**RBC Test Cases**

**Vector Component**

**Authors:**

Sam Green

Nick Hudson

Stanton Sievers

Jarrod Stormo

# Table of Contents

[Vector Append 3](#_Toc216518472)

[Vector Arithmetic 3](#_Toc216518473)

[Vector Binread 5](#_Toc216518474)

[Vector Clear 7](#_Toc216518475)

[Vector Create 8](#_Toc216518476)

[Vector Delete 11](#_Toc216518477)

[Vector Destroy 12](#_Toc216518478)

[Vector Dup 12](#_Toc216518479)

[Vector Expr 13](#_Toc216518480)

[Vector Index 20](#_Toc216518481)

[Vector Length 21](#_Toc216518482)

[Vector Merge 22](#_Toc216518483)

[Vector Names 22](#_Toc216518484)

[Vector Normalize 23](#_Toc216518485)

[Vector Offset 24](#_Toc216518486)

[Vector Populate 24](#_Toc216518487)

[Vector Random 25](#_Toc216518488)

[Vector Range 25](#_Toc216518489)

[Vector Search 26](#_Toc216518490)

[Vector Set 27](#_Toc216518491)

[Vector Seq 28](#_Toc216518492)

[Vector Sort 28](#_Toc216518493)

[Vector Split 30](#_Toc216518494)

[Vector Variable 30](#_Toc216518495)

# Vector Append

## Test Case 1

**Test Case ID –** RBC.vector.append.1

**Test Item –** The *append* function of the *vector* BLT component.

**Input Specification –** A list of numeric expressions or another vector.

**Output Specification –** The components of the list or the input vector are added to the end of the vector.

**Special Procedural Requirements –** The *vector create* command can create multiple vector instances.

**Inter-case Dependencies –** RBC.vector.create.1, RBC.vector.set.1

### Test Procedure – Vector Append: List of Numbers

**Purpose –** Ensure the *vector append* command works correctly when given a list of numbers.

**Special Requirements –** None

**TclTest –** RBC.vector.append.1.1

### Test Procedure – Vector Append: Another Vector

**Purpose –** Ensure the *vector append* command works correctly when given another vector.

**Special Requirements –** None

**TclTest –** RBC.vector.append.1.2

### Test Procedure – Vector Append: List with Characters

**Purpose –** Ensure the *vector append* command works correctly when given a list containing characters.

**Special Requirements –** None

**TclTest –** RBC.vector.append.1.3

### Test Procedure – Vector Append: Non-Existent Vector

**Purpose –** Ensure the *vector append* command works correctly when given a non-existent vector reference.

**Special Requirements –** None

**TclTest –** RBC.vector.append.1.4

# Vector Arithmetic

## Test Case 1

**Test Case ID –** RBC.vector.arithmetic.1

**Test Item –** The *arithmetic* operators (+, -, \*, /) of the *vector* BLT component.

**Input Specification –** Two vectors to operate on or a scalar value to operate on a vector.

**Output Specification –** The values in the first vector are the result of the operation by the other input.

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.vector.create.1, RBC.vector.set.1

### Test Procedure – Vector Arithmetic: Vector Addition

**Purpose –** Ensure the *addition* command works correctly with vectors under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.arithmetic.1.1

### Test Procedure – Vector Arithmetic: Scalar Addition

**Purpose –** Ensure the *addition* command works correctly with scalar inputs under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.arithmetic.1.2

### Test Procedure – Vector Arithmetic: Vector Subtraction

**Purpose –** Ensure the *subtraction* command works correctly with vectors under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.arithmetic.1.3

### Test Procedure – Vector Arithmetic: Scalar Subtraction

**Purpose –** Ensure the *subtraction* command works correctly with scalar inputs under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.arithmetic.1.4

### Test Procedure – Vector Arithmetic: Vector Multiplication

**Purpose –** Ensure the *multiplication* command works correctly with vectors under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.arithmetic.1.5

### Test Procedure – Vector Arithmetic: Scalar Multiplication

**Purpose –** Ensure the *multiplication* command works correctly with scalar inputs under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.arithmetic.1.6

### Test Procedure – Vector Arithmetic: Vector Division

**Purpose –** Ensure the *division* command works correctly with vectors under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.arithmetic.1.7

### Test Procedure – Vector Arithmetic: Scalar Division

**Purpose –** Ensure the *division* command works correctly with scalar inputs under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.arithmetic.1.8

# Vector Binread

## Test Case 1

**Test Case ID -** RBC.vector.binread.1

**Test Item -** The *binread* function of an instance of the *vector* BLT component.

**Input Specification** - A Tcl channel that points to binReadTestFile.txt and a length parameter

**Output Specification -** The vector instance contains length number of bytes read from binReadTestFile.txt.

**Special Procedural Requirements -** The *vector create* command can create a vector. *Open* can open a channel to a file.

**Inter-case Dependencies** - RBC.vector.create.1

### Test Procedure - Vector Binread Entire File

**Purpose –** Ensure the *vector binread* command reads the entire contents of a file when given no *length* parameter.

**Special Requirements –** None

**TclTest –** RBC.vector.binread.1.1

### Test Procedure - Vector Binread with Length

**Purpose –** Ensure the *vector binread* command reads *length* number of items from the channel.

**Special Requirements –** None

**TclTest –** RBC.vector.binread.1.2

## Test Case 2

**Test Case ID -** RBC.vector.binread.2

**Test Item -** The *binread* function of an instance of the *vector* BLT component.

**Input Specification** - A Tcl channel that points to binReadTestFile.txt and a the *swap* switch

**Output Specification -** The vector instance contains items read from binReadTestFile.txt in the opposite endianess of the host machine.

**Special Procedural Requirements -** The *vector create* command can create a vector. *Open* can open a channel to a file.

**Inter-case Dependencies** - RBC.vector.create.1

### Test Procedure - Vector Binread without Swap

**Purpose –** Ensure the *vector binread* command reads the entire contents of a file in the default endianess of the host machine.

**Special Requirements –** None

**TclTest –** RBC.vector.binread.2.1

### Test Procedure - Vector Binread with Swap

**Purpose –** Ensure the *vector binread* command reads the entire contents of a file in the opposite endianess of the host machine.

