ภาคผนวก

ภาคผนวก ก

ตารางอาณาเขตวิกฤตของชุดการแจกแจงพาวเวอร์เอกซ์โปเนนเชียล

ตาราง แสดงอาณาเขตวิกฤตของการแจกแจงชุดพาวเวอร์เอกซ์โปเนนเชียลในการทดสอบแบบเบย์เซียนในหน่วยของ σ เมื่อทำการทดสอบที่ θ มีค่าเป็น 0

	α						
β	.25	.10	.05	.025	.01	.005	.001
-1.00	0.87	1.39	1.56	1.65	1.70	1.71	1.73
-0.75	0.84	1.36	1.57	1.71	1.84	1.91	2.05
-0.50	0.80	1.35	1.61	1.81	2.03	2.16	2.41
-0.25	0.73	1.31	1.63	1.86	2.18	2.37	2.75
0.00	0.67	1.28	1.64	1.96	2.33	2.58	3.09
0.25	0.62	1.25	1.65	2.02	2.46	2.77	3.43
0.50	0.58	1.22	1.65	2.06	2.58	2.94	3.76
0.75	0.53	1.18	1.64	2.09	2.68	3.10	4.08
1.00	0.49	1.14	1.63	2.12	2.77	3.28	4.39

(G.E.P. Box and G.C. Tiao, 1973, p.162)

ภาคผนวก ข
โปรแกรมคอมพิวเตอร์

โปรแกรมที่ 1 TESTPOP.BAS

```
DECLARE SUB PosSkewGen (Mean!, Variance!, Numvar%, Var!())
DECLARE FUNCTION VVariance! (Var!())
DECLARE FUNCTION Zscore# (Z#)
DECLARE SUB UniformGen (Mean!, Variance!, Numvar%, Var!())
DECLARE SUB DoubleExpGen (Mean!, Variance!, Numvar%, Var!())
DECLARE FUNCTION Skewness! (Var!())
DECLARE FUNCTION Kurtosis! (Var!())
DECLARE FUNCTION SMean! (Var!())
DECLARE FUNCTION SVariance! (Var!())
DECLARE SUB NormalGen (Mean!, Variance!, Numvar%, Var!())
'****************** Simulation Population ****************
' $DYNAMIC
TYPE XX
   A  AS STRING * 5
   B  AS STRING * 5
   C  AS STRING * 2
END TYPE
DIM SHARED TT AS XX
CONST pi = 3.141593
DIM Var1![1), Var2![1), Var3![1), Var4![1), VAR5![1)
   CLS
   PRINT "Test parameter of population...."
   PRINT "Do you want to print on (P) Printer (F) File T-Pop.OUT (M) Monitor (S) Stop"
   PP$ = INPUT$(1)
   SELECT CASE UCASE$(PP$)
   CASE "P"
      OPEN "O", #2, "LPT1:"
      WIDTH #2, 255
      GOSUB PrintTtest
      CLOSE #2
   CASE "F"
      OPEN "O", #2, "T-Pop.OUT"
      WIDTH #2, 255
      GOSUB PrintTtest
      CLOSE #2
   CASE "M"
      GOSUB PrintTtest
   CASE "S"
   CASE ELSE
   END SELECT
CLOSE
END
PrintTtest:
CLS
GOSUB Header
```

```
REDIM Var1!(10000), Var2!(10000), Var3!(10000), Var4!(10000)
RANDOMIZE 13
NormalGen 0, 1, 10000, Var1!()
PRINT TAB(1); USING "##.  Normal "; 1;
PRINT TAB(25); USING "#.####"; SMean!(Var1!());
PRINT TAB(33); USING "##.####"; VVariance!(Var1!());
PRINT TAB(47); USING "##.####"; Skewness!(Var1!());
PRINT TAB(65); USING "##.####"; Kurtosis!(Var1!())
SELECT CASE UCASE$(PP$)
CASE "P", "F"
   PRINT #2, TAB(1); USING "##.  Normal "; 1;
   PRINT #2, TAB(25); USING "#.####"; SMean!(Var1!());
   PRINT #2, TAB(33); USING "##.####"; VVariance!(Var1!());
   PRINT #2, TAB(47); USING "##.####"; Skewness!(Var1!());
   PRINT #2, TAB(65); USING "##.####"; Kurtosis!(Var1!())
CASE ELSE
END SELECT
DoubleExpGen 0, 1, 10000, Var2!()
PRINT TAB(1); USING "##.  Double Exponential"; 2;
PRINT TAB(25); USING "#.####"; SMean!(Var2!());
PRINT TAB(33); USING "##.####"; VVariance!(Var2!());
PRINT TAB(47); USING "##.####"; Skewness!(Var2!());
PRINT TAB(65); USING "##.####"; Kurtosis!(Var2!())
SELECT CASE UCASE$(PP$)
CASE "P", "F"
   PRINT #2, TAB(1); USING "##.  Double Exponential"; 2;
   PRINT #2, TAB(25); USING "#.####"; SMean!(Var2!());
   PRINT #2, TAB(33); USING "##.####"; VVariance!(Var2!());
   PRINT #2, TAB(47); USING "##.####"; Skewness!(Var2!());
   PRINT #2, TAB(65); USING "##.####"; Kurtosis!(Var2!())
CASE ELSE
END SELECT

UniformGen 0, 1, 10000, Var3!()
PRINT TAB(1); USING "##.  Uniform"; 3;
PRINT TAB(25); USING "#.####"; SMean!(Var3!());
PRINT TAB(33); USING "##.####"; VVariance!(Var3!());
PRINT TAB(47); USING "##.####"; Skewness!(Var3!());
PRINT TAB(65); USING "##.####"; Kurtosis!(Var3!())
SELECT CASE UCASE$(PP$)
CASE "P", "F"
   PRINT #2, TAB(1); USING "##.  Uniform"; 3;
   PRINT #2, TAB(25); USING "#.####"; SMean!(Var3!());
   PRINT #2, TAB(33); USING "##.####"; VVariance!(Var3!());
   PRINT #2, TAB(47); USING "##.####"; Skewness!(Var3!());
   PRINT #2, TAB(65); USING "##.####"; Kurtosis!(Var3!())
CASE ELSE
END SELECT
```

```
PosSkewGen 0, 1, 10000, Var4!()
PRINT TAB(1); USING "##.  Positive skewness"; 4;
PRINT TAB(25); USING "#.####"; SMean!(Var4!());
PRINT TAB(33); USING "##.####"; VVariance!(Var4!());
PRINT TAB(47); USING "##.####"; Skewness!(Var4!());
PRINT TAB(65); USING "##.####"; Kurtosis!(Var4!())
PRINT
"=============================================================================
====="
SELECT CASE UCASE$(PP$)
CASE "P", "F"
   PRINT #2, TAB(1); USING "##.  Positive skewness"; 4;
   PRINT #2, TAB(25); USING "#.####"; SMean!(Var4!());
   PRINT #2, TAB(33); USING "##.####"; VVariance!(Var4!());
   PRINT #2, TAB(47); USING "##.####"; Skewness!(Var4!());
   PRINT #2, TAB(65); USING "##.####"; Kurtosis!(Var4!())
   PRINT #2,
"=============================================================================
====="
CASE ELSE
END SELECT
RETURN
Header:
PRINT "TABLE :  testing of Population"
PRINT
"=============================================================================
====="
PRINT "No.  Population I      Mean   Variance   Coeff. Skewness   Coeff. Kurtosis"
PRINT
"=============================================================================
====="
SELECT CASE UCASE$(PP$)
CASE "P", "F"
   PRINT #2, "TABLE :  testing of Population"
   PRINT #2,
"=============================================================================
====="
   PRINT #2, "No.  Population I      Mean   Variance   Coeff. Skewness   Coeff. Kurtosis"
   PRINT #2, "##.                 #.####  ##.####      ##.####          ##.####    "
   PRINT #2,
"=============================================================================
====="
CASE ELSE
END SELECT

RETURN
END
```

```
REM $STATIC
SUB DoubleExpGen (Mean!, Variance!, Numvar%, Var!())
'parameter Mean!=Mean of population
'       Variance!=Variance of population
'       Numvar%=number of case
'       Var!()=Value of Data to return
Number% = 0
DO
   RandomVar! = RND(1)
   Z1! = Mean! - (SQR(Variance!) * LOG(RandomVar!)) / SQR(2)
   Number% = Number% + 1
   Var!(Number%) = Z1!
   IF Number% >= Numvar% THEN EXIT DO
   RandomVar! = RND(1)
   Z2! = Mean! + (SQR(Variance!) * LOG(RandomVar!)) / SQR(2)
   Number% = Number% + 1
   Var!(Number%) = Z2!
   IF Number% >= Numvar% THEN EXIT DO
LOOP

END SUB

FUNCTION Kurtosis! (Var!())
   Kur! = 0
   Mean! = SMean!(Var!())
   Sigma! = SQR(VVariance(Var!()))
   FOR Number% = 1 TO UBOUND(Var!)
