PROGRAM QUEUE (INPUT, OUTPUT)

AUTHOR:
CHESTER CHIA-ANN LEE
DEPARTMENT OF COMPUTER SCIENCE
THE UNIVERSITY OF TEXAS AT AUSTIN
AUSTIN, TEXAS, 78712

PROGRAM QUEUE EMPLOYS MARKOV PROCESS TO OBTAIN STEADY-STATE PROBABILITY, THROUGHPUT, AVERAGE QUEUE LENGTH, AND STANDARD DEVIATION OF AVG, Q-LENGTH FOR EACH QUEUE IN A CLOSED CYCLE QUEUING NETWORK WHERE SERVICE DISCIPLINE IN EACH QUEUE IS FIRST-COME-FIRST-SERVED. EACH QUEUE IN THE NETWORK HAS ONE OR MORE SERVERS, MEAN SERVICE TIME OF EACH SERVER IS EITHER EXPONENTIAL, HYPEREXPONENTIAL, OR HYPO-EXPONENTIAL. A JOB LEAVING QUEUE I MAY NOT JOIN QUEUE I AT ONCE.

LET:
ND = NO. OF QUEUES IN THE NETWORK
NJ = NO. OF JOBS IN THE NETWORK
NTYPE(I) = TYPE OF DEVICE(S) IN QUEUE I, VALUE OF NTYPE(I) IS 0 IF MEAN SERVICE TIME OF QUEUE I IS EXPONENTIAL, 1 IF IT IS HYPEREXPONENTIAL, AND 2 IF IT IS HYPOEXPONENTIAL.
NST(I) = NO. OF BRANCHES OR STATIONS OF QUEUE I
NDV(I) = NO. OF DEVICES IN QUEUE I
M(N1,S1,S2,...SN) = STATE WHERE NI IS THE NO. OF JOBS IN QUEUE I AND S1,S2,...SN ARE NOS. OF JOBS IN STATIONS 1,2,...,K OF QUEUE I, NOTE THAT THERE IS NO SI IN QUEUE I IF QUEUE I IS EXPONENTIAL
N = TOTAL NO. OF STATES
P(I) = STEADY-STATE PROBABILITY OF STATE I
W(I,J) = PROBABILITY OF A JOB LEAVING QUEUE I FOR QUEUE J
NF(I) = FIRST POSITION OF QUEUE I IN STATE M
NS(I) = NO. OF JOBS IN QUEUE I
MT = TOTAL NO. OF STATIONS OR BRANCHES IN THE NETWORK
U(I) = MEAN SERVICE TIME OF STATION I
V(I) = ARRIVAL PROBABILITY FOR STATION I
R(I) = MEAN SERVICE TIME OF QUEUE I
UT(I,J) = PROBABILITY THAT THERE ARE I JOBS IN QUEUE J
QL(I) = AVERAGE QUEUE LENGTH OF QUEUE I
SD(I) = STANDARD DEVIATION OF AVG, Q-LENGTH FOR QUEUE I
TP(I) = THROUGHPUT FROM QUEUE I

NOTE:
USER OF THIS PROGRAM IS RESPONSIBLE TO GIVE VALUES OF ND, NJ, NTYPE(I), NST(I), NDV(I), W(I,J), U(I), V(I), AND MT.

COMMON M(210,10), NST(5), NTYPE(5), NF(5), NS(5), NDV(5), P(210), N, ND, NJ, MT, \( M(5,5) \), U(10), V(10), UT(8,5), QL(5), SD(5), TP(5), R(5)

READ IN DATA FOR THE NETWORK

READ 10, ND, NJ, MT
10 FORMAT(4012)
READ 10, (NST(I), I=1, ND)
READ 10, (NTYPE(I), I=1,ND)
READ 10, (NDV(I), I=1,ND)
130 READ 140, ((W(I,J), J=1,ND), I=1,ND)
140 FORMAT(10F8.4)
   READ 140, (U(I), I=1,MT)
   READ 140, (V(I), I=1,MT)