**Special Requirements –** None

**TclTest –** RBC.vector.binread.2.2

## Test Case 3

**Test Case ID -** RBC.vector.binread.3

**Test Item -** The *binread* function of an instance of the *vector* BLT component.

**Input Specification** - A Tcl channel that points to binReadTestFile.txt and the *at index* switch

**Output Specification -** The vector instance contains the entire contents of binReadTestFile.txt starting at *index* in the vector. All items in the vector preceding *index* are unchanged.

**Special Procedural Requirements -** The *vector create* command can create a vector. *Open* can open a channel to a file.

**Inter-case Dependencies** - RBC.vector.create.1, RBC.vector.set.1

### Test Procedure - Vector Binread with At Index

**Purpose –** Ensure the *vector binread* command reads the entire contents of a file into the vector starting at *index.*

**Special Requirements –** None

**TclTest –** RBC.vector.binread.3.1

### Test Procedure - Vector Binread with At Index

**Purpose –** Ensure the *vector binread* command leaves all items preceding *index* unchanged.

**Special Requirements –** None

**TclTest –** RBC.vector.binread.3.2

## Test Case 4

**Test Case ID -** RBC.vector.binread.4

**Test Item -** The *binread* function of an instance of the *vector* BLT component.

**Input Specification** - A Tcl channel that points to binReadTestFile.txt and the *format* switch with “i1", "i2", "i4", "u1”, "u2", "u4", "r4", or “r8” as the *format* parameter

**Output Specification -** The following table represents the expected outputs given the *format* parameter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1 | 2 | 4 | 8 |
| i | {0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 -22.0 97.0} | {0.0 0.0 0.0 0.0 0.0 0.0 0.0 25066.0} | {0.0 0.0 0.0 1642725376.0} | N/A |
| u | {0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 22.0 97.0} | {0.0 0.0 0.0 0.0 0.0 0.0 0.0 25066.0} | {0.0 0.0 0.0 1642725376.0} | N/A |
| r | N/A | N/A | {0.0 0.0 0.0 5.39567264156e+020} | {0.0 4.67887034632e+163} |

**Special Procedural Requirements -** The *vector create* command can create a vector. *Open* can open a channel to a file.

**Inter-case Dependencies** - RBC.vector.create.1

### Test Procedure - Vector Binread with Format i1

**Purpose –** Ensure the *vector binread* command can read the i1 format.

**Special Requirements –** None

**TclTest –** RBC.vector.binread.4.1

### Test Procedure - Vector Binread with Format i2

**Purpose –** Ensure the *vector binread* command can read the i2 format.

**Special Requirements –** None

**TclTest –** RBC.vector.binread.4.2

### Test Procedure - Vector Binread with Format i4

**Purpose –** Ensure the *vector binread* command can read the i4 format.

**Special Requirements –** None

**TclTest –** RBC.vector.binread.4.3

### Test Procedure - Vector Binread with Format u1

**Purpose –** Ensure the *vector binread* command can read the u1 format.

**Special Requirements –** None

**TclTest –** RBC.vector.binread.4.4

### Test Procedure - Vector Binread with Format u2

**Purpose –** Ensure the *vector binread* command can read the u2 format.

**Special Requirements –** None

**TclTest –** RBC.vector.binread.4.5

### Test Procedure - Vector Binread with Format u4

**Purpose –** Ensure the *vector binread* command can read the u4 format.

**Special Requirements –** None

**TclTest –** RBC.vector.binread.4.6

### Test Procedure - Vector Binread with Format r4

**Purpose –** Ensure the *vector binread* command can read the r4 format.

**Special Requirements –** None

**TclTest –** RBC.vector.binread.4.7

### Test Procedure - Vector Binread with Format r8

**Purpose –** Ensure the *vector binread* command can read the r8 format.

**Special Requirements –** None

**TclTest –** RBC.vector.binread.4.8

# Vector Clear

## Test Case 1

**Test Case ID –** RBC.vector.clear.1

**Test Item –** The *clear* function of the *vector* BLT component.

**Input Specification –** None

**Output Specification –** The size of the vector’s Tcl array is 1.

**Special Procedural Requirements –** The *vector create* command can create a vector instance.

**Inter-case Dependencies –** RBC.vector.create.1, RBC.vector.set.1

### Test Procedure – Vector Clear

**Purpose –** Ensure the *vector clear* command works correctly.

**Special Requirements –** The vector must contain more than 1 component.

**TclTest –** RBC.vector.clear.1.1

# Vector Create

## Test Case 1

**Test Case ID** - RBC.vector.create.1

**Test Item –** This test case will test the *vector create vecName?(size)?* construct to ensure that a vector is created properly

**Input Specification –**

Create a vector three different ways:

1. No size or index parameters
2. Size parameter
3. Index parameters

**Output Specification –**

Three vectors should be created:

1. Vector with no components
2. Vector with *size* components
3. Vector with *lastIndex* – *firstIndex* + 1 components indexed starting at *firstIndex*

**Special Procedural Requirements – *none***

**Inter-case dependencies –** ***none***

### Test Procedure – Vector Create: No Size

**Purpose –** Ensure that creating a vector with no *size* parameter creates a new vector with zero elements (i.e. a length of zero).

