"hello"		shortcut for f("hello")			
unction o		ody [return values] end simple call, possibly retu			as additional first argument self
1 uoŋəun	vane (args) body	return values] end [return values] end	sportcut	ul = <i>9mbn.</i> 1 1	
n notionu n notionu		tnen values] end Letnen values] end Letnen values] end	defines f	etion as local	igns to global variable name to chunk igned to variable f
= 8sm} =	=10; "yes", "no"} "choice", {"yes",	{{;,'ou''			ue t.x, t.y, t[1], t[2] s tables as fields
(006-]} =	$$\hat{t} = [000], \hat{c} = \{010\}$		hash tabl	fields are t["	vo elements (no space wasted)
= {}	"no", "?"} "yes", [2] = "no",	[3] =3 }	s əlqmis	ıy; elements a	nd assigns it to t rre t[1], t[2], t[3] . explicit fields
леак					statement in block.
$= u p \lambda $ To	dxə nntil e xp start, end [, step] do iterator do blo		numerics	for loop; var	strue; exp is in loop scope. is local to loop.
dxə əlidv	dxə liəslə} əbold n	uə [уэора əsрә] [сүзора шәц) с	condition loop as l	s daces local s l execution g as <i>exp</i> is tru	ər
s = 3 2	tructures st	rings expected, numbers are	inte ot betre	("26" = s) 22")	
z = 1	nu7.	z is not defined it is nil , so umbers expected, strings are	erted to nur	$\operatorname{GLS}\left(\mathbf{g}=2\right)$	(1
te = 1.	re" fo	or missing values on right ha estroys a ; its contents are eli	ess si lin əl	pəu	
, b, c = 1, g, b = b, a g, b = d, 5,	AS	wap values: right hand side i sois bnahthagir :eses we supl scess values on right hand si	uated befor		
$\mathbf{c} = \mathbf{c} \mathbf{p} = \mathbf{c}$	ЭĮ		after the fi		ent types. Local variables are so that local $\mathbf{a} = S$).
sdots) 1	n true (not false or	urns last evaluated value) r nil), returns last evaluated	(
o garins) .	ncatenation, right -	associative)	=<	=~	==
101		# (Jength of st		% 1) -	nnary)
	s, decreasing pr sociative, math libr				
psckslas mapell psckslas	" string delin	niters; interpret escapes.	/ əui	ti line string; return sq. bracket	escape sequences are ignored. /t horiz. tab /v vert. tab /ddd decimal (up to 3 digits)
nil" Vote: for t	"boolean"	"string" d false count as false; everyt	"fable"	oitonnî"	n" "thread" "userdata"
eing any s	equence of upperca			the line. ry function	(()əd/i; (
••• -	comment to end of rved" (by conventio		լոա [=[t (zero or multiple '=' are valid) 5; Lua ignores whole first line if this
ocsj	break do nil not	else elseif or repeat	rn ther	for frue	function if in until while
^ ! թա ւ սթլչ [enters inter	ename (loads and executes in active mode after loading ar ion information	<i>linos</i> gaituo		
! און השטע אוט אוט און אוט	executes the requires the requires file reters inters	active mode after loading ar	ng <i>sta</i> ts, can Iready don cuting <i>scri</i> f	lillum bəsu əc	
na [option stats stats stats stats stationalit	executes the requires the requires file reters inters	nts]] secutes script from standar te Lua statements in the liter ename (loads and executes ir active mode after loading ar	t (no args a ng <i>stats</i> , cal lready don cuting <i>scri</i> p	iyinm bəsu əc Owed)	ple times on the same line
nsmman na (option grions e stats e stats ii	d line syntax [script [argumen] loads and e executes the process of the syntax requires file enters inters enters inters enters inters	ecutes script from standante. Lua statements in the liter ename (loads and executes in active mode after loading ar	t (no args a ng <i>stats</i> , cal lready don cuting <i>scri</i> p	iyinm bəsu əc Owed)	ple times on the same line
returing to the comman comman comman comman comman comman comman comman comman commen	ms fields name and ms field func ms fields source, si line syntax s] [script [argumen loads and e executes thi requires file requires file	d namewhat hort_src, what and linedeff The stand-s is [List statements in the liter ename (loads and executes is ename (loads after loading ar active mode after loading ar	n re n fine args s ng stats, ca lready don cuting scrip	iyinm bəsu əc Owed)	ple times on the same line
ptions fine comman comm	ms fields name and ms field func ms fields source, si line syntax s] [script [argumen loads and e executes thi requires file requires file	the function itself d namewhat and linedeff store. what and linedeff The stand-store. The standard is secutes script from standard in the liter statements in the liter statements in the liter statements.	I re u re u re n on a re t (no a res s a s rats, ca t (no a res s a s rats, ca tready don cuting scrip	rns field nup rpreter seused multi	ple times on the same line
unc unc unc options f cents cents f filename f f f f f f f f f f f f f	or debug-getinfing fields name and mar field func ma fields source, significant fine syntax significant furgument facript (argument facript (argument facript)) loads and executes this field facript facript (argument facript) requires filed facritics facript f	meaning of name: "globa number of upvalues of the function itself to (character codes for hort_src, what and linedeff The Stand-standard of the statements in the liter ename (loads and executes is crive mode after loading an active mode after loading and active mode active	cal", "met vion I re u re ng stats, ca It (no args a ng stats, ca lready don cuting scrip	ms field" or ms field curr rpretter rpretter	rentline ple times on the same line
anne anne anne anne anne anne anne anne	or debug.getinfing fields name and mar field func mar fields source, significant fields source, significant fields source, significant fields	"Lua" = Lua function, "C name of function, if avail meaning of name: "globing of name: "globing of character codes for the function itself anamewhat hort_src, what and linedeff [The stand-stand namewhat] The standard of Lua statements in the liter secutes acript from standard of Lua statements in the liter secutes scrive mode after loading an active mode after loading and active mode ac	function, " or a reason scal", "met stron nment w) I re u re t (no args a ng stats, ca lready don cuting scrip	ain" = part of degreess if pode, "field" or ms field curp ras field curp ras field curp oved)	rii rentline ple times on the same line
ource indecfined indec	or debug-getinfins fields name and mar field func ma fields source, significant fine syntax si [script [argumen loads and e executes thing the syntax and e enters interesting the syntax and e executes thing the syntax and e executes thing the syntax and e executes thing the syntax and e executes the syntax and e executes thing the syntax and e executes the	name of file (prefixed by short version of source, in of source where the line of source where the line of source where the line of unction, if avai name of function, if avai number of upvalues of the function itself the function itself anamewhat hort_src, what and linedeff the line line statements in the liter will line statements in the liter scenutes script from standard to Lua statements in the liter scenute scrive mode after loading an active mode after loading and active mode after loading an active mode after loading and active mode activ	50 characte on was defi function, " or a reson oral", "meti xion I re u re t (no args a ng stafs, ca lready don cuting scrip	sin" = part of de guess if po de, "field" or ms field curr ms field nup ms field nup owed)	sasible rentline rentline rentline ple times on the same line
ource hort_src inedefine hort_src inedefine hort src innedefine hort inne ho	ids for debug.getinfing felds name and ms fields name and ms fields source, single source, singl	name of file (prefixed by short version of source, in of source where the line of source where the line of source where the line of unction, if avai name of function, if avai number of upvalues of the function itself the function itself anamewhat hort_src, what and linedeff the line line statements in the liter will line statements in the liter scenutes script from standard to Lua statements in the liter scenute scrive mode after loading an active mode after loading and active mode after loading an active mode after loading and active mode activ	or string who characte on was defifunction, "or a reason ocal", "met strion or a reason or a stars, can be int	re the function sin" = part of ain, "field" or ms field curp ms field curp ms field curp or great	n was defined ranin chunk ranin ranin rentline
ource hort_src inedefine hort_src inedefine hort src innedefine hort inne ho	ebug library function descriptions of the second of the se	second argument) or "con returns current hook fund ons are not optimised for eff petinfo name of file (prefixed by short eversion of source, in the of source, in the of source where the "Lua" = Lua function, if avai name of function, if avai name of function, if avai number of upvalues of the function, if avai number of upvalues of the function itself the function itself and mamewhat the function itself the function itself the function itself the function itself and included for character codes for the function itself and included for character, what and linedeff the function itself the function itself and linedeff the function itself included after loading and accine a statements in the liter ending and control in the liter for a statements in the liter for a statements in the liter for a statements in the liter accine a series in the liter for a statements in the liter for a statement in the liter for a	see debug.g. mask and shou ar string who on was cide. function, "nor a reason cide." "In reason in a stars, can it (no args a stars, can it (no args a stars, can it can be in the interpretation.	tinfo(2) insidential and be used in not be used in the function sin" = part of the guess if poder, "field" or ms field and field and poded) Tpreter	e h() for info (not for "tsil_retum"). sbug.sethook() n normal operation. masin chunk sssible "" rentline ""
lebug, getl dots: the double:	ebug library function columness for debug.gg. or debug.getinfing fileds name and ma field function fileds source, signification for the syntax si [script [argument for source, signification for source fo	function return, "I" = ney receive the event type as second argument) or "con returns current hook function returns current hook function ons are not optimised for efficiency and returns current hook function on source, being a foot version of source, bline of source where the "Lua" = Lua function, if avail meaning of name: "global number of function, if avail the function itself number of upvalues of the function itself and present the function itself the function itself and the function itself and character codes for the function itself and in the function itself. The statements in the liter is statements in the liter for the statements in the liter is connected after loading and active mode after loading active mode active mode after loading active mode	time; also, rgument: " rgument: " rgument: " rgument and c	number n wil "i", "return", indo(2) insid met set with de mot be used i sin" = part of geness if po ans field curp ms field curp re field curp	l call h() every n instructions; h() w "tail return", "line" (tine number as e h() for info (not for "tail_return"). ebug.sethook() n normal operation. n was defined ssaible rentline "" rentline "" rentline "" rentline
lebug.getll lebug.getll loc: the d loc: the d loc: the d loci the	ook ([h, m [, n]]) book () chug library functio das for debug.getinf ms fields name and ms fields source, si si [script [argumen toads and es enters inters toads and es enters inters requires file	nil if is out of range. sets function in as hook, of function return, "" = new receive the event type as second argument) or "con returns current hook function are not optimised for efficients of the function of source, in the of source where the short version of source, in the of source where the mame of function, if avail mane of function, if avail the function itself of the function itself the funct	for events; since, also, sugarkents; since, also, and ebug.; and shours; sand shours; sand shours; sand shours; sand shours; sugarkent when the sugarkent in th	wen in string (number n wil in", "return", info(2) insid int set with de not be used i sin" = part of sin" = part of sin" = part of de guess if po ins field curr ms field curr ms field nup owed) owed)	mask) m: "c" = function call, "r" = "call h() every n instructions; h() w "call h() every n instructions; h() w "call return", "line" (line number as "bug.sethook() n normal operation. n was defined sasible "" rentline "" ple times on the same line
lebug.settlebug.	pvalue (f, i, v) look (lh, m [, n]]) ebug library functio lds for debug-getinfi ms fields name and ms fields source, si ms fields source, si si [script [argumen fields source, si fine syntax fine synta	at stack level n (1= called assigns value v to the up assigns value v to the up nil if i is out of range. The event type as second argument) or "con return, "" = nev receive the event type as second argument) or "con returns current hook function argument) or "con are not optimised for eff income are not optimised for eff and of source where the "Lua" = Lua function, if avair not of source, in the function, if avair number of function, if avair number of unction, if avair number of unction itself. The statement of the unction is a number of included and line definition in the liter of accounted	ins nil if i at index i (i) for events; line; also, rgument; ", se debug; mask and c nor string wh or string wh or a reson function," nor a reson nor a reson nor a reson function," nor a reson in no args a nor a reson function," nor a reson in no args a rents, call, "met no args a rents, call, "met and a rents, call, "met in no args a cuinner soril rents, call, "met and a rents, call, "met and a rents, call, "met and a rents, call, " cuinner soril cu	out of range, om 1, in order wen in string (number n wil II", "return", tinfo(2) insid int set with de not be used i sin" = part of le guess if po le guess if po d", "field" or ms field curr ms field curr rans field curr or o	raises error if n is out of range. of appearance) of function f ; returns [mask) m : "c" = function call, "r" = [call h () every n instructions; h () w "tail return", "line" (line number as e h () for info (not for "tail_return"), n normal operation. n main chunk n was defined rentline rentline rentline
lebug.settlebug.	ook ([h, m [, n]]) book () chug library functio das for debug.getinf ms fields name and ms fields source, si si [script [argumen toads and es enters inters toads and es enters inters requires file	returns nil if i is out of ra assigns value v to the loop as stack level n (1= callet assigns value v to the loop as stack level n (1= callet assigns value v to the loop assigns value v to the upper assignment v to the upper assigns value v to the upper assin the upper assigns value v to the upper assigns value v to t	of call stace inhe at index i (i) ari index i (i) for events i line; also, remement: " or atring when seed and should an attended and should an attended and should an attended and should are attended and attended and are are	prepended by i (from I, in order out of range, out of range, out of range, out of range in order in string (in in, in order in will in, "return", info(2) insid out set with de not be used i or or be used if po de used if po de used if po de used if in in in it is field out in it is field in it is f	was order of appearance) of the function raises error if n is out of range. of appearance) of function f; returns "c" = function of instensial return", "line" (line number as bug, sethok) or info (not for "tail_return"). I call h() every n instructions; h() was defined in normal operation. I main chunk assible readine
lebug, settlebug, sett	pvalue (f, i, v) nook ([h, m [, n]]) ebug library functio lds for debug.getinfi ms fields name and ms fields name and ms fields source, si line syntax liscript [argument requires file requires file requires file requires file	range. returns name and value of returns a string with trace assigns value w to the local sasigns value w to the local sasigns value w to the local sasigns value w to the up assigns value of the vent type as second argument) or "con are not optimised for eff and on a see not optimised for eff and of the column to the function itself and the function itself. The statements in the liter was statements in the liter (order assistements in the liter ename (loads and executes in sective mactive mode after loading and active mode	of call stace at inde at inde at inde at inde arms mil if i at index i (for events; but i (for events;	i (from 1, in or prepended by i (from 1, in or i (from 1, in order out of range, om 1, in order in ore	was defined in the function f; was order of appearance) of the function order of appearance) of the function raises error if n is out of range. I call h() every n instructions f; returns a list return", "line" (line number as bug.sethook() n normal operation. I was defined on the for "tail_return"). n was defined n normal operation. rentline rentline
lebug.geft lebug.seft lebug.seft lebug.seft lebug.seft lebug.geft	eback ([msg]) pvalue (f, i, v) nook ([h, m [, n]]) ebug library function ook (or debug-getinf ms fields name and ms fields source, si si [script [argumen fine syntax	function at stack level norange. range. returns a string with trace returns a string with trace assigns value v to the local assigns value v to the upper assigns value v to the volument of the cource where the value of source viter of columning of name. "Eloba income of source of the the the value of the threshold of the three values of values	I variable a lilet); return alue at inde of call stac inde ar index i (if inc. also, suments.") To string who is the area of a reason function, "nor a reason function, "nor a reason in a reason function, "nor ar index and con was definition area of indexidents." To a reason function, "nor area of a reason function, "nor area of indexidents." To a reason function area area of a reason function, "nor area of indexidents." To a reason function area of indexidents. "In a reason function, "nor area of indexidents." To a reason function area of indexidents. "In a reason function area of indexidents." To a reason function area of indexidents. The indexidents are indexidents are indexidents are indexidents. The indexidents are indexidents are indexidents are indexidents are indexidents. The indexidents are indexidents are indexidents are indexidents. The indexidents are indexidents are indexidents are indexidents are indexidents. The indexidents are indexidents are indexidents are indexidents are indexidents. The indexidents are indexidents are indexidents are indexidents are indexidents. The indexidents are indexidents are indexidents are indexidents are indexidents are indexidents. The indexidents are indexidents are indexidents are indexidents are indexidents are indexidents are indexidents. The indexidents are indexidents are indexidents are indexidents are indexidents are indexidents. The indexidents are indexidents are indexidents are indexidents are indexidents are indexidents are indexidents. The indexidents are indexidents are indexidents are indexidents are indexidents are indexidents are indexidents. The indexidents are indexidents are indexidents are indexidents are indexidents are indexidents are indexidents. The indexidents are indexidents are indexidents are indexidents are indexiden	ndex i (from ndex i (from ndi if i is out i (from 1, in out out of range, i (from 1, in out of range, in out of range, in out of range, in order n wil ill, "return", info(2) insid into be used in or be used in not be used in not be used in not be used in not be used if not be used if not be used in not be	I, in order of appearance) of the of range, raises error if n is out of of range, raises error if n is out of order of appearance) of function f ; was order of appearance) of the function f ; returns raises error if n is out of range. It also of overy n instructions; h () we have the function f ; returns also error if n is out of range. It also every n instructions; h () we have the instructions; h () we have the instructions; h () was defined normal operation. In main chunk In was defined