CALL SUBROUTINE FCFS FOR QUEUING ANALYSIS

CALL FCFS

PRINT RESULTS

NJ=NJ+1
PRINT 125
125 FORMAT(1H1//32X,3HCP,6X,5HI/O 1,5X,5HI/O 2,5X,5HI/O 3,5X,
1      5HI/O 4//30X,5(7H---------,3X))
PRINT 135, (NDV(I), I=1,ND)
135 FORMAT(/10X,16HSERVERS IN QUEUE,7X,5(I1,9X))
PRINT 942
942 FORMAT(/10X,16HPROB. OF ARRIVAL/)
   DO 946 I=1,ND
   PRINT 945, (W(I,J), J=1,ND)
945 FORMAT(31X,5(F5.3,5X))
946 CONTINUE
   PRINT 948, (R(I), I=1,ND)
948 FORMAT(/10X,17HMEAN SERVICE TIME,5(4X,F5.3,1X))
PRINT 950
950 FORMAT(/10X,11HUTILIZATION/)
   DO 960 I=1,NJ
   J=I-1
   PRINT 955, J, (UT(I,K), K=1,ND)
955 FORMAT(12X,9H JOB,9X,5(F5.3))
960 CONTINUE
   PRINT 965, (QL(I), I=1,ND)
965 FORMAT(/10X,17HMG, QUEUE LENGTH,5(4X,F5.3,1X))
PRINT 980, (SD(I), I=1,ND)
980 FORMAT(/10X,17HSTD. DEV. OF Q=LN,5(4X,F5.3,1X))
PRINT 982, (TP(I), I=1,ND)
982 FORMAT(/10X,1CHTHROUGHPUT,10X,5(F7.2,3X))
STOP
END
SUBROUTINE FCFS

SUBROUTINE FCFS GENERATES ALL STATES OF A GIVEN NETWORK, COMPUTES TRANSITIONS BETWEEN ANY TWO STATES, AND COMPUTES STEADY-STATE PROBABILITIES OF EACH STATE. DEVICE UTILIZATION, THROUGHPUT, AVG. QUEUE LENGTH, AND STANDARD DEVIATION OF AVG. Q-LENGTH OF EACH QUEUE IN THE NETWORK ARE ALSO COMPUTED.

VARIABLES USED ARE AS FOLLOWS:
A(I, J) = TRANSITION RATE FROM STATE I TO STATE J
LIN = INDEX OF QUEUE THAT A JOB ENTERS
LOUT = INDEX OF QUEUE THAT A JOB LEAVES
KIN = INDEX OF BRANCH OR STATION THAT A JOB ENTERS
KOUT = INDEX OF BRANCH OR STATION THAT A JOB LEAVES
DIAG = A(I, I)

COMMON M(210,10), NST(5), NTYPE(5), NF(5), NS(5), NDV(5), P(210), N, ND, 1, NJ, MT, L(5,5), U(10), V(10), UT(8,5), GL(5), SD(5), TP(5), R(5)
DIMENSION A(210,210)

INITIALIZE N AND POSITION THE FIRST INDEX OF EACH QUEUE IN STATE M

N=0
LAST=ND=1
NF(1)=1
DO 25 I=2, ND
IF (NTYPE(I-1) .EQ. 0) GO TO 20
NF(I)=NF(I-1)+NST(I-1)+1
GO TO 25
20 NF(I)=NF(I-1)+NST(I-1)
25 CONTINUE
NT=NF(ND)+NST(ND)-1
IF (NTYPE(ND) .GT. 0) NT=NT+1

GENERATE ALL POSSIBLE COMBINATIONS OF JOBS IN EACH QUEUE

DO 30 I=1, LAST
30 NS(I)=0
IX=LAST
IL=IX+1
40 NN=0
DO 50 J=1, LAST
50 NN=NN+NS(J)
NS(ND)=NJ=NN

CALL SUBROUTINE PERMUTE TO GENERATE POSSIBLE STATES FOR GIVEN NS(I)

CALL PERMUTE
IF (NS(1) .EQ. NJ) GO TO 100
IF (NS(ND) .EQ. 0) GO TO 60
NS(IX)=NS(IX)+1
GO TO 40
60 IX=IX+1
IL=IX+1
JL=0
IF (IX .LE. 1) GO TO 80
DO 70 I=1,IL
70 JL=JL+NS(I)
80 IF (JL+NS(IX)+1 .GT. NJ) GO TO 60
IR=IX+1
DO 90 J=IR,LAST
90 NS(J)=0
NS(IX)=NS(IX)+1
IX=LAST
IL=IX-1
GO TO 40

ALL POSSIBLE STATES ARE EXHAUSTED

100 CONTINUE

RE-POSITION OF ARRIVAL PROB. AND MEAN SERVICE RATE FOR EACH STATION

DO 160 I=1,MT
160 U(I)=1.0/U(I)
JT=MT
DO 180 I=1,ND
IF (NTYPE(I) .EQ. 0) GO TO 180
JT=JT+1
J=JT
170 IF (J .EQ. NF(I)) GO TO 180
U(J)=U(J-1)
V(J)=V(J-1)
J=J-1
GO TO 170
180 CONTINUE