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.1.1

### Test Procedure – Vector Create: Size

**Purpose –** Ensure that creating a vector with a *size* parameter create a new vector with the size parameter number of elements

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.1.2

### Test Procedure – Vector Create: Size of Zero

**Purpose –** Ensure that creating a vector with a *size* parameter of zero creates a new vector with zero elements.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.1.3

### Test Procedure – Vector Create: Component Initialization to 0.0

**Purpose –** Ensures that creating a vector with a *size* parameter greater than zero creates a new vector with size components all initialized to 0.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.1.4

### Test Procedure – Vector Create: Identical FirstIndex and LastIndex

**Purpose –** Ensures that creating a vector with identical range parameters creates a new vector with 1 element.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.1.5

### Test Procedure – Vector Create: Different FirstIndex and LastIndex

**Purpose –** Ensures that creating a vector with *firstIndex* and *lastIndex* range parameters creates a new vector with *lastIndex*-*firstIndex*+1 items.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.1.6

### Test Procedure – Vector Create: FirstIndex Offset

**Purpose –** Ensures that creating a vector with *firstIndex* and *lastIndex* range parameters creates a new vector whose index starts at *firstIndex*.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.1.7

### Test Procedure – Vector Create: Component Initialization to 0.0

**Purpose –** Ensures that creating a vector with *firstIndex* and *lastIndex* range parameters creates a new vector with *lastIndex*-*firstIndex*+1 components, all of which are initialized to 0.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.1.8

### Test Procedure – Vector Create: Command Creation

**Purpose –** Ensures that creating a vector also creates a command with the same name as the vector.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.1.9

### Test Procedure – Vector Create: Variable Creation

**Purpose –** Ensures that creating a vector also creates a variable with the same name as the vector.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.1.10

### Test Procedure – Vector Create: Vector Name Creation

**Purpose –** Ensures that newly created vector shows up in the vector names command.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.1.11

## Test Case 2

**Test Case ID** - RBC.vector.create.2

**Test Item –** This test case will test the *vector create vecName -variable varName* command

**Input Specification –** A valid *varName*

**Output Specification –** A variable *varName* must exist that references the newly created vector instance

**Special Procedural Requirements – *none***

**Inter-case dependencies –** RBC.vector.create.1

### Test Procedure – Vector Create: Variable Switch Variable Creation

**Purpose –** Ensures that creating a vector with the variable switch creates a variable.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.2.1

### Test Procedure – Vector Create: Variable Switch Vector Variable Non-Creation

**Purpose –** Ensures that creating a vector with the variable switch causes the vector name to not also be a variable reference.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.2.2

## Test Case 3

**Test Case ID** - RBC.vector.create.3

**Test Item –** This test case will test the *vector create vecName -command cmdName* command

**Input Specification –** A valid *cmdName*

**Output Specification –** A command *cmdName* must exist that references the commands for the newly created vector instance

**Special Procedural Requirements – *none***

**Inter-case dependencies –** RBC.vector.create.1

### Test Procedure – Vector Create: Command Switch Variable Existence

**Purpose –** Ensures that creating a vector with the command switch does not remove the variable reference that exists with the vector name.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.3.1

### Test Procedure – Vector Create: Command Switch Variable Non-Creation

**Purpose –** Ensures that creating a vector with the command switch does not create a variable reference.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.3.2

### Test Procedure – Vector Create: Command Switch Command Creation

**Purpose –** Ensures that creating a vector with the command switch creates a Tcl command.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.3.3

## Test Case 4

**Test Case ID** - RBC.vector.create.4

**Test Item –** This test case will test the *vector create vecName -watchunset boolean* command

**Input Specification –** *vecName* should be unset using the Tcl *unset* command

**Output Specification –** If *boolean* is true, the vector represented by *vecName* should still exist, otherwise, it should no longer exist

**Special Procedural Requirements – *none***

**Inter-case dependencies –** RBC.vector.create.1

### Test Procedure – Vector Create: Watchunset Switch True

**Purpose –** Ensures that creating a vector with the watchunset switch set to true causes the vector to be destroyed when the variable name (in this case the same as the vector name) is unset.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.4.1

### Test Procedure – Vector Create: Watchunset Switch False

**Purpose –** Ensures that creating a vector with the watchunset switch set to false does not destroy the vector when the variable name (in this case the same as the vector name) is unset.

**Special Requirements –** ***none***

**TclTest –** RBC.vector.create.4.2

# Vector Delete

## Test Case 1

**Test Case ID –** RBC.vector.delete.1

**Test Item –** The *delete* function of the *vector* BLT component.

**Input Specification –** One or more vector indices.

**Output Specification –** A compacted vector with the elements corresponding to the given indices removed.

**Special Procedural Requirements –** The *vector create* command can create a vector instance.

**Inter-case Dependencies –** RBC.vector.create.1, RBC.vector.length.1, RBC.vector.set.1

### Test Procedure – Vector Delete: Single Index

**Purpose –** Ensure the *vector delete* command works correctly when given a single valid vector index.

**Special Requirements –** None

**TclTest –** RBC.vector.delete.1.1

### Test Procedure – Vector Delete: Multiple Indices

**Purpose –** Ensure the *vector delete* command works correctly when given multiple valid vector indices.

**Special Requirements –** None

**TclTest –** RBC.vector.delete.1.2

### Test Procedure – Vector Delete: Invalid Index

**Purpose –** Ensure the *vector delete* command works correctly when given an invalid vector index.

**Special Requirements –** None

**TclTest –** RBC.vector.delete.1.3

### Test Procedure – Vector Delete: Valid and Invalid Indices

**Purpose –** Ensure the *vector delete* command works correctly when given both a valid and an invalid vector index.

**Special Requirements –** None

**TclTest –** RBC.vector.delete.1.4

# Vector Destroy

## Test Case 1

**Test Case ID –** RBC.vector.destroy.1

**Test Item –** The *destroy* function of the *vector* BLT component.

**Input Specification –** One or more existing vector names.

**Output Specification –** The named vectors and their corresponding Tcl commands are destroyed.

**Special Procedural Requirements –** The *vector create* command can create multiple vector instances.

**Inter-case Dependencies –** RBC.vector.create.1, RBC.vector.create.3, RBC.vector.length.1

### Test Procedure – Vector Destroy: Single Names, No Command

**Purpose –** Ensure the *vector destroy* command works correctly when given a single existing vector name and no *command* has been assigned during the *vector create*.

**Special Requirements –** None

**TclTest –** RBC.vector.destroy.1.1

### Test Procedure – Vector Destroy: Multiple Names, No Command

**Purpose –** Ensure the *vector destroy* command works correctly when given a multiple existing vector names and no *commands* have been assigned during the *vector create*.

**Special Requirements –** None

**TclTest –** RBC.vector.destroy.1.2

### Test Procedure – Vector Destroy: Single Name, With Command

**Purpose –** Ensure the *vector destroy* command works correctly when given a single existing vector name and a *command* has been assigned during the *vector create*.