lebug.getil lebug.getil lebug.getil lebug.setil lebug.setil lebug.setil lebug.setil lebug.setil lebug.setil lebug.setil lebug.getil lebug.setil lebug.getil lebug.	eback ([msg]) ook (h, m, v) nook (h, m, n]]) ebug library functio ebug library functio or debug-getinf ms fields source, si ms fields source, si si [script [argumen and field func ms fields source, si fine syntax si [script [argumen field func ms fields source, si fine syntax si [script [argumen field func ms fields source, si fine syntax si [script [argumen field func ms fields source, si fine syntax si [script [argumen field func ms fields source, si fine syntax si [script [argumen field func ms fields source, si fine syntax si [script [argumen field func	returns a table with informiny all devel (see Result.) groups of fields [default.] groups of fields [default.] function at stack level not range. returns name and value of returns name and value of returns assigns value v to the loos assigns of the vector of the upward of the cource where the line of cource where the line of cource of the function itself to the function itself to the function itself to the function itself. The statements in the liter was tatements in the liter of Lua statements in the liter of the the loos of the liter of the lit	for getinfo see Options see Options all let); return lable at inde at inde at index i (if	iow); charact low); charact low); charact low modes i (from l, in of frame l, in out of range, mu l, in out of range, mu l, in out of range, mu l, in order lange low in string (into l), inside the function out of range, mu l, in order lange low in string out of range, mu l, in order lange low in string out of range, mu l, in order lange low in string out of the function of low low in string low in the function of low in the l	I, in order of appearance) of the of range, raises error if n is out of of range, raises error if n is out of order of appearance) of function f ; was order of appearance) of the function f ; returns raises error if n is out of range. It also of overy n instructions; h () we have the function f ; returns also error if n is out of range. It also every n instructions; h () we have the instructions; h () we have the instructions; h () was defined normal operation. In main chunk In was defined
lebug.getil lebug.getil lebug.getil lebug.setil lebug.setil lebug.setil lebug.setil lebug.setil lebug.setil lebug.setil lebug.getil lebug.setil lebug.getil lebug.	ng () nfo (f [, w]) neach (n, i) pyalue (f, i) nook ([n, m, v) nook ([h, m [, n]]) ook ([h, m [, n]]) ook ([h, m [, n]]) ook (accupation ook (b, m, n) nook () nook (b, m, n) nook	directly. returns a table with inform returns a table with information by the folds [default; groups of fields [default; returns name and value of function at stack level no returns name and value of returns a string with traces a string of the up. In a stack level no (1 = called a string of the up. In a stack level no (1 = called a string of the up.) In a stack level no (1 = called a string of the up.) In a stack level no (1 = called a string of the up.) In a stack level no (1 = called a string and in a standard and in a	n for function for getingo of call state of call state index	or for function); characters we seinly of the modex i (from ni if is out if from 1, in of the month of the mo	ers in string w select one or more ow). 1, in order of appearance) of the of range, raises error if n is out of manage, raises error if n is out of the order of appearance) of function ft; mask) ms: "c" = function call, "t" = function call, "telums are error if n is out of range. I call h() every n instructions; h() w "tail return", "line" (line number as n mormal operation. I call h() every n instructions; h() w "tail return", "line" (line number as n mormal operation. I call h() every n instructions; h() w sasseble of function ft; return").
lebug, geti lebug,	ctions mfo (f [, w]) cocal (n, i) cocal (n, i, v) coc	enters interactive debugg directly. directly. groups of fields [default: groups of groups group	n for function for getting of getting of call flype control in variable a fler); return all et in call stace of call stace in call at index i (if if i	to exit); locally for for functional to the form of the form of form o	tion at level f [1 = caller], or nil if era in string w select one or more era in string w select one or more ow). 1, in order of appearance) of the of range, raises error if n is out of raises or of appearance) of function f; wask) m : "c" = function call, return; raises error if n is out of range. Taises error if n is out of range. of appearance) of function f; return; "line" (line number as lail return", "line" (line number as bug, sethook) I call h () every n instructions; h () we have defined in normal operation. I main chunk n was defined n mormal operation. rentline rentline
%S.	ctions recond (0061) rime zone name; rime zone name; rime zone name; rime (f, w]) reback ([msg]) reback ([msg]) reback ([h, m [, n])) reback ([h, m [, n]]) reducted fine recurrent and each specification of the specification	centers interactive debugg directly. The debugg directly. Gutter interactive debugg directly. groups of fields [default: groups of fields [default: function at stack level n. returns an stack level n. returns a string with trace designs value v to the loos assigns of name of tunction itself have the function itself of the function itself of the function itself have function itself and an amewhat The statements in the liter was tatements in the liter (loads and executes in the liter ename (loads and executes in sective materine (loads and executes in astitutes in the liter ename (loads and executes in astitutes in astitutes in the liter ename (loads and executes in astitutes in the liter ename (loads and executes in astitutes in astitute	n for function for getting of getting of call flype control in variable a fler); return all et in call stace of call stace in call at index i (if if i	to exit); locally for for functional to the form of the form of form o	tion at level f [1 = caller], or nil if era in string w select one or more era in string w select one or more ow). 1, in order of appearance) of the of range, raises error if n is out of raises or of appearance) of function f; wask) m : "c" = function call, return; raises error if n is out of range. Taises error if n is out of range. of appearance) of function f; return; "line" (line number as lail return", "line" (line number as bug, sethook) I call h () every n instructions; h () we have defined in normal operation. I main chunk n was defined n mormal operation. rentline rentline
%X %S %S %S %S %S %S %S %S %S %S %S %S %S	cifons citier AM or PN minute (00.59) second (00.61) second (00.61) minute (00.59) second (00.61) mine zone name. coal (n, i, v) myalue (f, i, v) coal (n, i	The debugg directly. The debugg directly. The debugg directly. The debugg directly. The debugg invalid level (see Result) groups of fields [default: groups of fields [default: function at stack level not returns and if i is out of range. The staing with trace assigns value v to the loos assigns value v to the up, assigns of the value of va	n for function for getingo for getingo for getingo for getingo for call stace Options, and call stace of call stace index in for events; the fine; stands for events; the fine; and shou for events; the fine; stands for events; the fine; stands for events; the fine; and shou for events; the fine; stands for events; the fine; and con the fine; the fin	To exit); locally and the bug] If or for function, character and it is out and it is out and it is out of range, and it is out of and	al variables cannot be accessed tion at level f [1 = caller], or nil if ers in string w select one or more ow). I, in order of appearance) of the order of appearance) of function f; order of appearance) of the function raises error if n is out of raises error if n is out of raises error if n is out of range. order of appearance) of the function f; ansel neum', "line" (line unmber as l'ail return", "line" (line number as l'ail return", "line" (line number as l'ail return"). abug.sethook() n normal operation. n was defined n main chunk n was defined n mean chunk ple times on the same line rentline
% % % % % % % % % % % % % % % % % % %	abbreviated wee hour (00.23) either AM or PN minute (00.29) either AM or PN erone (00.29) either AM or PN erone (00.29) either (1, w]) cook (2, w]	enters interactive debugg directly. The debugg directly. The debugg directly. The debugg directly. The debugg stouch as table with informatial devel (see Result). The groups of fields [default: a table with informatial devel (see Result). The stack level in tetums a stack level in the tetums as the own of the own the local stack level in the local assigns value v to the up assigns value v to the up the stack level in the liter (see where the stack level in the liter (see in the level in the level in the liter (see in the level in the level in the level in the level i	n for function for getting of getting of call flype control in variable a fler); return all et in call stace of call stace in call at index i (if if i	hour (01 to bug] to exit); location; character of or for function; character of character of the form and if it is out of range, and of range, and it in order in out of range, and it in order in out of range, and it in order in out of range, and it in order on out of range, and it in order on out of range, and it in order out of range, and it in order of the form in order order in out of range, and it in order of the form of the function of the form of the function of the form of the function of the used if or order of the function of the used in order of the following of th	al variables cannot be accessed tion at level f [1 = caller], or nil if ers in string w select one or more ow). I, in order of appearance) of the order of appearance) of function f; order of appearance) of the function raises error if n is out of raises error if n is out of raises error if n is out of range. order of appearance) of the function f; ansel neum', "line" (line unmber as l'ail return", "line" (line number as l'ail return", "line" (line number as l'ail return"). abug.sethook() n normal operation. n was defined n main chunk n was defined n mean chunk ple times on the same line rentline
% % % % % % % % % % % % % % % % % % %	week number (0) week day (0.6), weekday (0.6), abbreviated we hour (0059) either AM or PN minute (0059) second (0061) ime zone name, ime zone name, ook ([h, m [, n]]) abalue (f, i, v) ook ([h, m [, n]]) ing fold debug-getinf ms fields name and es and folds for debug-getinf ms fields source, si ing fields source, si ing fields source, si ing fields fine syntax ing fields source, si ing fields fine fields file ing fields file ing fields file ing f	The debugg centers interactive debugg as stack level nor invalid level (see Result) groups of fields level to the upon as stack level nor invalid level (see Result) groups of fields level to the look function at stack level nor invalid level (see Result) groups of fields level to the look function at stack level nor invalid level (see Result) groups of fields level nor returns a string with trace returns and stack level nor returns a string with trace assigns value v to the look assigns of the upon as are not optimised for eff look of the look as assigns of the look as as to the look as as a look as as look as as a look as as a look as as a look as as look as as a look as as a look as as a look as as a look as a look as as as a look as as a look as as as a look as as as a look as as as as a look as as as as a look as as as as as as a look as	over the control of the control of the control of call state of control of call state of control of call state of call of call of call of call state of call of	full weekd hour (01 to hour (01 to exit); location of the control of the	al variables cannot be accessed ion at level f [1 = caller], or nil if ion at level f [1] = caller], or nil if ever in string w select one or more of range, raises error if n is out of unction f; order of appearance) of function f; returns order of appearance) of the function f; returns order of appearance) of the function f; returns raises error if n is out of range. I call h() every n instructions; h() w "tail return", "line" (line number as len) for info (not for "tail_return"), abug.sethook() I call h() every n instructions; h() w "tail return", "line" (line number as bug.sethook() I call h() every n instructions; h() w "tail return"), abug.sethook() I call h() every n instructions; h() w "tail return"), abug.sethook() I call h() every n instructions; h() w "tail return"), abug.sethook() I call h() every n instructions; h() w "tail return"), abug.sethook() I call h() every n instructions; h() w "tail return"), abug.sethook() I call h() every n instructions; h() w "tail return"), abug.sethook() I call h() every n instructions; h() w "tail return"), abug.sethook() I call h() every n instructions; h() w "tail return"), abug.sethook() I call h() every n instructions; h() w "tail return"), abug.sethook() I call h() every n instructions; h() w "tail return"), abug.sethook() I call h() every n instructions (instructions) I call h() every n i
% % % % % % % % % % % % % % % % % % %	day of month (0) week damy of month (0) week damy (0.6.59) either AM or PN minute (00.59) econd (00.61) minute (00.59) second (00.61) minute (00.59) ctions minute (00.59) ctions minute (00.59) ctions minute (00.59) ctions my fields for debug.getint my fields name and my fields file trequires file requires file req	The debugg centers interactive debugg as stack level nor invalid level (see Result) groups of fields level to the upon as stack level nor invalid level (see Result) groups of fields level to the look function at stack level nor invalid level (see Result) groups of fields level to the look function at stack level nor invalid level (see Result) groups of fields level nor returns a string with trace returns and stack level nor returns a string with trace assigns value v to the look assigns of the upon as are not optimised for eff look of the look as assigns of the look as as to the look as as a look as as look as as a look as as a look as as a look as as look as as a look as as a look as as a look as as a look as a look as as as a look as as a look as as as a look as as as a look as as as as a look as as as as a look as as as as as as a look as	%A %I %I %I %I %I %I %I %I %I	week num full week hour (01 lebug] to cxit); loca for for func flow); charact low); charact low) lowed low) lowed low) lowed lowed low) lowed	lay name (locale) lay name (locale) lay aniables cannot be accessed lon at level f [I = caller], or nil if ers in string w select one or more cers in string w select one or more cers in string w select one or more lof range, raises error if n is out of raises error if n is out of tange. order of appearance) of function f; raises error if n is out of range. order of appearance) of function f; raises error if n is out of range. order of appearance) of function f; raises error if n is out of range. order of appearance) of function f; mask) m. "c" = function raises error if n is out of range. order of appearance) of function f; raises error if n is out of range. order of appearance) of function f; raises error if n is out of range. anset in it is out of range. anset in it is out of range. raises error if n is out of range. order of appearance) of the function raises error if n is out of range. anset if n is out of range. raises error if n is out of range. order of appearance) of the function raises error if n is out of range. raises error if n is out of range. order of appearance) of the function raises error if n is out of range. raises error if n is out of range. order of appearance) of the function raises error if n is out of range. raise
%64 %64 %64 %64 %64 %65 %65 %65 %66 %66 %66 %66 %66 %66 %66	abbreviated more day of month (0) week flumb(c. (0), abbreviated week flumber (0), abbreviated week flumber (0), abbreviated (0), abbreviated (0), abbreviated (0), abbreviated (0), abbreviated (1), abbrevi	if any min name (locale) 10.35, Sunday-based 10.	% W % % W % W % W % W % W % W % W % W %	week num full week hour (01 lebug] to cxit); loca for for func flow); charact low); charact low) lowed low) lowed low) lowed lowed low) lowed	of month (locale) aber (0153), Monday-based day name (locale) day name (locale) al variables cannot be accessed al variables cannot be accessed al variables cannot be accessed tion at level f [I = caller], or nil if ers in string w select one or more of range, raises error if n is out of range. Taises error if n is out of range. order of appearance) of function f; returns, order of appearance) of the function f; ansk) m: "c" = function call, "r" = order of appearance) of the function f; ansk) m. "c" = function call, "r" = order of appearance) of the function f; ansk) mag order of appearance) of the function f; ansk) mage, raises error if n is out of range. order of appearance) of function f; mage of an include function f; ansist return". "ansin chunk page defined n mas defined assible rentline rentline rentline rentline

