rmlem240_05, "50,0.6=" & rmlem250_06, "70,0.8=" & rmlem270_08
```


End Function

Function mse_binomial3()

```
mlem320_01 = (s_M320_01 / loop_test)
rmlem320_01 = Round(Sqr(mlem320_01), 4)
```

```
mlem330_03 = (s_M330_03 / loop_test)
rmlem330_03 = Round(Sqr(mlem330_03), 4)
```

```
mlem340_05 = (s_M340_05 / loop_test)
rmlem340_05 = Round(Sqr(mlem340_05), 4)
```

```
mlem350_06 = (s_M350_06 / loop_test)
rmlem350_06 = Round(Sqr(mlem350_06), 4)
```

```
mlem370_08 = (s_M370_08 / loop_test)
rmlem370_08 = Round(Sqr(mlem370_08), 4)
```

```
Print "M3: RMSE BI 20,0.1=" & rmlem320_01, "30,0.3=" & rmlem330_03, "40,0.5="
& rmlem340_05, "50,0.6=" & rmlem350_06, "70,0.8=" & rmlem370_08
```

End Function

Function mse_binomial4()

```
mlem420_01 = (s_M420_01 / loop_test)
rmlem420_01 = Round(Sqr(mlem420_01), 4)
```

```
mlem430_03 = (s_M430_03 / loop_test)
rmlem430_03 = Round(Sqr(mlem430_03), 4)
```

```
mlem440_05 = (s_M440_05 / loop_test)
rmlem440_05 = Round(Sqr(mlem440_05), 4)
```

```
mlem450_06 = (s_M450_06 / loop_test)
rmlem450_06 = Round(Sqr(mlem450_06), 4)
```

```
mlem470_08 = (s_M470_08 / loop_test)
rmlem470_08 = Round(Sqr(mlem470_08), 4)
```

```
Print "M4: RMSE BI 20,0.1=" & rmlem420_01, "30,0.3=" & rmlem430_03, "40,0.5="
& rmlem440_05, "50,0.6=" & rmlem450_06, "70,0.8=" & rmlem470_08
```

End Function

Function mse_binomial5()

```
mlem520_01 = (s_M520_01 / loop_test)
rmlem520_01 = Round(Sqr(mlem520_01), 4)
```

```
mlem530_03 = (s_M530_03 / loop_test)
rmlem530_03 = Round(Sqr(mlem530_03), 4)
```

```
mlem540_05 = (s_M540_05 / loop_test)
rmlem540_05 = Round(Sqr(mlem540_05), 4)
```

```
mlem550_06 = (s_M550_06 / loop_test)
rmlem550_06 = Round(Sqr(mlem550_06), 4)
```

```
mlem570_08 = (s_M570_08 / loop_test)
rmlem570_08 = Round(Sqr(mlem570_08), 4)
```

```
Print "M5: RMSE BI 20,0.1=" & rmlem520_01, "30,0.3=" & rmlem530_03, "40,0.5="
& rmlem540_05, "50,0.6=" & rmlem550_06, "70,0.8=" & rmlem570_08
```

End Function

Function normal1()

