STICHTING
MATHEMATISCH CENTRUM
2e BOERHAAVESTRAAT 49
AMSTERDAM

REKENAFDELING

MR 56

# A PERT program in ALGOL 60

by

J. Nederkoorn

February 1963

Table of contents

## 1. Introduction

The purpose of this report is to demonstrate, that the notions underlying PERT are so simple and straightforward, that anyone with a reasonable command of ALGOL 60 and a basic understanding of PERT may easily write his own readable and flexible computer program adapted to the special circumstances of any project.

No new ideas concerning PERT are brought forward. In fact I consider it very doubtful if in the host of papers about PERT, PEP, CPM, CPS etc. following the original publication [1] any new ideas emerged.

The program to be described is certainly not presented as exemplary. To a large extent its form was dictated by the form of some input tapes prepared previously and by the decision to write a single run program.

I wish to thank Messrs. J. Kriens and F. Göbel of the Statistical Department of the Mathematical Centre for procuring this occasion of popularizing ALGOL 60 and for explaining PERT to me, and Mr. Ch. Harmse for his help in preparing this report.

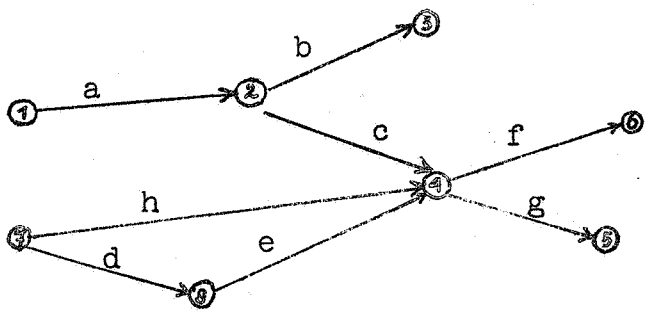
[1] D. Malcolm, J. Roseboom, C.Clark and W. Fazar, Application of a Technique for Research and Development Program Evaluation, Opns. Res. 7, 646-669, 1959.

[2] E.W. Dijkstra, A. Primer of ALGOL 60 Programming, A.P.I.C. Studies in Data Processing, nr. 2, Ac. Pr., London 1962.

## 2. Description of the ALGOL 60 program

### 2.1. Definitions

A project is described by an oriented graph consisting of vectors and nodes.
Every job corresponds with a vector. For every job either its expected duration or 3 estimates (m1 optimistic, m2 most probable, m3 pessimistic) of the duration are supposed to be known. In the latter case PERT calculates the expected duration as
ed = 1/6 (m1 + 4m2 + m3)




In the program nodes are used only for administrating dependences of jobs. All interest is focussed on the jobs.

In the above diagram job a is an immediate predecessor of jobs b and c, which means that there exists a technical dependence by force of which the execution of jobs b and c must wait until job a has been finished.

Suppose jobs d, e and h have expected durations 3, 5 and 10 hours respectively. Since jobs h and d can start at the beginning of the project (t=0), the following statements hold:

est (h) = 0 (est = early start time)
est (d) = 0
est (e) = 3

The early start time of a job is the maximum of the early start times of its immediate predecessors, each being increased with its expected duration.

In formula:

$$est(x) = \max_i \left(est(pred_i(x)) + ed(pred_i(x))\right)$$

Here $pred_i(x)$ means the $i^{th}$ predecessor of x.
If a job x has no predecessor at all, est(x) = 0.

Any job of the network may be appointed as "completing job" for a special run of the program. This means: during this run the program only considers as relevant the jobs preceding this one either immediately or indirectly.

If we choose job g to be the "completing job", the jobs b and f become irrelevant. (Should we wish to extend our calculations to the whole project, then, in the above example, we would have to add some fictive jobs and nodes.) Since job e may be postponed freely for 2 hours without delaying the starting time of any other job:

fs (e) = 2 (fs = free slack)
fs (d) = 0
fs (h) = 0

The free slack of a job depends on its own early start time and the early start time of its immediate followers. This latter quantity, of course, is equal for all immediate followers of one job.

In formula:
$fs(x) = est\ (foll(x)) - est(x) - ed(x)$

Here foll(x) means any job immediately following job x.

By postponing the execution of all jobs as much as possible without nevertheless delaying the execution of job g, we get

lst(h)=0 (lst=late start time)
lst(d)=2
lst(e)=5
lft(h)=10 (lft = late finish time)
lft(d)=5
lft(e)=10

The late finish time of a job is the minimum of the late start times of its immediate followers (denoted by $foll_i(x)$). In formula:

$$lft(x) = \min_i\ (lft\ (foll_i(x)) - ed(foll_i(x))$$

If there are no followers at all, the late finish time of a job will be equal to its early finish time. This will be the case for the completing job only.

Since not only e, but also d may be postponed, it becomes clear, that the concept of free slack does not suffice to describe the essential features of the graph.
We define

$ts(x) = lst(x) - est(x)$ (ts = total slack)
$ts(d) = 2$

By definition the <u>critical path</u> is made up by the chain of all jobs with total slack 0.

