

TVfacilities.rtf Breakdown of facilities in Tabula Vigilans (AE)

[To help analyse the task of recreating Tabula Vigilans in Python]

Mathematical Functions (these should be covered in Python; equivalents for 'dimensions' and 'dimsize' ?)

abs(x)	absolute value of x
arccosine(x)	the arccosine of x
arcsine(x)	the arcsine of x
arctangent(x)	the arctangent of x
arg(n)	the numerical value of the nth commandline argument
args(n)	the string value of the nth commandline argument
cosine(x)	the cosine of x
dimensions(TABLE)	the number of dimensions a table has
dimsize(TABLE, dimension_number)	the size of the dimension indicated
int(x)	the integer value of x
gamma()	returns a number between 0 and 1 representing Gamma probability
gauss()	returns a number between 0 and 1 representing Gauss probability
log10(x)	the base 10 logarithm of x
natlog(x)	the natural logarithm of x
power(x, n)	x to the nth power
rand()	a random number between 0 and 1
random(x, y)	a random number between x and y
round(x)	x rounded to the nearest integer
sine(x)	the sine of x
sqrt(x)	the square root of x
tangent(x)	the tangent of x
try(rule)	the status of executing the rule

Control Flow (most of this is covered in Python; not so sure about 'trigger', 'wait' or 'spawn'; there is an equivalent for 'system')

break	break out of a 'for' or 'while' loop with or without using a label
call	move between procedures
cls	clears the screen
continue	continue with a 'for' or 'while' loop with or without using a label
end	ends performance
for	counter-controlled action sequence
if / else	conditional branch (< > == != <= >= &&)
loop	return control-flow to the start of the current procedure
return	without a label, this means return to the last procedure; with a label, it means search for the most recent call with that label, and return to it
switchon/case	to avoid chains of if...then...else statements
trigger	set a trigger cell to activate when a timevalue has elapsed
wait	complete stay of execution for the duration of the input cell
while	condition-controlled action sequence
#include	a way to combine different TV scripts
system	create a system call to an external program
spawn	spawn a new external program process to run in parallel with the TV script

Tables (The table functions have to be studied to see which are covered by normal array/list handling, and which encapsulate musically useful operations involving more than one table, and how these would be realised in Python.)

Tables can be sized (e.g., [20]) or, if read from an external file with fill_table, unsized (e.g., []); they can be multi-dimensional (e.g., five rows and two columns: [5][2])

Tables are declared outside the main loop, which means that they cannot be resized dynamically, which is one of the main limitations of TV.

Re indexing and fractional indexing

- indexing supports x++ or x-- (automatically increment / decrement on next pass of the loop)
- negative numbers can be used as indices (e.g., [-1])
- indices are used to access and change specific index values in a multi-dimensional table
- operations between tables are in fact using table pointers
- TV supports fractional indexing, i.e., index values lying between 0 and 1; for example, 0.5 would be half-way through the number of items in a table; it is notated [|x|]

compare	compare two input tables and write to an output table
copy_table	copy the contents of one table to another
dimensions	return the number of dimensions of a table
dimsize	return the number of cells in a table dimension (specifying the dimension)
embed	embed two one-dimensional tables
fill_table	read in the values in a file into a declared table
fold	time-based (multiplicative) embedding of one-dimensional tables
generate table	output table results from increment / decrement operations on the input
interp_table	interpolates a value between each corresponding value in two tables
mult_table table	multiply each value in a table by its corresponding index value in 2nd
offset_table	offset the contents of a table by a value
perm	permute the contents of a 1-D table, randomly or by step or skip
scale_table	scale each value in a table by a constant value
shift	shift contents of a one-dimensional table one place to the left or right
sort	sort a table into ascending or descending order
subst	place in an output table values drawn statistically from either of two input tables
sum_table	sum successive values in two input tables
table	declare a table, with or without sized dimensions
xad / xar table	extract adjacent differences / ratios from input table and write to output

MIDI (most of this should be handled by a MIDI library; perhaps midichord and schedule are TV-specific)

control-out	send a MIDI control message
midichord	output a chord to designated MIDI channel
midiecho	output immediately (echo) a MIDI event
midiin	collect and store any available MIDI input data; includes note, pitchbend, aftertouch and controller events
midiout	output a MIDI message (chan, note velocity, duration [,num_notes]); usually used with try(), as in 'x = try(midiout ...)'
midiset	set instruments to specific MIDI channels
pitchbend	send a MIDI pitchbend message
schedule	schedule a MIDI event for later performance

Text handling (messages to console, opening, reading from and writing to external files; use to create output text or breakpoint files)

fill_table	fill a declared table with data from named external file
storefile [n]	open named file for writing to; number additional files 1, 2 etc.
stori / store	store integer / store floating point values in first (unnumbered) file
storf n	use this to store values in file n (to precision specified by store_digits)
storstr (newline)	store a string (contained in double quotes); accepts \t (tab) and \n
store_digits	decimal precision to use with the numbered storefiles
close_storefiles it	close all open storefiles (you can write to a file, close it, and then re-open and access its data with fill_table ...)
messag1	print message (contained in double quotes) only the first time it is called; (messages cannot display the contents of a variable)
message	print the message again as many times as it is called
probi / probe	display an integer or floating point value on the console
print	print input argument to the console, with optional formatting fields

cls clear the screen

Cells (specific operations on)

alllocked	test whether all of a list of cells are locked (use with try())
anylocked	test whether any of a list of cells are locked (use with try())
lock	lock one or more cells (variable cannot be changed while cell is locked)
unlock	unlocks previously locked cells
copy	copy the value of an input cell to one or more output cells
swap	swap the values of two cells
local by	declare a cell to be local (private) to the current procedure (they are global default)

Arguments (from command line; can TV test whether the command line has the correct number of arguments required by a given TV script?)

arg	employ a value passed from the command line (e.g. $x = \text{arg}(2)$)
args	employ a string value passed from the command line (e.g. a filename)
showargs	display a command line argument string (e.g., showargs args(1))

And some others

fail	rule used for testing and debugging
lin	create a linear time-varying output between two specified values, with a direction option for min -> max or max to min
seg	create a linear time-varying output between values supplied by the user
lintrans	implements a combined multiply and add in a single operation
mouse	outputs normalised coordinates of the mouse position
pop	fractal algorithm which employs an output cell and an input value
time	sets a real-time counter