Math Lib Vector Operation Case Study Results				
219	Yes's	82	No's	
72.80%		27.20%		
Op	**Originally Safe?**	**Original Syntax**	**Location (best available)**	**Analysis**
ref	NO	(vector-ref new-js (+ k shift))	.../array/array-broadcast.rkt:34:31	Yes - made safe by writing code in a different way
ref	YES	(vector-ref old-ds k)	.../array/array-broadcast.rkt:35:31	Yes - works w/o changes
set!	NO	(vector-set! old-js k old-jk)	.../array/array-broadcast.rkt:37:16	Yes - made safe by writing code in a different way
ref	NO	(vector-ref ds1 k)	.../array/array-broadcast.rkt:58:23	Yes - fixed with minor dynamic checks
ref	NO	(vector-ref ds2 k)	.../array/array-broadcast.rkt:59:23	Yes - fixed with minor dynamic checks
set!	NO	(vector-set! new-ds k (if (let-values (((or-part) (= dk1 (quote 0)))) (if or-part or-part (#%expression (= dk2 (quote 0))))) (let-values () (fail)) (let-values () (fxmax dk1 dk2))))	.../array/array-broadcast.rkt:60:11	Yes - fixed with minor dynamic checks
ref	NO	(vector-ref ds1 k)	.../array/array-broadcast.rkt:72:23	Yes - fixed with minor dynamic checks
ref	NO	(vector-ref ds2 k)	.../array/array-broadcast.rkt:73:23	Yes - fixed with minor dynamic checks
set!	NO	(vector-set! new-ds k (if (= dk1 dk2) (let-values () dk1) (if (if (= dk1 (quote 1)) (#%expression (> dk2 (quote 0))) (quote #f)) (let-values () dk2) (if (if (= dk2 (quote 1)) (#%expression (> dk1 (quote 0))) (quote #f)) (let-values () dk1) (let-values () (fail))))))	.../array/array-broadcast.rkt:74:11	Yes - fixed with minor dynamic checks
ref	YES	(vector-ref ds k)	.../array/array-fft.rkt:32:16	Yes - works w/o changes
ref	NO	(vector-ref ds (quote 0))	.../array/array-fft.rkt:36:5	Yes - fixed with minor code changes
set!	NO	(vector-set! js (quote 0) j0)	.../array/array-fft.rkt:36:5	Yes - fixed with minor code changes
set!	NO	(vector-set! js i ji)	.../array/array-fft.rkt:36:5	Yes - fixed with minor code changes

ref	NO	(vector-ref ds (quote 0))	.../array/array-fft.rkt:36:5	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 1) j1)	.../array/array-fft.rkt:36:5	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 1))	.../array/array-fft.rkt:36:5	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ds i)	.../array/array-fft.rkt:36:5	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! js (quote 0) j0)	.../array/array-fft.rkt:36:5	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref ds k)	.../array/array-parallel.rkt:21:31	Yes - works w/o changes
ref	NO	(vector-ref js k)	.../array/array-parallel.rkt:22:73	Yes - fixed with minor dynamic checks
set!	NO	(vector-set! js k jk)	.../array/array-parallel.rkt:25:20	Yes - fixed with minor dynamic checks
set!	NO	(vector-set! js k (quote 0))	.../array/array-parallel.rkt:28:20	Yes - fixed with minor dynamic checks
set!	NO	(vector-set! vs j v)	.../array/array-parallel.rkt:32:17	No - requires non-linear math.
ref	YES	(vector-ref ds i)	.../array/array-print.rkt:72:24	Yes - works w/o changes
set!	NO	(vector-set! js i ji)	.../array/array-print.rkt:76:17	Duplicate - from macro usage
ref	NO	(vector-ref js k)	.../array/array-unfold.rkt:29:23	No - unsafe, caller must verify safety of ops
ref	NO	(vector-ref zs j)	.../array/fcarray-struct.rkt:40:69	No - too complex.
ref	NO	(vector-ref zs j)	.../array/fcarray-struct.rkt:41:69	No - too complex.
ref	NO	(vector-ref ds (quote 0))	.../array/fcarray-struct.rkt:51:4	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 0))	.../array/fcarray-struct.rkt:51:4	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 0) j0)	.../array/fcarray-struct.rkt:51:4	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 1) j1)	.../array/fcarray-struct.rkt:51:4	Duplicate - from macro usage
set!	NO	(vector-set! js i ji)	.../array/fcarray-struct.rkt:51:4	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 1))	.../array/fcarray-struct.rkt:51:4	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ds i)	.../array/fcarray-struct.rkt:51:4	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! js (quote 0) j0)	.../array/fcarray-struct.rkt:51:4	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs j)	.../array/flarray-struct.rkt:34:66	No - too complex.

ref	NO	(vector-ref ds (quote 0))	.../array/flarray-struct.rkt:43:4	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 0))	.../array/flarray-struct.rkt:43:4	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 0) j0)	.../array/flarray-struct.rkt:43:4	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 1) j1)	.../array/flarray-struct.rkt:43:4	Duplicate - from macro usage
set!	NO	(vector-set! js i ji)	.../array/flarray-struct.rkt:43:4	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 1))	.../array/flarray-struct.rkt:43:4	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ds i)	.../array/flarray-struct.rkt:43:4	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! js (quote 0) j0)	.../array/flarray-struct.rkt:43:4	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref js k)	.../array/typed-array-constructors.rkt:21:62	No - would require dep struct types (unimplemented feature)
ref	NO	(vector-ref js (quote 0))	.../array/typed-array-constructors.rkt:49:33	No - would require dep struct types (unimplemented feature)
ref	NO	(vector-ref js (quote 1))	.../array/typed-array-constructors.rkt:50:33	No - would require dep struct types (unimplemented feature)
ref	NO	(vector-ref js (quote 0))	.../array/typed-array-constructors.rkt:55:33	No - would require dep struct types (unimplemented feature)
ref	NO	(vector-ref js i)	.../array/typed-array-constructors.rkt:58:34	No - would require dep struct types (unimplemented feature)
ref	NO	(unsafe-vector-ref js i)	.../array/typed-array-convert.rkt:1:0	No - too complex.
ref	NO	(vector-ref vec j_i)	.../array/typed-array-convert.rkt:1:0	No - too complex.
set!	NO	(unsafe-vector-set! vec i lst)	.../array/typed-array-convert.rkt:1:0	Yes - successfully made safe with annotations
ref	YES	(vector-ref vec (quote 0))	.../array/typed-array-convert.rkt:1:0	Yes - works w/o changes
ref	YES	(vector-ref vec i)	.../array/typed-array-convert.rkt:1:0	Yes - works w/o changes
ref	YES	(vector-ref vec i)	.../array/typed-array-convert.rkt:1:0	Yes - works w/o changes
ref	YES	(vector-ref ds i)	.../array/typed-array-convert.rkt:33:22	Yes - works w/o changes
set!	NO	(vector-set! js i ji)	.../array/typed-array-convert.rkt:40:27	Yes - successfully made safe with annotations
ref	YES	(vector-ref ds i)	.../array/typed-array-convert.rkt:53:22	Yes - works w/o changes
set!	NO	(vector-set! js i ji)	.../array/typed-array-convert.rkt:60:27	Yes - successfully made safe with annotations

