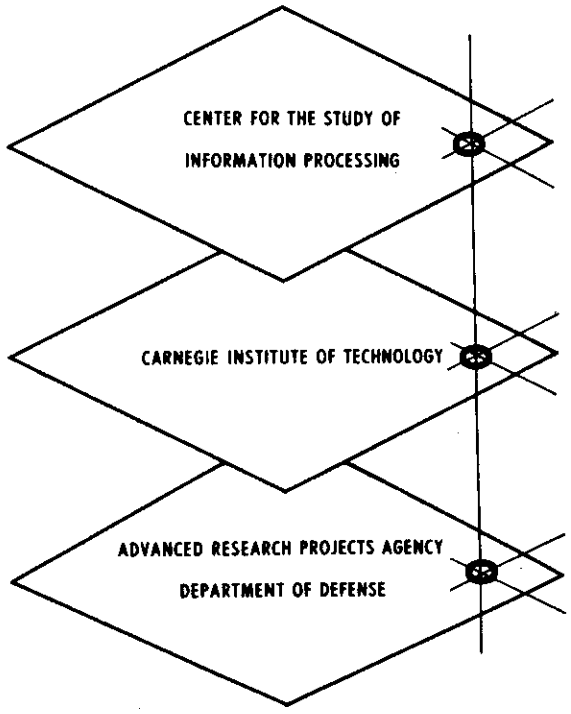
ALGOL-20

A LANGUAGE MANUAL

JANET W. FIERST, EDITOR
DAVID M. BLOCHER
ROBERT T. BRADEN
ARTHUR EVANS JR.
RICHARD B. GROVE



ALGOL-20

A LANGUAGE MANUAL

JANET W. FIERST, EDITOR
DAVID M. BLOCHER
ROBERT T. BRADEN
ARTHUR EVANS JR.
RICHARD B. GROVE

FIRST PRINTING FEBRUARY 1965

THIS WORK WAS SUPPORTED BY THE
ADVANCED RESEARCH PROJECTS AGENCY OF THE
OFFICE OF THE SECRETARY OF DEFENSE:
CONTRACT SD-146

CARNEGIE INSTITUTE OF TECHNOLOGY

Acknowledgements

The construction of the programming system described here is the result of the combined effort of many people. The following were involved with coding the translator: David M. Blocher, Arthur Evans, Jr., Janet W. Fierst, Richard B. Grove and Carol H. Thompson. Ronald R. Bushyager, III, wrote WHAT, the assembly language processor included in the translator. Charles L. Thornton wrote a table loader (the "Meta-compiler") which is an essential part of the process of assembling the translator. Grove wrote the relocator and the librarian. Special thanks are owed to Robert T. Braden for acting as the conscience of the group with many useful suggestions on "ALGOL esthetics". The entire task was directed by Evans.

This document has been edited by Fierst, who also did much of the writing. The following people, in addition, contributed to the writing and editing of the document: Blocher, Braden, Evans and Grove. Ronald P. Hackleman wrote many of the relocatable library routine descriptions appearing in Chapter 5. The typing has been done by Edythe Simmons, and Robert D. Smith contributed materially with his editorial assistance.

PREFACE

ALGOL-20 is a realization of the international language ALGOL-60. The international language, although a valuable vehicle for the description of algorithms, does not really become useful until it is implemented on computers. However, each implementer has found it necessary in some cases and desirable in others to make changes in the language. Further, additions to the language such as input/output are necessary. This ALGOL-20 Manual is a description of the realization of ALGOL as implemented at Carnegie Institute of Technology.

Two additional documents are needed to complete the description of Carnegie Tech ALGOL. One is a description of ALIBN - the librarian used in connection with the two libraries. The other document describes the assembly language - WHAT - which is built into the Algol translator. The user may include assembly code as part of his program, as described in the WHAT manual. These manuals are currently in preparation.

The internal operation of the translator has not been adequately described. However, An ALGOL 60 Compiler, by A. Evans, which was printed in Annual Review of Automatic Programming, Volume 4, Pergamon Press, describes part of the translator. A preliminary version of the format language used was described in A Format Language, by Alan J. Perlis, in Comm. ACM, 7(Feb. 1964), pp. 89-96.

