ch	function name	python / c pseudo-code
s	rcore_t_s	t=stack.pop()
S	rcore_s_t	stack.append(t)
f	rcore_t_f	t=tf
F	rcore_f_t	tf=t
g	rcore_t_g	t=tg
G	rcore_g_t	tg=t
h	rcore_t_h	t=th
H	rcore_h_t	th=t
i	rcore_t_i	t=ti
I	rcore_i_t	ti=t
k	rcore_t_k	t=tk
K	rcore_k_t	tk=t
l	rcore_t_l	t=tl
L	rcore_l_t	tl=t
c	rcore_t_c	t=cstack.pop()
C	rcore_c_t	cstack.append(t)
z	rcore_zte	if(t==0) tk=1; else tk=0
Z	rcore_ztg	if(t>0) tk=1; else tk=0
j	rcore_jnz_r	if(tk!=0) jump to line# (line+t)
J	rcore_jnz_a	if(tk!=0) jump to line# t
q	rcore_quit	sys.exit(t)
Q	rcore_quit_ifkz	if(tk==0) sys.exit(t)
n	rcore_not_tk	tk = NOT tk
o	rcore_or_tk	tk = tk OR tl
a	rcore_and_tk	tk = tk AND tl
x	rcore_xor_tk	tk = tk XOR tl
N	rcore_not_tl	tl = NOT tl
O	rcore_or_tl	tl = tk OR tl
A	rcore_and_tl	tl = tk AND tl
X	rcore_xor_tl	tl = tk XOR tl
_	rcore_t_zero	t=0
^	rcore_t_inc	t++
v	rcore_t_dec	t--
<	rcore_t_shl	t=t<<1
>	rcore_t_shr	t=t>>1
|	rcore_t_abs	t=abs(t)
-	rcore_t_flipsign	t=t*-1

ch	function name	python / c pseudo-code
+	rmath_t_tl_add	t=t+tl
*	rmath_t_tl_mul	t=t*tl
/	rmath_t_tl_idiv	t=t//tl
%	rmath_t_tl_mod	t=t%tl
p	rmath_t_tl_pow	t=floor(pow(t,tl))
P	rmath_t_tl_log	t=floor(log(t,tl))

ch	function name	python / c pseudo-code
u	rxtra_t_uptime_s	t=floor(time.monotonic())
U	rxtra_t_uptime_ns	t=time.monotonic_ns()%1000000000
R	rxtra_t_randseed	random.seed(t)
r	rxtra_t_randint	t=random.randint(0,t-1)
w	rxtio_t_in_char	t=sys.stdin.read(1)
W	rxtio_t_in_int	inputs a decimal int and stores it to t
y	rxtio_t_out_char	sys.stdout.write(t)
Y	rxtio_t_out_int	outputs t in decimal form
m	rxtio_t_in_hex	inputs a hexadecimal int and stores it to t
M	rxtio_t_out_hex	outputs t in hexadecimal form

ch	function name	python / c pseudo-code
B	rbarr_t_b_tl	t=b[tl]
B	rbarr_b_t_tl	b[tl]=t
c	rbarr_t_wordnum_tl_b	get wordnum from t%4+2 chars in array b starting at index tl ; store wordnum to t
C	rbarr_t_varnum_tl_b	get varnum from t%4+1 chars in array b starting at index tl ; store varnum to t
d	rbarr_word_b	make new word with name (in wordnum form) in b[t] and valid chars starting at b[t+1]
D	rbarr_var_b	make new var with name (in varnum format) in b[t] and value of b[t+1]
e	rmeta_run_word_t	runs word with name (in wnum format) in t; t=(sum of return vals)%256 (usually 0). Note: any word whose code contains the char 'e' will not run properly here.
E	rmeta_t_var	value of var with name (in vnum format) in t is stored to t.