	returns next index, value pair or nil when finished.
(t [, inx])	if inx is nil [default] returns first index, value pair of table t; if inx is the previous index
(t) sairs	returns an iterator getting key, value pairs of table t in an unspecified order
(1) szingi	returns an iterator getting index, value pairs of array t in numerical order
terators	
nubsck (t)	returns $\mathbf{t}[\mathbf{I}]\mathbf{t}[\mathbf{n}]$ (n = $\mathbf{H}\mathbf{t}$) as separate values
	invalid; for base 10 accepts full format (e.g. "1.5e6").
tonumber (x [, b])	converts string \mathbf{x} representing a number in base \mathbf{b} [236, default: 10] to a number, or nil if
(x) gairtsot	converts x to a string, using t 's metatable's tostring if available
(x) əd \dagan	returns the type of x as a string (e.g. "nil", "string"); see Types above.
	arguments it received after index
select (index,)	returns the arguments after argument number index or (if index is "#") the total number of
nformation and conversio	u
assert (v [, msg])	calls error(msg) if v is nil or false [default msg: "assertion failed!"]
	level n [default: 1, current function]
error (msg [, n])	terminates the program or the last protected call (e.g. pcall()) with error message msg quoting
print (args)	prints each of the passed args to stdout using tostring() (see below)
Simple output and error fe	ведряск
	of h () as ertor message, if any.
хЬсяц (ұ, h)	as peall() but passes error handler h instead of extra args; returns as peall() but with the result
pcall (f [, args])	calls f() in protected mode; returns true and function results or false and error message.
loadstring (s [, name])	loads string s (with chunk name set to name); return values like load().