set!	YES	(vector-set! veci ji (i-loop (+ i (quote 1))))	.../array/typed-array-convert.rkt:61:27	Yes - works w/o changes
ref	NO	(vector-ref ds k)	.../array/typed-array-fold.rkt:28:15	No - would require dep struct types (unimplemented feature)
set!	NO	(vector-set! old-js k jk)	.../array/typed-array-fold.rkt:35:22	Yes - made safe by writing code in a different way
ref	NO	(vector-ref (Array-shape arr) k)	.../array/typed-array-fold.rkt:64:15	No - would require dep struct types (unimplemented feature)
set!	NO	(vector-set! jks i jk)	.../array/typed-array-indexing.rkt:168:18	Yes - successfully made safe with annotations
ref	NO	(vector-ref ds k)	.../array/typed-array-indexing.rkt:185:17	Yes - fixed with minor code changes
ref	NO	(vector-ref js (quote 0))	.../array/typed-array-indexing.rkt:263:24	Yes - fixed with minor dynamic checks
set!	NO	(vector-set! js (quote 0) (vector-ref (vector-ref old-jss (quote 0)) j0))	.../array/typed-array-indexing.rkt:264:13	Yes - made safe by writing code in a different way
ref	NO	(vector-ref (vector-ref old-jss (quote 0)) j0)	.../array/typed-array-indexing.rkt:264:38	No - too complex.
ref	NO	(vector-ref (vector-ref old-jss (quote 0)) j0)	.../array/typed-array-indexing.rkt:264:38	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 0))	.../array/typed-array-indexing.rkt:264:57	Yes - made safe by writing code in a different way
ref	NO	(vector-ref old-jss (quote 0))	.../array/typed-array-indexing.rkt:264:57	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 0))	.../array/typed-array-indexing.rkt:264:57	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 0))	.../array/typed-array-indexing.rkt:264:57	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 0))	.../array/typed-array-indexing.rkt:264:57	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 0))	.../array/typed-array-indexing.rkt:264:57	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-array-indexing.rkt:266:13	Yes - made safe by writing code in a different way
ref	NO	(vector-ref js (quote 0))	.../array/typed-array-indexing.rkt:272:24	Yes - made safe by writing code in a different way
ref	NO	(vector-ref js (quote 1))	.../array/typed-array-indexing.rkt:273:24	Yes - made safe by writing code in a different way

set!	NO	(vector-set! js (quote 0) (vector-ref (vector-ref old-jss (quote 0)) j0))	.../array/typed-array-indexing.rkt:274:13	Yes - made safe by writing code in a different way
ref	NO	(vector-ref (vector-ref old-jss (quote 0)) j0)	.../array/typed-array-indexing.rkt:274:38	No - too complex.
ref	NO	(vector-ref (vector-ref old-jss (quote 0)) j0)	.../array/typed-array-indexing.rkt:274:38	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 0))	.../array/typed-array-indexing.rkt:274:57	Yes - made safe by writing code in a different way
ref	NO	(vector-ref old-jss (quote 0))	.../array/typed-array-indexing.rkt:274:57	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 0))	.../array/typed-array-indexing.rkt:274:57	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 0))	.../array/typed-array-indexing.rkt:274:57	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 0))	.../array/typed-array-indexing.rkt:274:57	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 0))	.../array/typed-array-indexing.rkt:274:57	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! js (quote 1) (vector-ref (vector-ref old-jss (quote 1)) j1))	.../array/typed-array-indexing.rkt:275:13	Yes - made safe by writing code in a different way
ref	NO	(vector-ref (vector-ref old-jss (quote 1)) j1)	.../array/typed-array-indexing.rkt:275:38	No - too complex.
ref	NO	(vector-ref (vector-ref old-jss (quote 1)) j1)	.../array/typed-array-indexing.rkt:275:38	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 1))	.../array/typed-array-indexing.rkt:275:57	Yes - made safe by writing code in a different way
ref	NO	(vector-ref old-jss (quote 1))	.../array/typed-array-indexing.rkt:275:57	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 1))	.../array/typed-array-indexing.rkt:275:57	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 1))	.../array/typed-array-indexing.rkt:275:57	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 1))	.../array/typed-array-indexing.rkt:275:57	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss (quote 1))	.../array/typed-array-indexing.rkt:275:57	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-array-indexing.rkt:277:13	Yes - made safe by writing code in a different way

set!	NO	(vector-set! js (quote 1) j1)	.../array/typed-array-indexing.rkt:278:13	Yes - made safe by writing code in a different way
ref	NO	(vector-ref new-js i)	.../array/typed-array-indexing.rkt:288:34	No - too complex.
ref	NO	(vector-ref (vector-ref old-jss i) new-ji)	.../array/typed-array-indexing.rkt:289:34	No - too complex.
ref	NO	(vector-ref old-jss i)	.../array/typed-array-indexing.rkt:289:53	No - too complex.
ref	NO	(vector-ref old-jss i)	.../array/typed-array-indexing.rkt:289:53	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jss i)	.../array/typed-array-indexing.rkt:289:53	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! old-js i old-ji)	.../array/typed-array-indexing.rkt:290:19	Yes - made safe by writing code in a different way
ref	NO	(vector-ref ds (quote 0))	.../array/typed-array-indexing.rkt:59:4	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 0))	.../array/typed-array-indexing.rkt:59:4	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-array-indexing.rkt:59:4	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 1) j1)	.../array/typed-array-indexing.rkt:59:4	Duplicate - from macro usage
set!	NO	(vector-set! js i ji)	.../array/typed-array-indexing.rkt:59:4	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 1))	.../array/typed-array-indexing.rkt:59:4	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ds i)	.../array/typed-array-indexing.rkt:59:4	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-array-indexing.rkt:59:4	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref js k)	.../array/typed-array-sequence.rkt:106:33	Yes - made safe by writing code in a different way
ref	NO	(vector-ref js k)	.../array/typed-array-sequence.rkt:106:33	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref arrs jk)	.../array/typed-array-sequence.rkt:108:44	No - call is *actually* potentially unsafe
ref	NO	(vector-ref arrs jk)	.../array/typed-array-sequence.rkt:108:44	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref ds k)	.../array/typed-array-sequence.rkt:71:20	Duplicate - from other (overloaded function types, etc)

ref	YES	(vector-ref ds k)	.../array/typed-array-sequence.rkt:71:20	Yes - works w/o changes
ref	YES	(vector-ref ds k)	.../array/typed-array-sequence.rkt:89:20	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref ds k)	.../array/typed-array-sequence.rkt:89:20	Yes - works w/o changes
ref	NO	(vector-ref vs (unsafe-array-index->value-index ds js))	.../array/typed-array-struct.rkt:102:22	No - too complex.
ref	NO	(vector-ref vs (unsafe-array-index->value-index ds js))	.../array/typed-array-struct.rkt:133:27	No - too complex.
ref	NO	(vector-ref ds k)	.../array/typed-array-struct.rkt:155:4	Yes - successfully made safe with annotations
ref	NO	(vector-ref ds (quote 0))	.../array/typed-array-struct.rkt:155:4	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 0))	.../array/typed-array-struct.rkt:155:4	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-array-struct.rkt:155:4	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 1) j1)	.../array/typed-array-struct.rkt:155:4	Duplicate - from macro usage
set!	NO	(vector-set! js i ji)	.../array/typed-array-struct.rkt:155:4	Duplicate - from macro usage
set!	NO	(vector-set! vs j ((#%expression (lambda (js j) (let-values (((js) (vector-copy-all js))) (delay (lambda () (proc js)))))) js j))	.../array/typed-array-struct.rkt:155:4	Duplicate - from macro usage
set!	NO	(vector-set! vs j ((#%expression (lambda (js j) (let-values (((js) (vector-copy-all js))) (delay (lambda () (proc js)))))) js j))	.../array/typed-array-struct.rkt:155:4	Duplicate - from macro usage
set!	NO	(vector-set! vs j ((#%expression (lambda (js j) (let-values (((js) (vector-copy-all js))) (delay (lambda () (proc js)))))) js j))	.../array/typed-array-struct.rkt:155:4	Duplicate - from macro usage
set!	NO	(vector-set! vs j ((#%expression (lambda (js j) (let-values (((js) (vector-copy-all js))) (delay (lambda () (proc js)))))) js j))	.../array/typed-array-struct.rkt:155:4	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 1))	.../array/typed-array-struct.rkt:155:4	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ds i)	.../array/typed-array-struct.rkt:155:4	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-array-struct.rkt:155:4	Duplicate - from other (overloaded function types, etc)
ref	YES	(unsafe-vector-ref vec pos)	.../array/typed-array-struct.rkt:163:20	Yes - works w/o changes