## 2.2 Program output

The program punches a tape consisting of one or three lists. Since this tape contains the necessary lay out symbols, the Flexowriter on reading the tapes will produce a printed copy of the lists.

List 1 ( LIST OF RELEVANT JOBS )

All relevant jobs occur in this list in a systematic order. The principles of this ordering are:

1. lower total slack precedes higher total slack. If total slacks are equal, the decision depends on
2. early start times. If early start times are equal also, then
3. lower job number precedes higher job number.

A line of list 1 consists of the following numbers:

a. job number
b. number of begin node of the job
c. number of end node of the job
d. expected duration, measured in hours, days or weeks. Throughout one specific run of the program all calculations are in one of these units of time, chosen beforehand. In calculations including times the program takes heed of exactly one decimal after the point.
e. capacities. With each job some (at most 5) non-negative integers may be associated denoting the demands on behalf of the job to be made on different kinds of resources (equipment, crafts, money etc.) The number of capacities to be regarded by the program has to be specified beforehand. The problem of levelling these requirements by suitable delays of particular jobs is not solved by the program but on purpose left to management. No general, reliable algorithm has come into our notice for solving this problem.

Some of the following numbers denote durations, converted however by the program into calendar data. In this conversion some rounding up takes place, by which times or durations not exactly corresponding with the starting point of a working day are converted into the date of the next working-day. For details see description input-tape.

f. t(i) = early start time

g. p(i) = planned start time
With each job a planned start time and a planned finish time may be associated, in order to compare the results of the PERT calculations with previous planning efforts.

h. T(i) = late start time

i. t(j) = early finish time

j. p(j) = planned finish time

k. T(j) = late finish time

l. a(ij)= free slack

m. A(ij)= total slack

n. sigma= $\sigma$ = standard deviation, calculated here as
$^1/6\ (m_3-m_1)$ if 3 estimates of job duration were given.
Otherwise sigma = 0.

o. sigmasqu = $\sigma^2$ = variance.
For the use and significance of these quantities see [1].

p. Finally the t(i) ..... T(j) are repeated in unconverted form.

List 2 (early start times) and 3 (late start times)

In order to provide a survey of the demands made by the project on the available resources 2 extreme cases are considered:

List 2. every job is started and completed as soon as possible;

List 3. every job is delayed as far as possible without delaying the completion of the „completing job".

Of course, the punching of lists 2 and 3 is skipped when the number of capacities given for each job has been specified to be zero.
From here on we explain only list 2 to avoid repetitions.
A line of list 2 consists of the following numbers:

a. early start time, converted to a calendar date;
b. early start time in unconverted form. No two lines give the same value for b, but repetitions in the column a may occur, because of rounding off in conversion;
c. one or more integers (5 at most) denoting sums of capacities required in the period immediately following the early start time mentioned under b. Between two start times, of course, finish times may occur, not coinciding with start times; these times are not mentioned in the list; of course, they reduce the total capacities required. So only maximum requirements become visible in the list, not the minima.
d. job numbers of the relevant jobs contributing to the sums of capacities in the previous columns.

## 2.3. Description of ALGOL text

The program is written in ALGOL 60, with due observance of the restrictions imposed by the X1-translator of Dijkstra and Zonneveld. For in- and output some extensions had to be used. Input uses the standard function „read", the value of which is always made equal to the next number on the tape. Principal output procedure is ABSFIXP (n,m,x) which punches x in fixed point decimal notation, with n digits before and m after the point. Other output procedures are PUTEXT (punching text), RUNOUT (punching blank tape), TAPEND (punching interrogation mark and blank tape), STOPCODE (punching stopcode and blank tape),

PUSPACE (punching a given number of spaces) and PUNLCR (punching the Flexowriter symbol „new line, carriage return").

Most of the data are stored in the two dimensional array D, which may be considered as consisting of nj rows and nc+11 columns. Each row corresponds with a job.

Column:

| | |
|---|---|
| 1 | numbers of begin nodes; also used for storing markers during the punching of lists 2 and 3; |
| 2 | numbers of end nodes; |
| 3 | expected durations; |
| 4 | early start times; |
| 5 | free slack; |
| 6 | late finish times; |
| 7 | standard deviations ( $\sigma$ ); |
| 8 | capacities, only if nc $\neq$ 0; |
| ⋮ | |
| 7+nc | |
| 8+nc | planned start times — only if c2 $\neq$ 0; |
| 9+nc | planned finish times |
| 10+nc | working space for procedure „order". |
| 11+nc | |