**Special Requirements –** At least one vector instance with a corresponding Tcl command exists.

**TclTest –** RBC.vector.destroy.1.3

### Test Procedure – Vector Destroy: Multiple Names, With Command

**Purpose –** Ensure the *vector destroy* command works correctly when given a multiple existing vector names and a *commands* have been assigned during the *vector create*.

**Special Requirements –** More than one vector instance with corresponding Tcl commands exist.

**TclTest –** RBC.vector.destroy.1.4

# Vector Dup

## Test Case 1

**Test Case ID –** RBC.vector.dup.1

**Test Item –** The *dup* function of an instance of the *vector* BLT component.

**Input Specification –** A destination vector name, *destName.*

**Output Specification –** *destName* should contain all contents of the *vector* instance.

**Special Procedural Requirements –** The *vector create* command can create a vector instance.

**Inter-case Dependencies –** RBC.vector.create.1

### Test Procedure – Vector Dup: Destination Exists

**Purpose –** Ensure the *vector dup* command overwrites the contents of an existing *destName* vector with the contents of the instance vector.

**Special Requirements –** None

**TclTest –** RBC.vector.dup.1.1

### Test Procedure – Vector Dup: Destination Doesn’t Exist

**Purpose –** Ensure the *vector dup* command creates a new vector with the contents of the instance vector when *destName* does not exist.

**Special Requirements –** None

**TclTest –** RBC.vector.dup.1.2

# Vector Expr

## Test Case 1

**Test Case ID –** RBC.vector.expr.1

**Test Item –** The *expr* function of the *vector* BLT component.

**Input Specification –**

**Output Specification –**

**Special Produral Requirements –**

**Inter-case Dependencies –** RBC.vector.create.1

### Test Procedure – Vector Expr: Unary Minus

**Purpose –** Ensure the *vector expr* command works correctly with unary minus under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.1

### Test Procedure – Vector Expr: Logical Not

**Purpose –** Ensure the *vector expr* command works correctly with logical not under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.2

### Test Procedure – Vector Expr: Exponentiation

**Purpose –** Ensure the *vector expr* command works correctly with exponentiation under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.3

### Test Procedure – Vector Expr: Multiplication

**Purpose –** Ensure the *vector expr* command works correctly with multiplication under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.4

### Test Procedure – Vector Expr: Division

**Purpose –** Ensure the *vector expr* command works correctly with division under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.5

### Test Procedure – Vector Expr: Modulo

**Purpose –** Ensure the *vector expr* command works correctly with modulo under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.6

### Test Procedure – Vector Expr: Subtraction

**Purpose –** Ensure the *vector expr* command works correctly with subtraction under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.7

### Test Procedure – Vector Expr: Addition

**Purpose –** Ensure the *vector expr* command works correctly with addition under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.8

### Test Procedure – Vector Expr: Shift Left

**Purpose –** Ensure the *vector expr* command works correctly with vector shift left one index.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.9

### Test Procedure – Vector Expr: Shift Right

**Purpose –** Ensure the *vector expr* command works correctly with vector shift right one index.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.10

### Test Procedure – Vector Expr: Less Than

**Purpose –** Ensure the *vector expr* command works correctly with less than under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.11

### Test Procedure – Vector Expr: Greater Than

**Purpose –** Ensure the *vector expr* command works correctly with greater than under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.12

### Test Procedure – Vector Expr: Less Than or Equal

**Purpose –** Ensure the *vector expr* command works correctly with less than or equal under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.13

### Test Procedure – Vector Expr: Greater Than or Equal

**Purpose –** Ensure the *vector expr* command works correctly with greater than or equal under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.14

### Test Procedure – Vector Expr: Logical And

**Purpose –** Ensure the *vector expr* command works correctly with logical and under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.15

### Test Procedure – Vector Expr: Logical Or

**Purpose –** Ensure the *vector expr* command works correctly with logical or under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.16

### Test Procedure – Vector Expr: Inverse Cosine

**Purpose –** Ensure the *vector expr acos* command correctly calculates the inverse cosine of each element in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.17

### Test Procedure – Vector Expr: Inverse Sine

**Purpose –** Ensure the *vector expr asin* command correctly calculates the inverse sine of each element in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.18

### Test Procedure – Vector Expr: Inverse Tangent

**Purpose –** Ensure the *vector expr atan* command correctly calculates the inverse tangent of each element in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.19

### Test Procedure – Vector Expr: Ceiling

**Purpose –** Ensure the *vector expr ceil* command correctly calculates the ceiling of each element in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.20

### Test Procedure – Vector Expr: Cosine

**Purpose –** Ensure the *vector expr cos* command correctly calculates the cosine of each element in a vector under several different conditions.

**Special Requirements –** Elements given must be in radians.

**TclTest –** RBC.vector.expr.1.21

### Test Procedure – Vector Expr: Hyperbolic Cosine

**Purpose –** Ensure the *vector expr cosh* command correctly calculates the hyperbolic cosine of each element in a vector under several different conditions.

**Special Requirements –** Elements given must be in radians.

**TclTest –** RBC.vector.expr.1.22

### Test Procedure – Vector Expr: Exponentiation Base e

**Purpose –** Ensure the *vector expr exp* command correctly calculates the exponentiation of the base *e* of each element in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.23

### Test Procedure – Vector Expr: Floor

**Purpose –** Ensure the *vector expr floor* command correctly calculates the floor of elements in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.24

### Test Procedure – Vector Expr: Natural Logarithm

**Purpose –** Ensure the *vector expr log* command correctly calculates the natural logarithm (base *e*) of elements in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.25

### Test Procedure – Vector Expr: Logarithm Base 10

**Purpose –** Ensure the *vector expr log10* command correctly calculates the logarithm (base 10) of elements in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.26

### Test Procedure – Vector Expr: Absolute Value

**Purpose –** Ensure the *vector expr abs* command correctly calculates the absolute value of elements in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.27

### Test Procedure – Vector Expr: Sine

**Purpose –** Ensure the *vector expr sin* command correctly calculates the sine of elements in a vector under several different conditions.