      Kur! = Kur! + (Var!(Number%) - Mean!) ^ 4
   NEXT Number%
   Kurtosis! = Kur! / (UBOUND(Var!) * (Sigma! ^ 4))
END FUNCTION

SUB NormalGen (Mean!, Variance!, Numvar%, Var!())
Number% = 0
DO
   RandomVar1! = RND(1)
   RandomVar2! = RND(1)
   Z1! = Mean! + SQR(-2 * LOG(RandomVar1!)) * (COS(2 * pi * RandomVar2!)) * SQR
(Variance!)
   Number% = Number% + 1
   Var!(Number%) = Z1!
   IF Number% >= Numvar% THEN EXIT DO
   Z2! = Mean! + SQR(-2 * LOG(RandomVar1!)) * (SIN(2 * pi * RandomVar2!)) * SQR(Variance!)
   Number% = Number% + 1
   Var!(Number%) = Z2!
   IF Number% >= Numvar% THEN EXIT DO
LOOP
END SUB
```

```
SUB PosSkewGen (Mean!, Variance!, Numvar%, Var!())
'parameter Mean!=Mean of population
'        Variance!=Variance of population
'        Numvar%=number of case
'        Var!()=Value of Data to return
   NormalGen Mean!, Variance!, Numvar%, Var!()
   Max% = UBOUND(Var!)
   'TransForm Normal to Posistive Skewness Distribution
   'Funct y=-0.3268+1.16050961x+0.2909708x2-0.0886191x3
   FOR I% = 1 TO Max%
      X! = Var!(I%)
      Var!(I%) = -.3268 + (1.16050961# * X!) + (.2909708# * (X! ^ 2)) - (8.861910000000001D-02 *
(X! ^ 3))
   NEXT I%
END SUB

FUNCTION Skewness! (Var!())
   SK! = 0
   Mean! = SMean!(Var!())
   Sigma! = SQR(VVariance(Var!()))
   FOR Number% = 1 TO UBOUND(Var!)
      SK! = SK! + (Var!(Number%) - Mean!) ^ 3
   NEXT Number%
   Skewness! = SK! / (UBOUND(Var!) * (Sigma! ^ 3))
END FUNCTION

FUNCTION SMean! (Var!())
   M! = 0
   FOR Number% = 1 TO UBOUND(Var!)
      M! = M! + Var!(Number%)
   NEXT Number%
   SMean! = M! / UBOUND(Var!)
END FUNCTION

FUNCTION SVariance! (Var!())
   M! = SMean!(Var!())
   S! = 0
   FOR Number% = 1 TO UBOUND(Var!)
      S! = S! + (Var!(Number%) - M!) ^ 2
   NEXT Number%
   SVariance! = S! / (UBOUND(Var!) - 1)
END FUNCTION

SUB UniformGen (Mean!, Variance!, Numvar%, Var!())
'parameter Mean!=Mean of population
'        Variance!=Variance of population
'        Numvar%=number of case
'        Var!()=Value of Data to return
Min! = Mean! - SQR(3) * SQR(Variance!)
Max! = Mean! + SQR(3) * SQR(Variance!)
```

```
Number% = 0
DO
   RandomVar! = RND(1)
   Z1! = Min! + (Max! - Min!) * RandomVar!
   Number% = Number% + 1
   Var!(Number%) = Z1!
LOOP UNTIL Number% >= Numvar%
END SUB

FUNCTION VVariance! (Var!())
   M! = SMean!(Var!())
   S! = 0
   FOR Number% = 1 TO UBOUND(Var!)
      S! = S! + (Var!(Number%) - M!) ^ 2
   NEXT Number%
   VVariance! = S! / (UBOUND(Var!))
END FUNCTION

FUNCTION Zscore# (Z#)
' The formula used from Abramowitz & Stegun (1972) fomula 2

IF Z# >= -14.14 AND Z# <= 14.14 THEN
   Y# = (.7071067812#) * Z#
ELSE
   Y# = 10
END IF
A1# = 7.052307839999999D-02
A2# = .0422820123#
A3# = .0092705272#
A4# = .0001520143#
A5# = .0002765672#
A6# = .0000430638#
Zscore# = 1 - .5 * (1 + A1# * Y# + A2# * (Y# ^ 2) + A3# * (Y# ^ 3) + A4# * (Y# ^ 4) + A5# * (Y#
^ 5) + A6# * (Y# ^ 6)) ^ (-16)
END FUNCTION
```

โปรแกรมที่ 2 CR_RND.BAS

```
OPEN "O", #1, "STRT.RND"
RANDOMIZE TIMER
FOR I& = 1 TO 72000
   PRINT #1, USING "#,###"; (INT(RND(1) * 1000!)); (INT(RND(1) * 1000!))
NEXT I&
CLOSE
END
```

โปรแกรมที่ 3 **T-TEST.BAS**

```
DECLARE SUB PosSkewGen (Mean!, Variance!, Numvar%, Var!())
DECLARE SUB RRT (Order&, Round%, No1%, No2%)
DECLARE SUB CutRnD (No%)
DECLARE FUNCTION VVariance! (Var!())
DECLARE FUNCTION tunpair! (VAR1!(), VAR2!(), Mean1!, Mean2!, df%)
DECLARE FUNCTION ttest! (Var!(), Mean!, df%)
DECLARE FUNCTION tpair! (VAR1!(), VAR2!(), Mean1!, Mean2!, df%)
DECLARE FUNCTION RandInt% (lower%, Upper%)
DECLARE SUB TestRandom (Numvar%, NoClass%, Chisquare!)
DECLARE SUB QuickSort (Low AS INTEGER, High AS INTEGER, SortArray!())
DECLARE FUNCTION Zscore# (Z#)
DECLARE FUNCTION tprob# (t#, k%)
DECLARE FUNCTION tprop# (t0#, k1 AS INTEGER)
DECLARE SUB UniformGen (Mean!, Variance!, Numvar%, Var!())
DECLARE SUB DoubleExpGen (Mean!, Variance!, Numvar%, Var!())
DECLARE FUNCTION Skewness! (Var!(), Mean!, Sigma!)
DECLARE FUNCTION Kurtosis! (Var!(), Mean!, Sigma!)
DECLARE FUNCTION SMEAN! (Var!())
DECLARE FUNCTION SVariance! (Var!())
DECLARE SUB NormalGen (Mean!, Variance!, Numvar%, Var!())
'******************** Simulation normal distribution ****************
' $DYNAMIC
TYPE XX
   A  AS STRING * 5
   B  AS STRING * 5
   C  AS STRING * 2
END TYPE
DIM SHARED TT AS XX
CONST pi = 3.141593
DIM VAR1!(1), VAR2!(1), VAR3!(1), VAR4!(1), VAR5!(1)
   CLS
   PRINT "Do you want to print on (P) Printer (F) File T-test.OUT (M) Monitor (S) Stop"
   PP$ = INPUT$(1)
   SELECT CASE UCASE$(PP$)
   CASE "P"
      OPEN "O", #2, "LPT1:"
      WIDTH #2, 255
      GOSUB PrintTtest
      CLOSE #2
   CASE "F"
      OPEN "O", #2, "T-test.OUT"
      WIDTH #2, 255
      GOSUB PrintTtest
      CLOSE #2
   CASE "M"
      GOSUB PrintTtest
   CASE "S"
   CASE ELSE
```

```
    END SELECT
  CLOSE
  END
  PrintTtest:
  CLS
  Lno% = 0
  GOSUB Header
  VIEW PRINT 8 TO 25
  OPEN "I", #1, "THES.DAT"
  DO WHILE NOT EOF(1)
    Sum05! = 0: Sum01! = 0
    P7505! = 0: P105! = 0: P1505! = 0
    P7501! = 0: P101! = 0: P1501! = 0
    LINE INPUT #1, NN$
    Order& = VAL(LEFT$(NN$, 2))
    Pop1% = VAL(MID$(NN$, 3, 1))
    Pop2% = VAL(MID$(NN$, 4, 1))
    n1% = VAL(MID$(NN$, 5, 2))
    n2% = VAL(MID$(NN$, 7, 2))
    Mean1! = VAL(MID$(NN$, 9, 5))
    Mean2! = VAL(MID$(NN$, 14, 5))
    V1! = VAL(MID$(NN$, 19, 5))
    V2! = VAL(MID$(NN$, 24, 5))
    GOSUB ttest
  LOOP
  VIEW PRINT 1 TO 25
  RETURN
  ttest:
  FOR Round% = 1 TO 1000
    RRT Order&, Round%, No1%, No2%
    REDIM VAR1!(n1%), VAR2!(n2%), VAR3!(n1%), VAR4!(n1%), VAR5!(n1%)
    RANDOMIZE Round%
    CutRnD No1%
    SELECT CASE Pop1%
    CASE 1
      NormalGen Mean1!, V1!, n1%, VAR1!()
    CASE 2
      DoubleExpGen Mean1!, V1!, n1%, VAR1!()
    CASE 3
      UniformGen Mean1!, V1!, n1%, VAR1!()
    CASE 4
      PosSkewGen Mean1!, V1!, n1%, VAR1!()
    CASE ELSE
    END SELECT
    VV! = SQR((V1! / n1%) + (V2! / n2%))
    FOR kk% = 1 TO n1%
      VAR3!(kk%) = (.75) * VV! + VAR1!(kk%)
      VAR4!(kk%) = (1!) * VV! + VAR1!(kk%)
      VAR5!(kk%) = (1.5) * VV! + VAR1!(kk%)
    NEXT kk%
```

```
    RANDOMIZE (1000 + Round%)
    CutRnD No2%
    SELECT CASE Pop2%
    CASE 1
      NormalGen Mean2!, V2!, n2%, VAR2!()
    CASE 2
      DoubleExpGen Mean2!, V2!, n2%, VAR2!()
    CASE 3
      UniformGen Mean2!, V2!, n2%, VAR2!()