Arthur Evans, Jr.
January, 1965

Introduction

The manual is organized as follows: Chapter 0 is a ready reference containing in summary form the information the experienced programmer will need. It is not suitable for reading by itself, but is useful for reference to particular points. Chapter 1 is an introduction which includes bibliographical citations to several introductory texts on ALGOL, for the programmer who does not yet know the language. Chapter 2 describes in considerable detail how the local system differs from the international language. Chapter 3 contains a detailed description of the input/output system provided at Carnegie Tech. A format language of some sophistication is defined. Chapter 4 contains a description of system statements - those statements used to communicate to the translator information which is not part of the ALGOL language. Chapter 5 contains a description of the two libraries available to the translator and contains descriptions of the routines currently in the libraries. Chapter 6 is a collection of miscellaneous topics, including keypunch conventions, error codes, etc. Chapter 7 includes the ALGOL-60 report as revised in 1962 and a summarized list of differences between ALGOL-60 and local ALGOL.

Page numbers are of the form AL.m.n, where m indicates the chapter number and n is the page in the chapter. Two chapters -- three and six -- are divided into sub-chapters distinguished by lower case letters, e.g., Chapter 3b. The sub-chapters are also paged individually so that the first page of Chapter 3b is AL.3b.1, immediately following AL.3a.2.

CONTENTS

Chapter 0 - ALGOL READY REFERENCE

ALGOL READY REFERENCE

ALGOL Notes and Error Messages (Chapter 6b)

Compile Errors

Phase I Errors (Each of these errors terminates Phase II.)

0: The program does not start with begin.
1: A statement starts with an illegal character or an illegal reserved word.
2: A statement starts with an identifier followed by an illegal character.
3: In an expression an operand was expected and was not found.
4: In an expression a binary operator was expected and was not found.
(Possibly caused by a semicolon missing after the preceding statement.)
5: A "]" does not have a matching "[".
6: An array element has been used illegally.
7: A ":" has appeared incorrectly.
8: A "←" or ":=" has appeared incorrectly.
9: A ")" does not have a matching "(".
10: A "," has appeared incorrectly.
11: then has appeared without if.
12: else has appeared without then.
13: Characters are still in the stack after a ";" or an end.
14: A procedure statement is followed by other than end, else, or ";".
15: for is not followed by an identifier.
16: The for variable is not followed by a "←" or ":=".
17: step has appeared without for.
18: until has appeared without step.
19: while has appeared without for.
20: do has appeared without for.
21: go to is not followed by an identifier or "(" or if.
22: go to if...then...is not followed by else.
23:
24: An obscure error in a go to statement.
25: An impossible error after begin. ("⊢" is not the second element in the stack. See Error 98.)
26: own is followed by something other than <type>.
27: An array declaration does not specify subscript bounds.
28: The identifier list of a declaration is not followed by a ";".
29: switch is not followed by an identifier.
30: The identifier of a switch declaration is not followed by a "←" or ":=".
31:
32: procedure is not followed by an identifier.
33: A procedure identifier is not followed by "(" or ";".
34: A formal parameter list is not followed by ")".
35: The ")" following a parameter list is not followed by a ";".
36: The identifier list in a specification is not followed by a ";".
37: An identifier did not follow the "," in an identifier list.
38: The illegal construction "then if" has occurred.
39: A switch with more than one subscript position has been used.
40: The value part of a procedure declaration was not followed by a ";".

ALGOL READY REFERENCE

41: The name of a permanent subroutine (such as "SIN") is not followed by "(".
42: There is an extra "," or else a missing ":" in an array declaration.
43: More begin's than end's have occurred when the end-of-file is reached.
44: Impossible - see Error 98.
45: max or min is not followed by "(".
46: In an array declaration the identifier list is not followed by "[".
47: Array specifier has subscript bounds, which it should not.
48: library is not followed by <type> or procedure.
49:

Phase I Errors (format and name statements) (Each of these errors terminates Phase II.)