**Special Requirements –** Elements given must be in radians.

**TclTest –** RBC.vector.expr.1.28

### Test Procedure – Vector Expr: Hyperbolic Sine

**Purpose –** Ensure the *vector expr sinh* command correctly calculates the hyperbolic sine of elements in a vector under several different conditions.

**Special Requirements –** Elements given must be in radians.

**TclTest –** RBC.vector.expr.1.29

### Test Procedure – Vector Expr: Square Root

**Purpose –** Ensure the *vector expr sqrt* command correctly calculates the square root of elements in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.30

### Test Procedure – Vector Expr: Trigonometric Tangent

**Purpose –** Ensure the *vector expr tan* command correctly calculates the trigonometric tangent of elements in a vector under several different conditions.

**Special Requirements –** Elements given must be in radians.

**TclTest –** RBC.vector.expr.1.31

### Test Procedure – Vector Expr: Hyperbolic Tangent

**Purpose –** Ensure the *vector expr tanh* command correctly calculates the hyperbolic tangent of elements in a vector under several different conditions.

**Special Requirements –** Elements given must be in radians.

**TclTest –** RBC.vector.expr.1.32

### Test Procedure – Vector Expr: Round

**Purpose –** Ensure the *vector expr round* command correctly rounds each element in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.33

### Test Procedure – Vector Expr: Average Deviation

**Purpose –** Ensure the *vector expr adev* command correctly calculates the average deviation of elements in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.34

### Test Procedure – Vector Expr: Kurtosis

**Purpose –** Ensure the *vector expr kurtosis* command correctly calculates the kurtosis (degree of peakedness or fourth moment) of elements in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.35

### Test Procedure – Vector Expr: Length with Elements

**Purpose –** Ensure the *vector expr length* command correctly calculates the length of a vector containing elements.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.36

### Test Procedure – Vector Expr: Maximum

**Purpose –** Ensure the *vector expr maximum* command correctly calculates the maximum element in a vector.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.37

### Test Procedure – Vector Expr: Mean

**Purpose –** Ensure the *vector expr mean* command correctly calculates the mean of the elements in a vector.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.38

### Test Procedure – Vector Expr: Median

**Purpose –** Ensure the *vector expr median* command correctly calculates the median of the elements in a vector.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.39

### Test Procedure – Vector Expr: Minimum

**Purpose –** Ensure the *vector expr minimum* command correctly calculates the minimum element in a vector.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.40

### Test Procedure – Vector Expr: First Quartile

**Purpose –** Ensure the *vector expr q1* command correctly calculates the first quartile of a vector.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.41

### Test Procedure – Vector Expr: Third Quartile

**Purpose –** Ensure the *vector expr q3* command correctly calculates the third quartile of a vector.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.42

### Test Procedure – Vector Expr: Product of Vector Entries

**Purpose –** Ensure the *vector expr prod* command correctly calculates the product the elements in a vector.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.43

### Test Procedure – Vector Expr: Standard Deviation

**Purpose –** Ensure the *vector expr sdev* command correctly calculates the standard deviation of elements in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.44

### Test Procedure – Vector Expr: Skewness

**Purpose –** Ensure the *vector expr skew* command correctly calculates the skewness (third moment) of elements in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.45

### Test Procedure – Vector Expr: Sum of Vector Entries

**Purpose –** Ensure the *vector expr sum* command correctly calculates the sum of the elements in a vector.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.46

### Test Procedure – Vector Expr: Variance

**Purpose –** Ensure the *vector expr var* command correctly calculates the variance of elements in a vector under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.47

### Test Procedure – Vector Expr: Normalize

**Purpose –** Ensure the *vector expr norm* command correctly calculates a normalized version of a vector.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.48

### Test Procedure – Vector Expr: Sort

**Purpose –** Ensure the *vector expr sort* command correctly sorts a vector. under several different conditions.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.49

### Test Procedure – Vector Expr: Produce Random Vector

**Purpose –** Ensure the *vector expr random* command correctly produces a random vector of a given length.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.50

### Test Procedure – Vector Expr: Length without Elements

**Purpose –** Ensure the *vector expr length* command correctly calculates the length of a vector containing no elements.