    CASE 4
      PosSkewGen Mean1!, V2!, n2%, VAR2!()
    CASE ELSE
    END SELECT
    tvalue# = tunpair!(VAR1!(), VAR2!(), Mean1!, Mean2!, df%)
    IF tprob#(ABS(tvalue#), n1% + n2% - 2) > .975# THEN
      Sum05! = Sum05! + 1
    END IF
    IF tprob#(ABS(tvalue#), n1% + n2% - 2) > .995# THEN
      Sum01! = Sum01! + 1
    END IF
    tvalue# = tunpair!(VAR3!(), VAR2!(), Mean1!, Mean2!, df%)
    IF tprob#(ABS(tvalue#), n1% + n2% - 2) > .975# THEN
      P7505! = P7505! + 1
    END IF
    IF tprob#(ABS(tvalue#), n1% + n2% - 2) > .995# THEN
      P7501! = P7501! + 1
    END IF
    tvalue# = tunpair!(VAR4!(), VAR2!(), Mean1!, Mean2!, df%)
    IF tprob#(ABS(tvalue#), n1% + n2% - 2) > .975# THEN
      P105! = P105! + 1
    END IF
    IF tprob#(ABS(tvalue#), n1% + n2% - 2) > .995# THEN
      P101! = P101! + 1
    END IF
    tvalue# = tunpair!(VAR5!(), VAR2!(), Mean1!, Mean2!, df%)
    IF tprob#(ABS(tvalue#), n1% + n2% - 2) > .975# THEN
      P1505! = P1505! + 1
    END IF
    IF tprob#(ABS(tvalue#), n1% + n2% - 2) > .995# THEN
      P1501! = P1501! + 1
    END IF
  NEXT Round%
  PRINT USING "##   "; Order&;
  SELECT CASE Pop1%
  CASE 1
    PRINT "Normal";
  CASE 2
    PRINT "Double Exponential";
  CASE 3
    PRINT "Uniform";
```

```
CASE 4
   PRINT "Positive Skewness";
CASE ELSE
END SELECT
PRINT TAB(25); USING "#.##"; Mean1!;
PRINT TAB(33); USING "##.##"; V1!;
PRINT TAB(41); USING "##"; n1%;
PRINT TAB(45); USING "###.###"; (Sum05! / 1000);
PRINT TAB(58); USING "#.###"; P7505! / 1000;
PRINT TAB(67); USING "#.###"; P105! / 1000;
PRINT TAB(76); USING "#.###"; P1505! / 1000
PRINT TAB(6);
SELECT CASE Pop2%
CASE 1
   PRINT "Normal";
CASE 2
   PRINT "Double Exponential";
CASE 3
   PRINT "Uniform";
CASE 4
   PRINT "Positive Skewness";
CASE ELSE
END SELECT
PRINT TAB(25); USING "#.##"; Mean2!;
PRINT TAB(33); USING "##.##"; V2!;
PRINT TAB(41); USING "##"; n2%;
PRINT TAB(45); USING "###.###"; (Sum01! / 1000);
PRINT TAB(58); USING "#.###"; P7501! / 1000;
PRINT TAB(67); USING "#.###"; P101! / 1000;
PRINT TAB(76); USING "#.###"; P1501! / 1000
SELECT CASE UCASE$(PP$)
CASE "P", "F"
   PRINT #2, USING "##   "; Order&;
   SELECT CASE Pop1%
   CASE 1
      PRINT #2, "Normal";
   CASE 2
      PRINT #2, "Double Exponential";
   CASE 3
      PRINT #2, "Uniform";
   CASE 4
      PRINT "Positive Skewness";
   CASE ELSE
   END SELECT
   PRINT #2, TAB(25); USING "#.##"; Mean1!;
   PRINT #2, TAB(33); USING "##.##"; V1!;
   PRINT #2, TAB(41); USING "##"; n1%;
   PRINT #2, TAB(45); USING "###.###"; (Sum05! / 1000);
   PRINT #2, TAB(58); USING "#.###"; P7505! / 1000;
   PRINT #2, TAB(67); USING "#.###"; P105! / 1000;
```

```
    PRINT #2, TAB(76); USING "#.###"; P1505! / 1000
    PRINT #2, TAB(6);
    SELECT CASE Pop2%
    CASE 1
       PRINT #2, "Normal";
    CASE 2
       PRINT #2, "Double Exponential";
    CASE 3
       PRINT #2, "Uniform";
    CASE 4
       PRINT "Positive Skewness";
    CASE ELSE
    END SELECT
    PRINT #2, TAB(25); USING "#.##"; Mean2!;
    PRINT #2, TAB(33); USING "##.##"; V2!;
    PRINT #2, TAB(41); USING "##"; n2%;
    PRINT #2, TAB(45); USING "###.###"; (Sum01! / 1000);
    PRINT #2, TAB(58); USING "#.###"; P7501! / 1000;
    PRINT #2, TAB(67); USING "#.###"; P101! / 1000;
    PRINT #2, TAB(76); USING "#.###"; P1501! / 1000
    Lno% = Lno% + 2
    IF Lno% >= 55 THEN
       Lno% = 0
       PRINT #2,
"=============================================================================
========="
       PRINT #2, CHR$(12)
       GOSUB Header
    END IF
CASE ELSE
END SELECT
RETURN

Header:
PRINT "TABLE :  t-test "
PRINT
"=============================================================================
========="
PRINT "                                  Type I Error     Power of Test"
PRINT "No.  Population I      Mean  Variance  n   Sig .05     0.75"; CHR$(229); "    1.00"; CHR$
(229); "    1.50"; CHR$(229)
PRINT "    Population II     Mean  Variance     Sig .01     Sig.05  Sig.05   Sig.05"
PRINT "                                          Sig.01   SIg.01   Sig.01"
PRINT
"=============================================================================
========="
SELECT CASE UCASE$(PP$)
CASE "P", "F"
   PRINT #2, "TABLE :  t-test "
```

```
   PRINT #2,
"=============================================================================
========="
   PRINT #2, "                                    Type I Error     Power of Test"
   PRINT #2, "No.  Population I      Mean  Variance  n  Sig .05     0.75"; CHR$(229); "    1.00";
CHR$(229); "    1.50"; CHR$(229)
   PRINT #2, "    Population II     Mean  Variance     Sig .01    Sig.05  Sig.05  Sig.05"
   PRINT #2, "                                          Sig.01   SIg.01  Sig.01"
   PRINT #2,
"=============================================================================
========="
CASE ELSE
END SELECT

RETURN
END

REM $STATIC
SUB CutRnD (No%)
FOR I% = 1 TO (No% - 1)
   RR! = RND(1)
NEXT I%
END SUB

SUB DoubleExpGen (Mean!, Variance!, Numvar%, Var!())
'parameter Mean!=Mean of population
'         Variance!=Variance of population
'         Numvar%=number of case
'         Var!()=Value of Data to return
Number% = 0
DO
   RandomVar! = RND(1)
   Z1! = Mean! - (SQR(Variance!) * LOG(RandomVar!)) / SQR(2)
   Number% = Number% + 1
   Var!(Number%) = Z1!
   IF Number% >= Numvar% THEN EXIT DO
   RandomVar! = RND(1)
   Z2! = Mean! + (SQR(Variance!) * LOG(RandomVar!)) / SQR(2)
   Number% = Number% + 1
   Var!(Number%) = Z2!
   IF Number% >= Numvar% THEN EXIT DO
LOOP

END SUB

FUNCTION Kurtosis! (Var!(), Mean!, Sigma!)
   Kur! = 0
   FOR Number% = 1 TO UBOUND(Var!)
      Kur! = Kur! + (Var!(Number%) - Mean!) ^ 4
   NEXT Number%
```

```
   Kurtosis! = Kur! / (UBOUND(Var!) * (Sigma! ^ 4))
END FUNCTION

' Sub Routine for Genteration of Normal Distribution By Box & Muller method
' MEAN!      : Mean of Normal Distribution
' VARIANCE!  : Variance of Normal Distribution
' Randomvar1! : Random variable no.1
' Randomvar2! : Random variable no.2
' Z1!,Z2!    : Normal Random variables
' Store variabes = VAR!()
' Number o variabe = Numvar%
SUB NormalGen (Mean!, Variance!, Numvar%, Var!())
Number% = 0
DO
   RandomVar1! = RND(1)
   RandomVar2! = RND(1)
   Z1! = Mean! + SQR(-2 * LOG(RandomVar1!)) * (COS(2 * pi * RandomVar2!)) * SQR
(Variance!)