•	Settlen
dofile ([filename])	loads and executes the contents of filename [default: standard input]; returns its returned
require (pkgname)	loads a package, raises error if it can't be loaded
Loading and executing	
^AEKSION	global variable containing the interpreter's version (e.g. "Lua 5.1")
$^{-}$ e	global variable whose value is the global environment (that is, $\mathbf{G} = \mathbf{G} = \mathbf{G}$)
	environment has a field fenv , raises an error. Returns function \mathbf{f} if $\mathbf{f} \sim 0$.
setfenv (f, t)	sets environment for function ${f r}$ (or function at level ${f r}$, $0=$ current thread); if the original
	instead.
:	at level \mathbf{f} (I = current [default], $0 = \text{global}$); if the environment has a field \mathbf{f} fenv, returns that
getfenv ([f])	if \mathbf{f} is a function, returns its environment; if \mathbf{f} is a number, returns the environment of function

loads a chunk (with chunk name set to \mathbf{name}) using function \mathbf{func} to get its pieces; returns

compiled chunk as function (or ${\bf nil}$ and error message). loads file ${\bf filename};$ return values like ${\bf load}().$

loadfile (filename)

load (func [, chunkname])

Environment and global variables

The base library [no prefix]

	0.4!1 0004 041		[~;70
-metatable	sets value to be returned by getmetatable()		
	from C)		values; $\mathbf{k}\mathbf{v}_{i} = \text{poth}$.
၁ <u>g</u>	sets finalizer h(ud) for userdata (has to be set	әрош	table mode: $\mathbf{k}_{i} = \text{weak keys}$; $\mathbf{k}_{i} = \text{weak}$
	opject as a function)		brint()
_call	sets handler h(f,) for function call (using the	gninteot	sets handler h(a) to convert to string, e.g. for
	field		existing field
xəpui_	sets handler $\mathbf{h}(\mathbf{t}, \mathbf{k})$ for access to non-existing	xəbniwən	sets handler $\mathbf{h}(\mathbf{t}, \mathbf{k}, \mathbf{v})$ for assignment to non-
	(əl on ii) '=<'		
11_	sets handler h(a, b) for '<', '>' and possibly '<=',	ə <u>r</u>	sets handler $\mathbf{h}(\mathbf{a}, \mathbf{b})$ for '<=', '>='
_concat	sets handler h(a, b) for ''	рэ	sets handler $\mathbf{h}(\mathbf{s}, \mathbf{p})$ for $==$, $\sim=$
wun [_]	sets handler h(a) for unary '-'	uəլ	sets handler h(a) for the # operator (userdata)
pom_	set handler h(a, b) for '%'	wod	sets handler h(a, b) for '^ '
ans_ 'nnp_	sets figurated $\mathbf{n}(\mathbf{s}, \mathbf{p})$ for $+$ and for ordary -	AID 'InIII'	sets figurater $\pi(a, b)$ for a sind for 7

sets handler $\mathbf{h}(\mathbf{a},\mathbf{b})$ for ' \wedge '	wod	'%' 101 (d	set handler h(a,	pow
sets handler h(a, b) for '*' and for '/'	vib ,lum	b) for '+' and for binary '-'	sets handler h(a,	qns—'ppe—
		and userdata)	lds (for tables	Metatable fie
spotpame	ut invoking meta	returns boolean (t1 == t2) witho	(7	rawequal (t1, t
spoų	гуокіпд теtатеt	sets $\mathbf{t}[\mathbf{i}] = \mathbf{v}$ on a table without in		rawset (t, i, v)
S	ing metamethod	gets t[i] of a table without invok		rawget (i, i)
netatable or nil	metatable or t 's r	returns metatable field of t 's	(1	getmetatable (
s ametatable field, and returns t	s t 's metatable ha	sets mt as metatable for t, unless	(1m t	setmetatable (t

(v i t) toswer	short a stable without invoking metamethods
rawget (t, i)	gets t[i] of a table without invoking metamethods
getmetatable (t)	returnsmetatable field of t's metatable or t's metatable or mil
setmetatable (t, mt)	sets mt as metatable for t, unless t's metatable has a _metatable field, and returns t
Metatable operations (bas	e library required)
х:шоле (2, -3)	object call: shortcut for $\mathbf{x.move}(\mathbf{x}, 2, -3)$
(x) 1.1	calling a function assigned to field f of table t
$\{\mathbf{f} = \mathbf{g}, \mathbf{f} = \mathbf{x}\}$ T	shortcut for $f(\{x = 3, y = 4\})$
[[uoos noa əəs]] J	sportcut for f([[see you soon]])

Lua is a language designed and implemented by Roberto lerusalimschy, Luiz Henrique de Figueiredo and Waldemar Celes; for details see lua.org. Drafts of this reference card (for Lua 5.0) were produced by Enrico Colombini <erix@erix.ir> in 2004 and updated by Thomas Lauer <htps://documes.com/ence.com/ence.com/enced by Enrico Colombini cerix@erix.ir> in 2004. 2008 and 2009. Comments, praise or blame please to the lua-l mailing list. This reference card can be used and distributed according to the terms of the Lua 5.1 license.

The compiler

arg argli in the prompt for interactive mode; can be changed by assigning a new value.

PROMPT[2] contain the prompt for interactive mode; can be changed by assigning a new value.

defines search path for dynamic libraries (e.g. .g. oo. .g.) with "?" replaced by the module name

if this holds a string in the form @filename loads and executes filename, else executes the string itself defines search path for Lua modules, with "o" replaced by the module name

performs syntax and integrity checking only, does not output bytecode strips debug information; line numbers and local names are lost.

Note: compiled chunks are portable between machines having the same word size.

sends output to filename [default: luac.out]

produces a listing of the compiled bytecode

prints version information stops parsing options

compiles from standard input

set the prompts for interactive mode

Λ-

d-

I-

อนเซนอนูป **0-**

snoitq0

Command line syntax luac [options]

Special Lua variables

Recognized environment variables

stops parsing options

_PROMPT[2]