CONSTRUCT TRANSITION MATRIX

DO 600 I=1,N

INITIALIZE MAIN DIAGONAL ELEMENT

DIAG=0.0
DO 560 J=1,N
IF (I .EQ. J) GO TO 560
KT=0
PP=1.0
DO 500 IC=1,ND
ID=NF(IC)
JC=ID+1
JD=JC+NST(ID)-1

CHECK FOR NO. OF JOBS IN EACH QUEUE OF STATES I AND J

IF (IABS(M(I,ID)-M(J,ID))-1) 200,300,550

NO. OF JOBS IN QUEUE IC OF STATE I EQUALS TO QUEUE IC OF STATE J

200 IF (NTYPE(IC)=1) 500,210,250

SERVER IN QUEUE IC IS HYPEREXPONENTIALLY DISTRIBUTED
DO 220 JE=JC,JD
   IF (MI,JE) .NE. M(J,JE)) GO TO 550
220  CONTINUE
      GO TO 500

SERVER IN QUEUE IC IS HYPOEXPONENTIALLY DISTRIBUTED

250  IX=0
     JCC=JD-1
     DO 255 JF=JC,JCC
     IF (MI,JF) .EQ. M(J,JF)) GO TO 255
     IF (MI,JF)-M(J,JF) .EQ. 1 .AND. M(J,JF+1)=M(I,JF+1) .EQ. 1) GO
     1 TO 252
     GO TO 500
252  IX=IX+1
     KOUT=JF
     255  CONTINUE

CHECK FOR TRANSITION BETWEEN STAGES

     IF (IX=1) 500,260,550

GENERATE TRANSITION RATE

260  KT=KT+2
     LOUT=IC
     LIN=IC
     PP=PP*M(I,KOUT)*U(KOUT)
     GO TO 500

THERE IS A POSSIBLE TRANSITION BETWEEN STATES I AND J

300  IF (MI,ID) .GT. M(J,ID)) GO TO 400

A JOB LEAVING STATE J ENTERS STATE I

     IF (NTYPE(IC)=1) 310,330,360

JOB ENTERES HYPOEXPONENTIALLY DISTRIBUTED QUEUE

310  KT=KT+1
     LIN=IC
     GO TO 500

ENTERING HYPER-EXP. DISTRIBUTED QUEUE

330  IX=0
     DO 340 KA=JC,JD

CHECK FOR PERMISSIBILITY OF TRANSITION

     IF (MI,ID) .LT. NDV(IC)) GO TO 335
     IF (MI,KA) .NE. M(J,KA)) GO TO 550
     GO TO 340
335  IF (MI,KA) .GT. M(J,KA)) GO TO 550
     IF (M(J,KA)-M(I,KA)=1) 340,337,550
337  IX=IX+1$ KIN=KA
340 CONTINUE
LIN=IC
KT=KT+1
IF (IX=1) 500,350,550
350 PP=PP+V(KIN)
GO TO 500

JOB ENTERING HYPO=EXP. DISTRIBUTED QUEUE

360 IF (M(J,JC)=M(I,JC)) 370,370,550

CHECK FOR PERMISSIBILITY OF TRANSITION

370 JCC=JC+1
DO 380 JF=JCC,JD
IF (M(I,JF),NE.,M(J,JF)) GO TO 550
380 CONTINUE
KT=KT+1
LIN=IC
GO TO 500

A JOB LEAVING STATE I ENTERS STATE J

400 IF (NTYPE(IC)=1) 410,430,470

COMPUTE TRANSITION RATE FOR EXPONENTIAL QUEUE

410 LOUT=IC
KT=KT+1
IF (M(I,IC)=NDV(IC)) 415,420,420
415 KK=M(I,IC) $ GO TO 425
420 KK=NDV(IC)
425 PP=PP*U(ID)*KK
GO TO 500

COMPUTE TRANSITION RATE FOR HYPEREXPONENTIAL QUEUE

430 IF (M(I,IC),GT.,NDV(IC)) GO TO 440
LX=0
DO 435 JF=JC,JD
IF (M(I,JF)=M(J,JF)) 550,435,432
432 LX=LX+1 $ KK=JF
435 CONTINUE
IF (LX,NE.,1) GO TO 550
KT=KT+1
LOUT=IC
PP=PP*U(KK)*M(I,KK)
GO TO 500

440 RATE=0,0
LI=0 $ LJ=0
DO 460 JF=JC,JD
IF (IABS(M(I,JF)-M(J,JF))=1) 445,450,550
445 IF (M(I,JF),EQ.,0) GO TO 460
IF (LI,NE.,0) OR (LJ,NE.,0) GO TO 460
RATE=RATE+U(JF)*V(JF)*M(I,JF)
GO TO 460
450 IF (M(I,JF),LT.,M(J,JF)) GO TO 455
LI=LI+1
IF (LI .GT. 1) GO TO 550
KOUT=JF
GO TO 460