ref	NO	(vector-ref vs (unsafe-array-index->value-index ds js))	.../array/typed-array-struct.rkt:166:13	No - too complex.
ref	NO	(vector-ref ds (quote 0))	.../array/typed-array-struct.rkt:215:2	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 0))	.../array/typed-array-struct.rkt:215:2	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-array-struct.rkt:215:2	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 1) j1)	.../array/typed-array-struct.rkt:215:2	Duplicate - from macro usage
set!	NO	(vector-set! js i ji)	.../array/typed-array-struct.rkt:215:2	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 1))	.../array/typed-array-struct.rkt:215:2	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ds i)	.../array/typed-array-struct.rkt:215:2	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-array-struct.rkt:215:2	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ds (quote 0))	.../array/typed-array-struct.rkt:24:9	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 0))	.../array/typed-array-struct.rkt:24:9	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-array-struct.rkt:24:9	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 1) j1)	.../array/typed-array-struct.rkt:24:9	Duplicate - from macro usage
set!	NO	(vector-set! js i ji)	.../array/typed-array-struct.rkt:24:9	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 1))	.../array/typed-array-struct.rkt:24:9	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ds i)	.../array/typed-array-struct.rkt:24:9	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-array-struct.rkt:24:9	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ds (quote 0))	.../array/typed-array-struct.rkt:37:2	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 0))	.../array/typed-array-struct.rkt:37:2	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-array-struct.rkt:37:2	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 1) j1)	.../array/typed-array-struct.rkt:37:2	Duplicate - from macro usage
set!	NO	(vector-set! js i ji)	.../array/typed-array-struct.rkt:37:2	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 1))	.../array/typed-array-struct.rkt:37:2	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ds i)	.../array/typed-array-struct.rkt:37:2	Duplicate - from other (overloaded function types, etc)

set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-array-struct.rkt:37:2	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ds (quote 0))	.../array/typed-array-struct.rkt:99:21	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 0))	.../array/typed-array-struct.rkt:99:21	Duplicate - from macro usage
ref	NO	(vector-ref ds k)	.../array/typed-array-struct.rkt:99:21	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-array-struct.rkt:99:21	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 1) j1)	.../array/typed-array-struct.rkt:99:21	Duplicate - from macro usage
set!	NO	(vector-set! js i ji)	.../array/typed-array-struct.rkt:99:21	Duplicate - from macro usage
set!	NO	(vector-set! vs j ((#%expression (lambda (js j) (old-f js))) js j))	.../array/typed-array-struct.rkt:99:21	Duplicate - from macro usage
set!	NO	(vector-set! vs j ((#%expression (lambda (js j) (old-f js))) js j))	.../array/typed-array-struct.rkt:99:21	Duplicate - from macro usage
set!	NO	(vector-set! vs j ((#%expression (lambda (js j) (old-f js))) js j))	.../array/typed-array-struct.rkt:99:21	Duplicate - from macro usage
set!	NO	(vector-set! vs j ((#%expression (lambda (js j) (old-f js))) js j))	.../array/typed-array-struct.rkt:99:21	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 1))	.../array/typed-array-struct.rkt:99:21	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ds i)	.../array/typed-array-struct.rkt:99:21	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-array-struct.rkt:99:21	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref ds k)	.../array/typed-array-transform.rkt:102:37	Yes - works w/o changes
ref	YES	(vector-ref ds k)	.../array/typed-array-transform.rkt:103:68	Yes - works w/o changes
ref	NO	(vector-ref js (quote 0))	.../array/typed-array-transform.rkt:149:28	No - too complex.
ref	NO	(vector-ref ds k)	.../array/typed-array-transform.rkt:170:48	No - too complex.
set!	NO	(vector-set! new-ds k new-dk)	.../array/typed-array-transform.rkt:189:7	Yes - fixed with minor dynamic checks
set!	NO	(vector-set! new-ds k new-dk)	.../array/typed-array-transform.rkt:189:7	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! old-procs jk proc)	.../array/typed-array-transform.rkt:201:34	No - too complex.

set!	NO	(vector-set! old-procs jk proc)	.../array/typed-array-transform.rkt:201:34	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! old-jks jk i)	.../array/typed-array-transform.rkt:202:34	No - too complex.
set!	NO	(vector-set! old-jks jk i)	.../array/typed-array-transform.rkt:202:34	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref js k)	.../array/typed-array-transform.rkt:208:29	Yes - fixed with minor dynamic checks
ref	NO	(vector-ref js k)	.../array/typed-array-transform.rkt:208:29	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! js k (vector-ref old-jks jk))	.../array/typed-array-transform.rkt:209:18	Yes - fixed with minor dynamic checks
set!	NO	(vector-set! js k (vector-ref old-jks jk))	.../array/typed-array-transform.rkt:209:18	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jks jk)	.../array/typed-array-transform.rkt:209:43	No - requires non-linear math.
ref	NO	(vector-ref old-jks jk)	.../array/typed-array-transform.rkt:209:43	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jks jk)	.../array/typed-array-transform.rkt:209:43	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-jks jk)	.../array/typed-array-transform.rkt:209:43	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref old-procs jk)	.../array/typed-array-transform.rkt:210:29	No - requires non-linear math.
ref	NO	(vector-ref old-procs jk)	.../array/typed-array-transform.rkt:210:29	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! js k jk)	.../array/typed-array-transform.rkt:211:18	Yes - fixed with minor dynamic checks
set!	NO	(vector-set! js k jk)	.../array/typed-array-transform.rkt:211:18	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! old-js (vector-ref perm i) (vector-ref js i))	.../array/typed-array-transform.rkt:44:35	Duplicate - from macro usage
ref	NO	(vector-ref perm i)	.../array/typed-array-transform.rkt:45:55	No - too complex.
ref	NO	(vector-ref perm i)	.../array/typed-array-transform.rkt:45:55	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref js i)	.../array/typed-array-transform.rkt:46:55	No - too complex.

ref	NO	(vector-ref js i)	.../array/typed-array-transform.rkt:46:55	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref new-ds i0)	.../array/typed-array-transform.rkt:61:20	Yes - works w/o changes
ref	NO	(vector-ref new-ds i1)	.../array/typed-array-transform.rkt:62:20	Yes - works w/o changes
set!	NO	(vector-set! new-ds i0 j1)	.../array/typed-array-transform.rkt:63:9	Yes - works w/o changes
set!	NO	(vector-set! new-ds i1 j0)	.../array/typed-array-transform.rkt:64:9	Yes - works w/o changes
ref	NO	(vector-ref js i0)	.../array/typed-array-transform.rkt:69:31	Yes - made safe by writing code in a different way
ref	NO	(vector-ref js i1)	.../array/typed-array-transform.rkt:70:31	Yes - made safe by writing code in a different way
set!	NO	(vector-set! js i0 j1)	.../array/typed-array-transform.rkt:71:20	Yes - made safe by writing code in a different way
set!	NO	(vector-set! js i1 j0)	.../array/typed-array-transform.rkt:72:20	Yes - made safe by writing code in a different way
set!	NO	(vector-set! js i0 j0)	.../array/typed-array-transform.rkt:74:20	Yes - made safe by writing code in a different way
set!	NO	(vector-set! js i1 j1)	.../array/typed-array-transform.rkt:75:20	Yes - made safe by writing code in a different way
ref	NO	(vector-ref vs j)	.../array/typed-mutable-array.rkt:22:49	No - too complex.
set!	NO	(vector-set! vs j v)	.../array/typed-mutable-array.rkt:23:61	No - too complex.
set!	NO	(vector-set! vs j ((#%expression (lambda (js j) (g js))) js j))	.../array/typed-mutable-array.rkt:51:27	No - too complex.
ref	NO	(vector-ref ds (quote 0))	.../array/typed-mutable-array.rkt:51:27	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 0))	.../array/typed-mutable-array.rkt:51:27	Duplicate - from macro usage
ref	NO	(vector-ref ds k)	.../array/typed-mutable-array.rkt:51:27	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-mutable-array.rkt:51:27	Duplicate - from macro usage
set!	NO	(vector-set! js (quote 1) j1)	.../array/typed-mutable-array.rkt:51:27	Duplicate - from macro usage
set!	NO	(vector-set! js i ji)	.../array/typed-mutable-array.rkt:51:27	Duplicate - from macro usage
set!	NO	(vector-set! vs j ((#%expression (lambda (js j) (g js))) js j))	.../array/typed-mutable-array.rkt:51:27	Duplicate - from macro usage
set!	NO	(vector-set! vs j ((#%expression (lambda (js j) (g js))) js j))	.../array/typed-mutable-array.rkt:51:27	Duplicate - from macro usage

