Eidg. Technische Hochschule
Institut für angewandte Mathematik
Zürich

Zürich, February 12, 1960

Prof. Dr. H. Rutishauser

A. P. Naur
Regnecentralen
Valby, Copenhagen

Remarks to the final draft and other documents

A. P. Naur's letter of Febr. 4

1) I agree to replace the name list by the complementary value list. In addition to P. Naur's reasons I might say that replacement by name is the ordinary way of call whereas call by value is the more involved operation and therefore this should be mentioned.

2) Reasons for 4.2.4. of the draft report:
begin
Mathematically the 4 basic arithmetic operations are defined for real numbers as well as for integers, except that the integer division is undefined if there is a reminder $\neq$ o (Undefinedness if divisor = o is understood). Therefore we ought to have the following definitions:

op(real,real) = real
op(real,integer) = real
op(integer,integer) = integer; with the only exception that
integer/integer is defined if and only if there is no reminder.

For several reasons the arithmetic operations cannot be defined that way, in addition it is obviously more practical to define

integer/integer = real, as we did.

It should be kept in mind however, that this deviation from classical mathematical definitions should be corrected at the earliest convenience, an ideal opportunity being the assignment of the result or the evaluation of the addressfunction in case the expression occurs in a subscript position. This automatically leads to the rule, that

integer := E

is defined if and only if the realvalued expression E on the right hand side has an integral value. Thus the accordance with century-old mathematical usage is reestablished.

If for practical reasons (roundoff errors, difficulty to check integrality of E) this rule cannot be enforced in ALGOL, then we can only relax the rule, i.e. the final rule must define a meaning to the above assignment at least if E is integral-valued. This automatically leads to 4.2.4.

Of course we have the operation ÷ , but there are also other operations which lead to integral values of type real and for these cases ÷ is no help. In addition I feel obliged to point out that for the man who frequently computes in the domain of integers, like number theorists, the operation ÷ is insufficient insofar, as there an operation div(a,b) yields:(c,d) is needed, where c and d are quotient and reminder. Thus on the whole, I feel that the operation ÷ which has been introduced very hastily, is not a very brilliant achievment and probably should be reconsidered and turned into a really fine instrument. I do not make here any proposal in this direction, but it should be kept in mind that this is a weak point of ALGOL.

Besides my reasons given above, I can say that I have quite a long experience in programming and teaching programming, which allows me to extrapolate that undefinedness of

integer := real

will be a terrible trap for programmers, simply because "semantic difference" of real and integer is so little. If a real number must be integer for mathematical reasons, then I simply refuse to blame the programmer who forgets the entier function, in addition the entier function is absolutely distasting in such cases. Let me give some examples which would lead to undefined situation without 4.2.4. In these examples n and m always are integer-declared, x a real-declared variable, whereas j may be either way.

l1)
```
 :
for j := abs(n-m) step 1 until m+n do  x[j] := sqrt(j) ;
 :
```
l2)
```
 :
for x := 1 step 1 until n do
begin
 :
for j := x step 1 until n do  xx[j] := 0
end x ;
```
l3) x[(-1)↑n] where n=negative.

l4) a[n×(n-1)/2, 0])
l5) x[sin(n×pi/2), Q[3,u,4]]) from P. Naur's report.

It is true, that the same measure which helps to make such examples legal, also saves the fool who writes n := sqrt(n) or a[7.8] from getting rubbish, but are we really obliged to give the fool the rubbish which he calls for?

There are a few further reasons connected with the compiler-construction, but generally rejecting such reasons I cannot honestly use them here in my favour.

end 2) ;

3) I was trying to write some examples already in the train from Paris to Zurich in order to mail them as soon as possible. But then I found contradictions and had to give it up.

4) I agree with P. Naur's wish concerning conditional expressions.

B. Remarks to the report.

ad 3.4.1. It was agreed in Paris (though probably not unanimously) to require brackets around relations except if a relation stands isolated, e.g. if x=o then. In fact the examples in 3.4.2. look so horrible that cosmetics must be applied. The necessary changes in 3.4.1. are:

Remove ⟨relation⟩| from the list following ⟨Boolean primary⟩::= and insert it in the list following ⟨Boolean expression⟩::=

A further change is needed in 3.4.6.1. :

....of precedence hold:

first ¬
second ∧
third ∨
forth >
fifth ≡

For relational operators no precedence is defined, i.e. a Boolean operator can operate on a relation if and only if the latter is enclosed by brackets.