All other comments and explanations can be found in the program text itself.

```
begin   comment PERT program by J Nederkoorn, R 856, code-nr JNE 090163/2876;

        boolean b; integer c0, c00, c1, c2, nc, nj, ncj, i, j, K, k, l, m, n, p; real m1, m2, m3; integer array cap[1 : 5];

CONTROL TAPE:
length calendar tape:                                                   c0:= read;
number of working hours a day 1 if irrelevant:                          c00:= 100 × read;

        begin   integer array CAL[1 : 2, 1 : c0];
                for i:= 1 step 1 until c0 do begin CAL[1, i]:= read × 10; CAL[2, i]:= read end;

        choosing first 1 or last 2 calendar date as point of reference:                                 c1:= read;
        determining whether 3 durations or 1 mean duration given:                                       b:= 3 = read;
        number of capacities:                                                                           nc:= read;
        planned start and finish time for each job given as data 1 or as durations 2 or not given 0 :   c2:= read;
        number completing job:                                                                          ncj:= read;
        number of jobs in network:                                                                      nj:= read;

                begin   integer array D[1 : nc + 11, 1 : nj];

                INPUT:  for i:= 1 step 1 until nj do
                        begin   D[1, i]:= read; D[2, i]:= read; m1:= read; D[4, i]:= D[5, i]:= D[6, i]:= -1;
                                if b then begin m2:= read; m3:= read; D[3, i]:= (m1 + 4 × m2 + m3) × 1.666 6667; D[7, i]:= (m3 - m1) × 16.666 667 end
                                     else begin D[3, i]:= 10 × m1; D[7, i]:= 0 end;
                                for j:= 1 step 1 until nc do D[7 + j, i]:= read;
                                if c2 ≠ 0 then begin D[8 + nc, i]:= read × 10; D[9 + nc, i]:= read × 10 end
                        end;

                TOOLS:  begin   integer procedure est (a); value a; integer a;
                                begin   integer p, q, r;
                                        p:= 0; for q:= 1 step 1 until nj do
                                        begin   if D[2, q] = D[1, a] then
                                                begin   r:= (if D[4, q] = -1 then est (q) else D[4, q]) + D[3, q];
                                                        if r > p then p:= r; D[5, q]:= r
                                                end
                                        end compare immediate predecessors; est:= D[4, a]:= p;
                                        for q:= 1 step 1 until nj do begin if D[2, q] = D[1, a] then D[5, q]:= p - D[5, q] end free slack
                                end recursive calculation of early start time;

                                integer procedure lft (a); value a; integer a;
                                begin   integer p, q, r;
                                        p:= K; for q:= 1 step 1 until nj do
                                        begin   if D[2, a] = D[1, q] ∧ D[4, q] ≠ -1 then
                                                begin   r:= (if D[6, q] = -1 then lft (q) else D[6, q]) - D[3, q];
                                                        if r < p then p:= r
                                                end
                                        end compare immediate followers; lft:= D[6, a]:= p
                                end recursive calculation of late finish time;

                                integer procedure cal (a); value a; integer a;
                                begin   integer i, j, l;
                                        if c1 = 2 then a:= a + CAL[1, c0] - K; i:= 1; j:= c0;
                                DECR:   l:= (i + j) : 2;
                                        if a < CAL[1, l] then begin j:= l; goto DECR end;
                                        if a = CAL[1, l] ∨ a < CAL[1, l + 1] then goto READY;
                                INCR:   i:= l + 1; goto DECR;
                                READY:  i:= a × 10 : c00; rounded number of days:
                                        a:= if a = i × c00 : 10 then i else i + 1;
                                        cal:= if a = 10 × CAL[1, l + 1] : c00 then CAL[2, l + 1] else CAL[2, l] + 4 × (a - 10 × CAL[1, l] : c00)
                                end searching calendar date in array CAL;
```

```
        procedure order (cr1, cr2, RN); integer cr1, cr2, RN;
        begin   integer kk, ll, rn; RN:= 0; j:= nc + 10; k:= nc + 11;
                for i:=  1 step 1 until nj do D[j, i]:= if D[4, i] ≥ 0 then cr1 else -1;
                for ll:= 1 step 1 until nj do
                begin   if D[j, ll] ≥ 0 then
                        begin   rn:= 0; for kk:= 1 step 1 until nj do
                                begin   if      if D[j, kk] < D[j, ll] ∧ D[j, kk] ≠ -1
                                                then true
                                                else    if D[j, kk] = D[j, ll]
                                                        then    (if D[cr2, kk] < D[cr2, ll]
                                                                 then true