set!	NO	(vector-set! vs j ((#%expression (lambda (js j) (g js))) js j))	.../array/typed-mutable-array.rkt:51:27	Duplicate - from macro usage
ref	NO	(vector-ref ds (quote 1))	.../array/typed-mutable-array.rkt:51:27	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ds i)	.../array/typed-mutable-array.rkt:51:27	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! js (quote 0) j0)	.../array/typed-mutable-array.rkt:51:27	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref ds i)	.../array/typed-utils.rkt:113:22	Yes - works w/o changes
ref	NO	(vector-ref js i)	.../array/typed-utils.rkt:114:22	Yes - fixed with minor dynamic checks
set!	NO	(vector-set! new-js i ji)	.../array/typed-utils.rkt:116:18	Yes - successfully made safe with annotations
ref	YES	(vector-ref vec (quote 0))	.../array/typed-utils.rkt:128:54	Yes - works w/o changes
ref	NO	(vector-ref vec i)	.../array/typed-utils.rkt:131:39	Yes - successfully made safe with annotations
ref	NO	(vector-ref vec i)	.../array/typed-utils.rkt:131:39	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! new-vec i (vector-ref vec i))	.../array/typed-utils.rkt:131:9	Yes - successfully made safe with annotations
set!	NO	(vector-set! new-vec i (vector-ref vec (+ i (quote 1))))	.../array/typed-utils.rkt:135:14	Yes - successfully made safe with annotations
ref	YES	(vector-ref vec (+ i (quote 1)))	.../array/typed-utils.rkt:135:44	Yes - works w/o changes
ref	YES	(vector-ref vec (+ i (quote 1)))	.../array/typed-utils.rkt:135:44	Yes - works w/o changes
ref	NO	(vector-ref vec i)	.../array/typed-utils.rkt:145:36	Yes - fixed with minor code changes
ref	NO	(vector-ref vec i)	.../array/typed-utils.rkt:145:36	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! dst-vec i (vector-ref vec i))	.../array/typed-utils.rkt:145:6	Yes - fixed with minor code changes
ref	YES	(vector-ref vec i)	.../array/typed-utils.rkt:150:40	Yes - works w/o changes
ref	YES	(vector-ref vec i)	.../array/typed-utils.rkt:150:40	Yes - works w/o changes
set!	NO	(vector-set! dst-vec i+1 (vector-ref vec i))	.../array/typed-utils.rkt:150:8	Yes - successfully made safe with annotations
ref	NO	(vector-ref visited k)	.../array/typed-utils.rkt:173:25	Yes - successfully made safe with annotations

set!	NO	(vector-set! visited k (quote #t))	.../array/typed-utils.rkt:174:31	Yes - successfully made safe with annotations
set!	NO	(vector-set! new-ds i (vector-ref ds k))	.../array/typed-utils.rkt:175:18	Yes - successfully made safe with annotations
ref	YES	(vector-ref ds k)	.../array/typed-utils.rkt:175:47	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref ds k)	.../array/typed-utils.rkt:175:47	Yes - works w/o changes
set!	NO	(vector-set! new-perm i k)	.../array/typed-utils.rkt:176:18	Yes - successfully made safe with annotations
ref	NO	(vector-ref js k)	.../array/typed-utils.rkt:202:19	Yes - successfully made safe with annotations
ref	NO	(vector-ref ds k)	.../array/typed-utils.rkt:203:19	Yes - successfully made safe with annotations
set!	NO	(vector-set! js k (quote 0))	.../array/typed-utils.rkt:206:17	Yes - successfully made safe with annotations
set!	NO	(vector-set! js k jk)	.../array/typed-utils.rkt:209:17	Yes - successfully made safe with annotations
ref	YES	(vector-ref js (quote 0))	.../array/typed-utils.rkt:23:72	Yes - works w/o changes
set!	NO	(vector-set! new-js i (vector-ref js i))	.../array/typed-utils.rkt:25:42	Yes - successfully made safe with annotations
ref	YES	(vector-ref js i)	.../array/typed-utils.rkt:25:71	Yes - works w/o changes
ref	YES	(vector-ref js i)	.../array/typed-utils.rkt:25:71	Yes - works w/o changes
ref	YES	(vector-ref ds i)	.../array/typed-utils.rkt:36:39	Yes - works w/o changes
ref	YES	(vector-ref ds i)	.../array/typed-utils.rkt:52:24	Yes - works w/o changes
set!	NO	(vector-set! new-ds i di)	.../array/typed-utils.rkt:53:33	Yes - successfully made safe with annotations
ref	YES	(vector-ref ds i)	.../array/typed-utils.rkt:63:24	Yes - works w/o changes
ref	NO	(vector-ref js i)	.../array/typed-utils.rkt:64:24	Yes - made safe by writing code in a different way
ref	NO	(vector-ref ds i)	.../array/typed-utils.rkt:75:54	Yes - successfully made safe with annotations
set!	NO	(vector-set! js i (fxquotient j s))	.../array/typed-utils.rkt:76:23	Yes - successfully made safe with annotations
ref	YES	(vector-ref ds i)	.../array/typed-utils.rkt:98:22	Yes - works w/o changes
ref	YES	(vector-ref js i)	.../array/typed-utils.rkt:99:22	Yes - works w/o changes

ref	YES	(vector-ref vec i)	.../distributions/impl/walker-table.rkt:52:21	Yes - works w/o changes
ref	NO	(vector-ref positive-ds i)	.../flonum/expansion/expansion-exp-reduction.rkt:51:27	No - requires non-linear math.
ref	NO	(vector-ref base^ks i)	.../flonum/expansion/expansion-exp-reduction.rkt:52:21	No - requires non-linear math.
ref	NO	(vector-ref negative-ds i)	.../flonum/expansion/expansion-exp-reduction.rkt:59:27	No - requires non-linear math.
ref	NO	(vector-ref base^ks i)	.../flonum/expansion/expansion-exp-reduction.rkt:60:21	No - requires non-linear math.
ref	NO	(vector-ref flexp2s (fl->exact-integer (fl+ x (quote 1074.0))))	.../flonum/flonum-exp.rkt:105:13	No - requires non-linear math.
ref	YES	(unsafe-vector-ref vec pos)	.../flonum/flvector.rkt:86:2	Duplicate - from macro usage
set!	NO	(unsafe-vector-set! vs i v)	.../flonum/flvector.rkt:91:2	No - would require changes to Math lib API
set!	NO	(vector-set! vec-fs m (* (- (/ (+ m (quote 1)) (+ m (quote 2)))) (+ (* (/ (- m (quote 1)) (* (quote 3) m)) (vector-ref vec-fs (- m (quote 1)))) (#%expression (let-values (((start) (quote 3)) ((end) m) ((inc) (quote 1))) (if (if (real? start) (if (real? end) (#%expression (real? inc)) (quote #f)) (quote #f)) (void) (let-values () (in-range start end inc))) ((letrec-values (((for-loop) (lambda (sum pos) (if (#%expression (< pos end)) (let-values (((j) pos)) (if (#%expression (quote #t)) (let-values (((sum) (let-values (((sum) sum)) (let-values (((sum) (let-values () (+ sum (/ (* (vector-ref vec-fs (- j (quote 1))) (vector-ref vec-fs (+ m (quote 1) (- j)))) (+ m (quote 2) (- j))))))) (values sum))))) (if (#%expression (quote #t)) (for-loop sum (+ pos inc)) sum)) sum)) sum)))) for-loop) (quote 0) start))))))	.../functions/incomplete-gamma/gamma-temme.rkt:27:4	Yes - successfully made safe with annotations
ref	NO	(vector-ref vec-fs (- m (quote 1)))	.../functions/incomplete-gamma/gamma-temme.rkt:30:34	Yes - successfully made safe with annotations
ref	NO	(vector-ref vec-fs (- m (quote 1)))	.../functions/incomplete-gamma/gamma-temme.rkt:30:34	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vec-fs (- j (quote 1)))	.../functions/incomplete-gamma/gamma-temme.rkt:32:26	Yes - successfully made safe with annotations
ref	NO	(vector-ref vec-fs (- j (quote 1)))	.../functions/incomplete-gamma/gamma-temme.rkt:32:26	Duplicate - from other (overloaded function types, etc)