```
M1y10_1 = -1 * Log(answer)
```

```
If answer <= 1 / 2 Then
```

```
    M1z0_1 = (M1y10_1 - 0) / 1
```

```
Else
```

```
    M1z0_1 = (-M1y10_1 - 0) / 1
```

```
End If
```

```
s_M1z0_1 = s_M1z0_1 + M1z0_1
```

```
normal1_01(i) = M1z0_1
```

```
M1y11_3 = -1 * Log(answer)
```

```
If answer <= 1 / 2 Then
```

```
    M1z1_3 = (M1y11_3 - 1) / 3
```

```
Else
```

```
    M1z1_3 = (-M1y11_3 - 1) / 3
```

```
End If
s_M1z1_3 = s_M1z1_3 + M1z1_3
normal1_13(i) = M1z1_3

M1y12_5 = -1 * Log(answer)
If answer <= 1 / 2 Then

    M1z2_5 = (M1y12_5 - 2) / 5

Else

    M1z2_5 = (-M1y12_5 - 2) / 5

End If
s_M1z2_5 = s_M1z2_5 + M1z2_5
normal1_25(i) = M1z2_5

M1y13_6 = -1 * Log(answer)
If answer <= 1 / 2 Then

    M1z3_6 = (M1y13_6 - 3) / 6

Else

    M1z3_6 = (-M1y13_6 - 3) / 6

End If
s_M1z3_6 = s_M1z3_6 + M1z3_6
normal1_36(i) = M1z3_6

M1y14_5 = -1 * Log(answer)
If answer <= 1 / 2 Then

    M1z4_5 = (M1y14_5 - 4) / 5

Else

    M1z4_5 = (-M1y14_5 - 4) / 5

End If
s_M1z4_5 = s_M1z4_5 + M1z4_5
normal1_45(i) = M1z4_5
```

End Function

Function normal2()

```
M2y10_1 = -1 * Log(answer)
If answer <= 1 / 2 Then
```

$$M2z0_1 = (M2y10_1 - 0) / 1$$

Else

$$M2z0_1 = (-M2y10_1 - 0) / 1$$

End If

$$s_M2z0_1 = s_M2z0_1 + M2z0_1$$

$$\text{normal2_01}(i) = M2z0_1$$

$$M2y11_3 = -1 * \text{Log}(\text{answer})$$

If answer <= 1 / 2 Then

$$M2z1_3 = (M2y11_3 - 1) / 3$$

Else

$$M2z1_3 = (-M2y11_3 - 1) / 3$$

End If

$$s_M2z1_3 = s_M2z1_3 + M2z1_3$$

$$\text{normal2_13}(i) = M2z1_3$$

$$M2y12_5 = -1 * \text{Log}(\text{answer})$$

If answer <= 1 / 2 Then

$$M2z2_5 = (M2y12_5 - 2) / 5$$

Else

$$M2z2_5 = (-M2y12_5 - 2) / 5$$

End If

$$s_M2z2_5 = s_M2z2_5 + M2z2_5$$

$$\text{normal2_25}(i) = M2z2_5$$

$$M2y13_6 = -1 * \text{Log}(\text{answer})$$

If answer <= 1 / 2 Then

$$M2z3_6 = (M2y13_6 - 3) / 6$$

Else

$$M2z3_6 = (-M2y13_6 - 3) / 6$$

End If

$$s_M2z3_6 = s_M2z3_6 + M2z3_6$$

$$\text{normal2_36}(i) = M2z3_6$$

$M2y14_5 = -1 * \text{Log}(\text{answer})$

If $\text{answer} \leq 1 / 2$ Then

$$M2z4_5 = (M2y14_5 - 4) / 5$$

Else

$$M2z4_5 = (-M2y14_5 - 4) / 5$$

End If

$s_M2z4_5 = s_M2z4_5 + M2z4_5$

$\text{normal2_45}(i) = M2z4_5$

End Function

