Whitesmiths, Ltd.
C Programmers' Manual

Edition 2.2
March 1983

Whitesmiths, Ltd.

C PROGRAMMERS' MANUAL

Edition: 2.2

Date: March 1983

The C language was developed at Bell Laboratories by Dennis Ritchie; Whitesmiths, Ltd. has endeavored to remain as faithful as possible to his language specification. The external specifications of the Idris operating system, and of most of its utilities, are based heavily on those of UNIX, which was also developed at Bell Laboratories by Dennis Ritchie and Ken Thompson. Whitesmiths, Ltd. gratefully acknowledges the parentage of many of the concepts we have commercialized, and we thank Western Electric Co. for waiving patent licensing fees for use of the UNIX protection mechanism.

The successful implementation of Whitesmiths´ compilers, operating systems, and utilities, however, is entirely the work of our programming staff and allied consultants.

For the record, UNIX is a trademark of Bell Laboratories; IAS, RSTS/E, VAX, VMS, P/OS, PDP-11, RT-11, RSX-11M, and nearly every other term with an 11 in it all are trademarks of Digital Equipment Corporation; CP/M is a trademark of Digital Research Co.; MC68000 and VERSAdos are trademarks of Motorola Inc.; ISIS and iRMX are trademarks of Intel Corporation; A-Natural and Idris are trademarks of Whitesmiths, Ltd. C is not.

Copyright (c) 1978, 1979, 1980, 1981, 1982, 1983

by Whitesmiths, Ltd.

All rights reserved.

C PROGRAMMERS' MANUAL

SECTIONS

I. The C Language

II. Portable C Runtime Library

III. C System Interface Library

IV. C Machine Interface Library

SCOPE

This manual describes the C programming language, as implemented by Whitesmiths, Ltd., and the various library routines that make up the machine independent C environment. Section I introduces the C language, and Section II details the numerous functions callable from C to extend the power of the language. Section III lists the functions that interface to a given operating system, while Section IV describes the functions that interface to a given machine architecture. The distinction between Sections II, III, and IV, while of considerable importance to implementors, is probably academic to most programmers -- all functions in all of these sections are present as described on all systems supported by Whitesmiths, Ltd.

For documentation of the programming utilities, or information on each implementation of system or machine dependent features of the C support software, see the C Interface Manual for the appropriate machine.

TABLE OF CONTENTS

I. The C Language

I - 1	**Introduction**	the C compiler
I - 3	**Syntax**	syntax rules for C
I - 7	**Identifiers**	naming things in C
I - 11	**Declarations**	declaring names in C
I - 15	**Initializers**	giving values to data
I - 17	**Statements**	the executable code
I - 19	**Expressions**	computing values in C
I - 24	**Constants**	compile time arithmetic
I - 25	**Preprocessor**	lines that begin with #
I - 27	**Style**	rules for writing good C code
I - 32	**Portability**	writing portable code
I - 35	**Differences**	comparative anatomy
I - 37	**Diagnostics**	compiler complaints

II. Portable C Runtime Library

II - 1	**Conventions**	using C with the standard libraries
II - 4	**std.h**	standard header file
II - 6	**Files**	I/O using the standard libraries
II - 17	**Runtime**	command line processing using the standard libraries
II - 21	**FIO**	the file input/output structure
II - 22	**V7lib**	a library for UNIX support under the standard compiler
II - 26	**_memerr**	no memory error condition
II - 27	**abs**	find absolute value
II - 28	**alloc**	allocate space on the heap
II - 29	**amatch**	look for anchored match of regular expression
II - 31	**arctan**	arctangent
II - 32	**bldks**	build key schedule from key
II - 33	**btod**	convert buffer to double
II - 34	**btoi**	convert buffer to integer
II - 35	**btol**	convert buffer to long
II - 36	**btos**	convert buffer to short integer
II - 37	**buybuf**	allocate a cell and copy in text buffer
II - 38	**cmpbuf**	compare two buffers for equality
II - 39	**cmpstr**	compare two strings for equality
II - 40	**cos**	cosine in radians
II - 41	**cpybuf**	copy one buffer to another
II - 42	**cpystr**	copy multiple strings
II - 43	**decode**	convert arguments to text under format control