ref	NO	(vector-ref vec-fs (+ m (quote 1) (- j)))	.../functions/incomplete-gamma/gamma-temme.rkt:33:26	Yes - successfully made safe with annotations
ref	NO	(vector-ref vec-fs (+ m (quote 1) (- j)))	.../functions/incomplete-gamma/gamma-temme.rkt:33:26	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! ij (quote 1) j)	.../matrix/matrix-basic.rkt:105:13	Yes - made safe by writing code in a different way
set!	NO	(vector-set! ij (quote 1) (quote 0))	.../matrix/matrix-basic.rkt:107:13	Yes - made safe by writing code in a different way
ref	NO	(vector-ref js (quote 0))	.../matrix/matrix-basic.rkt:130:16	Yes - made safe by writing code in a different way
ref	NO	(vector-ref ij (quote 0))	.../matrix/matrix-basic.rkt:145:19	Yes - made safe by writing code in a different way
ref	NO	(vector-ref ij (quote 1))	.../matrix/matrix-basic.rkt:146:19	Yes - made safe by writing code in a different way
ref	NO	(vector-ref ij (quote 0))	.../matrix/matrix-basic.rkt:161:19	Yes - made safe by writing code in a different way
ref	NO	(vector-ref ij (quote 1))	.../matrix/matrix-basic.rkt:162:19	Yes - made safe by writing code in a different way
ref	NO	(vector-ref rows i0)	.../matrix/matrix-basic.rkt:461:17	Yes - fixed with minor code changes
ref	NO	(vector-ref rows i1)	.../matrix/matrix-basic.rkt:462:17	Yes - fixed with minor code changes
set!	NO	(vector-set! ij (quote 0) i)	.../matrix/matrix-basic.rkt:89:13	Yes - made safe by writing code in a different way
set!	NO	(vector-set! ij (quote 0) (quote 0))	.../matrix/matrix-basic.rkt:91:13	Yes - made safe by writing code in a different way
set!	NO	(vector-set! vs (fx+ res-i i) k)	.../matrix/matrix-constructors.rkt:105:10	No - too complex.
set!	NO	(vector-set! is (fx+ res-i i) (let-values (((val) i) ((pred) index?)) (if (pred val) val (error (format (quote Assertion ~a failed on ~v) pred val)))))	.../matrix/matrix-constructors.rkt:106:10	No - too complex.
set!	NO	(vector-set! hs (fx+ res-j j) k)	.../matrix/matrix-constructors.rkt:108:10	No - too complex.
set!	NO	(vector-set! js (fx+ res-j j) (let-values (((val) j) ((pred) index?)) (if (pred val) val (error (format (quote Assertion ~a failed on ~v) pred val)))))	.../matrix/matrix-constructors.rkt:109:10	No - too complex.
ref	NO	(vector-ref ij (quote 0))	.../matrix/matrix-constructors.rkt:116:16	Yes - made safe by writing code in a different way

ref	NO	(vector-ref ij (quote 1))	.../matrix/matrix-constructors.rkt:117:16	Yes - made safe by writing code in a different way
ref	NO	(vector-ref vs i)	.../matrix/matrix-constructors.rkt:118:16	No - requires non-linear math.
ref	NO	(vector-ref hs j)	.../matrix/matrix-constructors.rkt:119:20	No - requires non-linear math.
ref	NO	(vector-ref procs v)	.../matrix/matrix-constructors.rkt:120:26	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref is i)	.../matrix/matrix-constructors.rkt:121:24	No - requires non-linear math.
ref	NO	(vector-ref js j)	.../matrix/matrix-constructors.rkt:122:24	No - requires non-linear math.
set!	NO	(vector-set! ij (quote 0) iv)	.../matrix/matrix-constructors.rkt:123:13	Yes - made safe by writing code in a different way
set!	NO	(vector-set! ij (quote 1) jv)	.../matrix/matrix-constructors.rkt:124:13	Yes - made safe by writing code in a different way
set!	NO	(vector-set! ij (quote 0) i)	.../matrix/matrix-constructors.rkt:126:13	Yes - made safe by writing code in a different way
set!	NO	(vector-set! ij (quote 1) j)	.../matrix/matrix-constructors.rkt:127:13	Yes - made safe by writing code in a different way
ref	YES	(vector-ref as (quote 0))	.../matrix/matrix-constructors.rkt:139:11	Yes - works w/o changes
ref	NO	(vector-ref js (quote 0))	.../matrix/matrix-constructors.rkt:51:19	Yes - made safe by writing code in a different way
ref	NO	(vector-ref js (quote 1))	.../matrix/matrix-constructors.rkt:52:19	Yes - made safe by writing code in a different way
ref	NO	(vector-ref js (quote 0))	.../matrix/matrix-constructors.rkt:67:22	No - too complex.
ref	NO	(vector-ref js (quote 1))	.../matrix/matrix-constructors.rkt:68:24	No - too complex.
ref	NO	(vector-ref vs i)	.../matrix/matrix-constructors.rkt:68:51	No - too complex.
ref	YES	(unsafe-vector-ref vec pos)	.../matrix/matrix-constructors.rkt:85:27	Duplicate - from macro usage
set!	NO	(vector-set! js k (vector-ref ij (quote 0)))	.../matrix/matrix-conversion.rkt:102:34	No - too complex.
ref	NO	(vector-ref ij (quote 0))	.../matrix/matrix-conversion.rkt:102:59	Yes - made safe by writing code in a different way
ref	NO	(vector-ref ij (quote 0))	.../matrix/matrix-conversion.rkt:102:59	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref xss (quote 0))	.../matrix/matrix-conversion.rkt:139:34	Yes - works w/o changes
ref	YES	(vector-ref xss i)	.../matrix/matrix-conversion.rkt:143:54	Yes - works w/o changes
ref	YES	(vector-ref ds k)	.../matrix/matrix-conversion.rkt:75:38	Yes - works w/o changes
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/matrix-expt.rkt:26:14	No - too complex.

ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/matrix-expt.rkt:26:14	No - too complex.
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/matrix-expt.rkt:26:14	No - too complex.
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/matrix-expt.rkt:26:14	No - too complex.
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/matrix-expt.rkt:26:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/matrix-expt.rkt:26:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/matrix-expt.rkt:26:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/matrix-expt.rkt:26:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/matrix-expt.rkt:26:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/matrix-expt.rkt:26:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/matrix-expt.rkt:26:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/matrix-expt.rkt:26:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/matrix-expt.rkt:26:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/matrix-expt.rkt:26:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/matrix-expt.rkt:26:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/matrix-expt.rkt:26:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage

ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/matrix-expt.rkt:28:12	Duplicate - from macro usage
ref	NO	(vector-ref ij (quote 0))	.../matrix/matrix-expt.rkt:44:20	No - too complex.
ref	NO	(vector-ref ij (quote 0))	.../matrix/matrix-expt.rkt:44:20	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ij (quote 0))	.../matrix/matrix-expt.rkt:44:20	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ij (quote 0))	.../matrix/matrix-expt.rkt:44:20	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ij (quote 1))	.../matrix/matrix-expt.rkt:44:45	No - too complex.
ref	NO	(vector-ref ij (quote 1))	.../matrix/matrix-expt.rkt:44:45	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ij (quote 1))	.../matrix/matrix-expt.rkt:44:45	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ij (quote 1))	.../matrix/matrix-expt.rkt:44:45	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-gauss-elim.rkt:63:50	No - too complex.
ref	NO	(vector-ref rows i)	.../matrix/matrix-gauss-elim.rkt:63:50	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-gauss-elim.rkt:63:50	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-gauss-elim.rkt:63:50	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-gauss-elim.rkt:63:50	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-gauss-elim.rkt:63:50	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-gauss-elim.rkt:63:50	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-gauss-elim.rkt:63:50	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vss i)	.../matrix/matrix-gram-schmidt.rkt:28:35	Yes - works w/o changes
ref	YES	(vector-ref vss i)	.../matrix/matrix-gram-schmidt.rkt:28:35	Yes - works w/o changes
ref	YES	(vector-ref vss i)	.../matrix/matrix-gram-schmidt.rkt:28:35	Yes - works w/o changes
ref	YES	(vector-ref vss i)	.../matrix/matrix-gram-schmidt.rkt:28:35	Yes - works w/o changes