Function normal3()

$M3y10_1 = -1 * \text{Log}(\text{answer})$

If $\text{answer} \leq 1 / 2$ Then

$$M3z0_1 = (M3y10_1 - 0) / 1$$

Else

$$M3z0_1 = (-M3y10_1 - 0) / 1$$

End If

$s_M3z0_1 = s_M3z0_1 + M3z0_1$

$\text{normal3_01}(i) = M3z0_1$

$M3y11_3 = -1 * \text{Log}(\text{answer})$

If $\text{answer} \leq 1 / 2$ Then

$$M3z1_3 = (M3y11_3 - 1) / 3$$

Else

$$M3z1_3 = (-M3y11_3 - 1) / 3$$

End If

$s_M3z1_3 = s_M3z1_3 + M3z1_3$

$\text{normal3_13}(i) = M3z1_3$

$M3y12_5 = -1 * \text{Log}(\text{answer})$

If $\text{answer} \leq 1 / 2$ Then

$$M3z2_5 = (M3y12_5 - 2) / 5$$

Else

$$M3z2_5 = (-M3y12_5 - 2) / 5$$

End If

$$s_M3z2_5 = s_M3z2_5 + M3z2_5$$

$$\text{normal3_25}(i) = M3z2_5$$

$$M3y13_6 = -1 * \text{Log}(\text{answer})$$

If answer <= 1 / 2 Then

$$M3z3_6 = (M3y13_6 - 3) / 6$$

Else

$$M3z3_6 = (-M3y13_6 - 3) / 6$$

End If

$$s_M3z3_6 = s_M3z3_6 + M3z3_6$$

$$\text{normal3_36}(i) = M3z3_6$$

$$M3y14_5 = -1 * \text{Log}(\text{answer})$$

If answer <= 1 / 2 Then

$$M3z4_5 = (M3y14_5 - 4) / 5$$

Else

$$M3z4_5 = (-M3y14_5 - 4) / 5$$

End If

$$s_M3z4_5 = s_M3z4_5 + M3z4_5$$

$$\text{normal3_45}(i) = M3z4_5$$

End Function

Function normal4()

$$M4y10_1 = -1 * \text{Log}(\text{answer})$$

If answer <= 1 / 2 Then

$$M4z0_1 = (M4y10_1 - 0) / 1$$

Else

$$M4z0_1 = (-M4y10_1 - 0) / 1$$

End If

$$s_M4z0_1 = s_M4z0_1 + M4z0_1$$

$$\text{normal4_01}(i) = M4z0_1$$

$M4y11_3 = -1 * \text{Log}(\text{answer})$

If answer $\leq 1 / 2$ Then

$$M4z1_3 = (M4y11_3 - 1) / 3$$

Else

$$M4z1_3 = (-M4y11_3 - 1) / 3$$

End If

$s_M4z1_3 = s_M4z1_3 + M4z1_3$

normal4_13(i) = M4z1_3

$M4y12_5 = -1 * \text{Log}(\text{answer})$

If answer $\leq 1 / 2$ Then

$$M4z2_5 = (M4y12_5 - 2) / 5$$

Else

$$M4z2_5 = (-M4y12_5 - 2) / 5$$

End If

$s_M4z2_5 = s_M4z2_5 + M4z2_5$

normal4_25(i) = M4z2_5

$M4y13_6 = -1 * \text{Log}(\text{answer})$

If answer $\leq 1 / 2$ Then

$$M4z3_6 = (M4y13_6 - 3) / 6$$

Else

$$M4z3_6 = (-M4y13_6 - 3) / 6$$

End If

$s_M4z3_6 = s_M4z3_6 + M4z3_6$

normal4_36(i) = M4z3_6

$M4y14_5 = -1 * \text{Log}(\text{answer})$

If answer $\leq 1 / 2$ Then

$$M4z4_5 = (M4y14_5 - 4) / 5$$

Else

$$M4z4_5 = (-M4y14_5 - 4) / 5$$

End If