                                                                 else    if D[cr2, kk] = D[cr2, ll] then kk ≤ ll else false)
                                                        else false
                                        then rn:= rn + 1
                                end searching lower or equal elements; if rn > RN then RN:= rn; D[k, rn]:= ll
                        end
                end finding order numbers of relevant jobs
        end order;

    CALCULATIONS:
        K:= est (ncj) + D[3, ncj];
        for i:= 1 step 1 until nj do begin if D[4, i] ≠ -1 ∧ D[6, i] = -1 then lft (i) end;
        ordering jobs as to total slack:
        order (D[6, i] - D[4, i] - D[3, i], 4, 1);

    FIRST HEADING:
        RUNOUT; PUNLCR; PUTEXT (
¢PERT  program R856 , 090163 ,  LIST   OF   RELEVANT   JOBS

  job   begin   end   time        capacities          t(i)   p(i)   T(i)   t(j)   p(j)   T(j)   a(ij)  A(ij)   sigma sigmasqu
  nr    node    node          1   2   3   4   5     start  data          finish  data            slack¢);
        LIST1:  for i:= 1 step 1 until l do
                begin   j:= D[k, i];
                        PUNLCR; ABSFIXP (3, 0, j); ABSFIXP (6, 0, D[1, j]); ABSFIXP (6, 0, D[2, j]); ABSFIXP (4, 1, D[3, j]/10);
                        for m:= 1 step 1 until 5 do begin if m ≤ nc then ABSFIXP (3, 0, D[m + 7, j]) else PUSPACE (5) end;
                        ABSFIXP (6, 0, cal (D[4, j]));
                        if c2 = 0 then PUSPACE (8)
                                  else if c2 = 1 then ABSFIXP (6, 0, D[nc + 8, j] : 10)
                                                 else ABSFIXP (6, 0, cal (D[nc + 8, j]));
                        ABSFIXP (6, 0, cal (D[6, j] - D[3, j]));
                        ABSFIXP (6, 0, cal (D[4, j] + D[3, j]));
                        if c2 = 0 then PUSPACE (8)
                                  else if c2 = 1 then ABSFIXP (6, 0, D[nc + 9, j] : 10)
                                                 else ABSFIXP (6, 0, cal (D[nc + 9, j]));
                        ABSFIXP (6, 0, cal (D[6, j]));
                        ABSFIXP (3, 1, if D[5, j] = -1 then 0 else D[5, j]/10);
                        ABSFIXP (3, 1, D[nc + 10, j]/10);
                        ABSFIXP (3, 2, D[7, j]/100);
                        ABSFIXP (3, 2, D[7, j] ↑ 2 / 10↑4);
                end;
                PUNLCR; PUNLCR;PUTEXT (¢
  job   t(i)    p(i)    T(i)    t(j)    p(j)    T(j)
  nr    start times             finish times
¢);
```

```
                PUNLCR; for i:= 1 step 1 until 1 do
                begin   j:= D[k, i]; PUNLCR; ABSFIXP (3, 0, j);
                        ABSFIXP (5, 1, D[4, j]/10);
                        if c2 ≠ 2 then PUSPACE (9)
                                  else ABSFIXP (5, 1, D[nc + 8, j]/10);
                        ABSFIXP (5, 1, (D[6, j] - D[3, j])/10);
                        ABSFIXP (5, 1, (D[4, j] + D[3, j])/10);
                        if c2 ≠ 2 then PUSPACE (9)
                                  else ABSFIXP (5, 1, D[nc + 9, j]/10);
                        ABSFIXP (5, 1, D[6, j]/10)
                end;

        CONTROL OUTPUT:
                PUNLCR; PUNLCR;  if c00 ≠ 100 then begin PUTEXT (⟨Number of working hours a day⟩); ABSFIXP (4, 2, c00/100); PUNLCR end;
                if c1 = 1 then PUTEXT (⟨First⟩) else PUTEXT (⟨Last⟩); PUTEXT (⟨calendar date chosen as point of reference⟩); PUNLCR;
                PUTEXT (⟨Number completing job⟩); ABSFIXP (5, 0, ncj); if nc = 0 then goto END OF PROGRAM;

        SECOND HEADING:
                PUNLCR; PUNLCR; PUTEXT (⟨Early start  times   Capacities⟩); PUSPACE (7 × nc - 6); PUTEXT (⟨Job numbers⟩); PUNLCR;
                order ( D[4, i], 4, 1); m:= 0;

        LIST 2 or 3:
                k:= -1; for i:= 1 step 1 until 1 do
                begin   if D[nc + 10, D[nc + 11, i]] > k then
                        begin   PUNLCR; k:= D[nc + 10, D[nc + 11, i]]; ABSFIXP (6, 0, cal (k)); ABSFIXP (7, 1, k/10);
                                for j:= 1 step 1 until nj do D[1, j]:= -1;
                                cap[1]:= cap[2]:= cap[3]:= cap[4]:= cap[5]:= 0;
                                for j:= 1 step 1 until nj do
                                begin   if      if D[nc + 10, j] ≤ k
                                                then D[nc + 10, j] + D[3, j] > k ∧ D[4, j] ≠ -1
                                                else false
                                        then begin for p:= 1 step 1 until nc do cap[p]:= cap[p] + D[7 + p, j]; D[1, j]:= +1 end
                                end preparation of line;
                                for p:= 1 step 1 until nc do ABSFIXP (5, 0, cap[p]); PUSPACE (6);
                                for j:= 1 step 1 until nj do begin if D[1, j] = +1 then ABSFIXP (3, 0, j) end
                        end
                end; m:= m + 1; if m = 1 then
                begin

        THIRD HEADING:  PUNLCR; PUNLCR;
                        PUTEXT (⟨Late  start  times   Capacities⟩); PUSPACE (7 × nc - 6); PUTEXT (⟨Job numbers⟩); PUNLCR;
                        order ((D[6, i] - D[3, i]), 4, 1);
                        goto LIST 2 or 3
                end third list;

        END OF PROGRAM: PUNLCR; TAPEND; STOPCODE
        end calculation and output

    end input etc.

end control tape etc.

end

PERT program
```