**Special Requirements –** None

**TclTest –** RBC.vector.expr.1.51

# Vector Index

## Test Case 1

**Test Case ID –** RBC.vector.index.1

**Test Item –** The *index* function of the *vector* BLT component.

**Input Specification –** A vector index.

**Output Specification –** The vector value that corresponds to the given index.

**Special Procedural Requirements –** The *vector create* command can create a vector instance.

**Inter-case Dependencies –** RBC.vector.create.1, RBC.vector.set.1

### Test Procedure – Vector Index: Valid Index

**Purpose –** Ensure the *vector index* command works correctly when given a valid vector index.

**Special Requirements –** None

**TclTest –** RBC.vector.index.1.1

### Test Procedure – Vector Index: Invalid Index

**Purpose –** Ensure the *vector index* command works correctly when given an invalid vector index.

**Special Requirements –** None

**TclTest –** RBC.vector.index.1.2

## Test Case 2

**Test Case ID –** RBC.vector.index.2

**Test Item –** The *index* function of the *vector* BLT component.

**Input Specification –** A vector index and a numerical value.

**Output Specification –** The vector value at the given index is set to the given value.

**Special Procedural Requirements –** The *vector create* command can create a vector instance.

**Inter-case Dependencies –** RBC.vector.create.1, RBC.vector.set.1

### Test Procedure – Vector Index: Valid Index, Valid Value

**Purpose –** Ensure the *vector index* command works correctly when given a valid vector index and a valid value.

**Special Requirements –** None

**TclTest –** RBC.vector.index.2.1

### Test Procedure – Vector Index: Invalid Index, Valid Value

**Purpose –** Ensure the *vector index* command works correctly when given an invalid vector index.

**Special Requirements –** None

**TclTest –** RBC.vector.index.2.2

### Test Procedure – Vector Index: Valid Index, Invalid Value

**Purpose –** Ensure the *vector index* command works correctly when given a valid vector index and an invalid value.

**Special Requirements –** None

**TclTest –** RBC.vector.index.2.3

# Vector Length

## Test Case 1

**Test Case ID –** RBC.vector.length.1

**Test Item –** The *length* function of the *vector* BLT component.

**Input Specification –** A vector of a certain length.

**Output Specification –** The value of the vector’s length

**Special Procedural Requirements –**None

**Inter-case dependencies –** RBC.vector.create.1

### Test Procedure – Vector Length: No Offset

**Purpose –** Ensure the *vector length* command works correctly when a vector with no offset is used.

**Special Requirements –** None

**TclTest –** RBC.vector.length.1.1

### Test Procedure – Vector Names: With Offset

**Purpose –** Ensure the *vector length* command works correctly when a vector with an offset is used.

**Special Requirements –** None

**TclTest –** RBC.vector.length.1.2

## Test Case 2

**Test Case ID –** RBC.vector.length.2

**Test Item –** The *length* function of the *vector* BLT component.

**Input Specification –** A vector of a certain length and a value to resize the input vector.

**Output Specification –** The input vector adjusted to the correct size.

**Special Procedural Requirements –**None

**Inter-case dependencies –** RBC.vector.create.1, RBC.vector.length.1

### Test Procedure – Vector Length: Return Value

**Purpose –** Ensure that the *length* returns the correct value

**Special Requirements –** None

**TclTest –** RBC.vector.names.2.1

### Test Procedure – Vector Length: Intact Offset

**Purpose –** Ensure the *length* will keep the vector’s offset intact.

**Special Requirements –** None

**TclTest –** RBC.vector.names.2.2

### Test Procedure – Vector Length: Expanding Vector

**Purpose –** Ensure the *length* will expand to the correct size.

**Special Requirements –** None

**TclTest –** RBC.vector.names.2.3

### Test Procedure – Vector Length: Expanding Vector keeps Data

**Purpose –** Ensure the *length* will keep its data and set new spaces to 0 when it expands.

**Special Requirements –** None

**TclTest –** RBC.vector.names.2.4

### Test Procedure – Vector Length: Shrinking Vector

**Purpose –** Ensure the *length* will shrink to the correct size.

**Special Requirements –** None

**TclTest –** RBC.vector.names.2.5

### Test Procedure – Vector Length: Shrinking Vector keeps Data

**Purpose –** Ensure the *length* will keep its data when it shrinks.

**Special Requirements –** None

**TclTest –** RBC.vector.names.2.6

# Vector Merge

## Test Case 1

**Test Case ID –** RBC.vector.merge.1

**Test Item –** The *merge* function of the *vector* BLT component.

**Input Specification –** A vector to merge into as well as vectors which to merge.

**Output Specification –** The first vector contains the index by index merge of the other vectors.

**Special Procedural Requirements –**None

**Inter-case dependencies –** RBC.vector.create.1

### Test Procedure – Vector Merge: Single Vector

**Purpose –** Ensure the *vector merge* command works correctly when a single vector is being merged.

**Special Requirements –** None

**TclTest –** RBC.vector.merge.1.1

### Test Procedure – Vector Merge: Multiple Vectors

**Purpose –** Ensure the *vector length* command works correctly when multiple vectors are being merged.

**Special Requirements –** None

**TclTest –** RBC.vector.merge.1.2

# Vector Names

## Test Case 1

**Test Case ID –** RBC.vector.names.1

**Test Item –** The *names* function of the *vector* BLT component.

**Input Specification –** A pattern.

**Output Specification –** The names of vector objects that match the pattern.

**Special Procedural Requirements –** The *vector create* command can create multiple vector instances.

**Inter-case dependencies –** RBC.vector.create.1

### Test Procedure – Vector Names: No Pattern

**Purpose –** Ensure the *vector names* command works correctly when no pattern is given.

**Special Requirements –** None

**TclTest –** RBC.vector.names.1.1

### Test Procedure – Vector Names: Exact Pattern

**Purpose –** Ensure the *vector names* command works correctly when an exact pattern is given.

**Special Requirements –** None

**TclTest –** RBC.vector.names.1.2

### Test Procedure – Vector Names: Wildcard Pattern

**Purpose –** Ensure the *vector names* command works correctly when a wildcard pattern is given.

**Special Requirements –** None

**TclTest –** RBC.vector.names.1.3

### Test Procedure – Vector Names: Incorrect Pattern

**Purpose –** Ensure the *vector names* command works correctly when an incorrect pattern is given.

**Special Requirements –** None

**TclTest –** RBC.vector.names.1.4

# Vector Normalize

## Test Case 1

**Test Case ID –** RBC.vector.normalize.1

**Test Item –** The *normalize* function of the *vector* BLT component.

**Input Specification –** A vector of numbers

**Output Specification –** An equivalent normalized vector.

**Special Procedural Requirements –** The *vector create* command can create a vector instance.

**Inter-case dependencies –** RBC.vector.normalize.1

### Test Procedure – Vector Normalize: Positive Numbers

**Purpose –** Ensure the *vector normalize* command works correctly when given a vector of positive numbers.

**Special Requirements –** None

**TclTest –** RBC.vector.normalize.1.1

### Test Procedure – Vector Normalize: Negative Numbers

**Purpose –** Ensure the *vector normalize* command works correctly when given a vector of negative numbers.

**Special Requirements –** None

**TclTest –** RBC.vector.normalize.1.2

### Test Procedure – Vector Normalize: Empty Vector

**Purpose –** Ensure the *vector normalize* command works correctly when given a vector with no numbers.