   Number% = Number% + 1
   Var!(Number%) = Z1!
   IF Number% >= Numvar% THEN EXIT DO
   Z2! = Mean! + SQR(-2 * LOG(RandomVar1!)) * (SIN(2 * pi * RandomVar2!)) * SQR(Variance!)
   Number% = Number% + 1
   Var!(Number%) = Z2!
   IF Number% >= Numvar% THEN EXIT DO
LOOP
END SUB

SUB PosSkewGen (Mean!, Variance!, Numvar%, Var!())
'parameter Mean!=Mean of population
'          Variance!=Variance of population
'          Numvar%=number of case
'          Var!()=Value of Data to return
   NormalGen Mean!, Variance!, Numvar%, Var!()
   Max% = UBOUND(Var!)
   'TransForm Normal to Posistive Skewness Distribution
   'Funct y=-0.3268+1.16050961x+0.2909708x2-0.0886191x3
   FOR I% = 1 TO Max%
      X! = Var!(I%)
      Var!(I%) = -.3268 + (1.16050961# * X!) + (.2909708# * (X! ^ 2)) - (8.861910000000001D-02 *
(X! ^ 3))
   NEXT I%
END SUB

SUB QuickSort (Low%, High%, SortArray!())

  IF Low% < High% THEN

    ' Only two elements in this subdivision; swap them if they are out of
    ' order, then end recursive calls:
```

```
    IF High% - Low% = 1 THEN
      IF SortArray!(Low%) > SortArray!(High%) THEN
        SWAP SortArray!(Low%), SortArray!(High%)
      END IF
    ELSE

      ' Pick a pivot element at random, then move it to the end:
      RandIndex = RandInt%(Low%, High%)
      SWAP SortArray!(High%), SortArray!(RandIndex)
      Partition! = SortArray!(High%)
      DO

        ' Move in from both sides towards the pivot element:
        I% = Low%: J% = High%
        DO WHILE (I% < J%) AND (SortArray!(I%) <= Partition!)
          I% = I% + 1
        LOOP
        DO WHILE (J% > I%) AND (SortArray!(J%) >= Partition!)
          J% = J% - 1
        LOOP

        ' If we haven't reached the pivot element, it means that two
        ' elements on either side are out of order, so swap them:
        IF I% < J% THEN
          SWAP SortArray!(I%), SortArray!(J%)
        END IF
      LOOP WHILE I% < J%

      ' Move the pivot element back to its proper place in the array:
      SWAP SortArray(I%), SortArray(High%)

      ' Recursively call the QuickSort procedure (pass the smaller
      ' subdivision first to use less stack space):
      IF (I% - Low%) < (High% - I%) THEN
        QuickSort Low%, I% - 1, SortArray!()
        QuickSort I% + 1, High%, SortArray!()
      ELSE
        QuickSort I% + 1, High%, SortArray!()
        QuickSort Low%, I% - 1, SortArray!()
      END IF
    END IF
  END IF

END SUB

FUNCTION RandInt% (lower%, Upper%) STATIC
  RandInt% = INT(RND * (Upper% - lower% + 1)) + lower%
END FUNCTION

SUB RRT (Order&, Round%, No1%, No2%)
```

```
FileNo% = FREEFILE
OPEN "R", #FileNo%, "STRT.RND", LEN(TT)
Record& = (Order& - 1) * 1000 + Round%
GET #FileNo%, Record&, TT
No1% = VAL(TT.A)
No2% = VAL(TT.B)
CLOSE #FileNo%
END SUB

FUNCTION Skewness! (Var!(), Mean!, Sigma!)
   SK! = 0
   FOR Number% = 1 TO UBOUND(Var!)
      SK! = SK! + (Var!(Number%) - Mean!) ^ 3
   NEXT Number%
   Skewness! = SK! / (UBOUND(Var!) * (Sigma! ^ 3))
END FUNCTION

FUNCTION SMEAN! (Var!())
   M! = 0
   FOR Number% = 1 TO UBOUND(Var!)
      M! = M! + Var!(Number%)
   NEXT Number%
   SMEAN! = M! / UBOUND(Var!)
END FUNCTION

FUNCTION SVariance! (Var!())
   M! = SMEAN!(Var!())
   S! = 0
   FOR Number% = 1 TO UBOUND(Var!)
      S! = S! + (Var!(Number%) - M!) ^ 2
   NEXT Number%
   SVariance! = S! / (UBOUND(Var!) - 1)
END FUNCTION

SUB TestRandom (Numvar%, NoClass%, Chisquare!)
REDIM Var!(1 TO Numvar%), ObsFrq%(1 TO NoClass%), Upperbound!(1 TO NoClass%)
FOR I% = 1 TO Numvar%
   Var!(I%) = INT(RND(1) * 1000000!)
NEXT I%
QuickSort 1, Numvar%, Var!()
ClassSize! = (Var!(Numvar%) - Var!(1)) / NoClass%
FOR I% = 1 TO NoClass%
   Upperbound!(I%) = Var!(1) + I% * ClassSize!
NEXT I%
FOR I% = 1 TO Numvar%
   J% = 1
   DO WHILE Var!(I%) > Upperbound!(J%)
      J% = J% + 1
   LOOP
   ObsFrq%(J%) = ObsFrq%(J%) + 1
```

```
NEXT I%
ExpectedFrq! = Numvar% / NoClass%
Chisquare! = 0
FOR J% = 1 TO NoClass%
   PRINT ObsFrq%(J%), ExpectedFrq!,
   Chisquare! = Chisquare! + (((ObsFrq%(J%) - ExpectedFrq!) ^ 2) / ExpectedFrq!)
   PRINT ((ObsFrq%(J%) - ExpectedFrq!) ^ 2) / ExpectedFrq!
NEXT J%
PRINT Chisquare!
END SUB

FUNCTION tpair! (VAR1!(), VAR2!(), Mean1!, Mean2!, df%)
'tpair = two variance equal
n1% = UBOUND(VAR1!)
n2% = UBOUND(VAR2!)
xbar1! = SMEAN!(VAR1!())
xbar2! = SMEAN!(VAR2!())
Variance1! = SVariance!(VAR1!())
Variance2! = SVariance!(VAR2!())
Sp! = (((n1% - 1) * Variance1!) + ((n2% - 1) * Variance2!)) / (n1% + n2% - 2)
tcal! = ((xbar1! - xbar2!) - (Mean1! - Mean2!)) / (SQR(Sp!) * SQR((1 / n1%) + (1 / n2%)))
tpair! = tcal!
df% = n1% + n2% - 2
END FUNCTION

FUNCTION tprob# (t#, k%)
'parameter tprob = probability when t<=t#
'       t#    = t-score value to fine prob.
'       k%    = degree of freedom
'
' if k%>30 then limit of t-score to  Z-score
IF k% > 30 THEN
   tprob# = Zscore#(t#)
   EXIT FUNCTION
END IF
' This function calculated by seeing Abramowitz & Stegun (1972)
Zeta# = ATN(t# / SQR(k%))
SELECT CASE k%
CASE 1
   A# = 2 * Zeta# / pi
CASE ELSE
   odd% = ((k% MOD 2) = 1)
   SELECT CASE odd%
   CASE -1  'odd
      IF k% = 3 THEN
         A# = 2 / pi * (Zeta# + SIN(Zeta#) * COS(Zeta#))
      ELSE
         CosZeta# = COS(Zeta#)
         FOR X% = 2 TO k% - 3 STEP 2
            TX# = 1
```

```
            FOR Y% = 2 TO X% STEP 2
               TX# = TX# * Y% / (Y% + 1)
            NEXT Y%
            TX# = TX# * ((COS(Zeta#)) ^ (X% + 1))
            CosZeta# = CosZeta# + TX#
         NEXT X%
         A# = 2 / pi * (Zeta# + SIN(Zeta#) * CosZeta#)
      END IF
   CASE ELSE 'even
      IF k% = 2 THEN
         A# = SIN(Zeta#)
      ELSE
         CosZeta# = 1
         FOR X% = 1 TO k% - 3 STEP 2
            TX# = 1
            FOR Y% = 1 TO X% STEP 2
               TX# = TX# * Y% / (Y% + 1)
            NEXT Y%
            TX# = TX# * ((COS(Zeta#)) ^ (X% + 1))
            CosZeta# = CosZeta# + TX#
         NEXT X%
         A# = SIN(Zeta#) * CosZeta#
      END IF
   END SELECT
END SELECT
tprob# = .5 * (1 + A#)
END FUNCTION

FUNCTION ttest! (Var!(), Mean!, df%)
n% = UBOUND(Var!)
xbar! = SMEAN!(Var!())
sd! = SQR(SVariance!(Var!()))
ttest! = (xbar! - Mean!) / (sd! / SQR(n%))
df% = n% - 1
END FUNCTION
FUNCTION tunpair! (VAR1!(), VAR2!(), Mean1!, Mean2!, df%)
'tpair = two variance not equal
n1% = UBOUND(VAR1!)
n2% = UBOUND(VAR2!)
xbar1! = SMEAN!(VAR1!())
xbar2! = SMEAN!(VAR2!())
Variance1! = SVariance!(VAR1!())
Variance2! = SVariance!(VAR2!())
Sp! = SQR((Variance1! / n1%) + (Variance2! / n2%))
tcal! = ((xbar1! - xbar2!) - (Mean1! - Mean2!)) / Sp!
tunpair! = tcal!
df% = ((Variance1! / n1% + Variance2! / n2%) ^ 2) / (((Variance1! / n1%) ^ 2) / (n1% - 1) +
((Variance2! / n2%) ^ 2) / (n2% - 1))
'PRINT Variance1!, Variance2!