50: A reserved input/output word is not followed by "(" .
51: A format list element starts with an illegal character. (Should be "<" or "→" or "$" or identifier).
52: "→" is missing: i.e., a replicator was expected but not found.
53: for is missing after "$".
54: "→" is not followed by "$" or an identifier.
55: ")" or ">" is not followed by ")" or ",".
56: A name statement or format statement is not followed by end, else or ";".
57: A replicator is not followed by "(" or "<".
58: "<" or "," is followed by an illegal character.
59: An integer is followed by an illegal character.
60: A format instruction is not followed by ">" or ",".
61: An illegal prefix to a numeric primary has been used.
62: An illegal numeric primary has been used.
63: "." appears in a numeric primary in a read statement.
64: In a numeric primary, E, F or S is not followed by an integer.
65:
66:
67:
68:
69:

Phase II Errors (Only those errors marked "*" turn off Phase II.)

*70: A reserved word which is not yet available has been used.
71: A label has been used but not defined. (The name of the label is printed prior to this error message)
72: An identifier has been used but not declared.
73: An identifier has been declared twice in this block.
74: An identifier in the value list is not a parameter.
75: An identifier which has been used as a procedure has not been declared to be one.
76: A subscripted identifier has not been declared to be an array or switch.
77: The program is too long.
78: A procedure identifier which is not a function designator has been used in an expression.

* Turns off Phase II.

79: An identifier which has been used as a switch has not been declared to be one.
80: An array identifier has been used without subscripts.
81: Too many index variables have been declared.
82: A label or array or switch has been called by value.
83: An identifier in a specification list is not a parameter.
84: In a procedure declaration a parameter is not specified.
85: In a procedure declaration a parameter is specified twice.
86: A procedure, switch or label appears on the left of a ":=" or "←".
*87: The W2 stack is too full.
88: More than 100 relocatable library procedures have been declared.
89: A constant has been used in place of an identifier, e.g., 33[k].
*90: A subscripted for variable has been used (this is not yet available in ALGOL-20).
*91: The next-command pointer is less than the base of the program.
92:
93:
94:
95:

Miscellaneous Errors

96:
97: A possible translator error - bring listing to Janet Fierst at the Computation Center.
98: Impossible: bring your listing to A. Evans at the Computation Center.
99: Same as 98.

Subscan Errors

100: A card column contains an illegal combination of punches.
101: Too many abcons or adcons have been used (numerical constants and alphanumeric string constants).
102: Too many decimal points appear in a number.
103: Too many "$_{10}$"s appear in a number.
104: An error has appeared in a parameter delimiter comment: ")<any string not containing:>:(".
105: An illegal bar ("|") variable has been used.
106: A constant has been used which is too large to fit into a real variable.
107: A "$_{10}$" is followed by something other than "+", "-", or <digit>.
108: A string goes over the end of a card.
*109: The symbol table has been exceeded.
110:
111:
112:
113:
114:

* Turns off Phase II.

ALGOL READY REFERENCE

System Statement Errors

115: An abcon system statement has occurred after code has been compiled.
116:
117: An abcon system statement has requested more space than there is in user memory.
118:
119: An illegal SY card has occurred. (This may be caused by a LIBRARY card after the symbolic library has been released.)
120: The library procedure nesting exceeds 5.
121:
122: WHAT has been called after it has been released.
123: An illegal segment statement has been used.
124: An SY LIBRARY card has asked for a routine not in the symbolic library.
125: A library procedure declaration has named a routine not in the relocatable library.

Notes

Note 1: end comment convention was used on preceding card. That is, everything was ignored up to ";", end, or else.
Note 2: A function designator has been used as a procedure statement.
Note 3: In an arithmetic or boolean expression, the construction if...then if has occurred. This is syntactically illegal but unambiguous, and is therefore accepted by the translator.
Note 4: An arithmetic (boolean) (designational) expression has been used where a simple arithmetic (boolean) (designational) expression should have been used.
Note 5: In a designational expression, the construction if...then if has occurred. This is syntactically illegal but unambiguous.
Note 6: Phase II has been turned off.
Note 7: The construction if...then for...do...else... which is legal in ALGOL 60 but illegal in ALGOL 62 has been used.
Note 8: TAB appears as a character.
Note 9: Fifty errors have been found on a single card; compilation has been terminated.