**Special Requirements –** None

**TclTest –** RBC.vector.normalize.1.3

# Vector Offset

## Test Case 1

**Test Case ID –** RBC.vector.offset.1

**Test Item –** The *offset* function of the *vector* BLT component.

**Input Specification –** A vector with an offset.

**Output Specification –** The value of the vector’s offset

**Special Procedural Requirements –**None

**Inter-case dependencies –** RBC.vector.create.1

### Test Procedure – Vector Merge: Non-Zero Offset

**Purpose –** Ensure the *vector offset* command returns the correct value.

**Special Requirements –** None

**TclTest –** RBC.vector.offset.1.1

## Test Case 2

**Test Case ID –** RBC.vector.offset.2

**Test Item –** The *offset* function of the *vector* BLT component.

**Input Specification –** A vector and a value to offset this vector to.

**Output Specification –** The value of the vector’s offset

**Special Procedural Requirements –**None

**Inter-case dependencies –** RBC.vector.create.1

### Test Procedure – Vector Merge: Correct Offset

**Purpose –** Ensure the *vector offset* command actually changes the vectors offset.

**Special Requirements –** None

**TclTest –** RBC.vector.offset.2.1

### Test Procedure – Vector Merge: Correct Data

**Purpose –** Ensure the *vector offset* command keeps the same data in the vector

**Special Requirements –** None

**TclTest –** RBC.vector.offset.2.2

# Vector Populate

## Test Case 1

**Test Case ID –** RBC.vector.populate.1

**Test Item –** The *populate* function of an instance of the *vector* BLT component.

**Input Specification –** A destination vector name, *destName* and a *density*

**Output Specification –** *destName* should contain all contents of the *vector* instance with *density* number of new components between each consecutive pair of components in the vector. These new components should have numerical values interpolated between every consecutive pair of pre-existing components.

**Special Procedural Requirements –** The *vector create* command can create a vector instance.

**Inter-case Dependencies –** RBC.vector.create.1, RBC.vector.set.1

### Test Procedure – Vector Populate: Destination Exists

**Purpose –** Ensure the *vector populate* command overwrites the contents of an existing *destName* vector with the contents of the instance vector and *density* new components between each pre-existing component.

**Special Requirements –** None

**TclTest –** RBC.vector.populate.1.1

### Test Procedure – Vector Populate: Destination Doesn’t Exist

**Purpose –** Ensure the *vector populate* command creates a new *destName* vector with the contents of the instance vector and *density* new components between each pre-existing component.

**Special Requirements –** None

**TclTest –** RBC.vector.populate.1.2

# Vector Random

## Test Case 1

**Test Case ID –** RBC.vector.random.1

**Test Item –** The *random* function of the *vector* BLT component.

**Input Specification –** A with a specified length.

**Output Specification –** A random vector with all elements between 0.0 and 1.0

**Special Procedural Requirements –** The *vector create* command can create a vector instance.

**Inter-case Dependencies –** RBC.vector.create.1, RBC.vector.set.1

### Test Procedure – Vector Random: Uninitialized Vector

**Purpose –** Ensure the *vector random* command works correctly when given vector with only a set length but uninitialized elements.

**Special Requirements –** None

**TclTest –** RBC.vector.random.1.1

### Test Procedure – Vector Random: Initialized Vector

**Purpose –** Ensure the *vector random* command works correctly when given vector that has elements initialized.

**Special Requirements –** None

**TclTest –** RBC.vector.random.1.2

# Vector Range

## Test Case 1

**Test Case ID –** RBC.vector.range.1

**Test Item –** The *range* function of the *vector* BLT component.

**Input Specification –** Two vector indices.

**Output Specification –** The set of elements between the given indices.

**Special Procedural Requirements –** The *vector create* command can create a vector instance.

**Inter-case Dependencies –** RBC.vector.create.1, RBC.vector.set.1

### Test Procedure – Vector Range: Valid Indices

**Purpose –** Ensure the *vector range* command works correctly when given two valid vector indices where the last index is greater than the first index.

**Special Requirements –** None

**TclTest –** RBC.vector.range.1.1

### Test Procedure – Vector Range: Invalid Index

**Purpose –** Ensure the *vector range* command works correctly when given an invalid vector index.

**Special Requirements –** None

**TclTest –** RBC.vector.range.1.2

### Test Procedure – Vector Range: Valid Indices, Reversed

**Purpose –** Ensure the *vector range* command works correctly when given two valid vector indices where the last index is less than the first index.

**Special Requirements –** None

**TclTest –** RBC.vector.range.1.3

### Test Procedure – Vector Range: Keyword

**Purpose –** Ensure the *vector range* command works correctly when given a valid vector index and the *end* keyword.

**Special Requirements –** None

**TclTest –** RBC.vector.range.1.4

# Vector Search

## Test Case 1

**Test Case ID –** RBC.vector.search.1

**Test Item –** The *search* function of the *vector* BLT component.

**Input Specification –** A list of numbers and a vector

**Output Specification –** The locations of the numbers provided in the given vector.

**Special Procedural Requirements –** The *vector create* command can create an instance of a vector.

**Inter-case Dependencies –** RBC.vector.create.1, RBC.vector.set.1

### Test Procedure – Vector Search: Single Number Multiple Instances

**Purpose –** Ensure the *vector search* command returns the locations of the given element.

**Special Requirements –** None

**TclTest –** RBC.vector.search.1.1

### Test Procedure – Vector Search: Single Number Single Instances

**Purpose –** Ensure the *vector search* command returns the location of the given element.

**Special Requirements –** None

**TclTest –** RBC.vector.search.1.2

### Test Procedure – Vector Search: Single Number Zero Instances

**Purpose –** Ensure the *vector search* command returns nothing.

**Special Requirements –** None

**TclTest –** RBC.vector.search.1.3

### Test Procedure – Vector Search: Range of Numbers A

**Purpose –** Ensure the *vector search* command returns the locations of all elements given a range of numbers for which to search. Numbers provided are in the vector.

**Special Requirements –** None

**TclTest –** RBC.vector.search.1.4

### Test Procedure – Vector Search: Range of Numbers B

**Purpose –** Ensure the *vector search* command returns the locations of all elements given a range of numbers for which to search. Numbers provided are not in the vector but elements within the vector are.

**Special Requirements –** None

**TclTest –** RBC.vector.search.1.5

### Test Procedure – Vector Search: Range of Numbers C

**Purpose –** Ensure the *vector search* command returns the locations of all elements given a range of numbers for which to search. Numbers provided are not in the vector and no elements within the vector are within the range.

**Special Requirements –** None

**TclTest –** RBC.vector.search.1.6

# Vector Set

## Test Case 1

**Test Case ID –** RBC.vector.set.1

**Test Item –** The *set* function of the *vector* BLT component.

**Input Specification –** A list of numeric expressions or another vector.

**Output Specification –** The components of the vector being set correspond to those in the list or the other vector’s components.

