

11iM-Λ[M-ΛT 24, 16 21:34 **hw09-CHECKED-output-trace** Page 1/8

```

1 dshin@acacia:checked$ checked -t /home/pl/hw09/tests/const-negative
2 Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.
3 > % =====
4 % negative constant. (value=-27)
5
6 -27
7
8 % =====
9 |+ exp=-27
10 || tenv=[x=int,v=int,i=int]
11 |- type=int
12 int
13 -27
14 >
15 dshin@acacia:checked$ checked -t /home/pl/hw09/tests/const-positive
16 Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.
17 > % =====
18 % positive constant. (value=12)
19
20 12
21
22 % =====
23 |+ exp=12
24 || tenv=[x=int,v=int,i=int]
25 |- type=int
26 int
27 12
28 >
29 dshin@acacia:checked$ checked -t /home/pl/hw09/tests/diff-nested-left
30 Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.
31 > % =====
32 % nested diff. (-11)
33
34 -(-(44,33),22)
35
36 % =====
37 |+ exp=-(-(44,33),22)
38 || tenv=[x=int,v=int,i=int]
39 ||+ exp=-(44,33)
40 ||| tenv=[x=int,v=int,i=int]
41 |||+ exp=44
42 |||| tenv=[x=int,v=int,i=int]
43 |||- type=int
44 |||+ exp=33
45 |||| tenv=[x=int,v=int,i=int]
46 |||- type=int
47 ||- type=int
48 ||+ exp=22
49 ||| tenv=[x=int,v=int,i=int]
50 ||- type=int
51 |- type=int
52 int
53 -11
54 >
55 dshin@acacia:checked$ checked -t /home/pl/hw09/tests/diff-nested-right
56 Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.
57 > % =====
58 % nested diff. (44)
59
60 -(55,-(22,11))
61
62 % =====
63 |+ exp=-(55,-(22,11))
64 || tenv=[x=int,v=int,i=int]
65 ||+ exp=55
66 ||| tenv=[x=int,v=int,i=int]

```

11iM-Λ[M-ΛT 24, 16 21:34 **hw09-CHECKED-output-trace** Page 2/8

```

67 ||- type=int
68 ||+ exp=-(22,11)
69 ||| tenv=[x=int,v=int,i=int]
70 |||+ exp=22
71 |||| tenv=[x=int,v=int,i=int]
72 |||- type=int
73 |||+ exp=11
74 |||| tenv=[x=int,v=int,i=int]
75 |||- type=int
76 ||- type=int
77 |- type=int
78 int
79 44
80 >
81 dshin@acacia:checked$ checked -t /home/pl/hw09/tests/if-false-1
82 Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.
83 > % =====
84 % if conditional. (value=4)
85
86 if zero?(1) then 3 else 4
87
88 % =====
89 |+ exp=if zero?(1) then 3 else 4
90 || tenv=[x=int,v=int,i=int]
91 ||+ exp=zero?(1)
92 ||| tenv=[x=int,v=int,i=int]
93 |||+ exp=1
94 |||| tenv=[x=int,v=int,i=int]
95 |||- type=int
96 ||- type=bool
97 ||+ exp=3
98 ||| tenv=[x=int,v=int,i=int]
99 ||- type=int
100 ||+ exp=4
101 ||| tenv=[x=int,v=int,i=int]
102 ||- type=int
103 |- type=int
104 int
105 4
106 >
107 dshin@acacia:checked$ checked -t /home/pl/hw09/tests/if-false-type-error
108 Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.
109 > % =====
110 % if conditional where false exp is not evaluated. (value=type-error)
111
112 if zero?(0) then 3 else foo
113
114 % =====
115 |+ exp=if zero?(0) then 3 else foo
116 || tenv=[x=int,v=int,i=int]
117 ||+ exp=zero?(0)
118 ||| tenv=[x=int,v=int,i=int]
119 |||+ exp=0
120 |||| tenv=[x=int,v=int,i=int]
121 |||- type=int
122 ||- type=bool
123 ||+ exp=3
124 ||| tenv=[x=int,v=int,i=int]
125 ||- type=int
126 ||+ exp=foo
127 ||| tenv=[x=int,v=int,i=int]
128 apply-tenv: foo is undefined
129 /home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/tenv.scm:13
:23: (eopl:error (quote apply-tenv) "~a is undefined" search-sym)
130 /home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1
5:4: (let-values ((val) (type-of-1 exp tenv))) (trace-check-exit val) val)