Changes in 3.4.2: Brackets must be inserted at appropriate places.

ad 4.1.1. I weakly remember an agreement allowing multiple labelling (and M. Woodger confirms this). However, the issue seems not important enough to make P. Naur the trouble of rewriting. I agree with the present form of 4.1.-

ad 4.3.5. Agreement with P. Naur: such a go to should be undefined.

ad 4.5. The present form does not allow labelling of conditional statements. I cannot believe that this was intended. The omission can be cured by the following change of 4.5.1.:

⟨ if statement⟩::=⟨if clause⟩⟨unconditional statement⟩|

⟨label⟩:⟨if clause⟩⟨unconditional statement⟩

ad 4.5.3.2. I do not fully understand the whole section, especially not the remark in brackets. To me the statement

if B1 then S1 else if B2 then S2 else if B3 then S3

(where the B are Boolean expressions, the S statements) is perfectly allowed, yet if B2 then S2 is a statement exactly between two else's. I would like a rewording like follows:

4.5.3.2. Effect of else.
The worddelimiter else is a connective which serves to couple two statements (the first of which must be an if statement), such that the if clause causes the execution of exactly one of these two statements(except for possible go to's from one of these statements into the other). else has no independent meaning without the preceding if statement. Semantics (B is a Boolean

expression, Su an unconditional, S an arbitrary statement):

If B then Su else S is executed as

aux := true ; if B then begin Su ; aux := false end;

if aux then S

Provided this is exactly what 4.5.3.2. means, the change needs not to be made at any cost, provided the sentence "Thus in particular..." is deleted.

ad 4.5.4. Delete this section because it is not true. Example:

if B1 then S1 else if B2 then go to L else if B3 then S3 is not equivalent to

if B1 then S1 else if B2 then go to L ; if B3 then S3 or did I misunderstand?

ad 4.6.5. I had the idea that the controlled variable of a for statement is automatically local to the for statement (which would automatically include all rules) but I agree also to the present form of 4.6.5., provided there are no other opponents.

ad 4.7.3.1. The wording seems to me slightly incorrect (but really only slightly). Please look at my remarks under F).

ad 5.4.5. This is obviously part a) of my proposal in document 1001 and of course I absolutely agree with P. Naur, since to my feeling, and experience, optional devices have only little value. In fact if I receive a procedure declaration then

either specifications are given

or I mail it back for completion.

Thus in the final end we have to give the specifications anyhow, the only difference being it requires more ink(and trouble) if we make the feature only optional. In addition we may suddenly realize the usefulness of the specifications for the compiler-construction, though they are not needed in principle.

Further missing brackets around relations:

p. 16, 157 mm below page number V := (Q>Y) ∧ z

p. 18, 26 mm " if (s < o) ∨ (P ≤ Q) then .....

p. 25, 200 mm " if (o ≤ u) ∧ (u ≤ 1) then .....

Everybody will admit that the construction s < 0 ∨ P ≤ Q is impossible by any standard. On the other hand I do not object (though recommend the use of brackets) to the fifth example in 3.4.2., since for an expression containing only Boolean variables in the proper sense there are at least long established precedence rules.

C. My own proposals (document 1001) are obsolete since part a) is taken care of by the proposal of P. Naur (see 5.4.5. of the draft report) whereas part b) is not feasible. real understood

D. Concerning the letters of A. Perlis (Jan 20) and P. Naur (Jan17) I still do not see clearly how the trouble with A (f) is resolved, if procedure A is global to the program P in which the call occurs. I would like to hear explanations how the compiler does the translation of P without having the declaration for A and therefore without any information whatsoever whether f is to be called by value or name. I still believe that it is important to translate a procedure or programm without at the same time translating all its subprocedures. (a remark in M. Woodger's letter seems to indicate how it may be done, but I do not see this in the draft report).

E. As far as I can see, the remarks of M. Woodger have now all been taken care of, as far as I can judge (I have no document 30). The syntax contains some errors for instance in Nr. 1,2,46, but these do not show up in P. Naur's report except the one mentioned in "ad 4.5.".

F. Unfortunately I cannot accept A.V. Wijngaardens proposal in his letter of Febr. 4. Parameters are no doubt global to the procedure body S and only in case of call by value they are local to a hypothetical block H (in the sense of "executed as if") which is defined only for defining the semantics of the procedure call. H contains the assignments before and after the execution of S(as well as declarations for auxiliary variables) which are needed to shield the actual parameters against changes inside S. Therefore these auxiliary variables (which in effect are the formal parameters) are local to H but still global to S which is a part of H. For this reason ....
For the execution, H is inserted in place of the procedure statement.

Of course I shall leave it to you to compile the final report. It would be nice, however, if ~~big~~ publications of that report could be obtained in the same form as the draft report. I could then send you the appropriate number of copies of the example Runge Kutta which I could attach to the big publication.

Sincerely yours,

i.V. H. R. Schwarz