**Special Procedural Requirements –** The *vector create* command can create multiple vector instances.

**Inter-case Dependencies –** RBC.vector.create.1

### Test Procedure – Vector Set: List of Numbers

**Purpose –** Ensure the *vector set* command works correctly when given a list of numbers.

**Special Requirements –** None

**TclTest –** RBC.vector.set.1.1

### Test Procedure – Vector Set: Another Vector

**Purpose –** Ensure the *vector set* command works correctly when given another vector.

**Special Requirements –** None

**TclTest –** RBC.vector.set.1.2

### Test Procedure – Vector Set: List with Characters

**Purpose –** Ensure the *vector set* command works correctly when given a list containing characters.

**Special Requirements –** None

**TclTest –** RBC.vector.set.1.3

### Test Procedure – Vector Set: Non-Existent Vector

**Purpose –** Ensure the *vector set* command works correctly when given a non-existent vector reference.

**Special Requirements –** None

**TclTest –** RBC.vector.set.1.4

# Vector Seq

## Test Case 1

**Test Case ID –** RBC.vector.seq.1

**Test Item –** The *seq* function of the *vector* BLT component.

**Input Specification –** A vector and inputs for the start, finish and step of the sequence.

**Output Specification –** A vector is resized to contain all values in the sequence between start and finished and incremented by step.

**Special Procedural Requirements –** None

**Inter-case Dependencies –** RBC.vector.create.1, RBC.vector.range.1

### Test Procedure – Vector Seq: No Step

**Purpose –** Ensure the *vector seq* command works correctly when given no steps.

**Special Requirements –** None

**TclTest –** RBC.vector.split.1.1

### Test Procedure – Vector Seq: Positive Step

**Purpose –** Ensure the *vector seq* command works correctly when given a positive step.

**Special Requirements –** None

**TclTest –** RBC.vector.seq.1.2

### Test Procedure – Vector Seq: Negative Step

**Purpose –** Ensure the *vector seq* command works correctly when given a negative step.

**Special Requirements –** None

**TclTest –** RBC.vector.seq.1.3

# Vector Sort

## Test Case 1

**Test Case ID –** RBC.vector.sort.1

**Test Item –** The *sort* function of the *vector* BLT component.

**Input Specification –** A vector of numbers

**Output Specification –** The components of the vector are sorted as specified.

**Special Procedural Requirements –** The vector create and set commands function.

**Inter-Case Dependencies –** RBC.vector.create.1, RBC.vector.set.1

### Test Procedure – Vector Sort: One Vector

**Purpose –** Ensure the *vector sort* command works correctly when given a single vector name.

**Special Requirements –** None

**TclTest –** RBC.vector.sort.1.1

### Test Procedure – Vector Sort: One Vector Reverse

**Purpose –** Ensure the *vector sort* command works correctly when given a single vector name and the reverse flag

**Special Requirements –** None

**TclTest –** RBC.vector.sort.1.2

## Test Case 2

**Test Case ID –** RBC.vector.sort.2

**Test Item –** The *sort* function of the *vector* BLT component.

**Input Specification –** Two vectors of numbers

**Output Specification –** The components of the vectors are sorted as specified.

**Special Procedural Requirements –** The vector create and set commands function.

**Inter-Case Dependencies –** RBC.vector.create.1, RBC.vector.set.1

### Test Procedure – Vector Sort: Two Vectors

**Purpose –** Ensure the *vector sort* command works correctly when given two vectors.

**Special Requirements –** None

**TclTest –** RBC.vector.sort.1.1

### Test Procedure – Vector Sort: Two Vectors Reverse

**Purpose –** Ensure the *vector sort* command works correctly when given two vectors and the reverse flage.

**Special Requirements –** None

**TclTest –** RBC.vector.sort.1.2

# Vector Split

## Test Case 1

**Test Case ID –** RBC.vector.split.1

**Test Item –** The *split* function of the *vector* BLT component.

**Input Specification –** One or more vector names.

**Output Specification –** The components of the vector are split into the input vector names by appending the first component to the first vector name, the ith component to the ith vector name, the ith + 1 component to the first vector name, etc.

**Special Procedural Requirements –** The *vector create* command can create multiple vector instances and the number of components in the vector can be split evenly into the given number of input vector names.

**Inter-case Dependencies –** RBC.vector.create.1, RBC.vector.set.1

### Test Procedure – Vector Split: One Vector Name

**Purpose –** Ensure the *vector split* command works correctly when given a single vector name.

**Special Requirements –** None

**TclTest –** RBC.vector.split.1.1

### Test Procedure – Vector Split: Multiple Vector Name

**Purpose –** Ensure the *vector split* command works correctly when given multiple vector names.

**Special Requirements –** None

**TclTest –** RBC.vector.split.1.2

### Test Procedure – Vector Split: Incorrect Number of Vector Names

**Purpose –** Ensure the *vector split* command works correctly when the components of the vector cannot be evenly split into the given number of vector names.

**Special Requirements –** None

**TclTest –** RBC.vector.split.1.3

# Vector Variable

## Test Case 1

**Test Case ID –** RBC.vector.variable.1

**Test Item –** The *variable* function of an instance of the *vector* BLT component.

**Input Specification –** A destination variable name, *varName*.

**Output Specification –** *varName* should map to the vector instance and any existing variable mappings to the vector instance should no longer exist.

**Special Procedural Requirements –** The *vector create* command can create a vector instance.

**Inter-case Dependencies –** RBC.vector.create.1, RBC.vector.set.1

### Test Procedure – Vector Variable: Variable Creation

**Purpose –** Ensure the *vector variable* command creates a new Tcl variable reference to the vector instance.

**Special Requirements –** None

**TclTest –** RBC.vector.variable.1.1

### Test Procedure – Vector Variable: Existing Variable Destruction

**Purpose –** Ensure the *vector variable* command removes any old Tcl variable reference to the vector instance.

**Special Requirements –** None

**TclTest –** RBC.vector.variable.1.2