```
s_M4z4_5 = s_M4z4_5 + M4z4_5  
normal4_45(i) = M4z4_5
```

End Function

Function normal5()

```
M5y10_1 = -1 * Log(answer)  
If answer <= 1 / 2 Then
```

```
    M5z0_1 = (M5y10_1 - 0) / 1
```

```
Else
```

```
    M5z0_1 = (-M5y10_1 - 0) / 1
```

```
End If
```

```
s_M5z0_1 = s_M5z0_1 + M5z0_1  
normal5_01(i) = M5z0_1
```

```
M5y11_3 = -1 * Log(answer)  
If answer <= 1 / 2 Then
```

```
    M5z1_3 = (M5y11_3 - 1) / 3
```

```
Else
```

```
    M5z1_3 = (-M5y11_3 - 1) / 3
```

```
End If
```

```
s_M5z1_3 = s_M5z1_3 + M5z1_3  
normal5_13(i) = M5z1_3
```

```
M5y12_5 = -1 * Log(answer)  
If answer <= 1 / 2 Then
```

```
    M5z2_5 = (M5y12_5 - 2) / 5
```

```
Else
```

```
    M5z2_5 = (-M5y12_5 - 2) / 5
```

```
End If
```

```
s_M5z2_5 = s_M5z2_5 + M5z2_5  
normal5_25(i) = M5z2_5
```

```
M5y13_6 = -1 * Log(answer)  
If answer <= 1 / 2 Then
```


$$M5z3_6 = (M5y13_6 - 3) / 6$$

Else

$$M5z3_6 = (-M5y13_6 - 3) / 6$$

End If

$$s_M5z3_6 = s_M5z3_6 + M5z3_6$$

$$\text{normal5_36}(i) = M5z3_6$$

$$M5y14_5 = -1 * \text{Log}(\text{answer})$$

If answer <= 1 / 2 Then

$$M5z4_5 = (M5y14_5 - 4) / 5$$

Else

$$M5z4_5 = (-M5y14_5 - 4) / 5$$

End If

$$s_M5z4_5 = s_M5z4_5 + M5z4_5$$

$$\text{normal5_45}(i) = M5z4_5$$

End Function

Function cal_est_normal1()

$$eN10_1 = s_M1z0_1 / \text{sample}$$

$$m10_1 = (eN10_1 - 0) ^ 2$$

$$s_m10_1 = s_m10_1 + m10_1$$

$$eN11_3 = s_M1z1_3 / \text{sample}$$

$$m1_1_3 = (eN11_3 - 1) ^ 2$$

$$s_m1_1_3 = s_m1_1_3 + m1_1_3$$

$$eN12_5 = s_M1z2_5 / \text{sample}$$

$$m12_5 = (eN12_5 - 2) ^ 2$$

$$s_m12_5 = s_m12_5 + m12_5$$

$$eN13_6 = s_M1z3_6 / \text{sample}$$

$$m13_6 = (eN13_6 - 3) ^ 2$$

$$s_m13_6 = s_m13_6 + m13_6$$

$$eN14_5 = s_M1z4_5 / \text{sample}$$

$$m14_5 = (eN14_5 - 4) ^ 2$$

$$s_m14_5 = s_m14_5 + m14_5$$

End Function

Function cal_est_normal2()

```
eN20_1 = s_M2z0_1 / sample
m20_1 = (eN20_1 - 0) ^ 2
s_m20_1 = s_m20_1 + m20_1

eN21_3 = s_M2z1_3 / sample
m2_1_3 = (eN21_3 - 1) ^ 2
s_m2_1_3 = s_m2_1_3 + m2_1_3

eN22_5 = s_M2z2_5 / sample
m22_5 = (eN22_5 - 2) ^ 2
s_m22_5 = s_m22_5 + m22_5

eN23_6 = s_M2z3_6 / sample
m23_6 = (eN23_6 - 3) ^ 2
s_m23_6 = s_m23_6 + m23_6

eN24_5 = s_M2z4_5 / sample
m24_5 = (eN24_5 - 4) ^ 2
s_m24_5 = s_m24_5 + m24_5
```

End Function

Function cal_est_normal3()

```
eN30_1 = s_M3z0_1 / sample
m30_1 = (eN30_1 - 0) ^ 2
s_m30_1 = s_m30_1 + m30_1

eN31_3 = s_M3z1_3 / sample
m3_1_3 = (eN31_3 - 1) ^ 2
s_m3_1_3 = s_m3_1_3 + m3_1_3

eN32_5 = s_M3z2_5 / sample
m32_5 = (eN32_5 - 2) ^ 2
s_m32_5 = s_m32_5 + m32_5

eN33_6 = s_M3z3_6 / sample
m33_6 = (eN33_6 - 3) ^ 2
s_m33_6 = s_m33_6 + m33_6

eN34_5 = s_M3z4_5 / sample
m34_5 = (eN34_5 - 4) ^ 2
s_m34_5 = s_m34_5 + m34_5
```

End Function

Function cal_est_normal4()

```
eN40_1 = s_M4z0_1 / sample
m40_1 = (eN40_1 - 0) ^ 2
s_m40_1 = s_m40_1 + m40_1

eN41_3 = s_M4z1_3 / sample
m4_1_3 = (eN41_3 - 1) ^ 2
s_m4_1_3 = s_m4_1_3 + m4_1_3

eN42_5 = s_M4z2_5 / sample
m42_5 = (eN42_5 - 2) ^ 2
s_m42_5 = s_m42_5 + m42_5

eN43_6 = s_M4z3_6 / sample
m43_6 = (eN43_6 - 3) ^ 2
s_m43_6 = s_m43_6 + m43_6

eN44_5 = s_M4z4_5 / sample
m44_5 = (eN44_5 - 4) ^ 2
s_m44_5 = s_m44_5 + m44_5
```

End Function

Function cal_est_normal5()