455 LJ=LJ+1
IF (LJ .GT. 1) GO TO 550
KIN=JF
460 CONTINUE
IF (LI .EQ. 0 .AND. LJ .EQ. 0) GO TO 465
RATE=U(KOUT)*V(KIN)*M(I,KOUT)
465 PP=RATE
KT=KT+1
LOUT=IC
GO TO 500

COMPUTE TRANSITION RATE FOR HYPOEXPONENTIAL QUEUE

470 IF (M(I,JD)=M(J,JD) .NE. 1) GO TO 550
IF (M(J,JC)=M(I,JC) .GT. 1) GO TO 550
JCC=JC+1
S JDD=JD=1
DO 475 JFF=JCC,JDD
475 IF (M(I,JFF) .NE. M(J,JFF)) GO TO 550
KT=KT+1
LOUT=IC
PP=PP*M(I,JD)*U(JD)
500 CONTINUE

CHECK IF MORE THAN ONE TRANSITION HAS OCCURED FOR STATES I AND J

IF (KT .NE. 2) GO TO 550

GENERATE ELEMENT A(I,J)

IF (LOUT .NE. LIN) GO TO 525
A(I,J)=PP S GO TO 540
525 A(I,J)=PP*K(LOUT,LIN)
540 CONTINUE
DIA=DIA+A(I,J)
GO TO 560
550 A(I,J)=0,0
560 CONTINUE
A(I,I)=DIA

REPLACE LAST COLUMN BY 1

A(I,N)=1,0
600 CONTINUE

INVERT TRANSITION MATRIX

DO 800 L=1,N
PIVOT=A(L,L)
800 CONTINUE

CHECK TO SEE IF MAIN DIAGONAL ELEMENT IS ZERO

IF (ABS(PIVOT) .LE. 1.0E-200) GO TO 985
A(L,L)=0,0
DO 700 I=1,N
IF (I .EQ. L) GO TO 700
A(I,L) = A(I,L) / PIVOT  
X = A(I,L)  
DO 650 J = 1, N  
650 A(I,J) = A(I,J) - X*A(L,J)  
700 CONTINUE  
DO 750 J = 1, N  
750 A(L,J) = A(L,J) / PIVOT  
A(L,L) = 1.0 / PIVOT  
800 CONTINUE  
NJJ = NJ + 1  
DO 910 KB = 1, ND  
DO 905 KA = 1, NJJ  
905 UT(KA, KB) = 0.0  
QL(KB) = 0.0  
910 CONTINUE  
DO 925 I = 1, N  

LAST ROW ELEMENTS ARE STEADY-STATE PROBABILITIES

P(I) = A(N,I)  

COMPUTE UTILIZATIONS OF EACH QUEUE

DO 920 J = 1, ND  
LA = NF(J)  
KS = M(I, LA) + 1  
UT(KS, J) = UT(KS, J) + P(I)  
920 CONTINUE  
925 CONTINUE  

COMPUTE AVERAGE QUEUE LENGTH OF EACH QUEUE

DO 935 I = 1, ND  
DO 930 J = 1, NJJ  
930 QL(I) = QL(I) + (J-1)*UT(J, I)  
935 CONTINUE  

COMPUTE THROUGHPUT OF EACH QUEUE

DO 940 I = 1, ND  
IA = NF(I)  
IF (NTYPE(I) .GT. 0) IA = IA + 1  
IB = IA + MST(I) - 1  
R(I) = 0.0  
DO 937 J = IA, IB  
937 R(I) = R(I) + V(J) / U(J)  
940 CONTINUE  
DO 960 I = 1, ND  
TP(I) = 0.0  
DO 955 J = 2, NJJ  
IF (J-1 - NDV(I)) 945, 945, 950  
945 TP(I) = TP(I) + UT(J, I) * (J-1) / R(I)  
GO TO 955  
950 TP(I) = TP(I) + UT(J, I) * NDV(I) / R(I)  
955 CONTINUE  
960 CONTINUE  

COMPUTE STANDARD DEVIATION OF AVG. Q=LENGTH OF EACH QUEUE
DO 975 I=1,ND
SD(I)=0.0
DO 970 J=1,NJ
970 SD(I)=SD(I)+UT(J+1,I)*J**2
SD(I)=SORT(SD(I)=QL(I)**2)
975 CONTINUE
GO TO 995
985 PRINT 990, L
990 FORMAT(//10X,1SHPIVOT=0 AT LOOP,I4)
995 CONTINUE
RETURN
END
FUNCTION NFACTOR(N)

IF (N=1) 10,10,20
10 NFACTOR=1
 RETURN
20 NFACTOR=1
 DO 30 I=2,N
30 NFACTOR=NFACTOR*I
 RETURN
END

SUBROUTINE PERMUTE

SUBROUTINE PERMUTE CONSTRUCTS ALL POSSIBLE STATES FOR FIXED NO. OF JOBS IN EACH QUEUE.