Run Errors

ADRP	address--opcode fault
CFLG	command flag error
EXP	EXP (x) called with $X > 160.116998$
EXPO	exponent overflow
LN	LN (x) called with $X \leq 0$
RAD1	upper < lower in a bound pair in an array declaration
RAD2	declared arrays exceed available space
READ	an error has occurred in reading a data card

ALGOL READY REFERENCE

SIN	the argument to SIN or COS is greater than 8↑21.
SQRT	SQRT (X) called with X < 0
TIMR	time limit exceeded
X↑A1	X = 0 and A ≤ 0
X↑A2	A * LN (X) > 160.116998
X↑A3	X ≤ 0 and A not integer valued

Keypunching (Chapter 6a)

```
                          111 1 1 111 11 2 222 2222223333333333...R
column  -   12 3 45678 9012 3 4 567 89 0 123 4567890123456789    M
WHAT        WH   .LOC.      F   OP.    M     Addr,Index;comment..
ALGOL       AL   ..................ALGOL text.................
system      SY   ..................system text................
comment     CO   ..................comment...................
```

RM is the right margin. (Initially RM = 72. It may be changed by SY RIGHT MARGIN - see Chapter 4.)

Precedence rules for operators and relations (Chapter 2)

↓ (performed first)

MOD

↑

+ - (unary)

* /

+ - (binary)

< > ¬ < ¬ > = ≠

¬

∧

∨ (performed last)

ALGOL READY REFERENCE

Format Instructions (Chapter 3)

Control Instructions

nC	Sets CP to column n (CP ← n).
nR	Moves CP n columns to the right (CP ← CP + n).
nL	Moves CP n columns to the left (CP ← CP - n).

Instructions for PRINT and PUNCH

Control and Alphanumerics

nE	Prints or punches the contents of the appropriate buffer, clears the buffer to blanks, prints or punches n-1 blank lines, and sets CP to left margin.
nW	Prints or punches the contents of the appropriate buffer n times. CP and the buffer are not changed.
P	Upspaces the paper to a new page and prints a page header on the first line. CP and the buffer are not changed. This instruction is ignored in PUNCH.
'<string>'	Stores the characters of the string into the appropriate buffer.
nB	Stores n blanks.
nQ	Stores n quotes (').
nA	Stores n characters which are taken from ↓ ((n+3)/4) names.
nT	Stores min (5, n) characters which are taken from one of the internal strings 'TRUE ', 'FALSE' or 'UNDEF', according to the boolean value of the corresponding name.

Numeric Instructions

1. Prefix

L$	Stores a dollar sign left-justified.
$	Stores a dollar sign immediately before the first digit.
L+(L-)	Stores the sign (sign if minus) left-justified or immediately after a '$' stored by "L$".

+(-)	Stores the sign (sign if minus) immediately before the first digit or a '$' stored by "$". If the number is negative, a minus sign is stored; if the number is non-negative, a plus sign (blank) is stored.

2. Numeric Primary

nD	Stores n digits or blanks. Leading (or trailing) zeros are replaced by blanks.
nZ	Stores n digits.
	Stores a decimal point, '.'

3. Suffix

L	Shifts the number until the left-most digit is non-zero (if possible) and stores the resultant exponent.
E(F)±n	Shifts the number until its exponent equals ± n. The resultant exponent is (is not) printed.
S±n	Shifts the number until the left-most digit is non-zero (if possible). The decimal point is inserted in the position to give an exponent equal to ±n and the exponent is stored. The numeric primary must be of the form "mD." or "mZ.".
H	Converts and stores the number in octal (base 8) instead of decimal.
K	Invokes special spacing. If the prefix contains "L$" or "$", the digits of the number are stored in groups of three separated by commas. If neither "L$" nor "$" appears, the digits are stored in groups of five, separated by blanks.
N	Suppresses error printing which may occur when left-most non-zero digits overflow the field specified by the numeric primary.
T	Truncates the number at the last digit stored. Normally, numbers are rounded by adding five to the first digit not printed.

ALGOL READY REFERENCE

Instructions for READ

Control and Alphanumeric Instructions

nE	Reads n card images into the READ buffer and sets CP to left margin. Only the last card image read is available after the instruction is executed.
nW	Functions as "nE" except that the card images are also listed on the printer.
'<string>'	Causes the n characters of the string to be stored, four per word, into the next $\downarrow((n+3)/4)$ names. If n is not a multiple of four, characters in the last name are stored right-justified. The CP and buffer are not changed.
nA	Scans the characters in the next n positions of the buffer and stores them as in the '<string>' instruction.
nT	Scans the characters in the next n positions of the buffer. If the first non-blank character is the letter "T", the value true is stored in the next name; otherwise, the value of the name is set to false. The corresponding name must be of type boolean or logic.