```
eN50_1 = s_M5z0_1 / sample
m50_1 = (eN50_1 - 0) ^ 2
s_m50_1 = s_m50_1 + m50_1

eN51_3 = s_M5z1_3 / sample
m5_1_3 = (eN51_3 - 1) ^ 2
s_m5_1_3 = s_m5_1_3 + m5_1_3

eN52_5 = s_M5z2_5 / sample
m52_5 = (eN52_5 - 2) ^ 2
s_m52_5 = s_m52_5 + m52_5

eN53_6 = s_M5z3_6 / sample
m53_6 = (eN53_6 - 3) ^ 2
s_m53_6 = s_m53_6 + m53_6

eN54_5 = s_M5z4_5 / sample
m54_5 = (eN54_5 - 4) ^ 2
s_m54_5 = s_m54_5 + m54_5
```

End Function

Function mse_normal1()

```
mlem10_1 = (s_m10_1 / loop_test)
rmlem10_1 = Round(Sqr(mlem10_1), 4)

mlem1_1_3 = (s_m1_1_3 / loop_test)
rmlem1_1_3 = Round(Sqr(mlem1_1_3), 4)

mlem12_5 = (s_m12_5 / loop_test)
rmlem12_5 = Round(Sqr(mlem12_5), 4)

mlem13_6 = (s_m13_6 / loop_test)
rmlem13_6 = Round(Sqr(mlem13_6), 4)

mlem14_5 = (s_m14_5 / loop_test)
rmlem14_5 = Round(Sqr(mlem14_5), 4)
```

```
Print "          "
Print "M1: RMSE NM 0,1=" & rmlem10_1, "1,3=" & rmlem1_1_3, "2,5=" &
rmlem12_5, "3,6=" & rmlem13_6, "4,5=" & rmlem14_5
```

End Function

Function mse_normal2()

```
mlem20_1 = (s_m20_1 / loop_test)
rmlem20_1 = Round(Sqr(mlem20_1), 4)

mlem2_1_3 = (s_m2_1_3 / loop_test)
rmlem2_1_3 = Round(Sqr(mlem2_1_3), 4)

mlem22_5 = (s_m22_5 / loop_test)
rmlem22_5 = Round(Sqr(mlem22_5), 4)

mlem23_6 = (s_m23_6 / loop_test)
rmlem23_6 = Round(Sqr(mlem23_6), 4)

mlem24_5 = (s_m24_5 / loop_test)
rmlem24_5 = Round(Sqr(mlem24_5), 4)
```

```
Print "M2: RMSE NM 0,1=" & rmlem20_1, "1,3=" & rmlem2_1_3, "2,5=" &
rmlem22_5, "3,6=" & rmlem23_6, "4,5=" & rmlem24_5
```

End Function

Function mse_normal3()

```
mlem30_1 = (s_m30_1 / loop_test)
rmlem30_1 = Round(Sqr(mlem30_1), 4)

mlem3_1_3 = (s_m3_1_3 / loop_test)
rmlem3_1_3 = Round(Sqr(mlem3_1_3), 4)

mlem32_5 = (s_m32_5 / loop_test)
rmlem32_5 = Round(Sqr(mlem32_5), 4)

mlem33_6 = (s_m33_6 / loop_test)
rmlem33_6 = Round(Sqr(mlem33_6), 4)

mlem34_5 = (s_m34_5 / loop_test)
rmlem34_5 = Round(Sqr(mlem34_5), 4)
```

```
Print "M3: RMSE NM 0,1=" & rmlem30_1, "1,3=" & rmlem3_1_3, "2,5=" &
rmlem32_5, "3,6=" & rmlem33_6, "4,5=" & rmlem34_5
```

End Function

Function mse_normal4()

```
mlem40_1 = (s_m40_1 / loop_test)
rmlem40_1 = Round(Sqr(mlem40_1), 4)

mlem4_1_3 = (s_m4_1_3 / loop_test)
rmlem4_1_3 = Round(Sqr(mlem4_1_3), 4)

mlem42_5 = (s_m42_5 / loop_test)
rmlem42_5 = Round(Sqr(mlem42_5), 4)

mlem43_6 = (s_m43_6 / loop_test)
rmlem43_6 = Round(Sqr(mlem43_6), 4)

mlem44_5 = (s_m44_5 / loop_test)
rmlem44_5 = Round(Sqr(mlem44_5), 4)
```

```
Print "M4: RMSE NM 0,1=" & rmlem40_1, "1,3=" & rmlem4_1_3, "2,5=" &
rmlem42_5, "3,6=" & rmlem43_6, "4,5=" & rmlem44_5
```

End Function

Function mse_normal5()