VARIABLES USED ARE AS FOLLOWS:
L(I,J) = TEMPORARY LOCATIONS FOR POSSIBLE STATES
IP(I) = REMAINING NO. OF STATES
LP(I) = NO. OF POSSIBLE COMBINATIONS FOR GIVEN NO. OF JOBS IN I TH. QUEUE
IP = TOTAL NO. OF POSSIBLE STATES

COMMON H(210,10),NST(5),NTYPE(5),NF(5),NS(5),NDV(5),P(210),N,ND,
1 NJ,MT,H(5,5),U(10),V(10),UT(8,5),QL(5),SD(5),TP(5),R(5)
DIMENSION L(10,10),IP(10),LP(10)

IP = 1
DO 70 I = 1, ND
NBGN = NF(I)
LAST = NF(I) + NST(I) - 1
IF (NTYPE(I) NE 0) LAST = LAST + 1

CHECK FOR ZERO JOB IN QUEUE

IF (NS(I) NE 0) GO TO 20
10 DO 15 J = NBGN, LAST
15 L(I,J) = 0
LP(I) = 1
GO TO 65

CONSTRUCT DIFFERENT COMBINATIONS OF JOBS IN EACH STATION OF QUEUE

20 IF (NTYPE(I) - 1) 22,25,25
22 L(I,NBGN) = NS(I)
LP(I) = 1
GO TO 65
25 IF (NS(I) - NDV(I)) 30,35,35
30 MJ = NS(I)
GO TO 40
35 MJ = NDV(I)
40 IA = 1
MA = NF(I) + 1
MB = LAST + 1
DO 42 IA = MA, MB
42 IX = MB
IL = IX + 1
45 MM = 0
DO 47 JA = MA, MB
47 MM = MM + L(I, JA)
L(IA,NBGN) = NS(I)
L(IA,LAST) = MJ = MM
IF (L(IA,MJ) EQ MJ) GO TO 64
IF (L(IA,LAST) EQ 0) GO TO 55
50 IA = IA + 1
DO 52 IB=NBGN,IL
52 L(IA,IB)=L(IA=1,IB)
   L(IA,IX)=L(IA=1,IX)+1
   GO TO 45
55 IX=IX+1
   IL=IL+1
   JL=0
   IF (IX ,EQ., MA) GO TO 59
   DO 57 IB=MA,IL
57 JL=JL+L(IA,IB)
59 IF (JL+L(IA,IX)+1 ,GT., MJ) GO TO 55
   IR=IX+1
   DO 60 JA=IR,MJ
60 L(IA+1,JA)=0
   IF (IX ,EQ., MA) GO TO 62
   DO 61 JA=NBGN,IL
61 L(IA+1,JB)=L(IA,JB)
62 L(IA+1,IX)=L(IA,IX)+1
   IX=MB
   IL=IX+1
   IA=IA+1
   GO TO 45
64 LP(I)=IA
65 IPP=IPP*LP(I)
70 CONTINUE

C C  GENERATE ALL POSSIBLE STATES
C
NB=N+1
N=N+1PP
IP(I)=IPP/LP(I)
DO 75 I=2,ND
75 IP(I)=IP(I=1)/LP(I)
DO 95 J=1,ND
JX=1
XX=1
KA=NF(J)
KB=KA+NST(J)-1
IF (NTYPE(J) ,GT., 0) KB=KB+1
DO 90 K=NB,N
IF (XX ,LE., IP(J)) GO TO 80
   XX=1
   JX=JX+1
   IF (JX ,LE., LP(J)) GO TO 80
   JX=1
80 DO 85 KK=KA,KB
85 M(K,KK)=L(JX,KK)
   XX=XX+1
90 CONTINUE
95 CONTINUE
RETURN
END
PROGRAM LOCABAL(INPUT,OUTPUT)

AUTHOR:
CHESTER CHIA-ANN LEE
DEPT. OF COMPUTER SCIENCE
UNIVERSITY OF TEXAS AT AUSTIN
AUSTIN, TEXAS

PROGRAM LOCABAL EMPLOYS LOCAL BALANCE EQUATIONS TO SOLVE QUEUING NET-WORK PROBLEMS. IT GENERATES STATES FOR THE NETWORK, OBTAINS THE TERMS OF ALL NODES IN THE NETWORK VIA TOPOLOGICAL SORT, COMPUTES STEADY-STATE PROBABILITIES OF ALL STATES, UTILIZATIONS, AVERAGE QUEUE LENGTHS, STANDARD DEVIATIONS OF AVG. QUE-LENGTHS, AND THROUGHPUTS OF ALL QUEUES IN THE NETWORK.