II - 44	**decrypt**	decode encrypted block of text
II - 45	**doesc**	process character escape sequences
II - 46	**dtento**	multiply double by a power of ten
II - 47	**dtoe**	convert double to buffer in exponential format
II - 48	**dtof**	convert double to buffer in fixed-point format
II - 49	**encode**	convert text to arguments under format control
II - 50	**encrypt**	encode block of text
II - 51	**enter**	enter a control region
II - 52	**errfmt**	format output to error file
II - 53	**error**	print error message and exit
II - 54	**exp**	exponential
II - 55	**fclose**	close a file controlled by FIO buffer
II - 56	**fcreate**	create a file and initialize a control buffer
II - 57	**fill**	propagate fill character throughout buffer
II - 58	**finit**	initialize an FIO control buffer
II - 59	**fioerr**	NULL FIO pointer condition
II - 60	**fopen**	open a file and initialize a control buffer
II - 61	**fread**	read until full count
II - 62	**free**	free space on the heap
II - 63	**frelst**	free a list of allocated cells
II - 64	**fwrite**	write and check
II - 65	**getbfiles**	collect files from command line
II - 66	**getc**	get a character from input buffer
II - 67	**getch**	get a character from input buffer stdin
II - 68	**getf**	read formatted input
II - 72	**getfiles**	collect text files from command line
II - 73	**getflags**	collect flags from command line
II - 76	**getfmt**	read formatted input from stdin
II - 77	**getin**	build ac and av list from STDIN
II - 78	**getl**	get a text line from input buffer
II - 79	**getlin**	get a text line from input buffer stdin
II - 80	**gtc**	get a character from input buffer
II - 81	**inbuf**	find occurrence in buffer of character in set
II - 82	**instr**	find occurrence in string of character in set
II - 83	**isalpha**	test for alphabetic character
II - 84	**isdigit**	test for digit
II - 85	**islower**	test for lowercase character
II - 86	**isupper**	test for uppercase character
II - 87	**iswhite**	test for whitespace character
II - 88	**itob**	convert integer to text in buffer
II - 89	**itols**	convert integer to leading low-byte string
II - 90	**leave**	leave a control region
II - 91	**lenstr**	find length of a string
II - 92	**ln**	natural logarithm
II - 93	**lower**	convert characters in buffer to lowercase
II - 94	**lstoi**	convert leading low-byte string to integer
II - 95	**lstol**	convert filesystem date to long
II - 96	**lstou**	convert leading low-byte string to unsigned short
II - 97	**ltob**	convert long to text in buffer
II - 98	**ltols**	convert long to filesystem date
II - 99	**mapchar**	map single character to printable representation
II - 100	**match**	match a regular expression
II - 101	**max**	test for maximum
II - 102	**min**	test for minimum

II - 103	**mkord**	make an ordering function
II - 105	**nalloc**	allocate space on the heap
II - 106	**notbuf**	find occurrence in buffer of character not in set
II - 107	**notstr**	find occurrence in string of character not in set
II - 108	**ordbuf**	compare two NUL padded buffers for lexical order
II - 109	**pathnm**	complete a pathname
II - 110	**pattern**	build a regular expression pattern
II - 112	**prefix**	test if one string is a prefix of the other
II - 113	**ptc**	put a character to output buffer
II - 114	**putc**	put a character to output buffer
II - 115	**putch**	put a character to output buffer stdout
II - 116	**putf**	output arguments formatted
II - 119	**putfmt**	output arguments formatted to stdout
II - 120	**putl**	put a text line to output buffer
II - 121	**putlin**	put a text line to output buffer stdout
II - 122	**putstr**	copy multiple strings to file
II - 123	**readerr**	read error condition
II - 124	**remark**	print non-fatal error message
II - 125	**round**	round real to integer
II - 126	**scnbuf**	scan buffer for character
II - 127	**scnstr**	scan string for character
II - 128	**sin**	sine in radians
II - 129	**sort**	sort items in memory
II - 131	**sqr**	square an argument
II - 132	**sqrt**	real square root
II - 133	**squeeze**	delete specified character from buffer
II - 134	**stdin**	the standard input control buffer
II - 135	**stdout**	the standard output control buffer
II - 136	**stob**	convert short to text in buffer
II - 137	**subbuf**	find occurrence of substring in buffer
II - 138	**substr**	find occurrence of substring
II - 139	**tolower**	convert character to lowercase if necessary
II - 140	**toupper**	convert character to uppercase if necessary
II - 141	**trunc**	truncate real to integer
II - 142	**usage**	output standard usage information
II - 143	**writerr**	write error condition

III. C System Interface Library

III - 1	**Cint**	C interface to operating system
III - 3	**main**	enter a C program
III - 4	**_pname**	program name
III - 5	**close**	close a file
III - 6	**create**	open an empty instance of a file
III - 7	**exit**	terminate program execution
III - 8	**lseek**	set file read/write pointer
III - 9	**onexit**	call function on program exit
III - 10	**onintr**	capture interrupts
III - 11	**open**	open a file
III - 12	**read**	read characters from a file

III - 13	**remove**	remove a file
III - 14	**sbreak**	set system break
III - 15	**uname**	create a unique file name
III - 16	**write**	write characters to a file

IV. C Machine Interface Library

IV - 1	**Conventions**	of the C machine interface library
IV - 2	**_addexp**	scale double exponent
IV - 3	**_domain**	report domain error
IV - 4	**_domerr**	domain error condition
IV - 5	**_dtens**	powers of ten
IV - 6	**_dzero**	double zero
IV - 7	**_fcan**	canonicalize floating point datum
IV - 8	**_frac**	extract integer from fraction part
IV - 9	**_huge**	largest double number
IV - 10	**_norm**	convert double to normalized text string
IV - 11	**_ntens**	number of powers of ten
IV - 12	**_poly**	compute polynomial
IV - 13	**_raise**	raise an exception
IV - 15	**_ranerr**	range error condition
IV - 16	**_range**	report range error
IV - 17	**_round**	round off a fraction string
IV - 18	**_stop**	end of stack area
IV - 19	**_tiny**	smallest double number
IV - 20	**_unpack**	extract fraction from exponent part
IV - 21	**_when**	handle exceptions