```
mlem50_1 = (s_m50_1 / loop_test)
rmlem50_1 = Round(Sqr(mlem50_1), 4)
```

```
mlem5_1_3 = (s_m5_1_3 / loop_test)
rmlem5_1_3 = Round(Sqr(mlem5_1_3), 4)
```

```
mlem52_5 = (s_m52_5 / loop_test)
rmlem52_5 = Round(Sqr(mlem52_5), 4)
```

```
mlem53_6 = (s_m53_6 / loop_test)
rmlem53_6 = Round(Sqr(mlem53_6), 4)
```

```
mlem54_5 = (s_m54_5 / loop_test)
rmlem54_5 = Round(Sqr(mlem54_5), 4)
```

```
Print "M5: RMSE NM 0,1=" & rmlem50_1, "1,3=" & rmlem5_1_3, "2,5=" &
rmlem52_5, "3,6=" & rmlem53_6, "4,5=" & rmlem54_5
```

End Function

Function est_std()

For i = 1 To sample

```
v10_1 = (normal1_01(i) - eN10_1) ^ 2
s_v10_1 = s_v10_1 + v10_1
```

```
v11_3 = (normal1_13(i) - eN11_3) ^ 2
s_v11_3 = s_v11_3 + v11_3
```

```
v12_5 = (normal1_25(i) - eN12_5) ^ 2
s_v12_5 = s_v12_5 + v12_5
```

```
v13_6 = (normal1_36(i) - eN13_6) ^ 2
s_v13_6 = s_v13_6 + v13_6
```

```
v14_5 = (normal1_45(i) - eN14_5) ^ 2
s_v14_5 = s_v14_5 + v14_5
```

```
v20_1 = (normal2_01(i) - eN20_1) ^ 2
s_v20_1 = s_v20_1 + v20_1
```

```
v21_3 = (normal2_13(i) - eN21_3) ^ 2
s_v21_3 = s_v21_3 + v21_3
```

$$\begin{aligned}v22_5 &= (\text{normal2_25}(i) - eN22_5)^2 \\s_v22_5 &= s_v22_5 + v22_5\end{aligned}$$

$$\begin{aligned}v23_6 &= (\text{normal2_36}(i) - eN23_6)^2 \\s_v23_6 &= s_v23_6 + v23_6\end{aligned}$$

$$\begin{aligned}v24_5 &= (\text{normal2_45}(i) - eN24_5)^2 \\s_v24_5 &= s_v24_5 + v24_5\end{aligned}$$

$$\begin{aligned}v30_1 &= (\text{normal3_01}(i) - eN30_1)^2 \\s_v30_1 &= s_v30_1 + v30_1\end{aligned}$$

$$\begin{aligned}v31_3 &= (\text{normal3_13}(i) - eN31_3)^2 \\s_v31_3 &= s_v31_3 + v31_3\end{aligned}$$

$$\begin{aligned}v32_5 &= (\text{normal3_25}(i) - eN32_5)^2 \\s_v32_5 &= s_v32_5 + v32_5\end{aligned}$$

$$\begin{aligned}v33_6 &= (\text{normal3_36}(i) - eN33_6)^2 \\s_v33_6 &= s_v33_6 + v33_6\end{aligned}$$

$$\begin{aligned}v34_5 &= (\text{normal3_45}(i) - eN34_5)^2 \\s_v34_5 &= s_v34_5 + v34_5\end{aligned}$$

$$\begin{aligned}v40_1 &= (\text{normal4_01}(i) - eN40_1)^2 \\s_v40_1 &= s_v40_1 + v40_1\end{aligned}$$

$$\begin{aligned}v41_3 &= (\text{normal4_13}(i) - eN41_3)^2 \\s_v41_3 &= s_v41_3 + v41_3\end{aligned}$$