END FUNCTION
```

```
SUB UniformGen (Mean!, Variance!, Numvar%, Var!())
'parameter Mean!=Mean of population
'         Variance!=Variance of population
'         Numvar%=number of case
'         Var!()=Value of Data to return
Min! = Mean! - SQR(3) * SQR(Variance!)
Max! = Mean! + SQR(3) * SQR(Variance!)
Number% = 0
DO
   RandomVar! = RND(1)
   Z1! = Min! + (Max! - Min!) * RandomVar!
   Number% = Number% + 1
   Var!(Number%) = Z1!
LOOP UNTIL Number% >= Numvar%
END SUB

FUNCTION VVariance! (Var!())
   M! = SMEAN!(Var!())
   S! = 0
   FOR Number% = 1 TO UBOUND(Var!)
      S! = S! + (Var!(Number%) - M!) ^ 2
   NEXT Number%
   VVariance! = S! / (UBOUND(Var!))
END FUNCTION

FUNCTION Zscore# (Z#)
' The formula used from Abramowitz & Stegun (1972) formula 2

IF Z# >= -14.14 AND Z# <= 14.14 THEN
   Y# = (.7071067812#) * Z#
ELSE
   Y# = 10
END IF
A1# = 7.052307839999999D-02
A2# = .0422820123#
A3# = .0092705272#
A4# = .0001520143#
A5# = .0002765672#
A6# = .0000430638#
Zscore# = 1 - .5 * (1 + A1# * Y# + A2# * (Y# ^ 2) + A3# * (Y# ^ 3) + A4# * (Y# ^ 4) + A5# * (Y#
^ 5) + A6# * (Y# ^ 6)) ^ (-16)
END FUNCTION
```

โปรแกรมที่ 4 **B-TEST.BAS**

```
DECLARE FUNCTION Expect3# (a#, b#, d%, VarXk!())
DECLARE FUNCTION Expect1# (a#, b#, d%, VarXk!(), SumXk!, SumXkSqr!)
DECLARE FUNCTION Power# (a%, c#)
DECLARE FUNCTION Expect2# (a#, b#, d%, VarXk!())
DECLARE FUNCTION JBNormal! (Delta!, SumXkSqr!, SumXk!, VarXk!())
DECLARE FUNCTION PNormal! (Delta!, SumXkSqr!, SumXk!, VarXk!())
DECLARE FUNCTION JBUniFrm! (Delta!, VarXk!())
DECLARE FUNCTION PUniFrm! (Delta!, VarXk!(), JB!)
DECLARE FUNCTION JBDblExp! (Delta!, VarXk!())
DECLARE FUNCTION PDblExp! (Delta!, VarXk!(), JB!)
DECLARE SUB PosSkewGen (Mean!, Variance!, Numvar%, Var!())
DECLARE SUB RRT (Order&, round%, No1%, No2%)
DECLARE SUB CutRnD (No%)
DECLARE FUNCTION VVariance! (Var!())
DECLARE FUNCTION RandInt% (lower%, Upper%)
DECLARE SUB QuickSort (Low%, High%, SortArray!())
DECLARE SUB UniformGen (Mean!, Variance!, Numvar%, Var!())
DECLARE SUB DoubleExpGen (Mean!, Variance!, Numvar%, Var!())
DECLARE FUNCTION Skewness! (Var!(), Mean!, Sigma!)
DECLARE FUNCTION Kurtosis! (Var!(), Mean!, Sigma!)
DECLARE FUNCTION SMEAN! (Var!())
DECLARE FUNCTION SVariance! (Var!())
DECLARE SUB NormalGen (Mean!, Variance!, Numvar%, Var!())
DECLARE SUB FindXk (VarA!(), VarB!(), VarXk!(), SumXk!, SumXkSqr!)
'******************* Baysian Test ****************
' $DYNAMIC
TYPE XX
  a  AS STRING * 5
  b  AS STRING * 5
  c  AS STRING * 2
END TYPE
DIM SHARED TT AS XX
CONST pi = 3.141593
DIM Var1!(1), Var2!(1), Var3!(1), Var4!(1), Var5!(1), Xk!(1), memArray#(1, 1)
  CLS
  PRINT "Do you want to print on (P) Printer (F) File B-test.OUT (M) Monitor (S) Stop"
  PP$ = INPUT$(1)
  SELECT CASE UCASE$(PP$)
  CASE "P"
    OPEN "O", #2, "LPT1:"
    WIDTH #2, 255
    GOSUB PrintTtest
    CLOSE #2
  CASE "F"
    OPEN "O", #2, "B-test.OUT"
    WIDTH #2, 255
    GOSUB PrintTtest
    CLOSE #2
```

```
    CASE "M"
      GOSUB PrintTtest
    CASE "S"
    CASE ELSE
    END SELECT
CLOSE
END
PrintTtest:
CLS
DO
   INPUT "You want to run in Level 1 or 2 "; Level%
LOOP UNTIL Level% = 1 OR Level% = 2
Lno% = 0
GOSUB Header
'VIEW PRINT 8 TO 25
OPEN "I", #1, "THES.DAT"
DO WHILE NOT EOF(1)
   Sum05! = 0: Sum01! = 0
   P7505! = 0: P1005! = 0: P1505! = 0
   P7501! = 0: P1001! = 0: P1501! = 0
   LINE INPUT #1, NN$
   Order& = VAL(LEFT$(NN$, 2))
   Pop1% = VAL(MID$(NN$, 3, 1))
   Pop2% = VAL(MID$(NN$, 4, 1))
   n1% = VAL(MID$(NN$, 5, 2))
   n2% = VAL(MID$(NN$, 7, 2))
   Mean1! = VAL(MID$(NN$, 9, 5))
   Mean2! = VAL(MID$(NN$, 14, 5))
   V1! = VAL(MID$(NN$, 19, 5))
   V2! = VAL(MID$(NN$, 24, 5))
   GOSUB Baytest
LOOP
VIEW PRINT 1 TO 25
RETURN
Baytest:
fdg% = 0
FOR round% = 1 TO 1000
   RRT Order&, round%, No1%, No2%
   REDIM Var1!(n1%), Var2!(n2%), Var3!(n1%), Var4!(n1%), Var5!(n1%)
   RANDOMIZE round%
   CutRnD No1%
   SELECT CASE Pop1%
   CASE 1
     NormalGen Mean1!, V1!, n1%, Var1!()
   CASE 2
     DoubleExpGen Mean1!, V1!, n1%, Var1!()
   CASE 3
     UniformGen Mean1!, V1!, n1%, Var1!()
   CASE 4
     PosSkewGen Mean1!, V1!, n1%, Var1!()
```

```
CASE ELSE
END SELECT
VV! = SQR((V1! / n1%) + (V2! / n2%))
FOR kk% = 1 TO n1%
   Var3!(kk%) = (.75) * VV! + Var1!(kk%)
   Var4!(kk%) = (1!) * VV! + Var1!(kk%)
   Var5!(kk%) = (1.5) * VV! + Var1!(kk%)
NEXT kk%
RANDOMIZE (1000 + round%)
CutRnD No2%
SELECT CASE Pop2%
CASE 1
   NormalGen Mean2!, V2!, n2%, Var2!()
CASE 2
   DoubleExpGen Mean2!, V2!, n2%, Var2!()
CASE 3
   UniformGen Mean2!, V2!, n2%, Var2!()
CASE 4
   PosSkewGen Mean1!, V2!, n2%, Var2!()
CASE ELSE
END SELECT

n% = n1% * n2%
IF Level% = 1 THEN
   IF Pop1% = 4 THEN
      PopPosterior% = 1
   ELSE
      PopPosterior% = Pop1%
   END IF
ELSE
   IF Pop2% = 4 THEN
      PopPosterior% = 1
   ELSE
      PopPosterior% = Pop2%
   END IF
END IF
SELECT CASE PopPosterior%
CASE 1
   REDIM VarXk!(n1% * n2%)
   FindXk Var1!(), Var2!(), VarXk!(), SumXk!, SumXkSqr!
   P0! = Expect1#(4#, -4#, 25, VarXk!(), SumXk!, SumXkSqr!)