ref	NO	(vector-ref rows i)	.../matrix/matrix-gram-schmidt.rkt:41:24	Yes - successfully made safe with annotations
ref	NO	(vector-ref rows i)	.../matrix/matrix-gram-schmidt.rkt:41:24	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-gram-schmidt.rkt:41:24	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-gram-schmidt.rkt:41:24	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-gram-schmidt.rkt:57:22	Yes - successfully made safe with annotations
ref	NO	(vector-ref rows i)	.../matrix/matrix-gram-schmidt.rkt:57:22	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-gram-schmidt.rkt:57:22	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-gram-schmidt.rkt:57:22	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref rows i)	.../matrix/matrix-gram-schmidt.rkt:62:31	Yes - works w/o changes
ref	YES	(vector-ref rows i)	.../matrix/matrix-gram-schmidt.rkt:62:31	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref rows i)	.../matrix/matrix-gram-schmidt.rkt:62:31	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref rows i)	.../matrix/matrix-gram-schmidt.rkt:62:31	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! ys (+ (* l m) i) y_li)	.../matrix/matrix-lu.rkt:57:20	No - too complex.
set!	NO	(vector-set! ys (+ (* l m) i) y_li)	.../matrix/matrix-lu.rkt:57:20	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! ys (+ (* l m) i) y_li)	.../matrix/matrix-lu.rkt:57:20	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! ys (+ (* l m) i) y_li)	.../matrix/matrix-lu.rkt:57:20	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows l)	.../matrix/matrix-lu.rkt:59:40	No - would require dep struct types (unimplemented feature)
ref	NO	(vector-ref rows l)	.../matrix/matrix-lu.rkt:59:40	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows l)	.../matrix/matrix-lu.rkt:59:40	Duplicate - from other (overloaded function types, etc)

ref	NO	(vector-ref rows l)	.../matrix/matrix-lu.rkt:59:40	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-lu.rkt:60:40	No - would require dep struct types (unimplemented feature)
ref	NO	(vector-ref rows i)	.../matrix/matrix-lu.rkt:60:40	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-lu.rkt:60:40	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/matrix-lu.rkt:60:40	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! ys j (one* (vector-ref ys j)))	.../matrix/matrix-lu.rkt:69:12	Yes - fixed with minor dynamic checks
set!	NO	(vector-set! ys j (one* (vector-ref ys j)))	.../matrix/matrix-lu.rkt:69:12	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! ys j (one* (vector-ref ys j)))	.../matrix/matrix-lu.rkt:69:12	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! ys j (one* (vector-ref ys j)))	.../matrix/matrix-lu.rkt:69:12	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ys j)	.../matrix/matrix-lu.rkt:69:36	Yes - fixed with minor dynamic checks
ref	NO	(vector-ref ys j)	.../matrix/matrix-lu.rkt:69:36	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ys j)	.../matrix/matrix-lu.rkt:69:36	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ys j)	.../matrix/matrix-lu.rkt:69:36	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ys j)	.../matrix/matrix-lu.rkt:69:36	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ys j)	.../matrix/matrix-lu.rkt:69:36	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ys j)	.../matrix/matrix-lu.rkt:69:36	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref ys j)	.../matrix/matrix-lu.rkt:69:36	Duplicate - from other (overloaded function types, etc)
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/matrix-operator-norm.rkt:154:38	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/matrix-operator-norm.rkt:154:38	Duplicate - from macro usage

ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/matrix-operator-norm.rkt:154:38	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/matrix-operator-norm.rkt:154:38	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/matrix-operator-norm.rkt:154:38	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/matrix-operator-norm.rkt:154:38	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/matrix-operator-norm.rkt:154:38	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/matrix-operator-norm.rkt:154:38	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/matrix-qr.rkt:45:35	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp1 (quote 0))	.../matrix/matrix-solve.rkt:35:14	No - requires non-linear math.
ref	NO	(unsafe-vector-ref temp1 (quote 0))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp1 (quote 0))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp1 (quote 0))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage

ref	NO	(unsafe-vector-ref temp1 (quote 1))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp1 (quote 1))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp1 (quote 1))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp1 (quote 1))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp1 (quote 2))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp1 (quote 2))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp1 (quote 2))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp1 (quote 2))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp1 (quote 3))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp1 (quote 3))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp1 (quote 3))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp1 (quote 3))	.../matrix/matrix-solve.rkt:35:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 0))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 0))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 0))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 0))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 1))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 1))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 1))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 1))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 2))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 2))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 2))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 2))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 3))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 3))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 3))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 3))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 4))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 4))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage

ref	NO	(unsafe-vector-ref temp14 (quote 4))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 4))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 5))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 5))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 5))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 5))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 6))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 6))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 6))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 6))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 7))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 7))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 7))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 7))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 8))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 8))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 8))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref temp14 (quote 8))	.../matrix/matrix-solve.rkt:38:14	Duplicate - from macro usage
ref	YES	(vector-ref ds (quote 0))	.../matrix/matrix-types.rkt:30:18	Yes - works w/o changes
ref	YES	(vector-ref ds (quote 1))	.../matrix/matrix-types.rkt:31:18	Yes - works w/o changes
ref	YES	(vector-ref ds (quote 0))	.../matrix/matrix-types.rkt:38:10	Yes - works w/o changes
ref	YES	(vector-ref ds (quote 1))	.../matrix/matrix-types.rkt:39:10	Yes - works w/o changes
ref	YES	(vector-ref ds (quote 0))	.../matrix/matrix-types.rkt:45:10	Yes - works w/o changes
ref	YES	(vector-ref ds (quote 1))	.../matrix/matrix-types.rkt:46:10	Yes - works w/o changes
ref	YES	(vector-ref ds (quote 0))	.../matrix/matrix-types.rkt:53:14	Yes - works w/o changes
ref	YES	(vector-ref ds (quote 1))	.../matrix/matrix-types.rkt:54:14	Yes - works w/o changes
ref	NO	(vector-ref (Array-shape arr) (quote 0))	.../matrix/matrix-types.rkt:59:31	No - would require dep struct types (unimplemented feature)
ref	NO	(vector-ref (Array-shape a) (quote 0))	.../matrix/matrix-types.rkt:64:22	No - would require dep struct types (unimplemented feature)

ref	NO	(vector-ref (Array-shape a) (quote 1))	.../matrix/matrix-types.rkt:69:22	No - would require dep struct types (unimplemented feature)
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data (fx+ i k))	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref arr-data i)	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data (fx+ j k))	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(unsafe-vector-ref brr-data j)	.../matrix/typed-matrix-arithmetic.rkt:75:27	Duplicate - from macro usage
ref	NO	(vector-ref arr-data (quote 0))	.../matrix/untyped-matrix-arithmetic.rkt:31:40	No - too complex.
ref	NO	(vector-ref ij (quote 0))	.../matrix/untyped-matrix-arithmetic.rkt:40:31	No - too complex.

ref	NO	(vector-ref ij (quote 1))	.../matrix/untyped-matrix-arithmetic.rkt:41:31	No - too complex.
ref	NO	(vector-ref rows i)	.../matrix/utils.rkt:114:16	No - would require dep struct types (unimplemented feature)
ref	NO	(vector-ref rows i)	.../matrix/utils.rkt:114:16	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/utils.rkt:114:16	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows i)	.../matrix/utils.rkt:114:16	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows l)	.../matrix/utils.rkt:118:22	No - would require dep struct types (unimplemented feature)
ref	NO	(vector-ref rows l)	.../matrix/utils.rkt:118:22	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows l)	.../matrix/utils.rkt:118:22	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref rows l)	.../matrix/utils.rkt:118:22	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref row_l j)	.../matrix/utils.rkt:119:21	No - would require dep struct types (unimplemented feature)
ref	NO	(vector-ref row_l j)	.../matrix/utils.rkt:119:21	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref row_l j)	.../matrix/utils.rkt:119:21	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref row_l j)	.../matrix/utils.rkt:119:21	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! row_l j (- x_lj x_lj))	.../matrix/utils.rkt:123:10	No - would require dep struct types (unimplemented feature)
set!	NO	(vector-set! row_l j (- x_lj x_lj))	.../matrix/utils.rkt:123:10	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! row_l j (- x_lj x_lj))	.../matrix/utils.rkt:123:10	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! row_l j (- x_lj x_lj))	.../matrix/utils.rkt:123:10	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref (vector-ref vss i) j)	.../matrix/utils.rkt:60:2	No - would require dep struct types (unimplemented feature)
ref	NO	(vector-ref vss i)	.../matrix/utils.rkt:60:21	No - would require dep struct types (unimplemented feature)