## 4. Preparing input tapes

The input tape is punched on a so-called Flexowriter. Most X1-installations running ALGOL-programs possess a Flexowriter of special type, slightly adapted to the needs of the ALGOL 60 reference language.

On the tape occur exclusively:

1. blank tape. The tapes begins and ends with about 40 cm blank tape. After each piece of blank tape (which may be freely interspersed everywhere) a case definition must follow.
2. Numbers, separated by so-called number separators. Number separators are: + or -, a comma, an interrogation mark, at least two successive spaces, tab signals, carriage return signals and comment.
3. Comment. This should be bracketed by apostrophes (').
   Consequently, an apostrophe must not occur inside comment.

After the last number on the tape an interrogation mark is required.

The tape consists of two parts:

I The control tape
II The network tape

### 4.1. The control tape

This piece of tape contains - next to comment, which may be added freely between numbers - the following numbers:

1.c0 length (i.e. number of number pairs ) of the calendar tape

2.c00 number of working hours a day. This number is constant for the whole network. If durations are calculated in days or weeks then c00 = 1.

3. The calendar tape. This consists of number pairs, for instance:

| | |
|---|---|
| 0 | 20163 |
| 3 | 70163 |
| 8 | 140163 |

etc.

or

| | |
|---|---|
| 0 | 20163 |
| 24.75 | 70163 |
| 66 | 140163 |

etc.

The integer numbers of the second column are calendar data, in an obvious notation. The first column describes how many „units of working time" (hours, days or weeks) have elapsed at the beginning of the working day mentioned in the second column since the starting point of the calendar tape. Consequently the first number of column 0 is always zero. The other numbers of column 0 will be multiples of c00. These numbers may be reals. It makes no sense to specify them in more then two decimals after the point, since the program disregards higher precision. If calculations are in weeks, the numbers of column 1 will be 0,1,2,3 etc.

In column 2 all data must be mentioned which represent the beginning of an uninterrupted working-period of one or more days. Unless calculations are in weeks, such a period must be included entirely within one calendar month.
Generally, the numbers of column 2 will be Mondays.
If a holiday falls on Monday, then Tuesday will be selected.
Holidays on Friday have no effects on the calendar tape.
If one or more holidays divide a week in two parts, then an extra period should be added to the tape. Those extra periods become necessary also if a working period extends over two calendar months and calculations are in hours or days. In these cases the first working day of the second month is mentioned in column 2.

In running the PERT program one has to make a choice from 2 possibilities:

a. The project begins at a fixed starting point. This should be the starting point of the calendar tape and c1 = 1.
b. The project should be completed at a fixed moment. This moment should be represented in the last row of the calendar tape and c1 = 2.

In both cases one should take care that the calendar tape at the loose end extends far enough to cover the whole project.
Following the calendar tape should be punched:

4. c1 This will be 1 or 2, as mentioned before.
5. 3 if 3 estimates of duration were given for each job; or 1 in the other case.
6. nc number of capacities given for each job. ($0 \leqslant nc \leqslant 5$.)
7. 0 if no planned times will occur on network tape; in this case the columns p(i) and p(j) of the first output list will be left open by the program.

1 if planned times are given as calendar data on network tape (same code as calendar tape); in this case the columns p(i) and p(j) are only filled up in the first part of the first list viz. under „start data" and „finish data".
Under „start times" and „finish times" these columns will be left open.

2 if planned times are given as durations on network tape. The columns p(i) and p(j) are then filled up completely.

8. ncj number of completing job. Each job from the network may be appointed as completing job.
If one wishes to appoint a node (event) combining the ends of several jobs, then a fictive job with duration zero should be added.

9. nj number of jobs on the network tape.

4.2 The network tape

This tape immediately follows the control tape. Together, both tapes form one indivisible tape, the input tape.
The network tape consists of rows, i.e. sets of numbers, each set being of equal length and corresponding with a job.
The number of a job does not occur explicitly on the tape. It is equal to the ordinal number of its number set.
A row contains:

1. The number of the begin node of the job.
2. The number of the end node of the job.
( 1 and 2 are non-negative integers )
3. Either 3 estimates for job duration ( an optimistic, a probable and a pessimistic one) or 1 estimate, the expected duration. These are non-negative reals. The program neglects decimals after the point, except the first. See nr. 5 of control tape.

4. nc capacities. These are non-negative integers. See nr. 6 of control tape. ($0 \leq n \leq 5$).
5. Possibly (if $c1 \neq 0$) two planned times, either in the form of a calendar date or of a duration. See nr.7 of control tape.

The order of jobs is completely free. The numbers of nodes can be chosen freely.

If one wants to run the program periodically during the completion of a project, for each run finished jobs will have to be deleted from the tape and durations of jobs in progress and planned data will have to be revised.