VARIABLES ARE AS FOLLOWS:
INPUT(I,J)= INDEX OF NODES FOR A PATH IN THE NETWORK. NODES ARE NUMBERED FROM 1 TO N. SOURCE NODE IS 0 AND SINK NODE IS N+1, NO SOURCE NODE MAY APPEAR IN INPUT(I,1) AND NO SINK NODE MAY APPEAR IN INPUT(I,2)

*NOTE* NODES WITHIN A QUEUE MUST NUMBERED IN SUCCESSIVE ORDER

TRN(I) = NON-ZERO PROBABILITY FROM INPUT(I,1) TO INPUT(I,2)
NODE = NO OF NODES IN THE NETWORK INCLUDING SOURCE AND SINK
NJOB = NO OF JOBS IN THE SYSTEM
NO = NO OF QUEUES IN THE SYSTEM
MM(I,J) = STATE (N(1),N(2),...,N(NODE)) WHERE N(K) IS NO OF JOBS IN NODE K
NST(I) = NO OF STAGES OR BRANCHES IN QUEUE I
NB(I) = INDEX OF FIRST POSITION OF QUEUE I IN STATE M(J,I)
NJ(I) = NO OF JOBS IN QUEUE I
K = TOTAL NO OF STATES
LJ = NJOB+1
P(I) = STEADY-STATE PROBABILITY OF STATE I
 PTERM(I) = MEAN SERVICE TIME OF NODE I
 TERM(I) = PROBABILITY OF A JOB TRAVERSE FROM SOURCE TO NODE I
 SP(I) = MEAN SERVICE TIME OF QUEUE I
 UT(I,J) = PROBABILITY OF I-1 JOBS IN QUEUE J
 QL(I) = AVERAGE QUEUE LENGTH OF QUEUE I
 SD(I) = STANDARD DEVIATION OF AVG. Q-LNTH OF QUEUE I
 TP(I) = THROUGHPUT OF QUEUE I
 NDV(I) = NO OF SERVERS IN QUEUE I
 INDEX = 0 IF NETWORK CONTAINS A LOOP
 = 1 IF NETWORK IS FREE OF LOOP
 NEDGE = NO OF PATHS IN THE NETWORK

*NOTE* THIS PROGRAM APPLIES TO NETWORK WITH SINGLE SERVER QUEUES ONLY

COMMON MM(100,10),P(100),PTERM(10),TERM(10),UT(10,10),QL(10),
 SD(10),TP(10),NJ(10),NB(10),NST(10),SP(10),NJOB,NODE,NO,K
DIMENSION INPUT(100,2),TRN(100)
READ IN DATA AND STORE NETWORK PATHS

READ 10, NODE, NJOB, NQ
10 FORMAT(40I2)
READ 10, (NST(I), I=1, NQ)
READ 20, (SP(I), I=1, NQ)
READ 20, (PTERM(I), I=1, NODE)
20 FORMAT(10F8.3)
NEDGE=0
30 READ 40, LA, LB, PC
40 FORMAT(2I5,F5.3)
    IF (LA .EQ. 99999) GO TO 45
    NEDGE=NEDGE+1
    INPUT(NEDGE, 1)=LA
    INPUT(NEDGE, 2)=LB
    TRN(NEDGE)=PC
    GO TO 30
45 INDEX=0

CALL FOR TOPOLOGICAL SORTING
    CALL TOPOSRT(INPUT, TRN, NEDGE, INDEX)