LUA_PATH LUA_CPATH

	ollection				Pseudo-random nui	mhers
conectgarba	age (opt [, arg])	generic interface to the garbage	collector; opt	defines function performed.	math.random ([n [, m])	returns a pseudo-random number in range [0, 1] if no arguments given; in range [1, n] if n is
		les and the pack			math.randomseed (n)	given, in range [n , m] if both n and m are passed. sets a seed n for random sequence (same seed = same sequence)
module (nar	creates module name . If there is a table in package.loaded[name] , this table is the module. Otherwise, if there is a global table name , this table is the module. Otherwise creates a new table and sets it as the value of the global name and the value of package.loaded[name] .				The string library [string]	
			the global nam	e and the value of package.loaded[name].		end from 1 to #string, or from end of string if negative (index -1 refers to the last character). ets a metatable for strings where theindex field points to the string table. String functions can be used
	dlib (lib, func)	loads dynamic library lib (e.g	so or .dll) and	returns function func (or nil and error message)		e.g. string.len(s) can be written s:len(); literals have to be enclosed in parentheses, e.g. ("xyz"):len().
package.pat package.loa				for a Lua or C loader, respectively les are already loaded (see module)	Basic operations	returns the length of string gringly line ambedded rance (see also # operator)
package.pre	eload eall (module)	a table to store loaders for speci		ee require) ield referring to the global environment	string.len (s) string.sub (s, i [, j])	returns the length of string s , including embedded zeros (see also # operator) returns the substring of s from position i to j [default: -1] inclusive
раскаделес		The coroutine lib		· ·	string.rep (s, n) string.upper (s)	returns a string made of n concatenated copies of string s returns a copy of s converted to uppercase according to locale
coroutine.cı	reate (f)	creates a new coroutine with L	ua function f()	as body and returns it	string.lower (s)	returns a copy of s converted to lowercase according to locale
coroutine.re	esume (co, args)	starts or continues running cord if co calls coroutine.yield() or		ng <i>args</i> to it; returns true (and possibly values) alse and an error message.	Character codes string.byte (s [, i [, j]])	returns the platform-dependent numerical code (e.g. ASCII) of characters s[i], s[i+1],, s[j]. The
coroutine.yi	ield (args)		ng coroutine (n	ot from within C functions, metamethods or		default value for i is 1; the default value for j is i.
coroutine.st	. , ,	returns the status of coroutine of	o: either ''run	ning", "suspended" or "dead"	string.char (args) Function storage	returns a string made of the characters whose platform-dependent numerical codes are passed as args
coroutine.w			ua function f as	s body and returns a function; this function will	string.dump (f)	returns a binary representation of function f (), for later use with loadstring () (f () must be a Lua
		act as coroutine.resume() with any errors.	out the first ar	gument and the first return value, propagating	Formatting	function with no upvalues)
		The table lib	rary [ta	ble]		returns a copy of s where formatting directives beginning with '%' are replaced by the value of
table.insert table.remov		inserts v at numerical index i [c	lefault: after th		Formatting directive	arguments args, in the given order (see Formatting directives below)
		element or nil on empty table.		<u> </u>	% [flags] [field_width]	
	e.maxn (t) returns the largest positive numerical index of table t or zero if t has no positive indices e.sort (t [, cf]) sorts (in place) elements from t[1] to #t, using compare function cf(e1, e2) [default: '<']			compare function cf(e1, e2) [default: '<']	Formatting field typ	
table.concat	le.concat (t [, s [, i [, j]]]) returns a single string made by concatenating table elements $t[i]$ to $t[j]$ [default: $i = 1$, $j = \#t$] separated by string s ; returns empty string if no elements exist or $i > j$.				%d %o	decimal integer octal integer
		• •		•	%x %f	hexadecimal integer, uppercase if %X floating-point in the form [-]nnnn.nnnn
Basic oper	The mathematical library [math] Basic operations				%e %g	floating-point in exp. Form [-]n.nnnn e [+ -]nnn, uppercase if %E floating-point as %e if exp. < -4 or >= precision, else as %f ; uppercase if %G .
math.abs (x math.mod (()	returns the absolute value of \mathbf{x} returns the remainder of \mathbf{x} / \mathbf{y} as	a rounded-do	wn integer, for $\mathbf{v} \sim 0$	%c	character having the (system-dependent) code passed as integer
math.floor ((x)	returns x rounded down to the r	earest integer	, j	%s %q	string with no embedded zeros string between double quotes, with all special characters escaped
math.ceil (x math.min (a	args)	returns x rounded up to the near returns the minimum value from	n the args rece		%%	'%' character
math.max (args)	returns the maximum value from			Formatting flags	left-justifies within field_width [default: right-justify]
math.sqrt (x	al and logarithn x)	returns the square root of x , for	$\mathbf{x} >= 0$		+ (space)	prepends sign (only applies to numbers)
math.pow (x, y)	returns x raised to the power of global function added by the ma	y, i.e. x^y; if x		(space) #	prepends sign if negative, else blank space adds "0x" before %x, force decimal point for %e, %f, leaves trailing zeros for %g
pow (x, y) math.exp (x	()	returns e (base of natural logs) i	aised to the po		Formatting field wid	
math.log (x) math.log10		returns the natural logarithm of returns the base-10 logarithm of			n On	puts at least n (<100) characters, pad with blanks puts at least n (<100) characters, left-pad with zeros
Trigonome	etrical	_			.n	puts at least \mathbf{n} (<100) digits for integers; rounds to \mathbf{n} decimals for floating-point; puts no more than \mathbf{n} (<100) characters for strings.
math.deg (a math.rad (a		converts angle a from radians to converts angle a from degrees t			Formatting example	· · · · ·
math.pi math.sin (a)	1	constant containing the value of returns the sine of angle a (mea		c)	string.format("results: string.format("<%5d>	
math.cos (a))	returns the cosine of angle a (m	easured in radi	ans)	string.format("<%-5d: string.format("<%05d	>", 13) <13 >
math.tan (a) math.asin (x		returns the tangent of angle a (returns the arc sine of x in radia	ns, for x in [-1	, 1]	string.format("<%06.3	3d>", 13) < 013>
math.acos (z		returns the arc cosine of x in rac returns the arc tangent of x in ra		-1, 1]	string.format("<%f>" string.format("<%e>"	
math.atan2	(y, x)	similar to math.atan(y / x) but		and allowing $x = 0$	string.format("<%.4f> string.format("<%9.4f	
Splitting of math.frexp	n powers of 2	splits x into normalized fraction	and exponent	of 2 and returns both	string.format("<%c>"	,64) <@>
math.ldexp		returns $\mathbf{x} * (2 ^ y)$ with $\mathbf{x} = \text{norm}$			string.format("<%.4s> string.format("%q", [
Finding, re	eplacing, iteratir	ng (for the Patterns see be	low)		file:write (values)	writes each of the <i>values</i> (strings or numbers) to file , with no added separators. Numbers are
	(s, p [, i [, d]])	returns first and last position of	pattern p in str	ring s , or nil if not found, starting search at a results. If d is true, treat pattern as plain string.		written as text, strings can contain binary data (in this case, file may need to be opened in binary mode on some systems).
string.gmat	ch (s, p)	returns an iterator getting next of	occurrence of p	restrict. If u is true, treat pattern as plant string.	file:seek ([p] [, of])	sets the current position in file relative to p ("set" = start of file [default], "cur" = current, "end"
- 6-4	(F 7)	substring(s) matching the patter		ccurrences of pattern p (or its captures) replaced		= end of file) adding offset of [default: zero]; returns new current position in file .
string.gsub	(s, p, r[, n])	returns a copy of s with up to n	[default: all] o	contences of patient p (of its captures) replaced	file:flush ()	flushes any data still held in buffers to file
string.gsub	(s, p, r [, n])	by r if r is a string (r can include	le references to	captures in the form $\%n$). If r is a function r () is	Simple I/O	flushes any data still held in buffers to file
string.gsub	(s, p, r [, n])	by r if r is a string (r can include called for each match and received If r is a table, the captures are u	le references to ves captured su sed as fields in	contended of pattern \mathbf{p} (or its captures) replaced of captures in the form \mathbf{m} . If \mathbf{r} is a function \mathbf{r} () is obstrings; it should return the replacement string, it to the table. The function returns the number of		flushes any data still held in buffers to file sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file
string.gsub		by r if r is a string (r can include called for each match and receit If r is a table, the captures are usubstitutions made as second returns captures of pattern p in	le references to ves captured su sed as fields in sult. string s (or the	captures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if	Simple I/O io.input ([file])	flushes any data still held in buffers to file sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure.
string.matcl	h (s, p [, i])	by r if r is a string (r can include called for each match and receif r is a table, the captures are usubstitutions made as second reference captures of pattern p in p does not match s ; starts search	le references to ves captured su sed as fields in sult. string s (or the	captures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if	Simple I/O	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a
string.matcl Patterns ai General patte	h (s, p [, i]) nd pattern items ern format: pattern	by r if r is a string (r can include called for each match and receif If r is a table, the captures are usubstitutions made as second refereturns captures of pattern p in p does not match s ; starts search selection [and the content of the captures of pattern [and the captures of pattern p in p does not match s ; starts search selection [and the captures of pattern_items]	de references to wes captured su sed as fields in sult. string s (or the n at position i [captures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if default: 1].	Simple I/O io.input ([file]) io.output ([file]) io.close ([file])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file]
string.matcl	h (s, p [, i]) nd pattern items em format: pattern matches a single of matches zero or n	by r if r is a string (r can include called for each match and received if r is a table, the captures are usubstitutions made as second reterms captures of pattern p in p does not match s ; starts search s _item [pattern_items] character in the class cc (see Pattern characters in the class cc; materials and the class cc; materials are calculated as a second retermined and the class cc; materials are calculated as a second retermined as a second	de references to ves captured su sed as fields in sult. string s (or the n at position i [ern character of atchest longest	captures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if default: 1]. *classes* below*) sequence (greedy).	Simple I/O io.input ([file]) io.output ([file])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the
string.matcl Patterns al General patte cc cc* cc*	h (s, p [, i]) nd pattern items ern format: pattern matches a single o matches zero or n matches zero or n	by r if r is a string (r can include called for each match and received if r is a table, the captures are usubstitutions made as second reterms captures of pattern p in p does not match s ; starts search s item [pattern_items] character in the class cc (see Pattern characters in the class cc; more characters in the class cc; more characters in the class cc; more characters in the class cc; many contents and contents as a start of the class cc; more characters in the class cc; many contents as a start of the cl	de references to ves captured su sed as fields in sult. string s (or the n at position i [ern character of atchest longest atchest shortes	captures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if default: 1]. *classes* below) sequence (greedy). t sequence (non-greedy).	Simple I/O io.input ([file]) io.output ([file]) io.close ([file]) io.read (formats)	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read()
string.matcl Patterns ai General patte cc cc* cc- cc+ cc+	h (s, p [, i]) nd pattern items em format: pattern matches a single of matches zero or n matches zero or n matches zero or or matches zero or or	by r if r is a string (r can include called for each match and receif If r is a table, the captures are usubstitutions made as second refereturns captures of pattern p in p does not match s ; starts search search are true in the class <i>cc</i> (see <i>Patterneter Characters</i> in the class <i>cc</i> ; manore characters in the class <i>cc</i> ; manore characters in the class <i>cc</i> ; manore character in the class <i>cc</i> ; manore charact	de references to ves captured su sed as fields in sult. string s (or the n at position i [ern character of atchest longest atchest longest	captures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if default: 1]. *classes* below) sequence (greedy). t sequence (non-greedy).	io.input ([file]) io.output ([file]) io.close ([file]) io.read (formats) io.lines ([fn])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write()
string.matcl Patterns at General patte cc cc* cc- cc+ cc? %n %bxy	h (s, p [, i]) nd pattern items em format: pattern matches a single of matches zero or n matches zero or n matches zero or o matches the n-th matches the balan	by r if r is a string (r can include called for each match and receit If r is a table, the captures are usubstitutions made as second reterrors captures of pattern p in p does not match s ; starts search s : p does not match s ; starts search p does not match s ; starts search s distribution p does not match s ; starts search s distribution s distributio	de references to ves captured su sed as fields in sult. string s (or the in at position i [ern character of atchest longest atchest shortes techest longest techest longest techest longest ern captures) aracter y (e.g.	captures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if default: 1]. *classes* below) sequence (greedy). sequence (non-greedy). sequence (greedy).	Simple I/O io.input ([file]) io.output ([file]) io.close ([file]) io.read (formats) io.lines ([fn])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file
string.matcl Patterns as General patte cc cc* cc- cc+ cc? %n	h (s, p [, i]) nd pattern items em format: pattern matches a single of matches zero or n matches zero or n matches one or m matches the n-th matches the balan anchors pattern to	by r if r is a string (r can include called for each match and receive If r is a table, the captures are usubstitutions made as second references of pattern p in p does not match s ; starts search g increase of pattern in the class cc ; may note characters in the class cc ; may note character in the class cc captured string (n = 19, see Pattern g in the pattern g in the pattern g in the class cc captured string (n = 19, see Pattern g in the pattern g in the pattern g in the class cc captured string (n = 19, see Pattern g in the pattern g	de references to ves captured su sed as fields in sult. string s (or the en at position i [ern character of atchest longest atchest shortes tchest longest ern captures) aracter y (e.g. 9 tem in the patt	captures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if default: 1]. *Classes* below) sequence (greedy). It sequence (non-greedy). sequence (greedy). *Web() for nested parentheses)	Simple I/O io.input ([file]) io.output ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file ttility functions derr predefined file objects for stdin, stdout and stderr streams
string.matcl Patterns at General patte cc cc* cc- cc+ cc? %n %bxy ^ \$ Captures	h (s, p [, i]) nd pattern items em format: pattern, matches a single of matches zero or n matches zero or n matches zero or o matches zero or o matches the n-th o matches the balan anchors pattern to anchors pattern to	by r if r is a string (r can include called for each match and receive If r is a table, the captures are usubstitutions made as second references of pattern p in p does not match s ; starts search g included the pattern_items of pattern g included the pattern_items of pattern in the class <i>cc</i> ; managed the patte	de references to ves captured su sed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest tatchest longest techest longest term captures) aracter y (e.g. 4 tem in the pattern in t	captures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if default: 1]. **Classes* below) sequence (greedy). sequence (non-greedy). sequence (greedy). **Cob() for nested parentheses) ern m	io.input ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, mode	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file tility functions derr predefined file objects for stdin, stdout and stderr streams starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is "r", default) or to write data to (if mode is "w")
string.matcl Patterns at General patte cc cc* cc- cc+ cc? %n %bxy ^	h (s, p [, i]) nd pattern items em format: pattern matches a single of matches zero or n matches zero or n matches zero or or matches zero or or matches the n-th matches the balan anchors pattern to anchors pattern to	by r if r is a string (r can include called for each match and receive If r is a table, the captures are usubstitutions made as second restrums captures of pattern p in p does not match s ; starts search s : p the class could be considered in th	de references to ves captured su sed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest tatchest longest techest longest term captures) aracter y (e.g. 4 tem in the pattern in t	captures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if default: 1]. **Classes* below) sequence (greedy). sequence (non-greedy). sequence (greedy). **Cob() for nested parentheses) ern m	Simple I/O io.input ([file]) io.output ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, modelio.type (x)	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file stility functions derr predefined file objects for stdin, stdout and stderr streams