## 5. Some examples of input tapes and printed output

```
10      'co'
1       'working hours'

        'calender'

0                20163
3                70163
8               140163
13              210163
18              280163
22               10263
23               40263
28              110263
33              180263
38              250263

                        1       'c1'
                        1       '2 or 1'
                        0       'nc'
                        0       'c2'
                        10      'ncj'
                        12      'nj'

1000    1001    4.5     '1'
1004    1002    8.25    '2'
1002    1003    8.4     '3'
1001    1002    7       '4'
1005    1003    10      '5'
1006    1005    8.15    '6'
1004    1005    4       '7'
1006    1003    16.5    '8'
1000    1004    4.33    '9'
1003    1007    0       '10'
1001    1005    6       '11'
1000    1006    7       '12'
?
' TEST TAPE I'
```

'PERT program R856 , 090163 , LIST OF RELEVANT JOBS

| job nr | begin node | end node | time | capacities 1 | 2 | 3 | 4 | 5 | t(i) start | p(i) data | T(i) | t(j) finish | p(j) data | T(j) | a(ij) slack | A(ij) | sigma | sigmasqu |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 12 | 1000 | 1006 | 7.0 | | | | | | 20163 | | 20163 | 110163 | | 110163 | .0 | .0 | .00 | .00 |
| 6 | 1006 | 1005 | 8.1 | | | | | | 110163 | | 110163 | 240163 | | 240163 | .0 | .0 | .00 | .00 |
| 5 | 1005 | 1003 | 10.0 | | | | | | 240163 | | 240163 | 70263 | | 70263 | .0 | .0 | .00 | .00 |
| 10 | 1003 | 1007 | .0 | | | | | | 70263 | | 70263 | 70263 | | 70263 | .0 | .0 | .00 | .00 |
| 8 | 1006 | 1003 | 16.5 | | | | | | 110163 | | 150163 | 50263 | | 70263 | 1.6 | 1.6 | .00 | .00 |
| 9 | 1000 | 1004 | 4.3 | | | | | | 20163 | | 90163 | 90163 | | 150163 | .0 | 4.1 | .00 | .00 |
| 2 | 1004 | 1002 | 8.3 | | | | | | 90163 | | 150163 | 210163 | | 250163 | .0 | 4.1 | .00 | .00 |
| 3 | 1002 | 1003 | 8.4 | | | | | | 210163 | | 250163 | 310163 | | 70263 | 4.1 | 4.1 | .00 | .00 |
| 1 | 1000 | 1001 | 4.5 | | | | | | 20163 | | 90163 | 90163 | | 160163 | .0 | 4.6 | .00 | .00 |
| 11 | 1001 | 1005 | 6.0 | | | | | | 90163 | | 160163 | 170163 | | 240163 | 4.6 | 4.6 | .00 | .00 |
| 4 | 1001 | 1002 | 7.0 | | | | | | 90163 | | 160163 | 180163 | | 250163 | 1.1 | 5.2 | .00 | .00 |
| 7 | 1004 | 1005 | 4.0 | | | | | | 90163 | | 180163 | 150163 | | 240163 | 6.8 | 6.8 | .00 | .00 |

'

| job nr | t(i) start | p(i) times | T(i) | t(j) finish | p(j) times | T(j) |
|---|---|---|---|---|---|---|
| 12 | .0 | | .0 | 7.0 | | 7.0 |
| 6 | 7.0 | | 7.0 | 15.1 | | 15.1 |
| 5 | 15.1 | | 15.1 | 25.1 | | 25.1 |
| 10 | 25.1 | | 25.1 | 25.1 | | 25.1 |
| 8 | 7.0 | | 8.6 | 23.5 | | 25.1 |
| 9 | .0 | | 4.1 | 4.3 | | 8.4 |
| 2 | 4.3 | | 8.4 | 12.6 | | 16.7 |
| 3 | 12.6 | | 16.7 | 21.0 | | 25.1 |
| 1 | .0 | | 4.6 | 4.5 | | 9.1 |
| 11 | 4.5 | | 9.1 | 10.5 | | 15.1 |
| 4 | 4.5 | | 9.7 | 11.5 | | 16.7 |
| 7 | 4.3 | | 11.1 | 8.3 | | 15.1 |

'First''calendar date chosen as point of reference'
'Number completing job' 10
?

```
          11      'c0'
          1       'working hours'


          'calendar'

          0                20163
          1                70163
          2               140163
          3               210163
          4               280163
          5                40263
          6               110263
          7               180263
          8               250263
          9                10363
          10               40363

                   1      'c1'
                   3      '3 or 1'
                   3      'nc'
                   1      'c2'
                   9      'ncj'
                  11      'nj'

308     306     1   1   2       4   0   6          20163   70163
307     306     1   2   2       0   0   7          20163   70163
307     301     2   2   3       8   1   2          20163  140163
300     301     1   2   2       4   3   2          20163   70163
306     305     5   5   6       1   5   3          70163  110263
306     302     1   1   2       1   6   6          70163  140163
301     305     2   3   3       7   0   0         140163  280163
301     302     1   1   2       0   0   8          70163  280163
303     304     0   0   0       0   0   0         180263  180263
305     303     1   2   2       8   0   0         110263  180263
302     303     3   3   4       5   1   1         280163  280163
?
'TEST TAPE II'
```