$$\begin{aligned}v42_5 &= (\text{normal4_25}(i) - eN42_5)^2 \\s_v42_5 &= s_v42_5 + v42_5\end{aligned}$$

$$\begin{aligned}v43_6 &= (\text{normal4_36}(i) - eN43_6)^2 \\s_v43_6 &= s_v43_6 + v43_6\end{aligned}$$

$$\begin{aligned}v44_5 &= (\text{normal4_45}(i) - eN44_5)^2 \\s_v44_5 &= s_v44_5 + v44_5\end{aligned}$$

$$\begin{aligned}v50_1 &= (\text{normal5_01}(i) - eN50_1)^2 \\s_v50_1 &= s_v50_1 + v50_1\end{aligned}$$

$$\begin{aligned}v51_3 &= (\text{normal5_13}(i) - eN51_3)^2 \\s_v51_3 &= s_v51_3 + v51_3\end{aligned}$$

$$v52_5 = (\text{normal5_25}(i) - eN52_5)^2$$

$s_v52_5 = s_v52_5 + v52_5$

$v53_6 = (\text{normal5_36}(i) - eN53_6) ^ 2$

$s_v53_6 = s_v53_6 + v53_6$

$v54_5 = (\text{normal5_45}(i) - eN54_5) ^ 2$

$s_v54_5 = s_v54_5 + v54_5$

Next i

End Function

Function cal_variance()

$\text{var1_01} = s_v10_1 / (\text{sample} - 1)$

$\text{var1_13} = s_v11_3 / (\text{sample} - 1)$

$\text{var1_25} = s_v12_5 / (\text{sample} - 1)$

$\text{var1_36} = s_v13_6 / (\text{sample} - 1)$

$\text{var1_45} = s_v14_5 / (\text{sample} - 1)$

$\text{var2_01} = s_v20_1 / (\text{sample} - 1)$

$\text{var2_13} = s_v21_3 / (\text{sample} - 1)$

$\text{var2_25} = s_v22_5 / (\text{sample} - 1)$

$\text{var2_36} = s_v23_6 / (\text{sample} - 1)$

$\text{var2_45} = s_v24_5 / (\text{sample} - 1)$

$\text{var3_01} = s_v30_1 / (\text{sample} - 1)$

$\text{var3_13} = s_v31_3 / (\text{sample} - 1)$

$\text{var3_25} = s_v32_5 / (\text{sample} - 1)$

$\text{var3_36} = s_v33_6 / (\text{sample} - 1)$

$\text{var3_45} = s_v34_5 / (\text{sample} - 1)$

$\text{var4_01} = s_v40_1 / (\text{sample} - 1)$

$\text{var4_13} = s_v41_3 / (\text{sample} - 1)$

$\text{var4_25} = s_v42_5 / (\text{sample} - 1)$

$\text{var4_36} = s_v43_6 / (\text{sample} - 1)$

$\text{var4_45} = s_v44_5 / (\text{sample} - 1)$

$\text{var5_01} = s_v50_1 / (\text{sample} - 1)$

$\text{var5_13} = s_v51_3 / (\text{sample} - 1)$

$\text{var5_25} = s_v52_5 / (\text{sample} - 1)$

$\text{var5_36} = s_v53_6 / (\text{sample} - 1)$

$\text{var5_45} = s_v54_5 / (\text{sample} - 1)$

End Function

Function mse_variance()