   IF P0! >= (1.96 * VV!) THEN
      Sum05! = Sum05! + 1
   END IF
   IF P0! >= (2.58 * VV!) THEN
      Sum01! = Sum01! + 1
   END IF
   REDIM VarXk!(n1% * n2%)
   FindXk Var3!(), Var2!(), VarXk!(), SumXk!, SumXkSqr!
   P0! = Expect1#(4#, -4#, 25, VarXk!(), SumXk!, SumXkSqr!)
```

```
    IF P0! >= (1.96 * VV!) THEN
      P7505! = P7505! + 1
    END IF
    IF P0! >= (2.58 * VV!) THEN
      P7501! = P7501! + 1
    END IF
    REDIM VarXk!(n1% * n2%)
    FindXk Var4!(), Var2!(), VarXk!(), SumXk!, SumXkSqr!
    P0! = Expect1#(4#, -4#, 25, VarXk!(), SumXk!, SumXkSqr!)
    IF P0! >= (1.96 * VV!) THEN
      P1005! = P1005! + 1
    END IF
    IF P0! >= (2.58 * VV!) THEN
      P1001! = P1001! + 1
    END IF
    REDIM VarXk!(n1% * n2%)
    FindXk Var5!(), Var2!(), VarXk!(), SumXk!, SumXkSqr!
    P0! = Expect1#(4#, -4#, 25, VarXk!(), SumXk!, SumXkSqr!)
    IF P0! >= (1.96 * VV!) THEN
      P1505! = P1505! + 1
    END IF
    IF P0! >= (2.58 * VV!) THEN
      P1501! = P1501! + 1
    END IF
  CASE 2
    REDIM VarXk!(n1% * n2%)
    FindXk Var1!(), Var2!(), VarXk!(), SumXk!, SumXkSqr!
    P0! = Expect2#(4#, -4#, 25, VarXk!())
    IF P0! >= (2.12 * VV!) THEN
      Sum05! = Sum05! + 1
    END IF
    IF P0! >= (3.28 * VV!) THEN
      Sum01! = Sum01! + 1
    END IF
    REDIM VarXk!(n1% * n2%)
    FindXk Var3!(), Var2!(), VarXk!(), SumXk!, SumXkSqr!
    P0! = Expect2#(4#, -4#, 25, VarXk!())
    IF P0! >= (2.12 * VV!) THEN
      P7505! = P7505! + 1
    END IF
    IF P0! >= (3.28 * VV!) THEN
      P7501! = P7501! + 1
    END IF
    REDIM VarXk!(n1% * n2%)
    FindXk Var4!(), Var2!(), VarXk!(), SumXk!, SumXkSqr!
    P0! = Expect2#(4#, -4#, 25, VarXk!())
    IF P0! >= (2.12 * VV!) THEN
      P1005! = P1005! + 1
    END IF
    IF P0! >= (3.28 * VV!) THEN
```

```
      P1001! = P1001! + 1
    END IF
    REDIM VarXk!(n1% * n2%)
    FindXk Var5!(), Var2!(), VarXk!(), SumXk!, SumXkSqr!
    P0! = Expect2#(4#, -4#, 25, VarXk!())
    IF P0! >= (2.12 * VV!) THEN
      P1505! = P1505! + 1
    END IF
    IF P0! >= (3.28 * VV!) THEN
      P1501! = P1501! + 1
    END IF
  CASE 3
    REDIM VarXk!(n1% * n2%)
    FindXk Var1!(), Var2!(), VarXk!(), SumXk!, SumXkSqr!
    QuickSort 1, UBOUND(VarXk!), VarXk!()
    P0! = Expect3#(4#, -4#, 25, VarXk!())
    IF P0! >= (1.65 * VV!) THEN
      Sum05! = Sum05! + 1
    END IF
    IF P0! >= (1.71 * VV!) THEN
      Sum01! = Sum01! + 1
    END IF
    REDIM VarXk!(n1% * n2%)
    FindXk Var3!(), Var2!(), VarXk!(), SumXk!, SumXkSqr!
    QuickSort 1, UBOUND(VarXk!), VarXk!()
    P0! = Expect3#(4#, -4#, 25, VarXk!())
    IF P0! >= (1.65 * VV!) THEN
      P7505! = P7505! + 1
    END IF
    IF P0! >= (1.71 * VV!) THEN
      P7501! = P7501! + 1
    END IF
    REDIM VarXk!(n1% * n2%)
    FindXk Var4!(), Var2!(), VarXk!(), SumXk!, SumXkSqr!
    QuickSort 1, UBOUND(VarXk!), VarXk!()
    P0! = Expect3#(4#, -4#, 25, VarXk!())
    IF P0! >= (1.65 * VV!) THEN
      P1005! = P1005! + 1
    END IF
    IF P0! >= (1.71 * VV!) THEN
      P1001! = P1001! + 1
    END IF
    REDIM VarXk!(n1% * n2%)
    FindXk Var5!(), Var2!(), VarXk!(), SumXk!, SumXkSqr!
    QuickSort 1, UBOUND(VarXk!), VarXk!()
    P0! = Expect3#(4#, -4#, 25, VarXk!())
    IF P0! >= (1.65 * VV!) THEN
      P1505! = P1505! + 1
    END IF
    IF P0! >= (1.71 * VV!) THEN
```

```
        P1501! = P1501! + 1
      END IF
   CASE ELSE
   END SELECT
NEXT round%
PRINT USING "##   "; Order&;
SELECT CASE Pop1%
CASE 1
   PRINT "Normal";
CASE 2
   PRINT "Double Exponential";
CASE 3
   PRINT "Uniform";
CASE 4
   PRINT "Positive Skewness";
CASE ELSE
END SELECT
PRINT TAB(25); USING "#.##"; Mean1!;
PRINT TAB(33); USING "##.##"; V1!;
PRINT TAB(41); USING "##"; n1%;
PRINT TAB(45); "";
SELECT CASE PopPosterior%
CASE 1
   PRINT "Normal";
CASE 2
   PRINT "Double Exponential";
CASE 3
   PRINT "Uniform";
CASE ELSE
END SELECT
PRINT TAB(45 + 17); USING "###.###"; (Sum05! / 1000);
PRINT TAB(58 + 17); USING "#.###"; (P7505! / 1000);
PRINT TAB(67 + 17); USING "#.###"; (P1005! / 1000);
PRINT TAB(76 + 17); USING "#.###"; (P1505! / 1000)
PRINT TAB(6);
SELECT CASE Pop2%
CASE 1
   PRINT "Normal";
CASE 2
   PRINT "Double Exponential";
CASE 3
   PRINT "Uniform";
CASE 4
   PRINT "Positive Skewness";
CASE ELSE
END SELECT
PRINT TAB(25); USING "#.##"; Mean2!;
PRINT TAB(33); USING "##.##"; V2!;
PRINT TAB(41); USING "##"; n2%;
PRINT TAB(45 + 17); USING "###.###"; (Sum01! / 1000);
```

```
PRINT TAB(58 + 17); USING "#.###"; (P7501! / 1000);
PRINT TAB(67 + 17); USING "#.###"; (P1001! / 1000);
PRINT TAB(76 + 17); USING "#.###"; (P1501! / 1000)
SELECT CASE UCASE$(PP$)
CASE "P", "F"
  PRINT #2, USING "##   "; Order&;
  SELECT CASE Pop1%
  CASE 1
    PRINT #2, "Normal";
  CASE 2
    PRINT #2, "Double Exponential";
  CASE 3
    PRINT #2, "Uniform";
  CASE 4
    PRINT "Positive Skewness";
  CASE ELSE
  END SELECT
  PRINT #2, TAB(25); USING "#.##"; Mean1!;
  PRINT #2, TAB(33); USING "##.##"; V1!;
  PRINT #2, TAB(41); USING "##"; n1%;
  PRINT #2, TAB(45); "";
  SELECT CASE PopPosterior%
  CASE 1
    PRINT #2, "Normal";
  CASE 2
    PRINT #2, "Double Exponential";
  CASE 3
    PRINT #2, "Uniform";
  CASE ELSE
  END SELECT
  PRINT #2, TAB(45 + 17); USING "###.###"; (Sum05! / 1000);
  PRINT #2, TAB(58 + 17); USING "#.###"; (P7505! / 1000);
  PRINT #2, TAB(67 + 17); USING "#.###"; (P1005! / 1000);
  PRINT #2, TAB(76 + 17); USING "#.###"; (P1505! / 1000)
  PRINT #2, TAB(6);
  SELECT CASE Pop2%
  CASE 1
    PRINT #2, "Normal";
  CASE 2
    PRINT #2, "Double Exponential";
  CASE 3
    PRINT #2, "Uniform";
  CASE 4
    PRINT "Positive Skewness";
  CASE ELSE
  END SELECT
  PRINT #2, TAB(25); USING "#.##"; Mean2!;
  PRINT #2, TAB(33); USING "##.##"; V2!;
  PRINT #2, TAB(41); USING "##"; n2%;
  PRINT #2, TAB(45 + 17); USING "###.###"; (Sum01! / 1000);
```

```
   PRINT #2, TAB(58 + 17); USING "#.###"; (P7501! / 1000);
   PRINT #2, TAB(67 + 17); USING "#.###"; (P1001! / 1000);
   PRINT #2, TAB(76 + 17); USING "#.###"; (P1501! / 1000)
   Lno% = Lno% + 2
   IF Lno% >= 55 THEN
      Lno% = 0
      PRINT #2,
"=============================================================================
============================="
      PRINT #2, CHR$(12)
      GOSUB Header
   END IF
CASE ELSE
END SELECT
RETURN

Header:
PRINT "TABLE :  Baysian-test "
PRINT
"=============================================================================
============================="
PRINT "                                          Type I Error     Power of Test      "
PRINT "No.  Population I     Mean  Variance  n  Poterior distri.  Sig .05    0.75sd   1.00sd   1.50sd"
PRINT "    Population II     Mean  Variance                        Sig .01    Sig.05   Sig.05   Sig.05"
PRINT "                                                          Sig.01   SIg.01   Sig.01"
PRINT
"=============================================================================
============================="
SELECT CASE UCASE$(PP$)
CASE "P", "F"
   PRINT #2, "TABLE :  Baysian-test "
   PRINT #2,
"=============================================================================
============================="
   PRINT #2, "                                          Type I Error     Power of Test      "
   PRINT #2, "No.  Population I     Mean  Variance  n  Poterior distri.  Sig .05    0.75sd   1.00sd
1.50sd"
   PRINT #2, "    Population II     Mean  Variance                        Sig .01    Sig.05   Sig.05   Sig.05"
   PRINT #2, "                                                          Sig.01   SIg.01   Sig.01"
   PRINT #2,
"=============================================================================
============================="
CASE ELSE
END SELECT
RETURN
END

REM $STATIC
SUB CutRnD (No%)
FOR i% = 1 TO (No% - 1)
```

```
   RR! = RND(1)
NEXT i%
END SUB

SUB DoubleExpGen (Mean!, Variance!, Numvar%, Var!())
'parameter Mean!=Mean of population
'         Variance!=Variance of population
'         Numvar%=number of case
'         Var!()=Value of Data to return
number% = 0
DO
   RandomVar! = RND(1)
   Z1! = Mean! - (SQR(Variance!) * LOG(RandomVar!)) / SQR(2)
   number% = number% + 1
   Var!(number%) = Z1!