'PERT program R856 , 090163 , LIST OF RELEVANT JOBS

| job nr | begin node | end node | time | capacities 1 | 2 | 3 | 4 | 5 | t(i) start | p(i) data | T(i) | t(j) finish | p(j) data | T(j) | e(ij) slack' | A(ij) | sigma | sigmasqu |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 307 | 306 | 1.8 | 0 | 0 | 7 | | | 20163 | 20163 | 20163 | 140163 | 70163 | 140163 | .0 | .0 | .17 | .03 |
| 5 | 306 | 305 | 5.2 | 1 | 5 | 3 | | | 140163 | 70163 | 140163 | 180263 | 110263 | 180263 | .0 | .0 | .17 | .03 |
| 10 | 305 | 303 | 1.8 | 8 | 0 | 0 | | | 180263 | 110263 | 180263 | 10363 | 180263 | 10363 | .0 | .0 | .17 | .03 |
| 9 | 303 | 304 | .0 | 0 | 0 | 0 | | | 10363 | 180263 | 10363 | 10363 | 180263 | 10363 | .0 | .0 | .00 | .00 |
| 1 | 308 | 306 | 1.2 | 4 | 0 | 6 | | | 20163 | 20163 | 70163 | 140163 | 70163 | 140163 | .6 | .6 | .17 | .03 |
| 3 | 307 | 301 | 2.2 | 8 | 1 | 2 | | | 20163 | 20163 | 140163 | 210163 | 140163 | 40263 | .0 | 2.0 | .17 | .03 |
| 7 | 301 | 305 | 2.8 | 7 | 0 | 0 | | | 210163 | 140163 | 40263 | 40263 | 280163 | 180263 | 2.0 | 2.0 | .17 | .03 |
| 8 | 301 | 302 | 1.2 | 0 | 0 | 8 | | | 210163 | 70163 | 40263 | 280163 | 280163 | 110263 | .0 | 2.2 | .17 | .03 |
| 11 | 302 | 303 | 3.2 | 5 | 1 | 1 | | | 280163 | 280163 | 110263 | 180263 | 280163 | 10363 | 2.2 | 2.2 | .17 | .03 |
| 4 | 300 | 301 | 1.8 | 4 | 3 | 2 | | | 20163 | 20163 | 210163 | 140163 | 70163 | 40263 | .4 | 2.4 | .17 | .03 |
| 6 | 306 | 302 | 1.2 | 1 | 6 | 6 | | | 140163 | 70163 | 40263 | 210163 | 140163 | 110263 | .4 | 2.6 | .17 | .03 |

'

| job nr | t(i) start | p(i) times | T(i) | t(j) finish | p(j) times | T(j) |
|---|---|---|---|---|---|---|
| 2 | .0 | | .0 | 1.8 | | 1.8 |
| 5 | 1.8 | | 1.8 | 7.0 | | 7.0 |
| 10 | 7.0 | | 7.0 | 8.8 | | 8.8 |
| 9 | 8.8 | | 8.8 | 8.8 | | 8.8 |
| 1 | .0 | | .6 | 1.2 | | 1.8 |
| 3 | .0 | | 2.0 | 2.2 | | 4.2 |
| 7 | 2.2 | | 4.2 | 5.0 | | 7.0 |
| 8 | 2.2 | | 4.4 | 3.4 | | 5.6 |
| 11 | 3.4 | | 5.6 | 6.6 | | 8.8 |
| 4 | .0 | | 2.4 | 1.8 | | 4.2 |
| 6 | 1.8 | | 4.4 | 3.0 | | 5.6 |

'First''calendar date chosen as point of reference'
'Number completing job' 9

| 'Early start times | Capacities' | | | | 'Job numbers' | | | |
|---|---|---|---|---|---|---|---|---|
| 20163 | .0 | 16 | 4 | 17 | 1 | 2 | 3 | 4 |
| 140163 | 1.8 | 10 | 12 | 11 | 3 | 5 | 6 | |
| 210163 | 2.2 | 9 | 11 | 17 | 5 | 6 | 7 | 8 |
| 280163 | 3.4 | 13 | 6 | 4 | 5 | 7 | 11 | |
| 180263 | 7.0 | 8 | 0 | 0 | 10 | | | |
| 10363 | 8.8 | 0 | 0 | 0 | | | | |

| 'Late start times | Capacities' | | | | 'Job numbers' | | | |
|---|---|---|---|---|---|---|---|---|
| 20163 | .0 | 0 | 0 | 7 | 2 | | | |
| 70163 | .6 | 4 | 0 | 13 | 1 | 2 | | |
| 140163 | 1.8 | 1 | 5 | 3 | 5 | | | |
| 140163 | 2.0 | 9 | 6 | 5 | 3 | 5 | | |
| 210163 | 2.4 | 13 | 9 | 7 | 3 | 4 | 5 | |
| 40263 | 4.2 | 8 | 5 | 3 | 5 | 7 | | |
| 40263 | 4.4 | 9 | 11 | 17 | 5 | 6 | 7 | 8 |
| 110263 | 5.6 | 13 | 6 | 4 | 5 | 7 | 11 | |
| 180263 | 7.0 | 13 | 1 | 1 | 10 | 11 | | |
| 10363 | 8.8 | 0 | 0 | 0 | | | | |