string.match Patterns at General patte cc cc* cc- cc+ cc? %n %bxy ^ \$ Captures (pattern) ()	h (s, p [, i]) nd pattern items em format: pattern, matches a single of matches zero or n matches zero or n matches zero or of matches zero or of matches the n-th matches the balan anchors pattern to anchors pattern to stores substring n stores current strii	by r if r is a string (r can include called for each match and receit If r is a table, the captures are usubstitutions made as second reterrors captures of pattern p in p does not match s ; starts search p in the class cc; match a capture in the class cc; match a capture in the class cc; match a capture distring (n = 19, see Patth and the class cc; match a capture distring, must be the first in the class cc; match a capture distring, must be the last item and the class cc; match a capture distring, must be the last item and position as capture %19 and position as capture	de references to ves captured su sed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest tatchest longest techest longest term captures) aracter y (e.g. 4 tem in the pattern in t	captures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if default: 1]. **Classes* below) sequence (greedy). sequence (non-greedy). sequence (greedy). **Cob() for nested parentheses) ern m	Simple I/O io.input ([file]) io.output ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, modelio.type (x)])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file retility functions derr predefined file objects for stdin, stdout and stderr streams starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is "r", default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object returns a file object for a temporary file (deleted when program ends)
string.match Patterns at General patte cc cc* cc- cc+ cc? %n %bxy ^ \$ Captures (pattern) ()	h (s, p [, i]) nd pattern item: em format: pattern, matches a single of matches zero or n matches zero or n matches zero or o matches zero or o matches the n-th o matches the balan anchors pattern to anchors pattern to	by r if r is a string (r can include called for each match and receit If r is a table, the captures are usubstitutions made as second reterrors captures of pattern p in p does not match s ; starts search p in the class cc; match a capture in the class cc; match a capture in the class cc; match a capture distring (n = 19, see Patth and the class cc; match a capture distring, must be the first in the class cc; match a capture distring, must be the last item and the class cc; match a capture distring, must be the last item and position as capture %19 and position as capture	de references to ves captured su sed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest tatchest longest techest longest term captures) aracter y (e.g. 4 tem in the pattern in t	captures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if default: 1]. **Classes* below) sequence (greedy). sequence (non-greedy). sequence (greedy). **Cob() for nested parentheses) ern m	Simple I/O io.input ([file]) io.output ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, modelio.type (x)])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file stility functions derr predefined file objects for stdin, stdout and stderr streams
string.match Patterns at General patte cc cc* cc+ ccc %n %bxy ^ \$ Captures (pattern) () Pattern che . %a %c	h (s, p [, i]) nd pattern items em format: pattern matches a single of matches zero or n matches zero or n matches zero or or matches zero or or matches the neth of matches the balan anchors pattern to stores substring m stores current stri aracter classes any character any letter any control character	by r if r is a string (r can include called for each match and receive If r is a table, the captures are usubstitutions made as second resulting returns captures of pattern p in p does not match s ; starts search s ; item [pattern_items] character in the class cc (see Pattern characters in the class cc ; matched the control of the class cc ; matched the class cc ; matc	de references to ves captured su sed as fields ir sult. string s (or the en at position i [ern character of atchest longest atchest longest the strongest the strongest term in the pattern in the pat	captures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if default: 1]. *Classes* below) sequence (greedy). a sequence (non-greedy). sequence (greedy). *Wb() for nested parentheses) ern *Topening parentheses any non-letter any non-control character	Simple I/O io.input ([file]) io.output ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, mode io.type (x) io.tmpfile () Note: unless otherwise s	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file retility functions derr predefined file objects for stdin, stdout and stderr streams starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is "r", default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object returns a file object for a temporary file (deleted when program ends)
string.match Patterns at General patte cc cc* cc- cc+ cc? %n %bxy ^ \$ Captures (pattern) () Pattern che . %a %c %d %d	h (s, p [, i]) nd pattern items ern format: pattern matches a single of matches zero or n matches zero or n matches zero or o matches the n-th o matches the balan anchors pattern to anchors pattern to stores substring m stores current strin aracter classes any character any letter any control chara- any digit any lowercase lett	by r if r is a string (r can include called for each match and receive of the reach match and received if r is a table, the captures are usubstitutions made as second reterms captures of pattern p in p does not match s ; starts search s . item [pattern_items] character in the class cc (see Pattern characters in the class cc; manore characters in the class cc; manore characters in the class cc; manore character in the class cc captured string (n = 19, see Pattern character in the class cc; manore character in the class cc captured string from character x to choost art of string, must be the first it of end of string, must be the last item that the pattern as capture %19 , and position as capture certain the class capture when the pattern as capture captured string pattern as capture certain the class capture certain pattern as capture when the pattern as capture certain the class capture certain the class cc; manded the pattern as capture when the pattern as capture certain the class cc; manded the pattern as capture captured the pattern as capture when the pattern as capture certain the class cc; manded the pattern as capture captured the pattern as capture when the pattern as capture captured the pattern as captured the	de references to ves captured su seed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest atchest shortes techest longest tem in the pattern in the p	acaptures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if default: 1]. *classes below) *sequence (greedy). *sequence (greedy). *sequence (greedy). *obb() for nested parentheses) *em *m *Topening parentheses any non-letter any non-control character any non-digit any non-(lowercase letter)	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and usio.stdin, io.stdout, io.stdin, io.stdout, io.stdin, io.tmpfile () Note: unless otherwise serror instead.	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file retility functions derr predefined file objects for stdin, stdout and stderr streams
string.match Patterns al General patte cc cc* cc- cc+ cc? %n %bxy Captures (pattern) () Pattern che %a %c %d %l %p %s	h (s, p [, i]) nd pattern items ern format: pattern matches a single of matches zero or n matches zero or n matches zero or o matches the n-th matches the balan anchors pattern to stores substring n stores current stri aracter classes any character any letter any control characany digit	by r if r is a string (r can include called for each match and receit If r is a table, the captures are usubstitutions made as second refetures captures of pattern p in p does not match s ; starts search p the pattern_items captures of pattern p in the class cc (see Pattern or character in the class cc; make the character in the class cc captured string (n = 19, see Pattern of Start of String, must be the first in the class cc captured string, must be the last item of the control of string, must be the last item of the control of string as capture %1 and the control of string position as capture control of the contro	de references to ves captured su sed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest atchest shortes techest longest term captures) aracter y (e.g. of the mat position in the pattern in the	acaptures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if default: 1]. **Classes* below) sequence (greedy). sequence (greedy). sequence (greedy). **Ob() for nested parentheses) erm **Topening parentheses any non-letter any non-control character any non-clowercase letter) any non-punctuation character any non-punctuation character any non-whitespace character	io.input ([file]) io.output ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, mode io.type (x) io.tmpfile () Note: unless otherwise s error instead. System interaction os.execute (cmd)	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file stility functions derr predefined file objects for stdin, stdout and stderr streams starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is "r", default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object returns a file object for a temporary file (deleted when program ends) tated, the I/O functions return nil and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system shell to execute the string cmd as a command; returns a system-dependent status code.
string.match Patterns al General patte cc cc* cc- cc+ cc? %n %bxy ^ \$ Captures (pattern) () Pattern che . %a %c %d %l %l %p %s %u	h (s, p [, i]) nd pattern items em format: pattern matches a single of matches zero or n matches zero or n matches zero or or matches the n-th matches the balan anchors pattern to anchors pattern to stores substring m stores current strin aracter classes any character any letter any control charac any digit any lowercase lett any punctuation of any whitespace of any uppercase lett	by r if r is a string (r can include called for each match and receive If r is a table, the captures are usubstitutions made as second restrums captures of pattern p in p does not match s ; starts search s ; item [pattern_items] character in the class cc (see Pattern or characters in the class cc; make the characters in the class cc captured string (n = 19, see Pattern ced string from character x to choost art of string, must be the first in the class in the character and the character as capture %1 9 and position as capture ceter the character the cha	de references to ves captured su sed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest tachest shortes the string s (ern captures) aracter y (e.g., strem in the pattern in the p	acaptures in the form %n). If r is a function r () is abstrings; it should return the replacement string, to the table. The function returns the number of whole match if p specifies no captures) or nil if default: 1]. **Classes* below) sequence (greedy). sequence (greedy). sequence (greedy). **Ob() for nested parentheses) ern **Topening parentheses* any non-letter any non-control character any non-digit any non-quinctuation character any non-punctuation character any non-whitespace character any non-(uppercase letter) any non-(uppercase letter)	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and to io.stdin, io.stdout, io.st io.popen ([prog [, mode io.type (x)]) io.tmpfile () Note: unless otherwise serror instead. System interaction os.execute (cmd) os.exit ([code]) os.getenv (var)	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file retility functions derr predefined file objects for stdin, stdout and stderr streams starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is "r", default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object returns a file object for a temporary file (deleted when program ends) tated, the I/O functions return nil and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system shell to execute the string cmd as a command; returns a system-dependent status code. terminates the program returning code [default: success] returns a string with the value of the environment variable var or nil if no such variable exists
string.match Patterns at General patter cc cc* cc+ ccc %n %bxy ^ \$ Captures (pattern) () Pattern che %a %c %d %l %p %s %u %w %x	h (s, p [, i]) nd pattern item: em format: pattern. matches a single of matches zero or n matches zero or n matches zero or or matches zero or or matches the nether matches the balan anchors pattern to anchors pattern to stores substring matches the substring matches the substring matches the pattern to anchors pattern any letter any letter any letter any uncutation cany whitespace clany uppercase lettern any alphanumericany hexadecimal	by r if r is a string (r can include called for each match and receive If r is a table, the captures are usubstitutions made as second reterrors captures of pattern p in p does not match s ; starts search s . Item [pattern_items] character in the class cc (see Patternore characters in the class cc; manager characters in the class cc captured string (n = 19, see Patternore distring from character x to choost art of string, must be the first in the class cc; manager characters in the class cc; manager characters in the class cc; manager character string from character x to choost and consideration of string, must be the last item atching pattern as capture %1 In the class cc; manager character the character character character character digit	de references to ves captured su sed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest atchest shortest the string s (e.g., or the mat position i [ern character of atchest shortest the stringest atchest longest atchest longest atchest longest the mat pattern captures) aracter y (e.g., or the mat pattern in the pattern in th	any non-letter any non-control character any non-clowercase letter) any non-letter any non-clowercase letter) any non-clowercase letter) any non-clowercase letter) any non-cuppercase letter) any non-cuppercase letter) any non-chexadecimal digit)	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, mode io.type (x) io.tmpfile () Note: unless otherwise s error instead. System interaction os.execute (cmd) os.exit ([code])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file tility functions derr predefined file objects for stdin, stdout and stderr streams starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is "r", default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object returns a file object for a temporary file (deleted when program ends) tated, the I/O functions return nil and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system shell to execute the string cmd as a command; returns a system-dependent status code. terminates the program returning code [default: success] returns a string with the value of the environment variable var or nil if no such variable exists sets the locale described by string s for
string.match Patterns at General patte cc cc* cc- cc+ cc? %n %bxy ^ \$ Captures (pattern) () Pattern che . %a %c %d %l %p %s %u %s %u %x %z %x	h (s, p [, i]) nd pattern items em format: pattern, matches a single of matches zero or n matches zero or n matches zero or of matches zero or of matches the n-th of matches the balant anchors pattern to any letter any purctuation cany hypercase letter any alphanumeric any hexadecimal the byte value zer if x is a symbol the	by r if r is a string (r can include called for each match and received at the captures are usubstitutions made as second reserved returns captures of pattern p in p does not match s; starts search p does not match a class cc; match characters in the class cc; match characters in the class cc; match characters in the class cc; match character in the class cc; match character in the class cc; match character string, must be the first in the class cc; match p does not match in growth p does not	de references to ves captured su seed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest atchest shortes techest longest techest longest tem in the pattern in the	any non-letter any non-control character any non-clowercase letter) any non-clowercase letter) any non-clowercase letter) any non-cuppercase letter) any non-cuppercase letter) any non-cuppercase letter) any non-chexadecimal digit) any non-chexadecimal digit) any non-zero character if x not in ^\$()%.[]*+-? the character itself	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and usio.stdin, io.stdout, io.stio.popen ([prog [, mode io.type (x)] io.tmpfile () Note: unless otherwise serror instead. System interaction os.execute (cmd) os.exit ([code]) os.getenv (var) os.setlocale (s [, c])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file tility functions derr predefined file objects for stdin, stdout and stderr streams starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is "r", default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object returns a file object for a temporary file (deleted when program ends) tated, the I/O functions return nil and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system shell to execute the string cmd as a command; returns a system-dependent status code. terminates the program returning code [default: success] returns a string with the value of the environment variable var or nil if no such variable exists sets the locale described by string s for category c: "all", "collate", "ctype", "monetary", "numeric" or "time" [default: "all"]; returns the name of the locale or nil if it can't be set. deletes the file fn; in ca