Numeric Instructions

1. Numeric Primary

nD	Scans the number represented in the next n positions of the buffer. Blanks are ignored. Numbers are in free field format and may contain signs, decimal points, and exponents. Numbers preceded by a "/" are treated as octal quantities.
nZ	Scans as "nD" except that blanks are treated as zeros.

2. Suffix

H	Assumes the number read to be octal (base 8) instead of decimal.

E±n	Multiplies the number read by ten (eight) raised to the power ±n.
N	Causes illegal characters (such as letters) to be ignored.

3. Free Read

nF	Scans and concatenates n octal or decimal numbers in fields separated by commas. Blanks are ignored with the exception that if an entire field is blank, the corresponding name is unchanged. Numbers preceded by a "/" are treated as octal. A number field may be continued over the end of a card image. The last number in the data group must be followed by either a "," or a "*".

ALGOL READY REFERENCE

Library Routines and Standard Functions (Chapter 5)

Relocatable Routines (i.e., library procedures)

AND.CALL	sets the scratch pointer and enters AND
AND.FILE	assigns a logical file type to an AND file
DISC.READ	reads from disc or tape
DISC.WRITE	writes on disc or tape
GOOF.STAR	prints a run-error message
GO.SEG	slews to a segment
LINK	links to a segment
RUN.ERROR	sets up run-error recovery
SLEW	slews to a record
SYSTEM.DUMP	dumps an ALGOL program as a system

Symbolic Routines

AND.PUNCH	punches an AND record onto cards
BANSOLV	solves a system of linear equations whose coefficient matrix is a band matrix
COMDIV	computes the quotient of two complex numbers
CURVEFIT	determines the best least squares polynomial approximation to a given curve, with or without constraints
CURFIT	determines the best least squares polynomial approximation to a given curve, with constraints
ELIPS	computes the values of the complete elliptic integrals of the first and second kinds
FREQ	determines the frequency distribution of a given set of data
GAME	solves a finite, zero-sum, two-person game
GJR	computes the inverse of a given matrix
HERMJA	finds all the eigenvectors and eigenvalues of a given Hermitian matrix
JACOBI	finds all eigenvectors and eigenvalues of a given symmetric matrix
MULLER	finds the real and complex roots of a general equation of the form $f(z) = 0$

NEVILLE	computes approximate values of a tabulated function by interpolation
NORMRAN	computes a sequence of normally distributed pseudo-random numbers
PLOT	produces the graph of one to ten functions
RANDOM	computes a sequence of uniformly distributed pseudo-random numbers
SIM	performs numerical integration by Simpson's method
SORT1	sorts a list of numbers into ascending order

Standard Functions (i.e., built-in functions)

ABS	absolute value
ARCTAN	arctangent
BUFFERSET	redefines the input/output buffers
CLOCK	time since the last job-card minus parameter (in seconds)
COS	cosine
DEBUGPRINT	a fixed format print routine
ENTIER	the largest integer which is not greater than the parameter
EXP	exponential (e^x)
HALT	halt
LN	natural logarithm
MAX	maximum } (see page AL.2.10 in the ALGOL manual)
MIN	minimum } (see page AL.2.10 in the ALGOL manual)
PAGES	number of pages since the jobcard
PAUSE	saves the program for restart
PRINT	controls printing
SIGN	-1 if parameter negative, +1 if positive, 0 if zero
SIN	sine
SQRT	square root
TIME	time since midnight (in seconds)

MOD is an operator such that for integer m and n the quantity m MOD n is the remainder on dividing m by n.