```
mle_v10_1 = (var1_01 - 1) ^ 2  
rmle_v10_1 = Round(Sqr(mle_v10_1), 4)
```

```
mle_v11_3 = (var1_13 - 3) ^ 2  
rmle_v11_3 = Round(Sqr(mle_v11_3), 4)
```

```
mle_v12_5 = (var1_25 - 5) ^ 2  
rmle_v12_5 = Round(Sqr(mle_v12_5), 4)
```

```
mle_v13_6 = (var1_36 - 6) ^ 2  
rmle_v13_6 = Round(Sqr(mle_v13_6), 4)
```

```
mle_v14_5 = (var1_45 - 5) ^ 2  
rmle_v14_5 = Round(Sqr(mle_v14_5), 4)
```

```
mle_v20_1 = (var2_01 - 1) ^ 2  
rmle_v20_1 = Round(Sqr(mle_v20_1), 4)
```

```
mle_v21_3 = (var2_13 - 3) ^ 2  
rmle_v21_3 = Round(Sqr(mle_v21_3), 4)
```

```
mle_v22_5 = (var2_25 - 5) ^ 2  
rmle_v22_5 = Round(Sqr(mle_v22_5), 4)
```

```
mle_v23_6 = (var2_36 - 6) ^ 2  
rmle_v23_6 = Round(Sqr(mle_v23_6), 4)
```

```
mle_v24_5 = (var2_45 - 5) ^ 2  
rmle_v24_5 = Round(Sqr(mle_v24_5), 4)
```

```
mle_v30_1 = (var3_01 - 1) ^ 2  
rmle_v30_1 = Round(Sqr(mle_v30_1), 4)
```

```
mle_v31_3 = (var3_13 - 3) ^ 2  
rmle_v31_3 = Round(Sqr(mle_v31_3), 4)
```

```
mle_v32_5 = (var3_25 - 5) ^ 2  
rmle_v32_5 = Round(Sqr(mle_v32_5), 4)
```

```
mle_v33_6 = (var3_36 - 6) ^ 2  
rmle_v33_6 = Round(Sqr(mle_v33_6), 4)
```

```
mle_v34_5 = (var3_45 - 5) ^ 2  
rmle_v34_5 = Round(Sqr(mle_v34_5), 4)
```

```
mle_v40_1 = (var4_01 - 1) ^ 2  
rmle_v40_1 = Round(Sqr(mle_v40_1), 4)
```

```
mle_v41_3 = (var4_13 - 3) ^ 2  
rmle_v41_3 = Round(Sqr(mle_v41_3), 4)
```

```
mle_v42_5 = (var4_25 - 5) ^ 2  
rmle_v42_5 = Round(Sqr(mle_v42_5), 4)
```

```
mle_v43_6 = (var4_36 - 6) ^ 2  
rmle_v43_6 = Round(Sqr(mle_v43_6), 4)
```

```
mle_v44_5 = (var4_45 - 5) ^ 2  
rmle_v44_5 = Round(Sqr(mle_v44_5), 4)
```

```
mle_v50_1 = (var5_01 - 1) ^ 2  
rmle_v50_1 = Round(Sqr(mle_v50_1), 4)
```

```
mle_v51_3 = (var5_13 - 3) ^ 2  
rmle_v51_3 = Round(Sqr(mle_v51_3), 4)
```

```
mle_v52_5 = (var5_25 - 5) ^ 2  
rmle_v52_5 = Round(Sqr(mle_v52_5), 4)
```

```
mle_v53_6 = (var5_36 - 6) ^ 2  
rmle_v53_6 = Round(Sqr(mle_v53_6), 4)
```

```
mle_v54_5 = (var5_45 - 5) ^ 2  
rmle_v54_5 = Round(Sqr(mle_v54_5), 4)
```

```
Print "          "  
Print "M1: RMSE VAR 0,1=" & rmle_v10_1, "1,3=" & rmle_v11_3, "2,5=" &  
rmle_v12_5  
Print "3,6=" & rmle_v13_6, "4,5=" & rmle_v14_5  
Print "          "  
Print "M2: RMSE VAR 0,1=" & rmle_v20_1, "1,3=" & rmle_v21_3, "2,5=" &  
rmle_v22_5  
Print "3,6=" & rmle_v23_6, "4,5=" & rmle_v24_5  
Print "          "  
Print "M3: RMSE VAR 0,1=" & rmle_v30_1, "1,3=" & rmle_v31_3, "2,5=" &  
rmle_v32_5  
Print "3,6=" & rmle_v33_6, "4,5=" & rmle_v34_5  
Print "          "  
Print "M4: RMSE VAR 0,1=" & rmle_v40_1, "1,3=" & rmle_v41_3, "2,5=" &  
rmle_v42_5  
Print "3,6=" & rmle_v43_6, "4,5=" & rmle_v44_5  
Print "          "  
Print "M5: RMSE VAR 0,1=" & rmle_v50_1, "1,3=" & rmle_v51_3, "2,5=" &  
rmle_v52_5  
Print "3,6=" & rmle_v53_6, "4,5=" & rmle_v54_5
```

End Function