   IF number% >= Numvar% THEN EXIT DO
   RandomVar! = RND(1)
   Z2! = Mean! + (SQR(Variance!) * LOG(RandomVar!)) / SQR(2)
   number% = number% + 1
   Var!(number%) = Z2!
   IF number% >= Numvar% THEN EXIT DO
LOOP

END SUB

FUNCTION Expect1# (a#, b#, d%, VarXk!(), SumXk!, SumXkSqr!)
   Etm# = 0
   n% = UBOUND(VarXk!)
   REDIM memArray#(d%, 2)
   w# = a# - b#
   w# = w# / d%: hwidth# = w# / 2
   count% = 1
   j# = 0
   DO
      Delta# = b# + (w# * count% - hwidth#)
      c# = 0
      FOR i% = 1 TO n%
         meandiff# = SumXkSqr! - (2 * Delta# * SumXk!) + (n% * Delta# ^ 2)
         c# = c# + meandiff#
      NEXT i%
      number% = n%
      c# = n% * 10 / c#
      meandiff# = Power#(number% / 2, c#)
      j# = j# + (meandiff# * w#)
      memArray#(count%, 1) = meandiff#
      memArray#(count%, 2) = Delta#
      count% = count% + 1
   LOOP UNTIL count% > d%
   j# = 1 / j#
   count% = 1
```

```
    DO
      Etm# = Etm# + (j# * memArray#(count%, 1) * memArray#(count%, 2) * w#)
      count% = count% + 1
    LOOP UNTIL count% > d%
    Expect1# = Etm#
END FUNCTION

FUNCTION Expect2# (a#, b#, d%, VarXk!())
    Etm# = 0
    n% = UBOUND(VarXk!)
    REDIM memArray#(d%, 2)
    w# = a# - b#
    w# = w# / d%: hwidth# = w# / 2
    count% = 1
    j# = 0
    DO
      Delta# = b# + (w# * count% - hwidth#)
      c# = 0
      FOR i% = 1 TO n%
        meandiff# = ABS(VarXk!(i%) - Delta#)
        c# = c# + meandiff#
      NEXT i%
      number% = n%
      c# = n% * 10 / c#
      meandiff# = Power#(number%, c#)
      j# = j# + (meandiff# * w#)
      memArray#(count%, 1) = meandiff#
      memArray#(count%, 2) = Delta#
      count% = count% + 1
    LOOP UNTIL count% > d%
    j# = 1 / j#
    count% = 1
    DO
      Etm# = Etm# + (j# * memArray#(count%, 1) * memArray#(count%, 2) * w#)
      count% = count% + 1
    LOOP UNTIL count% > d%
    'PRINT USING "##.################"; Etm#
    Expect2# = Etm#
END FUNCTION

FUNCTION Expect3# (a#, b#, d%, VarXk!())
    Etm# = 0
    n% = UBOUND(VarXk!)
    REDIM memArray#(d%, 2)
    w# = a# - b#
    w# = w# / d%: hwidth# = w# / 2
    count% = 1
    j# = 0
    DO
      Delta# = b# + (w# * count% - hwidth#)
```

```
      c# = (VarXk!(n%) - VarXk!(1)) / 2 + ABS((VarXk!(n%) + VarXk!(1)) / 2 + Delta#)
      number% = n%
      c# = n% * 10 / c#
      meandiff# = Power#(number% * .025, c#)
      j# = j# + (meandiff# * w#)
      memArray#(count%, 1) = meandiff#
      memArray#(count%, 2) = Delta#
      count% = count% + 1
   LOOP UNTIL count% > d%
   j# = 1 / j#
   count% = 1
   DO
      Etm# = Etm# + (j# * memArray#(count%, 1) * memArray#(count%, 2) * w#)
      count% = count% + 1
   LOOP UNTIL count% > d%
   Expect3# = Etm#
END FUNCTION

SUB FindXk (VarA!(), VarB!(), VarXk!(), SumXk!, SumXkSqr!)
   SumXk! = 0: SumXkSqr! = 0
   n1% = UBOUND(VarA!)
   n2% = UBOUND(VarB!)
   FOR i% = 1 TO n1%
      FOR j% = 1 TO n2%
         VarXk!((i% - 1) * n2% + j%) = VarA!(i%) - VarB!(j%)
         SumXk! = SumXk! + VarXk!((i% - 1) * n2% + j%)
         SumXkSqr! = SumXkSqr! + ((VarXk!((i% - 1) * n2% + j%)) ^ 2)
         'PRINT USING "### "; i%; J%; ((i% - 1) * n2% + J%);
         'PRINT USING "#,###.####  "; VarA!(i%); VarB!(J%); VarXk!((i% - 1) * n2% + J%);
SumXk!; SumXkSqr!
      NEXT j%
   NEXT i%
END SUB

FUNCTION JBDblExp! (Delta!, VarXk!())
   n% = UBOUND(VarXk!)
   meandiff! = 0
   FOR ii% = 1 TO n%
      meandiff! = meandiff! + ABS(VarXk!(ii%) - Delta!)
   NEXT ii%
   JBDblExp! = meandiff! ^ (-n%)
END FUNCTION

FUNCTION JBNormal! (Delta!, SumXkSqr!, SumXk!, VarXk!())
   n% = UBOUND(VarXk!)
   JBNormal! = (SumXkSqr! - 2 * SumXk! * Delta! + n% * (Delta! ^ 2)) ^ (-n% / 2)
END FUNCTION

FUNCTION JBUniFrm! (Delta!, VarXk!())
   n% = UBOUND(VarXk!)
```

```
   m! = (VarXk!(n%) + VarXk!(1)) / 2
   h! = (VarXk!(n%) - VarXk!(1)) / 2
   JBUniFrm! = (h! + ABS(m! + Delta!)) ^ (-.025 * n%)
END FUNCTION

FUNCTION Kurtosis! (Var!(), Mean!, Sigma!)
   Kur! = 0
   FOR number% = 1 TO UBOUND(Var!)
      Kur! = Kur! + (Var!(number%) - Mean!) ^ 4
   NEXT number%
   Kurtosis! = Kur! / (UBOUND(Var!) * (Sigma! ^ 4))
END FUNCTION

' Sub Routine for Genteration of Normal Distribution By Box & Muller method
' MEAN!      : Mean of Normal Distribution
' VARIANCE!  : Variance of Normal Distribution
' Randomvar1! : Random variable no.1
' Randomvar2! : Random variable no.2
' Z1!,Z2!    : Normal Random variables
' Store variabes = VAR!()
' Number o variabe = Numvar%
SUB NormalGen (Mean!, Variance!, Numvar%, Var!())
number% = 0
DO
   RandomVar1! = RND(1)
   RandomVar2! = RND(1)
   Z1! = Mean! + SQR(-2 * LOG(RandomVar1!)) * (COS(2 * pi * RandomVar2!)) * SQR
(Variance!)
   number% = number% + 1
   Var!(number%) = Z1!