ALGOL READY REFERENCE

Library Routines and Standard Functions (Chapter 5)

Relocatable Routines (i.e., library procedures)

AND.CALL	sets the scratch pointer and enters AND
AND.FILE	assigns a logical file type to an AND file
DISC.READ	reads from disc or tape
DISC.WRITE	writes on disc or tape
GOOF.STAR	prints a run-error message
GO.SEG	slews to a segment
LINK	links to a segment
RUN.ERROR	sets up run-error recovery
SLEW	slews to a record
SYSTEM.DUMP	dumps an ALGOL program as a system

Symbolic Routines

AND.PUNCH	punches an AND record onto cards
BANSOLV	solves a system of linear equations whose coefficient matrix is a band matrix
COMDIV	computes the quotient of two complex numbers
CURVEFIT	determines the best least squares polynomial approximation to a given curve, with or without constraints
CURFIT	determines the best least squares polynomial approximation to a given curve, with constraints
ELIPS	computes the values of the complete elliptic integrals of the first and second kinds
FREQ	determines the frequency distribution of a given set of data
GAME	solves a finite, zero-sum, two-person game
GJR	computes the inverse of a given matrix
HERMJA	finds all the eigenvectors and eigenvalues of a given Hermitian matrix
JACOBI	finds all eigenvectors and eigenvalues of a given symmetric matrix
MULLER	finds the real and complex roots of a general equation of the form $f(z) = 0$

NEVILLE	computes approximate values of a tabulated function by interpolation
NORMRAN	computes a sequence of normally distributed pseudo-random numbers
PLOT	produces the graph of one to ten functions
RANDOM	computes a sequence of uniformly distributed pseudo-random numbers
SIM	performs numerical integration by Simpson's method
SORT1	sorts a list of numbers into ascending order

Standard Functions (i.e., built-in functions)

ABS	absolute value
ARCTAN	arctangent
BUFFERSET	redefines the input/output buffers
CLOCK	time since the last job-card minus parameter (in seconds)
COS	cosine
DEBUGPRINT	a fixed format print routine
ENTIER	the largest integer which is not greater than the parameter
EXP	exponential (e^x)
HALT	halt
LN	natural logarithm
MAX	maximum } (see page AL.2.10 in the ALGOL manual)
MIN	minimum } (see page AL.2.10 in the ALGOL manual)
PAGES	number of pages since the jobcard
PAUSE	saves the program for restart
PRINT	controls printing
SIGN	-1 if parameter negative, +1 if positive, 0 if zero
SIN	sine
SQRT	square root
TIME	time since midnight (in seconds)

MOD is an operator such that for integer m and n the quantity m MOD n is the remainder on dividing m by n.

ALGOL READY REFERENCE

Reserved Identifiers (Chapter 2)

ABS	(2)	GO TO	(1)	PRINT	(3-3.1ff)
ARCTAN	(2)	HALF	(3-2.5)	PROCEDURE	(1)
ARRAY	(1)	IF	(1)	PUNCH	(3-3.1ff)
BEGIN	(1)	INDEX	(3-2.5)	READ	(3-3.1ff)
BOOLEAN	(1)	INPUT	(3-NA)	REAL	(1)
COMMENT	(1)	INTEGER	(1)	SIGN	(2)
COS	(2)	LABEL	(1, 3-2.10)	SIN	(2)
DO	(1)	LIBRARY	(3-5.1ff)	SQRT	(2)
ELSE	(1)	LN	(2)	STEP	(1)
END	(1)	LOGIC	(3-2.5)	STRING	(1)
ENTIER	(2)	MAX	(3-2.9)	SWITCH	(1)
EXP	(2)	MIN	(3-2.9)	THEN	(1)
FALSE	(1)	MOD	(3-2.9)	TRUE	(1)
FOR	(1)	MONITOR	(3-NA)	UNTIL	(1)
FORWARD	(3-NA)	NAME	(3-3.1ff)	VALUE	(1)
GO	(1)	OUTPUT	(3-NA)	WHILE	(1)
GOTO	(1)	OWN	(1)		

(1) ALGOL 60 "built in" word.

(2) ALGOL 60 reserved function identifier.

(3) ALGOL 20 reserved word--see page reference. (NA means not now available.)