string.match Patterns at General patte cc cc* cc- cc+ cc? %n %bxy ^ \$ Captures (pattern) () Pattern che . %a %c %d %l %p %s %u %w %x %z	h (s, p [, i]) nd pattern items em format: pattern, matches a single of matches zero or n matches zero or n matches zero or of matches zero or of matches the n-th of matches the balant anchors pattern to any letter any purctuation cany hypercase letter any alphanumeric any hexadecimal the byte value zer if x is a symbol the	by r if r is a string (r can include called for each match and receive If r is a table, the captures are usubstitutions made as second refereturns captures of pattern p in p does not match s; starts search p does not match class cc; match p does not match class cc; match p does not match s capture does not character in the class cc; match p does not match p doe	de references to ves captured su seed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest atchest shortes techest longest techest longest tem in the pattern in the	any non-letter any non-control character any non-clowercase letter) any non-punctuation character any non-uppercase letter) any non-letter any non-control character any non-uppercase letter) any non-uppercase letter) any non-letterany non-uppercase letter) any non-uppercase letter) any non-zero character	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and usio.stdin, io.stdout, io.stio.popen ([prog [, mode io.type (x)] io.tmpfile () Note: unless otherwise serror instead. System interaction os.execute (cmd) os.exit ([code]) os.getenv (var) os.setlocale (s [, c])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file tility functions derr predefined file objects for stdin, stdout and stderr streams starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is "r", default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object returns a file object for a temporary file (deleted when program ends) tated, the I/O functions return nil and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system shell to execute the string cmd as a command; returns a system-dependent status code. terminates the program returning code [default: success] returns a string with the value of the environment variable var or nil if no such variable exists sets the locale described by string s for category c: "all", "collate", "ctype", "monetary", "numeric" or "time" [default: "all"]; returns the name of the locale or nil if it can't be set. deletes the file file in ca
string.match Patterns al General patte cc cc* cc- cc+ cc? %n %bxy ^ \$ Captures (pattern) () Pattern che %a %c %d %l %p %s %u %w %x [set] Pattern exe	h (s, p [, i]) nd pattern items em format: pattern, matches a single of matches zero or n matches zero or n matches one or m matches the n-th matches the balan anchors pattern to anchors pattern to stores substring m stores current strice ary control character any letter any control character any letter any punctuation of any whitespace of any uppercase lett any uppercase lett any letter any letter any letter any punctuation of any whitespace of any uppercase lett any letter any letter any punctuation of any whitespace of any uppercase lett any letter any letter any letter any letter any punctuation of any whitespace of any uppercase lett any alphanumeric any hexadecimal the byte value zer if x is a symbol th any character in a be a range [c1-c2]	by r if r is a string (r can include called for each match and received if r is a table, the captures are usubstitutions made as second references of pattern p in p does not match s; starts search p does not match a class cc; match characters in the class cc; match characters in the class cc; match characters in the class cc; match character in the class cc; match character in the class cc; match character string, must be the first in the class cc; match p does not match in the class cc; match p does not p does no	de references to ves captured su seed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest atchest shortes techest longest techest longest tem in the pattern in the	any non-letter any non-control character any non-clowercase letter) any non-letter any non-clowercase letter) any non-uncutation character any non-uncutation character any non-uncutation character any non-letter) any non-letter) any non-clowercase letter) any non-uncutation character any non-character if x not in ^\$()%.[]*+-? the character itself any character not in set	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, mode io.type (x) io.tmpfile () Note: unless otherwise s error instead. System interaction os.execute (cmd) os.exit ([code]) os.getenv (var) os.setlocale (s [, c]) os.remove (fn) os.rename (of, nf) os.tmpname ()	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file, usage as file:write() flushes any data still held in buffers to the default output file writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file writes to the default output file, usage as file:write() flushes any flushes any flushed in buffers to the default output file writes to the default output file, usage as file:write() flushes any flushes any flushed in buffers to the default output file writes to the default output file, usage as file:write() flushes any flushed in buffers to the default output file writes to the default output file, usage as file:write() flushes any flushed in buffers to the default output file writes to the default output file, usage as file:write() flushes any flushed in buffers to file in fil
string.match Patterns at General patter cc cc* cc- cc+ cc? %n %bxy ^ \$ Captures (pattern) () Pattern che . %a %c %d %l %b %s %u %w %x %s %c %ting.find('string	h (s, p [, i]) nd pattern items em format: pattern matches a single of matches zero or n matches zero or n matches zero or n matches the n-th matches the n-th matches the n-th matches the substring m stores substring m stores current strin aracter classes any character any letter any control character any letter any lowercase lett any punctuation of any whitespace of any uppercase lett any alphanumeric any hexadecimal the byte value zer if x is a symbol th any character in a be a range [c1-c2] amples "Lua is great!", "i" "Lua is great!", "i"	by r if r is a string (r can include called for each match and receive If r is a table, the captures are usubstitutions made as second references of pattern p in p does not match s; starts search p does not match p does not mat	de references to ves captured su sed as fields in sult. string s (or the in at position i [ern character of atchest longest atchest longest atchest shortes techest longest term in the pattern in the	any non-letter any non-control character any non-clowercase letter) any non-clowercase letter) any non-clowercase letter) any non-cuppercase letter) any non-cuppercase letter) any non-cuppercase letter) any non-chexadecimal digit) any non-chexadecimal digit) any non-zero character if x not in ^\$()%.[]*+-? the character itself	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, mode io.type (x) io.tmpfile () Note: unless otherwise s error instead. System interaction os.execute (cmd) os.exit ([code]) os.getenv (var) os.setlocale (s [, c]) os.remove (fn) os.rename (of, nf) os.tmpname () Date/time os.clock ()	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file ttility functions derr predefined file objects for stdin, stdout and stderr streams starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is "r", default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file on fil if x is not a file object returns a file object for a temporary file (deleted when program ends) tated, the I/O functions return file and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system shell to execute the string cmd as a command; returns a system-dependent status code. terminates the program returning code [default: success] returns a string with the value of the environment variable var or nil if no such variable exists sets the locale described by string s for category e: "all", "collate", "etype", "monetary", "numerie" or "time" [default: "all"]; returns the name of the locale or nil if it can't be set. deletes the file fir, in
string.match Patterns at General patte cc cc* cc- cc+ cc? %n %bxy ^ \$ Captures (pattern) () Pattern cha . %a %c %d %l %p %s %u %w %x [set] Pattern ex string.find('string.gsub('string	nd pattern itemsem format: pattern matches a single of matches zero or not matches zero or not matches zero or not matches the n-th of matches the n-th of matches the n-th of matches the pattern to anchors pattern any letter any letter any lowercase lettern any uppercase lettern any uppercase lettern any uppercase lettern any hexadecimal the byte value zer if x is a symbol the any character in a be a range [c1-c2] amples "Lua is great!", "Itua is great!", "I'Lua is	by r if r is a string (r can include called for each match and received at the captures are usubstitutions made as second results results as second results	le references to ves captured su sed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest atchest shortes the tern captures) arracter y (e.g. 4 tern in the pattern i	any non-letter any non-control character any non-clowercase letter) any non-lowercase letter) any non-lowercase letter) any non-punctuation character any non-control characte	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, mode io.type (x) io.tmpfile () Note: unless otherwise s error instead. System interaction os.execute (cmd) os.exit ([code]) os.getenv (var) os.setlocale (s [, c]) os.remove (fn) os.rename (of, nf) os.tmpname ()	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file flushes any data still held in buffers to the default output file starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is "r", default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object returns a file object for a temporary file (deleted when program ends) tated, the I/O functions return nil and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system shell to execute the string cmd as a command; returns a system-dependent status code. terminates the program returning code [default: success] returns a string with the value of the environment variable var or nil if no such variable exists sets the locale described by string s for category c: "all", "collate", "ctype", "monetary", "numeric" or "time" [default: sulr"]; returns the name of the locale or nil if it can't be set. deletes the file fn; in case of error returns ni
string.match Patterns at General patter cc cc* cc+ cc? %n %bxy Separatern () Pattern che %a %c %d %l %p %s %u %w %x %z %x [set] Pattern exa string.find(' string.gsub(' string.gsub(' string.gsub(' string.gsub(') string.gsub(')	h (s, p [, i]) nd pattern item: em format: pattern, matches a single of matches zero or n matches zer	by r if r is a string (r can include called for each match and received a string in a table, the captures are usubstitutions made as second references captures of pattern p in p does not match s; starts search p does not match class cc; match p does not match class cc; match p does not p does not match p does not p does	le references to ves captured su seed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest atchest shortes techest longest tem in the pattern in the p	any non-letter any non-control character any non-clowercase letter) any non-letter any non-lowercase letter) any non-letter any non-lowercase letter) any non-letter any non-control character any non-character any non-clowercase letter) any non-letter any non-clowercase letter) any non-letter any non-control character any non-character any non-character any non-character any non-character any non-character any non-character any non-digit any non-character any non-digit any non-character if x not in ^\$()%.[]*+-? the character itself any character not in set	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, mode io.type (x) io.tmpfile () Note: unless otherwise s error instead. System interaction os.execute (cmd) os.exit ([code]) os.getenv (var) os.setlocale (s [, c]) os.remove (fn) os.rename (of, nf) os.tmpname () Date/time os.clock ()	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object or beject) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fin for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fine given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file ittility functions derr predefined file objects for stdin, stdout and stderr streams starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is "r", default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object returns a file object for a temporary file (deleted when program ends) tated, the I/O functions return nil and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system shell to execute the string cmd as a command; returns a system-dependent status code. terminates the program returning code [default: success] returns a string with the value of the environment variable var or nil if no such variable exists sets the locale described by string s for category c: "all", "collate", "ctype", "monetary", "numeric" or "time" [default: "all"]; returns the name of the locale or nil if it can't be set. deletes the file fit, in case of error returns nil and error description. returns a string usable as n
string.match Patterns at General patter cc cc* cc- cc+ cc? %n %bxy \$ Captures (pattern) () Pattern che %a %c %d %l %p %s %u %w %x [set] Pattern exa string.find('string.gsub(string.	h (s, p [, i]) nd pattern items em format: pattern, matches a single of matches zero or n matches zero or n matches one or m matches the n-th matches the balan anchors pattern to anchors pattern to stores substring m stores current strice any control charae any digit any lowercase lett any punctuation of any whitespace of any uppercase lett any alphanumeric any hexadecimal the byte value zer if x is a symbol th any character in a be a range [c1-c2] amples "Lua is great!", "' "Lua is great!", "	by r if r is a string (r can include called for each match and receive If r is a table, the captures are usubstitutions made as second refereturns captures of pattern p in p does not match s; starts search p does not match class cc; match p does not match class cc; match p does not match p does not match p does not match p does not p does not match p does not match p does not p does not match p does not p	le references to ves captured su sed as fields in sult. string s (or the in at position i [ern character of atchest longest atchest longest atchest shortes techest longest term in the pattern in the	any non-letter any non-control character any non-clowercase letter) any non-letter any non-lowercase letter) any non-letter any non-lowercase letter) any non-letter any non-control character any non-character any non-clowercase letter) any non-letter any non-clowercase letter) any non-letter any non-control character any non-character any non-character any non-character any non-character any non-character any non-character any non-digit any non-character any non-digit any non-character if x not in ^\$()%.[]*+-? the character itself any character not in set	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, mode io.type (x) io.tmpfile () Note: unless otherwise s error instead. System interaction os.execute (cmd) os.exit ([code]) os.getenv (var) os.setlocale (s [, c]) os.remove (fn) os.rename (of, nf) os.tmpname () Date/time os.clock ()	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) (default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file titility functions derr predefined file objects for stdin, stdout and stderr streams predefined file objects for stdin, stdout and stderr streams predefined file objects for a separate process and returns a file handle that you can use to read data from (if mode is "r", default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object for a temporary file (deleted when program ends) tated, the I/O functions return nil and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system-shell to execute the string cmd as a command; returns a system-dependent status code. terminates the program returning code [default: success] returns a string with the value of the environment variable var or nil if no such variable exists sets the locale described by string s for category c: "all", "colla