?

```
          11      'c0'
          1       'working hours'

          'calendar'

          0              20163
          1              70163
          2             140163
          3             210163
          4             280163
          5              40263
          6             110263
          7             180263
          8             250263
          9              10363
          10             40363

                  2       'c1'
                  3       '3 of 1'
                  3       'nc'
                  1       'c2'
                 11       'ncj'
                 11       'nj'

308     306     1   1   2       4   0   6         20163   70163
307     306     1   2   2       0   0   7         20163   70163
307     301     2   2   3       8   1   2         20163  140163
300     301     1   2   2       4   3   2         20163   70163
306     305     5   5   6       1   5   3         70163  110263
306     302     1   1   2       1   6   6         70163  140163
301     305     2   3   3       7   0   0        140163  280163
301     302     1   1   2       0   0   8         70163  280163
303     304     0   0   0       0   0   0        180263  180263
305     303     1   2   2       8   0   0        110263  180263
302     303     3   3   4       5   1   1        280163  280163
?
'TEST TAPE  III'
```

'PERT program R856 , 090163 , LIST OF RELEVANT JOBS

| job nr | begin node | end node | time | capacities 1 | 2 | 3 | 4 | 5 | t(i) start | p(i) data | T(i) | t(j) finish | p(j) date | T(j) | a(ij) slack' | A(ij) | sigma | sigmasqu |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 307 | 301 | 2.2 | 8 | 1 | 2 | | | 280163 | 20163 | 280163 | 110263 | 140163 | 110263 | .0 | .0 | .17 | .03 |
| 8 | 301 | 302 | 1.2 | 0 | 0 | 8 | | | 110263 | 70163 | 110263 | 180263 | 280163 | 180263 | .0 | .0 | .17 | .03 |
| 11 | 302 | 303 | 3.2 | 5 | 1 | 1 | | | 180263 | 280163 | 180263 | 40363 | 280163 | 40363 | .0 | .0 | .17 | .03 |
| 2 | 307 | 306 | 1.8 | 0 | 0 | 7 | | | 280163 | 20163 | 280163 | 110263 | 70163 | 110263 | .0 | .4 | .17 | .03 |
| 4 | 300 | 301 | 1.8 | 4 | 3 | 2 | | | 280163 | 20163 | 280163 | 110263 | 70163 | 110263 | .4 | .4 | .17 | .03 |
| 6 | 306 | 302 | 1.2 | 1 | 6 | 6 | | | 110263 | 70163 | 110263 | 180263 | 140163 | 180263 | .4 | .4 | .17 | .03 |
| 1 | 308 | 306 | 1.2 | 4 | 0 | 6 | | | 280163 | 20163 | 40263 | 40263 | 70163 | 110263 | .6 | 1.0 | .17 | .03 |

'

| job nr | t(i) start | p(i) times | T(i) | t(j) finish | p(j) times | T(j) |
|---|---|---|---|---|---|---|
| 3 | .0 | | .0 | 2.2 | | 2.2 |
| 8 | 2.2 | | 2.2 | 3.4 | | 3.4 |
| 11 | 3.4 | | 3.4 | 6.6 | | 6.6 |
| 2 | .0 | | .4 | 1.8 | | 2.2 |
| 4 | .0 | | .4 | 1.8 | | 2.2 |
| 6 | 1.8 | | 2.2 | 3.0 | | 3.4 |
| 1 | .0 | | 1.0 | 1.2 | | 2.2 |

'Last''calendar date chosen as point of reference'
'Number completing job' 11

| 'Early start times | Capacities' | | | 'Job numbers' | | | |
|---|---|---|---|---|---|---|---|
| 280163 .0 | 16 | 4 | 17 | 1 | 2 | 3 | 4 |
| 110263 1.8 | 9 | 7 | 8 | 3 | 6 | | |
| 110263 2.2 | 1 | 6 | 14 | 6 | 8 | | |
| 180263 3.4 | 5 | 1 | 1 | 11 | | | |

| 'Late start times | Capacities' | | | 'Job numbers' | | | |
|---|---|---|---|---|---|---|---|
| 280163 .0 | 8 | 1 | 2 | 3 | | | |
| 280163 .4 | 12 | 4 | 11 | 2 | 3 | 4 | |
| 40263 1.0 | 16 | 4 | 17 | 1 | 2 | 3 | 4 |
| 110263 2.2 | 1 | 6 | 14 | 6 | 8 | | |
| 180263 3.4 | 5 | 1 | 1 | 11 | | | |

?