Privileged Identifiers (Chapter 6d)

ACC	real	Accumulator
CLOCK	integer procedure	(time in seconds since job-card) minus parameter
DAY	logic	'␣dd␣' dd = day of month
DEBUGPRINT	procedure	fixed format print routine
EPSILON	real	smallest positive number = 8↑-63
HALT	procedure	halt
INFINITY	real	largest positive number = (8↑14-1)*8↑63
MONTH	logic	'mmm␣' mmm = name of month
PAGES	integer procedure	number of pages since job-card
PAUSE	procedure	save for restart
PRINT	procedure	controls printing on teletype
TIME	integer procedure	time in seconds since midnight
YEAR	logic	'yy␣␣' yy = last two digits of year

¬ labels in WHAT and ALIBN

¬ 0	Complement 0 when accessed arithmetically
¬ 1	1 flag
¬ 2	2 flag
¬ 3	3 flag
¬ 4	Function variable for relocatable library procedures
¬ 5	INFINITY = (8↑14-1) * 8↑13 (Chapter 6d)
¬ 6	EPSILON = 8↑(-63) (Chapter 6d)
¬ 7	Dynamic block level - pointer to array stack (an index register)
¬ 8	
¬ 9	
¬ 10	Exit from FORMAT and NAME
¬ 11	Current NAME list
¬ 12	Current FORMAT list
¬ 13	NAME routine
¬ 14	PRINT routine
¬ 15	PUNCH routine
¬ 16	READ routine
¬ 17	FORMAT routine
¬ 18	Page-and-print-page-header routine
¬ 19	
¬ 20	go to <label> routine
¬ 21	RAD - run-time array declaration routine
¬ 22	begin administration routine
¬ 23	end administration routine
¬ 24	procedure begin administration routine
¬ 25	procedure end administration routine
¬ 26	GOOF* : Run Error routine
¬ 27	ADDR-OP routine
¬ 28	LINK routine
¬ 29	
¬ 30	Last location of user memory + 1
¬ 31	Contains base of compiled code
¬ 32	Contains end of relocated subroutines
¬ 33	Contains maximum location used by scalars
¬ 34	Base of array stack
¬ 35	Contains compile-time block level of current procedure
¬ 36	Contains run-time block level of current procedure
¬ 37	Base of run-time procedure nesting stack
¬ 38	Run-error recovery cell
¬ 39	Contains segment number
¬ 40	Contains physical right margin for READ
¬ 40+1	Contains physical right margin for PRINT
¬ 40+2	Contains physical right margin for PUNCH
¬ 41	RUN.ERROR switch for end
¬ 42	Run-time error printing mode switch (Chapter 5.RUNERROR)
¬ 43	
¬ 44	
¬ 45	

| variables in ALGOL

200	READ character pointer for standard buffer
201	READ right margin for standard buffer
202	READ left margin for standard buffer
203	
204	
205	PRINT and PUNCH character pointer for standard buffer
206	PRINT and PUNCH right margin for standard buffer
207	PRINT and PUNCH left margin for standard buffer
208	
209	
210	Format switch
211	NAME switch
212	Page counter
213	Page header switch
214	Up-space counter
215	Left-justify switch
250	Current READ buffer
251	Current PRINT buffer
252	Current PUNCH buffer

System Statements (Chapter 4)

Print Control for Compilation Listing

PAGE	Eject printer paper to top of next page.
LINE n	Skip n lines.
* SINGLE	Single space the listing.
* DOUBLE	Double space the listing.
* INDENT n or INDENT ± n	Set the left margin of the listing to n or K±n.
$ PRINT <parameter string>	Print or don't print selected parts of the listing.

Miscellaneous

RIGHT MARGIN n	Scan cards to column n for text.
LIBRARY <identifier>	Fetch <identifier> from the symbolic library.
† n ABCONS	Reserve space for n abcons and n abcons.
SEGMENT n_1, n_2	Treat this program as segment n_1 of length n_2.
RELEASE WHAT	WHAT will no longer be used. Free the space for compilation.
RELEASE SYMBOLIC LIBRARY	The symbolic library will no longer be used. Free the space for compilation.
DEBUG n	If n > 0 print the results produced by the three phases of the translator. If n=0 do not print the results.

* not printed

$ may not contain comments

† must occur before the first begin

THIS PUBLICATION IS AVAILABLE AT:

THE BOOK STORE
BAKER HALL
CARNEGIE INSTITUTE OF TECHNOLOGY
PITTSBURGH, PENNSYLVANIA 15213

THE SALE PRICE ($1.25)* REFLECTS THE COST
OF PRINTING AND DISTRIBUTION ONLY.

*(Add $.06 sales tax for purchase in Pa.)
*(Add $.20 postage for mail orders)