   IF number% >= Numvar% THEN EXIT DO
   Z2! = Mean! + SQR(-2 * LOG(RandomVar1!)) * (SIN(2 * pi * RandomVar2!)) * SQR(Variance!)
   number% = number% + 1
   Var!(number%) = Z2!
   IF number% >= Numvar% THEN EXIT DO
LOOP
END SUB

FUNCTION PDblExp! (Delta!, VarXk!(), JB!)
   n% = UBOUND(VarXk!)
   meandiff! = 0
   FOR ii% = 1 TO n%
      meandiff! = meandiff! + ABS(VarXk!(ii%) - Delta!)
   NEXT ii%
   PDblExp! = (JB! ^ (-1)) * Delta! * (meandiff! ^ (-n%))

END FUNCTION

FUNCTION PNormal! (Delta!, SumXkSqr!, SumXk!, VarXk!())
   n% = UBOUND(VarXk!)
```

```
   PNormal! = Delta! * (SumXkSqr! - 2 * SumXk! * Delta! + n% * (Delta! ^ 2)) ^ (-n% / 2)
END FUNCTION

SUB PosSkewGen (Mean!, Variance!, Numvar%, Var!())
'parameter Mean!=Mean of population
'          Variance!=Variance of population
'          Numvar%=number of case
'          Var!()=Value of Data to return
   NormalGen Mean!, Variance!, Numvar%, Var!()
   Max% = UBOUND(Var!)
   'TransForm Normal to Posistive Skewness Distribution
   'Funct y=-0.3268+1.16050961x+0.2909708x2-0.0886191x3
   FOR i% = 1 TO Max%
      X! = Var!(i%)
      Var!(i%) = -.3268 + (1.16050961# * X!) + (.2909708# * (X! ^ 2)) - (8.8619100000000001D-02 *
(X! ^ 3))
   NEXT i%
END SUB

FUNCTION Power# (a%, c#)
   d# = 1
   DO
      p% = 1
      r# = c#
      IF a% = 1 THEN
         d# = d# * c#
         a% = a% - 1
      ELSE
         DO WHILE (p% * 2 <= a%)
            r# = r# * r#
            p% = p% * 2
         LOOP
         d# = d# * r#
         a% = a% - p%
      END IF
   LOOP UNTIL a% = 0
   Power# = d#
END FUNCTION

FUNCTION PUniFrm! (Delta!, VarXk!(), JB!)
   n% = UBOUND(VarXk!)
   m! = (VarXk!(n%) + VarXk!(1)) / 2
   h! = (VarXk!(n%) - VarXk!(1)) / 2
   PUniFrm! = (JB! ^ (-1)) * (Delta!) * ((h! + ABS(m! + Delta!)) ^ (-.025 * n%))
END FUNCTION

SUB QuickSort (Low%, High%, SortArray!())

  IF Low% < High% THEN
```

```
    ' Only two elements in this subdivision; swap them if they are out of
    ' order, then end recursive calls:
    IF High% - Low% = 1 THEN
      IF SortArray!(Low%) > SortArray!(High%) THEN
        SWAP SortArray!(Low%), SortArray!(High%)
      END IF
    ELSE

      ' Pick a pivot element at random, then move it to the end:
      RandIndex = RandInt%(Low%, High%)
      SWAP SortArray!(High%), SortArray!(RandIndex)
      Partition! = SortArray!(High%)
      DO

        ' Move in from both sides towards the pivot element:
        i% = Low%: j% = High%
        DO WHILE (i% < j%) AND (SortArray!(i%) <= Partition!)
          i% = i% + 1
        LOOP
        DO WHILE (j% > i%) AND (SortArray!(j%) >= Partition!)
          j% = j% - 1
        LOOP

        ' If we haven't reached the pivot element, it means that two
        ' elements on either side are out of order, so swap them:
        IF i% < j% THEN
          SWAP SortArray!(i%), SortArray!(j%)
        END IF
      LOOP WHILE i% < j%

      ' Move the pivot element back to its proper place in the array:
      SWAP SortArray(i%), SortArray(High%)

      ' Recursively call the QuickSort procedure (pass the smaller
      ' subdivision first to use less stack space):
      IF (i% - Low%) < (High% - i%) THEN
        QuickSort Low%, i% - 1, SortArray!()
        QuickSort i% + 1, High%, SortArray!()
      ELSE
        QuickSort i% + 1, High%, SortArray!()
        QuickSort Low%, i% - 1, SortArray!()
      END IF
    END IF
  END IF

END SUB

FUNCTION RandInt% (lower%, Upper%) STATIC
  RandInt% = INT(RND * (Upper% - lower% + 1)) + lower%
END FUNCTION
```

```
SUB RRT (Order&, round%, No1%, No2%)
FileNo% = FREEFILE
OPEN "R", #FileNo%, "STRT.RND", LEN(TT)
Record& = (Order& - 1) * 1000 + round%
GET #FileNo%, Record&, TT
No1% = VAL(TT.a)
No2% = VAL(TT.b)
CLOSE #FileNo%
END SUB

FUNCTION Skewness! (Var!(), Mean!, Sigma!)
  SK! = 0
  FOR number% = 1 TO UBOUND(Var!)
    SK! = SK! + (Var!(number%) - Mean!) ^ 3
  NEXT number%
  Skewness! = SK! / (UBOUND(Var!) * (Sigma! ^ 3))
END FUNCTION

FUNCTION SMEAN! (Var!())
  m! = 0
  FOR number% = 1 TO UBOUND(Var!)
    m! = m! + Var!(number%)
  NEXT number%
  SMEAN! = m! / UBOUND(Var!)
END FUNCTION

FUNCTION SVariance! (Var!())
  m! = SMEAN!(Var!())
  S! = 0
  FOR number% = 1 TO UBOUND(Var!)
    S! = S! + (Var!(number%) - m!) ^ 2
  NEXT number%
  SVariance! = S! / (UBOUND(Var!) - 1)
END FUNCTION

SUB TestRandom (Numvar%, NoClass%, Chisquare!)
REDIM Var!(1 TO Numvar%), ObsFrq%(1 TO NoClass%), Upperbound!(1 TO NoClass%)
FOR i% = 1 TO Numvar%
  Var!(i%) = INT(RND(1) * 1000000!)
NEXT i%
QuickSort 1, Numvar%, Var!()
ClassSize! = (Var!(Numvar%) - Var!(1)) / NoClass%
FOR i% = 1 TO NoClass%
  Upperbound!(i%) = Var!(1) + i% * ClassSize!
NEXT i%
FOR i% = 1 TO Numvar%
  j% = 1
  DO WHILE Var!(i%) > Upperbound!(j%)
    j% = j% + 1
  LOOP
```

```
   ObsFrq%(j%) = ObsFrq%(j%) + 1
NEXT i%
ExpectedFrq! = Numvar% / NoClass%
Chisquare! = 0
FOR j% = 1 TO NoClass%
   PRINT ObsFrq%(j%), ExpectedFrq!,
   Chisquare! = Chisquare! + (((ObsFrq%(j%) - ExpectedFrq!) ^ 2) / ExpectedFrq!)
   PRINT ((ObsFrq%(j%) - ExpectedFrq!) ^ 2) / ExpectedFrq!
NEXT j%
PRINT Chisquare!
END SUB

SUB UniformGen (Mean!, Variance!, Numvar%, Var!())
'parameter Mean!=Mean of population
'         Variance!=Variance of population
'         Numvar%=number of case
'         Var!()=Value of Data to return
Min! = Mean! - SQR(3) * SQR(Variance!)
Max! = Mean! + SQR(3) * SQR(Variance!)
number% = 0
DO
   RandomVar! = RND(1)
   Z1! = Min! + (Max! - Min!) * RandomVar!
   number% = number% + 1
   Var!(number%) = Z1!
LOOP UNTIL number% >= Numvar%
END SUB

FUNCTION VVariance! (Var!())
   m! = SMEAN!(Var!())
   S! = 0
   FOR number% = 1 TO UBOUND(Var!)
      S! = S! + (Var!(number%) - m!) ^ 2
   NEXT number%
   VVariance! = S! / (UBOUND(Var!))
END FUNCTION
```

ประวัติผู้เขียน

ชื่อ	นายอนันต์ เดชพรม
วัน เดือน ปี เกิด	30 ตุลาคม 2506
ประวัติการศึกษา	สำเร็จชั้นมัธยมศึกษาตอนปลาย ที่โรงเรียนสูงเม่นชนูปถัมภ์ อำเภอสูงเม่น จังหวัดแพร่ เมื่อปี 2525
	สำเร็จการศึกษาระดับปริญญาตรี วิทยาศาสตรบัณฑิต สาขาวิชาสถิติ จากมหาวิทยาลัยเชียงใหม่ เมื่อปี 2528
ประวัติการทำงาน	ปี 2531 - ปัจจุบัน
	รับราชการเป็นเจ้าหน้าที่วิเคราะห์นโยบายและแผน กองแผนงาน สำนักงานอธิการบดี มหาวิทยาลัยเชียงใหม่