CHECK TO SEE IF NETWORK CONTAINS A LOOP
    IF (INDEX .EQ. 1) GO TO 100

CALL FOR GENERATING STATES
    CALL NSTATES

CALL FOR COMPUTING RESULTS
    CALL PSTATES

100 CONTINUE
STOP
END
SUBROUTINE TOPOSRT(INPUT, TRN, NEDGE, INDEX)

COMMON HH(100, 10), P(100), TTERM(10), BTERM(10), UT(10, 10), QU(10),
  1 SQ(10), TP(10), NJ(10), NA(10), NST(10), SP(10), NJOB, NODE, N0, K
INTEGER COUNT(20), TOP(20), QLINK(20), SUC(40), NEXT(40), R, F
DIMENSION INPUT(100, 2), TRN(100), LINK(20)
NP = NODE + 1
LIMIT = 99999
DO 10 I = 1, NODE
  QLINK(I) = 0
  COUNT(I) = 0
  TOP(I) = LIMIT
10 CONTINUE
  QLINK(NP) = 0
  DO 15 J = 1, NEDGE
    INPUT(J, 1) = INPUT(J, 1) + 1
    INPUT(J, 2) = INPUT(J, 2) + 1
15 CONTINUE
N = NODE
DO 20 K = 1, NEDGE
  KA = INPUT(K, 1)
  KB = INPUT(K, 2)
  COUNT(KB) = COUNT(KB) + 1
  SUC(K) = KB
  NEXT(K) = TOP(KA)
  TOP(KA) = K
20 CONTINUE
R = 0
NN = 0
DO 25 K = 1, NODE
  IF (COUNT(K) .NE. 0) GO TO 25
  QLINK(R + 1) = K
  R = K + 1
25 CONTINUE
F = QLINK(1)
30 NN = NN + 1
  LINK(NN) = F
  IF (F .EQ. 0) GO TO 50
  NN = 1
  L = TOP(F)
35 IF (L .EQ. LIMIT) GO TO 45
  KC = SUC(L)
  COUNT(KC) = COUNT(KC) + 1
  IF (COUNT(KC) .NE. 0) GO TO 40
  QLINK(R) = KC
  R = KC + 1
40 L = NEXT(L)
  GO TO 35
45 F = QLINK(F + 1)
GO TO 30
50 IF (N,N.EQ.,0) GO TO 60
INDEX=1
PRINT 55
55 FORMAT(///5X,*NETWORK CONTAINS A LOOP*)
RETURN
60 CONTINUE
BTERM(1)=1,0
DO 120 I=2,NODE
II=LINK(I)
BTERM(II)=0,0
DO 110 J=1,MBEDGE
IF (II,NE.,INPUT(J,2)) GO TO 110
JJ=INPUT(J,1)
BTERM(II)=BTERM(II)+BTERM(JJ)*TRN(J)
110 CONTINUE
120 CONTINUE
DO 130 I=1,MBEDGE
INPUT(I,1)=INPUT(I,1)-1
INPUT(I,2)=INPUT(I,2)+1
130 CONTINUE
NODE=NODE-1
DO 140 I=1,NODE
140 LINK(I)=LINK(I+1)-1
NODE=NODE-1
PRINT 150, (INPUT(I,J),J=1,2),TRN(I),I=1,NODE
150 FORMAT(1H1///10X,*NETWORK FLOW*///S5X,4HNODE,2X,2HTO,2X,4HNODE,2X,
1   *TRANSITION*///S5X,I3,7X,I3,6X,F5,3/)
PRINT 160, (LINK(I),I=1,NODE)
160 FORMAT(///5X,*ORDERED LINK LIST OF QUEUING NETWORK*///S5X,25IS/)
DO 170 J=1,NODE
170 BTERM(J)=BTERM(J+1)
PRINT 180, (BTERM(I),I=1,NODE)
180 FORMAT(///S5X,*BTERM*,*(*,I3,*),=*,F7,4))
RETURN
END
SUBROUTINE NSTATES

CC
CC THIS SUBROUTINE AUTOMATICALLY GENERATES ALL POSSIBLE STATES OF A
CC QUEUING NETWORK.
CC THE INDICES OF A STATE IS: (1,2,...,IL,IX,IR,...,LAST,NODE)
CC
CC
COMMON MM(1000,10),P(1000),PTERM(10),BTERM(10),UT(10,10),QL(10),
1 SD(10),TP(10),NJ(10),NB(10),NST(10),SP(10),NJOB,NODE,NQ,K
2 K=1
3 LAST=NODE=1
4 IX=LAST
5 IL=IX=1
6 DO 10 I=1,LAST
7 10 MM(K,I)=0
8 15 NF=0
9 DO 20 J=1,LAST
10 20 MM(K,J)=NF
11 NF=NF+MM(K,J)
12 MM(K,NODE)=NJOB=NF
13 IF (MM(K,1),EQ, NJOB) GO TO 50
14 IF (MM(K,NODE),EQ, 0) GO TO 35
15 K=K+1
16 DO 30 I=1,IL
17 30 MM(K,I)=MM(K-1,I)
18 MM(K,IX)=MM(K-1,IX)+1
19 GO TO 15
20 35 IX=IX+1
21 IL=IX=1
22 JL=0
23 IF (IX,GTE, 1) GO TO 43
24 DO 40 I=1,IL
25 40 JL=JL+MM(K,I)
26 IF (JL=MM(K,IX)+1,GTE, NJOB) GO TO 35
27 IR=IX+1
28 DO 45 J=IR,LAST
29 45 MM(K+1,J)=0
30 IF (IX,GTE, 1) GO TO 48
31 DO 47 JJ=1,IL
32 47 MM(K+1,JJ)=MM(K,JJ)
33 MM(K+1,IX)=MM(K,IX)+1
34 IX=LAST
35 IL=IX=1
36 K=K+1
37 GO TO 15
38 50 PRINT 55
39 55 FORMAT(/ /[5X,*QUEUING NETWORK STATES]*)
40 DO 65 I=1,K
41 PRINT 60, I, (MM(I,J), J=1,NODE)
42 60 FORMAT(5X,1H(I3,1H),30I4/)  
43 65 CONTINUE
44 RETURN
45 END
SUBROUTINE PSTATES