ref	NO	(vector-ref vss i)	.../matrix/utils.rkt:60:21	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vss i)	.../matrix/utils.rkt:60:21	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! global-bs (quote 0) (quote 1))	.../number-theory/bernoulli.rkt:15:0	Yes - successfully made safe with annotations
set!	NO	(vector-set! global-bs (quote 1) (quote -1/2))	.../number-theory/bernoulli.rkt:16:0	Yes - successfully made safe with annotations
set!	NO	(vector-set! E (quote 0) (quote 1))	.../number-theory/eulerian-number.rkt:19:5	Yes - successfully made safe with annotations
ref	NO	(vector-ref E j)	.../number-theory/eulerian-number.rkt:22:40	Yes - fixed with minor code changes
ref	NO	(vector-ref E j)	.../number-theory/eulerian-number.rkt:22:40	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! E j (+ (* (+ j (quote 1)) (vector-ref E j)) (* (- i j) (vector-ref E (- j (quote 1))))))	.../number-theory/eulerian-number.rkt:22:9	Yes - fixed with minor code changes
ref	NO	(vector-ref E (- j (quote 1)))	.../number-theory/eulerian-number.rkt:23:40	Yes - fixed with minor code changes
ref	NO	(vector-ref E (- j (quote 1)))	.../number-theory/eulerian-number.rkt:23:40	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref E k)	.../number-theory/eulerian-number.rkt:24:13	Yes - fixed with minor code changes
ref	NO	(vector-ref fact-table n)	.../number-theory/factorial.rkt:30:36	Yes - successfully made safe with annotations
set!	YES	(vector-set! ps (quote 0) (quote #f))	.../number-theory/number-theory.rkt:200:4	Yes - works w/o changes
set!	YES	(vector-set! ps (quote 1) (quote #f))	.../number-theory/number-theory.rkt:201:4	Yes - works w/o changes
ref	NO	(vector-ref ps n)	.../number-theory/number-theory.rkt:203:12	Yes - successfully made safe with annotations
set!	NO	(vector-set! ps m (quote #f))	.../number-theory/number-theory.rkt:205:10	Yes - successfully made safe with annotations
ref	YES	(vector-ref ps n)	.../number-theory/number-theory.rkt:210:11	Yes - works w/o changes
set!	NO	(vector-set! cache (quote 0) (quote 1))	.../number-theory/partitions.rkt:26:0	No - too complex.
ref	NO	(vector-ref cache n)	.../number-theory/partitions.rkt:76:17	No - involves invariants on mutable data

set!	NO	(vector-set! cache n pn)	.../number-theory/partitions.rkt:80:9	No - involves invariants on mutable data
set!	NO	(vector-set! mod60->bits x (let-values (((val) b) ((pred) byte?)) (if (pred val) val (error (format (quote Assertion ~a failed on ~v) pred val)))))	.../number-theory/small-primes.rkt:22:2	Yes - fixed with minor code changes
ref	NO	(vector-ref mod60->bits m)	.../number-theory/small-primes.rkt:25:23	Yes - fixed with minor code changes
set!	NO	(vector-set! T (quote 1) (quote 1))	.../number-theory/tangent-number.rkt:15:9	Yes - successfully made safe with annotations
set!	NO	(vector-set! T i (* (add1 i) (vector-ref T (add1 i))))	.../number-theory/tangent-number.rkt:19:13	Yes - successfully made safe with annotations
ref	NO	(vector-ref T (add1 i))	.../number-theory/tangent-number.rkt:19:42	Yes - successfully made safe with annotations
ref	NO	(vector-ref T (add1 i))	.../number-theory/tangent-number.rkt:19:42	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! T k (quote 0))	.../number-theory/tangent-number.rkt:20:11	Yes - successfully made safe with annotations
set!	NO	(vector-set! T i (+ (vector-ref T i) (vector-ref T (- i (quote 2)))))	.../number-theory/tangent-number.rkt:23:13	Yes - successfully made safe with annotations
ref	NO	(vector-ref T i)	.../number-theory/tangent-number.rkt:23:33	Yes - successfully made safe with annotations
ref	NO	(vector-ref T i)	.../number-theory/tangent-number.rkt:23:33	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref T (- i (quote 2)))	.../number-theory/tangent-number.rkt:23:50	Yes - fixed with minor code changes
ref	NO	(vector-ref T (- i (quote 2)))	.../number-theory/tangent-number.rkt:23:50	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref T (quote 0))	.../number-theory/tangent-number.rkt:24:9	Yes - works w/o changes
set!	NO	(unsafe-vector-set! vs i v)	.../polynomial/chebyshev.rkt:46:8	Duplicate - from macro usage
set!	NO	(unsafe-vector-set! vs i v)	.../polynomial/chebyshev.rkt:46:8	Duplicate - from macro usage
set!	NO	(unsafe-vector-set! vs i v)	.../polynomial/chebyshev.rkt:46:8	Duplicate - from macro usage
ref	YES	(vector-ref cs i)	.../polynomial/chebyshev.rkt:72:19	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref cs i)	.../polynomial/chebyshev.rkt:72:19	Duplicate - from other (overloaded function types, etc)

ref	YES	(vector-ref cs i)	.../polynomial/chebyshev.rkt:72:19	Yes - works w/o changes
ref	NO	(vector-ref cs (- i (quote 1)))	.../polynomial/chebyshev.rkt:84:34	Yes - successfully made safe with annotations
ref	NO	(vector-ref cs (- i (quote 1)))	.../polynomial/chebyshev.rkt:84:34	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref cs (- i (quote 1)))	.../polynomial/chebyshev.rkt:84:34	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref xs (quote 0))	.../statistics/hpd-interval.rkt:52:25	Yes - successfully made safe with annotations
ref	NO	(vector-ref xs (- n (quote 1)))	.../statistics/hpd-interval.rkt:52:43	Yes - successfully made safe with annotations
ref	NO	(vector-ref xs (quote 0))	.../statistics/hpd-interval.rkt:54:18	Yes - successfully made safe with annotations
ref	NO	(vector-ref xs i1)	.../statistics/hpd-interval.rkt:55:18	Yes - successfully made safe with annotations
ref	YES	(vector-ref xs i0)	.../statistics/hpd-interval.rkt:65:35	Yes - works w/o changes
ref	NO	(vector-ref xs i1)	.../statistics/hpd-interval.rkt:66:35	Yes - successfully made safe with annotations
ref	NO	(vector-ref vs p)	.../statistics/quickselect.rkt:21:16	Yes - successfully made safe with annotations
set!	NO	(vector-set! vs p (vector-ref vs end))	.../statistics/quickselect.rkt:22:2	Yes - successfully made safe with annotations
ref	NO	(vector-ref vs end)	.../statistics/quickselect.rkt:22:27	Yes - successfully made safe with annotations
ref	NO	(vector-ref vs end)	.../statistics/quickselect.rkt:22:27	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! vs end pivot)	.../statistics/quickselect.rkt:23:2	Yes - successfully made safe with annotations
ref	NO	(vector-ref vs start)	.../statistics/quickselect.rkt:26:22	Yes - successfully made safe with annotations
set!	NO	(vector-set! vs end v1)	.../statistics/quickselect.rkt:29:18	Yes - successfully made safe with annotations
set!	NO	(vector-set! vs start (vector-ref vs end))	.../statistics/quickselect.rkt:31:20	Yes - successfully made safe with annotations
ref	NO	(vector-ref vs end)	.../statistics/quickselect.rkt:31:49	Yes - successfully made safe with annotations