string.match Patterns at General patter cc cc* cc- cc+ cc? %n %bxy S Captures (pattern) () Pattern che %a %c %d %l %p %s %u %w %x %z %x [set] Pattern exe string.find(' string.gsub(string.gsub	h (s, p [, i]) nd pattern items em format: pattern matches a single of matches zero or n matches zero or n matches one or m matches the n-th matches the balan anchors pattern to anchors pattern to anchors pattern to stores substring m stores current strict aracter classes any character any letter any control character any letter any lowercase lett any punctuation of any whitespace of any uppercase lett any alphanumeric any hexadecimal the byte value zer if x is a symbol th any character in a be a range [c1-c2] amples "Lua is great!", "	by r if r is a string (r can include called for each match and receive If r is a table, the captures are usubstitutions made as second refereturns captures of pattern p in p does not match s; starts search p does not match class cc; match characters in the class cc; match character in the class cc; match character string from character x to choost at of string, must be the first in the p does not possible the string position as capture with the class cc; match in graph p does not	le references to ves captured su sed as fields in sult. string s (or the in at position i [ern character of atchest longest atchest longest atchest shortes techest longest term in the pattern in the	any non-letter any non-control character any non-clowercase letter) any non-letter any non-lowercase letter) any non-letter any non-lowercase letter) any non-letter any non-control character any non-character any non-clowercase letter) any non-letter any non-clowercase letter) any non-letter any non-control character any non-character any non-character any non-character any non-character any non-character any non-character any non-digit any non-character any non-digit any non-character if x not in ^\$()%.[]*+-? the character itself any character not in set	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, mode io.type (x) io.tmpfile () Note: unless otherwise s error instead. System interaction os.execute (cmd) os.exit ([code]) os.getenv (var) os.setlocale (s [, c]) os.remove (fn) os.rename (of, nf) os.tmpname () Date/time os.clock () os.time ([tt])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file, usage as file:write() flushes any data still held in buffers to the default output file writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file tillity functions derr predefined file objects for stdin, stdout and stderr streams starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is "r", default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object returns a file object for a temporary file (deleted when program ends) tated, the I/O functions return nil and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system shell to execute the string cmd as a command; returns a system-dependent status code. terminates the program returning code (default: success) returns a string with the value of the environment variable var or nil if no such variable exists sets the locale described by strings for category c: "all", "collate", "ctype", "monetary", "numeric" or "time" (default: "all"); returns the name of the locale or nil if it can't be set. deletes the file fn; in case of error re
string.match Patterns at General patter cc cc* cc+ cc- %n %bxy ^ \$ Captures (pattern) () Pattern cha %c %d %l %p %s %u %w %x %z %x [set] Pattern exa string.find(' string.gsub(string.gsub()	h (s, p [, i]) nd pattern items em format: pattern, matches a single of matches zero or n matches zero or n matches one or m matches the n-th matches the balan anchors pattern to anchors pattern to stores substring m stores current strice any control charae any digit any lowercase lett any punctuation of any whitespace of any uppercase lett any alphanumeric any hexadecimal the byte value zer if x is a symbol th any character in a be a range [c1-c2] amples "Lua is great!", "' "Lua is great!", "	by r if r is a string (r can include called for each match and receive If r is a table, the captures are usubstitutions made as second refereturns captures of pattern p in p does not match s; starts search p does not match class cc; match characters in the class cc; match characters in the class cc; match characters in the class cc; match character in t	le references to ves captured su sed as fields in sult. string s (or the in at position i [ern character of atchest longest atchest longest atchest shortes techest longest term in the pattern in the	any non-letter any non-control character any non-control character any non-lowercase letter) any non-punctuation character any non-punctuation character any non-punctuation character any non-lowercase letter) any non-letter any non-digit any non-letter any non-digit any non-digit any non-digit any non-digit any non-devace character any non-digit any character if x not in %\$()%.[]*+-? the character itself any character not in set 6	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, mode io.type (x) io.tmpfile () Note: unless otherwise s error instead. System interaction os.execute (cmd) os.exit ([code]) os.getenv (var) os.setlocale (s [, c]) os.remove (fn) os.rename (of, nf) os.tmpname () Date/time os.clock () os.time ([tt])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file whin name fn for reading and returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file flushes any data still held in buffers to the default output file writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file writes to the default output file, usage as file:write() flushes any file file objects for stdin, stdout and stderr streams starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is "", default) or to write data to (if mode is "w") returns a hile object for a temporary file (deleted when program ends) tated, the I/O functions return nil and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system shell to execute the string cmd as a command; returns a system-dependent status code. terminates the program returning code [default: success] returns a string with the value of the environment variable var or nil if no such varia
string.match Patterns at General patter cc cc* cc- cc+ cc? %n %bxy \$ Captures (pattern) () Pattern che %a %c %d %l %p %s %u %w %x %z %x [set] Pattern exa string.find(' string.gsub(string.gsub(string.gsub(string.gsub(string.gsub(string.gsub)string.gsub(string.gsub)string.gsub(string.gsub)string.gsub(string.gsub)string.gsub(string.gsub(string.gsub)string.gsub(string.gsub)string.gsub(s	h (s, p [, i]) nd pattern item: em format: pattern, matches a single of matches zero or n matches zer	by r if r is a string (r can include called for each match and received at the captures are usubstitutions made as second references are returns captures of pattern p in p does not match s; starts search proceeds are usually assumed to the class cc (see Pattern certain the class cc; match contacters in the class cc; match contacters are character in the class cc; match contacters are contacted string from character at the class cc; match contacters are contacted string, must be the first in the class cc; match contacters are character as capture with the contacter are contacter as capture with the class cc; match contacters are contacter as capture with the class cc; match contacters are contacter as capture with the class cc; match contacters are contacter as capture with the class cc; match contacters are contacter as capture with the class cc; match contacters are contacter as capture with the class cc; match contacters are contacter as capture with the class cc; match contacters are contacted as capture with the class cc; match class cc; match contacters are contacted as capture with the class cc; match clas	le references to ves captured su sed as fields in sult. string s (or the in at position i [ern character of atchest longest atchest longest atchest shortes techest longest term in the pattern in the	any non-letter any non-control character any non-control character any non-lowercase letter) any non-punctuation character any non-punctuation character any non-punctuation character any non-lowercase letter) any non-letter any non-digit any non-letter any non-digit any non-digit any non-digit any non-digit any non-devace character any non-digit any character if x not in %\$()%.[]*+-? the character itself any character not in set 6	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, mode io.type (x) io.tmpfile () Note: unless otherwise s error instead. System interaction os.execute (cmd) os.exit ([code]) os.getenv (var) os.setlocale (s [, c]) os.remove (fn) os.rename (of, nf) os.tmpname () Date/time os.clock () os.time ([tt])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object, the current one if no file given; raises error on failure. closes file (a file object) [default; closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file flushes any data still held in buffers to the default output file starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is ""," default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object returns a file object for a temporary file (deleted when program ends) tated, the I/O functions return nil and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system shell to execute the string cmd as a command; returns a system-dependent status code. terminates the program returning code [default: success] returns a string with the value of the environment variable var or nil if no such variable exists sets the locale described by strings for category c: "all", "collate", "ctype", "monetary", "numeric" or "time" (default: "all"); returns the name of the locale o
string.match Patterns at General patter cc cc* cc+ cc? %n %bxy \$ Captures (pattern) () Pattern che %a %c %d %l %p %s %u %w %x %z %x [set] Pattern exa string.find(' string.gsub(strin	h (s, p [, i]) nd pattern itemsem format: pattern, matches a single of matches zero or n matches zero or n matches zero or or matches zero or or matches the n-th matches the n-th matches pattern to anchors pattern any letter any elucitor any hexacter any uppercase letten any uppercase letten any alphanumeric any hexadecimal the byte value zer if x is a symbol the any character in a be a range [c1-c2] amples "Lua is great!", " ("Lua is great!", ")	by r if r is a string (r can include called for each match and received a substitutions made as second research returns captures of pattern p in p does not match s; starts search p does not match p does not match class cc; match p does not characters in the class cc; match p does not p does not match p does not p does n	le references to ves captured su seed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest atchest longest tendest shortes to the strongest tendest longest tendest in the pattern i	any non-letter any non-control character any non-lowercase letter) any non-unctuation character any non-unctuation character any non-letter any non-control character any non-lowercase letter) any non-letter any non-control character any non-whitespace character any non-letter) any non-letter any non-digit any non-letter) any non-to-precade letter) any non-to-precade let	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.st io.popen ([prog [, mode io.type (x) io.tmpfile () Note: unless otherwise s error instead. System interaction os.execute (cmd) os.exit ([code]) os.getenv (var) os.setlocale (s [, c]) os.remove (fn) os.rename (of, nf) os.tmpname () Date/time os.clock () os.time ([tt])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object or a file name; in the latter case the file is opened for writing in text mode. Returns a file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fin for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fin is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file flushes any data still held in buffers to the default output file tility functions derr prodefined file objects for stdin, stdout and stderr streams starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is "r", default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object returns a file object for a temporary file (deleted when program ends) tated, the I/O functions return nil and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system shell to execute the string cmd as a command; returns a system-dependent status code. terminates the program returning code [default: success] returns a string with the value of the environment variable var or nil if no such variable exists sets the locale described by string s for category e: "all", "collate", "ctype", "monetary", "numeric" or "time" [default: "all"]; returns the name of
string.match Patterns at General patter cc cc* cc- cc+ cc? %n %bxy \$ Captures (pattern) () Pattern che %a %c %d %l %p %s %u %w %x %z %x [set] Pattern exa string.find(' string.gsub(string.gsub(string.gsub(string.gsub(string.gsub(string.gsub)string.gsub(string.gsub)string.gsub(string.gsub)string.gsub(string.gsub)string.gsub(string.gsub(string.gsub)string.gsub(string.gsub)string.gsub(st	h (s, p [, i]) nd pattern itemsem format: pattern, matches a single of matches zero or n matches zero or n matches zero or or matches zero or or matches the n-th matches the n-th matches pattern to anchors pattern any letter any elucitor any hexacter any uppercase letten any uppercase letten any alphanumeric any hexadecimal the byte value zer if x is a symbol the any character in a be a range [c1-c2] amples "Lua is great!", " ("Lua is great!", ")	by r if r is a string (r can include called for each match and receive of the reach match and receive of the returns captures of pattern p in p does not match s; starts search p does not match class cc; match characters in the class cc; match characters in the class cc; match character i	de references to ves captured su sed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest atchest shortes techest longest term in the pattern in the p	any non-letter any non-control character any non-lowercase letter) any non-lunctuation character any non-unclustion character any non-whitespace character any non-whitespace character any non-unclexadecimal digit) any non-zero character if x not in ^\$()%.[]*+-? the character itself any character not in set	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.stio.popen ([prog [, modelio.type (x)]) io.tmpfile () Note: unless otherwise serror instead. System interaction os.execute (cmd) os.exit ([code]) os.getenv (var) os.setlocale (s [, c]) os.remove (fn) os.rename (of, nf) os.tmpname () Date/time os.clock () os.time ([tt])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file flushes any data still held in buffers to the default output file starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is ""," default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object returns a file object for a temporary file (deleted when program ends) tated, the I/O functions return nil and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system shell to execute the string cmd as a command; returns a system-dependent status code. terminates the program returning code [default: success] returns a string with the value of the environment variable var or nil if no such variable exists sets the locale described by strings for category c: "all", "collate", "ctype", "monetary", "numeric" or "time" (default: "all"); returns the name of the locale or nil
string.match Patterns at General patter cc cc* cc+ cc? %n %bxy \$ Captures (pattern) () Pattern che %a %c %d %l %p %s %u %w %x %z %x [set] Pattern exa string.find(' string.gsub(strin	th (s, p [, i]) Indepattern items em format: pattern matches a single of matches zero or not matches zero or not matches the n-th of matches th	by r if r is a string (r can include called for each match and receiviff r is a table, the captures are usubstitutions made as second reference returns captures of pattern p in p does not match s; starts search p does not match class cc; match characters in the class cc; match character string from character x to choost at of string, must be the first in the position of string, must be the last item provided by the position as capture with	le references to ves captured su sed as fields in sult. string s (or the mat position i [ern character of atchest longest atchest longest atchest shortest to the string s (or the mat position i [ern character of atchest longest atchest longest atchest longest atchest longest atchest longest tern captures) arracter y (e.g., strength of the string in the pattern in the patter	any non-letter any non-control character any non-lowercase letter) any non-lunctuation character any non-unclustion character any non-whitespace character any non-whitespace character any non-unclexadecimal digit) any non-zero character if x not in ^\$()%.[]*+-? the character itself any character not in set	io.input ([file]) io.close ([file]) io.close ([file]) io.read (formats) io.lines ([fn]) io.write (values) io.flush () Standard files and u io.stdin, io.stdout, io.stio.popen ([prog [, modelio.type (x)]) io.tmpfile () Note: unless otherwise serror instead. System interaction os.execute (cmd) os.exit ([code]) os.getenv (var) os.setlocale (s [, c]) os.remove (fn) os.rename (of, nf) os.tmpname () Date/time os.clock () os.time ([tt])	sets file as default input file; file can be either an open file object or a file name; in the latter case the file is opened for reading in text mode. Returns a file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object, the current one if no file given; raises error on failure. sets file as default output file (the current output file is not closed); file can be either an open file object, the current one if no file given; raises error on failure. closes file (a file object) [default: closes the default output file] reads from the default input file, usage as file:read() opens the file with name fn for reading and returns an iterator function to read line by line; the iterator closes the file when finished. If no fn is given, returns an iterator reading lines from the default input file. writes to the default output file, usage as file:write() flushes any data still held in buffers to the default output file flushes any data still held in buffers to the default output file starts program prog in a separate process and returns a file handle that you can use to read data from (if mode is ""," default) or to write data to (if mode is "w") returns the string "file" if x is an open file, "closed file" if x is a closed file or nil if x is not a file object returns a file object for a temporary file (deleted when program ends) tated, the I/O functions return nil and an error message on failure; passing a closed file object raises an The operating system library [os] calls a system shell to execute the string cmd as a command; returns a system-dependent status code. terminates the program returning code [default: success] returns a string with the value of the environment variable var or nil if no such variable exists sets the locale described by strings for category c: "all", "collate", "ctype", "monetary", "numeric" or "time" (default: "all"); returns the name of the locale or nil

file:lines ()

returns a value from **file** for each of the passed *formats*: "*n" = reads a number, "*a" = reads the whole **file** as a string from current position (returns "" at end of file), "*l" = reads a line (**nil** at end of file) [default], *n* = reads a string of up to *n* characters (**nil** at end of file) returns an iterator function for reading **file** line by line; the iterator does not close the file when finished.