COMMON MH(1000,10), P(1000), PTERM(10), BTERM(10), UT(10,10), QL(10),
        SD(10), TP(10), NJ(10), NB(10), NST(10), SP(10), NJOB, NODE, NG, K

INITIALIZING

PRINT 7
7 FORMAT(1H1///32X,3MCPU,6X,5HI/O 1,5X,5HI/O 2,5X,5HI/O 3,5X,
        1 5HI/O 4//30X,5(7H-------,3X))
LJ=NJOB+1
NB(I)=1
DO 20 I=2,NG
20 NB(I)=NB(I-1)*NST(I-1)
DO 50 I=1,NG
DO 40 J=1,LJ
40 UT(J,I)=0.0
QL(I)=0.0
SD(I)=P,0
TP(I)=0.0
50 CONTINUE

COMPUTE STEADY-STATE PROBABILITIES

G=0.0
DO 70 I=1,K
P(I)=1.0
DO 65 II=1,NG
NJ(II)=0
MA=NB(II) $ MB=MA+NST(II)=1
IF (MA+NE, MB) GO TO 55
NJ(II)=NJ(II)*MM(I,MA)
P(I)=P(I)*((BTERM(MA)*PTERM(MA))**NJ(II))
GO TO 65
55 DO 60 J=MA,MB
NJ(II)=NJ(II)*MM(I,J)
P(I)=P(I)*((PTERM(J)*BTERM(J))**MM(I,J)/NFACTOR(MM(I,J)))
60 CONTINUE
P(I)=P(I)*NFACTOR(NJ(II))
65 CONTINUE
G=G+P(I)
70 CONTINUE

COMPUTE PROBABILITY FOR JOBS IN QUEUE, AVG, Q-LENGTH, AND STANDARD
        DEVIATION FOR EACH QUEUE

DO 80 I=1,K
P(I)=P(I)/G
DO 75 J=1,NG
NJ(J)=0
75 CONTINUE
MA=NB(J)       \$         MB=MA+NST(J)=1
DO 72 JA=MA,MB
72 NJ(J)=NJ(J)+MH(I,JA)
    JJ=NJ(J)+1
    UT(JJ,J)=UT(JJ,J)+P(I)
    CONTINUE
80 CONTINUE
DO 90 I=1,NQ
    DO 85 J=1,LJ
    QL(I)=QL(I)+(J=1)*UT(J,I)
    SD(I)=SD(I)+UT(J,I)*(J-1)**2
85 CONTINUE
    SD(I)=SORT(SD(I)-QL(I)**2)
90 CONTINUE

COMPUTE THROUGHPUT FOR EACH QUEUE

    DO 95 I=1,NQ
        TP(I)=TP(I)+(1,0=UT(1,I))/SP(I)
    CONTINUE
    PRINT 100, (SP(I),I=1,NQ)
100 FORMAT(//10X,1HMEAN SERVICE TIME,5(4X,F5.3,1X))
    PRINT 105
105 FORMAT(//10X,1HUTILIZATION)
    DO 115 I=1,LJ
        J=I-1
        PRINT 110, J,(UT(J,II),II=1,NQ)
110 FORMAT(12X,II,4H JOB,9X,5(5X,F5.3))
    CONTINUE
    PRINT 120, (QL(I),I=1,NQ)
120 FORMAT(//10X,1HAVG QUEUE LENGTH,5(4X,F5.3,1X))
    PRINT 125, (SD(I),I=1,NQ)
125 FORMAT(//10X,1HSTD. DEVIATION,2X,5(5X,F5.3))
    PRINT 150, (TP(I),I=1,NQ)
150 FORMAT(//10X,1HTHROUGHPUT,6X,5(5X,F5.2))
RETURN
END