ref	NO	(vector-ref vs end)	.../statistics/quickselect.rkt:31:49	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! vs start pivot)	.../statistics/quickselect.rkt:34:11	Yes - successfully made safe with annotations
ref	NO	(vector-ref vs start+k)	.../statistics/quickselect.rkt:50:20	No - too complex.
ref	NO	(vector-ref vs start)	.../statistics/quickselect.rkt:52:11	No - too complex.
ref	NO	(vector-ref vs (quote 0))	.../statistics/statistics-utils.rkt:88:2	No - would require changes to Math lib API
set!	NO	(unsafe-vector-set! vs i v)	.../statistics/statistics-utils.rkt:88:2	Duplicate - from macro usage
set!	NO	(unsafe-vector-set! vs i v)	.../statistics/statistics-utils.rkt:96:4	Duplicate - from macro usage
ref	NO	(vector-ref vs (quote 0))	.../statistics/statistics-utils.rkt:96:4	Duplicate - from macro usage
ref	YES	(unsafe-vector-ref vec pos)	.../vector/vector-fft.rkt:31:18	Duplicate - from macro usage
ref	YES	(unsafe-vector-ref vec pos)	.../vector/vector-fft.rkt:32:18	Duplicate - from macro usage
set!	NO	(unsafe-vector-set! vs i v)	.../vector/vector-fft.rkt:36:5	Duplicate - from macro usage
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:112:36	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:112:36	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:112:36	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:112:36	Yes - works w/o changes
set!	YES	(vector-set! vs i (/ (vector-ref vs i) s))	.../vector/vector-mutate.rkt:112:8	Yes - works w/o changes
set!	YES	(vector-set! vs i (/ (vector-ref vs i) s))	.../vector/vector-mutate.rkt:112:8	Yes - works w/o changes
set!	YES	(vector-set! vs i (/ (vector-ref vs i) s))	.../vector/vector-mutate.rkt:112:8	Yes - works w/o changes
set!	YES	(vector-set! vs i (/ (vector-ref vs i) s))	.../vector/vector-mutate.rkt:112:8	Yes - works w/o changes
ref	NO	(vector-ref vs0 (quote 0))	.../vector/vector-mutate.rkt:135:34	Yes - fixed with minor code changes
ref	NO	(vector-ref vs0 (quote 0))	.../vector/vector-mutate.rkt:135:34	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 (quote 0))	.../vector/vector-mutate.rkt:135:34	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 (quote 0))	.../vector/vector-mutate.rkt:135:34	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:140:26	Yes - fixed with minor code changes

ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:140:26	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:140:26	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:140:26	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:141:26	Yes - fixed with minor code changes
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:141:26	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:141:26	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:141:26	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! vs0 i (- v0 (* v1 s)))	.../vector/vector-mutate.rkt:142:15	Yes - fixed with minor code changes
set!	NO	(vector-set! vs0 i (- v0 (* v1 s)))	.../vector/vector-mutate.rkt:142:15	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! vs0 i (- v0 (* v1 s)))	.../vector/vector-mutate.rkt:142:15	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! vs0 i (- v0 (* v1 s)))	.../vector/vector-mutate.rkt:142:15	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:154:16	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:154:16	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:154:16	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:154:16	Yes - works w/o changes
set!	YES	(vector-set! vs i (- x x))	.../vector/vector-mutate.rkt:155:6	Yes - works w/o changes
set!	YES	(vector-set! vs i (- x x))	.../vector/vector-mutate.rkt:155:6	Yes - works w/o changes
set!	YES	(vector-set! vs i (- x x))	.../vector/vector-mutate.rkt:155:6	Yes - works w/o changes
set!	YES	(vector-set! vs i (- x x))	.../vector/vector-mutate.rkt:155:6	Yes - works w/o changes
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:166:18	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:166:18	Duplicate - from other (overloaded function types, etc)

ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:166:18	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:166:18	Yes - works w/o changes
ref	NO	(vector-ref vs i0)	.../vector/vector-mutate.rkt:27:16	Yes - fixed with minor dynamic checks
ref	NO	(vector-ref vs i1)	.../vector/vector-mutate.rkt:28:30	Yes - fixed with minor dynamic checks
ref	NO	(vector-ref vs i1)	.../vector/vector-mutate.rkt:28:30	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! vs i0 (vector-ref vs i1))	.../vector/vector-mutate.rkt:28:4	Yes - fixed with minor dynamic checks
set!	NO	(vector-set! vs i1 tmp)	.../vector/vector-mutate.rkt:29:4	Yes - fixed with minor dynamic checks
set!	YES	(vector-set! vs i (* v (vector-ref vs i)))	.../vector/vector-mutate.rkt:37:17	Duplicate - from other (overloaded function types, etc)
set!	YES	(vector-set! vs i (* v (vector-ref vs i)))	.../vector/vector-mutate.rkt:37:17	Duplicate - from other (overloaded function types, etc)
set!	YES	(vector-set! vs i (* v (vector-ref vs i)))	.../vector/vector-mutate.rkt:37:17	Duplicate - from other (overloaded function types, etc)
set!	YES	(vector-set! vs i (* v (vector-ref vs i)))	.../vector/vector-mutate.rkt:37:17	Yes - works w/o changes
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:37:47	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:37:47	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:37:47	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:37:47	Yes - works w/o changes
set!	NO	(vector-set! vs0 i (+ (vector-ref vs0 i) (* (vector-ref vs1 i) v)))	.../vector/vector-mutate.rkt:55:17	Yes - fixed with minor code changes
set!	NO	(vector-set! vs0 i (+ (vector-ref vs0 i) (* (vector-ref vs1 i) v)))	.../vector/vector-mutate.rkt:55:17	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! vs0 i (+ (vector-ref vs0 i) (* (vector-ref vs1 i) v)))	.../vector/vector-mutate.rkt:55:17	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! vs0 i (+ (vector-ref vs0 i) (* (vector-ref vs1 i) v)))	.../vector/vector-mutate.rkt:55:17	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! vs0 i (+ (vector-ref vs0 i) (* (vector-ref vs1 i) v)))	.../vector/vector-mutate.rkt:55:17	Duplicate - from other (overloaded function types, etc)

set!	NO	(vector-set! vs0 i (+ (vector-ref vs0 i) (* (vector-ref vs1 i) v)))	.../vector/vector-mutate.rkt:55:17	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! vs0 i (+ (vector-ref vs0 i) (* (vector-ref vs1 i) v)))	.../vector/vector-mutate.rkt:55:17	Duplicate - from other (overloaded function types, etc)
set!	NO	(vector-set! vs0 i (+ (vector-ref vs0 i) (* (vector-ref vs1 i) v)))	.../vector/vector-mutate.rkt:55:17	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Yes - fixed with minor code changes
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:55:46	Duplicate - from other (overloaded function types, etc)

ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Yes - fixed with minor code changes
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:56:49	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs (quote 0))	.../vector/vector-mutate.rkt:80:26	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs (quote 0))	.../vector/vector-mutate.rkt:80:26	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs (quote 0))	.../vector/vector-mutate.rkt:80:26	Yes - works w/o changes
ref	YES	(vector-ref vs (quote 0))	.../vector/vector-mutate.rkt:80:26	Yes - works w/o changes

ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:82:61	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:82:61	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:82:61	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector-mutate.rkt:82:61	Yes - works w/o changes
ref	NO	(vector-ref vs0 (quote 0))	.../vector/vector-mutate.rkt:93:20	Yes - fixed with minor code changes
ref	NO	(vector-ref vs0 (quote 0))	.../vector/vector-mutate.rkt:93:20	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 (quote 0))	.../vector/vector-mutate.rkt:93:20	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 (quote 0))	.../vector/vector-mutate.rkt:93:20	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 (quote 0))	.../vector/vector-mutate.rkt:94:20	Yes - fixed with minor code changes
ref	NO	(vector-ref vs1 (quote 0))	.../vector/vector-mutate.rkt:94:20	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 (quote 0))	.../vector/vector-mutate.rkt:94:20	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 (quote 0))	.../vector/vector-mutate.rkt:94:20	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:97:29	Yes - fixed with minor code changes
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:97:29	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:97:29	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs0 i)	.../vector/vector-mutate.rkt:97:29	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:98:29	Yes - fixed with minor code changes
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:98:29	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:98:29	Duplicate - from other (overloaded function types, etc)
ref	NO	(vector-ref vs1 i)	.../vector/vector-mutate.rkt:98:29	Duplicate - from other (overloaded function types, etc)
ref	YES	(vector-ref vs i)	.../vector/vector.rkt:20:24	Yes - works w/o changes

set!	YES	(vector-set! vs i v)	.../vector/vector.rkt:23:20	Yes - works w/o changes