```

11iM-Λ[M-ΛT 24, 16 21:34	hw09-CHECKED-output-trace	Page 3/8
131	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:4 1:19: (let-values (((ty1) (type-of exp1 tenv)) ((ty2) (type-of exp2 tenv)) ((ty3) (type-of exp3 tenv))) (check-equal-type! ty1 (bool-type) exp1) (check-equal-ty pe! ty2 ty3 exp) ty2)	
132	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1 5:4: (let-values (((val) (type-of-1 exp tenv))) (trace-check-exit val) val)	
133	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:7:3 0: (type->string (check string))	
134	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:7:1 5: (format "~a~n" (type->string (check string)))	
135	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:7:6 : (display (format "~a~n" (type->string (check string))))	
136	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:5:4 : (let-values (((string) (readfile file))) (display string) (display (format "~a ~n" ((....)))) (set! trace-flag #f) (display (format))) (newline)	
137		
138		
139	=== context ===	
140	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1 3:2: type-of	
141	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:2 0:2: type-of-1	
142	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1 3:2: type-of	
143	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:4:2 : runfile	
144		
145	>	
146	dshin@acacia:checked\$ checked -t /home/pl/hw09/tests/let	
147	Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.	
148	> % =====	
149	% let. (value=3)	
150		
151	let x=3	
152	in x	
153		
154	% =====	
155	+ exp=let x=3 in x	
156	tenv=[x=int,v=int,i=int]	
157	+ exp=3	
158	tenv=[x=int,v=int,i=int]	
159	- type=int	
160	+ exp=x	
161	tenv=[x=int,x=int,v=int,i=int]	
162	- type=int	
163	- type=int	
164	int	
165	3	
166	>	
167	dshin@acacia:checked\$ checked -t /home/pl/hw09/tests/letrec-1	
168	Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.	
169	> % =====	
170	% letrecs. (value=32)	
171		
172	letrec int f(x:int)	
173	= -(x,1)	
174	in (f 33)	
175		
176	% =====	
177	+ exp=letrec int f(x:int)=-(x,1) in (f 33)	
178	tenv=[x=int,v=int,i=int]	
179	+ exp=-(x,1)	
180	tenv=[x=int,f=(int->int),x=int,v=int,i=int]	
181	+ exp=x	
182	tenv=[x=int,f=(int->int),x=int,v=int,i=int]	

11iM-Λ[M-ΛT 24, 16 21:34	hw09-CHECKED-output-trace	Page 4/8
183	- type=int	
184	+ exp=1	
185	tenv=[x=int,f=(int->int),x=int,v=int,i=int]	
186	- type=int	
187	- type=int	
188	+ exp=(f 33)	
189	tenv=[f=(int->int),x=int,v=int,i=int]	
190	+ exp=f	
191	tenv=[f=(int->int),x=int,v=int,i=int]	
192	- type=(int->int)	
193	+ exp=33	
194	tenv=[f=(int->int),x=int,v=int,i=int]	
195	- type=int	
196	- type=int	
197	- type=int	
198	int	
199	32	
200	>	
201	dshin@acacia:checked\$ checked -t /home/pl/hw09/tests/proc-1	
202	Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.	
203	> % =====	
204	% proc. (value=int->int)	
205		
206	proc(x:int) -(x,1)	
207		
208	% =====	
209	+ exp=proc(x:int)-(x,1)	
210	tenv=[x=int,v=int,i=int]	
211	+ exp=-(x,1)	
212	tenv=[x=int,x=int,v=int,i=int]	
213	+ exp=x	
214	tenv=[x=int,x=int,v=int,i=int]	
215	- type=int	
216	+ exp=1	
217	tenv=[x=int,x=int,v=int,i=int]	
218	- type=int	
219	- type=int	
220	- type=(int->int)	
221	(int->int)	
222	proc(x:int)-(x,1) [x=10,v=5,i=1]	
223	>	
224	dshin@acacia:checked\$ checked -t /home/pl/hw09/tests/proc-4	
225	Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.	
226	> % =====	
227	% proc. (value=(int->bool)->bool)	
228		
229	proc(f:(int->bool)) (f 3)	
230		
231	% =====	
232	+ exp=proc(f:(int->bool)) (f 3)	
233	tenv=[x=int,v=int,i=int]	
234	+ exp=(f 3)	
235	tenv=[f=(int->bool),x=int,v=int,i=int]	
236	+ exp=f	
237	tenv=[f=(int->bool),x=int,v=int,i=int]	
238	- type=(int->bool)	
239	+ exp=3	
240	tenv=[f=(int->bool),x=int,v=int,i=int]	
241	- type=int	
242	- type=bool	
243	- type=((int->bool)->bool)	
244	((int->bool)->bool)	
245	proc(f:(int->bool)) (f 3) [x=10,v=5,i=1]	
246	>	
247	dshin@acacia:checked\$ checked -t /home/pl/hw09/tests/proc-6	
248	Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.	

11iM-Λ[M-ΛT 24, 16 21:34	hw09-CHECKED-output-trace	Page 5/8
249	> % =====	
250	% proc. (value=(int->((int->(int->bool))->(int->bool))))	
251		
252	proc (x:int) proc (f:(int->(int->bool))) (f x)	
253		
254	% =====	
255	+ exp=proc(x:int)proc(f:(int->(int->bool))) (f x)	
256	tenv=[x=int,v=int,i=int]	
257	+ exp=proc(f:(int->(int->bool))) (f x)	
258	tenv=[x=int,x=int,v=int,i=int]	
259	+ exp=(f x)	
260	tenv=[f=(int->(int->bool)),x=int,x=int,v=int,i=int]	
261	+ exp=f	
262	tenv=[f=(int->(int->bool)),x=int,x=int,v=int,i=int]	
263	- type=(int->(int->bool))	
264	+ exp=x	
265	tenv=[f=(int->(int->bool)),x=int,x=int,v=int,i=int]	
266	- type=int	
267	type=(int->bool)	
268	- type=((int->(int->bool))->(int->bool))	
269	type=(int->((int->(int->bool))->(int->bool)))	
270	(int->((int->(int->bool))->(int->bool)))	
271	proc(x:int)proc(f:(int->(int->bool))) (f x) [x=10,v=5,i=1]	
272	>	
273	dshin@acacia:~\$ checked -t /home/pl/hw09/tests/proc-apply-1	
274	Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.	
275	> % =====	
276	% proc application. (value=29)	
277		
278	(proc(x:int) -(x,1) 30)	
279		
280	% =====	
281	+ exp=(proc(x:int)-(x,1) 30)	
282	tenv=[x=int,v=int,i=int]	
283	+ exp=proc(x:int)-(x,1)	
284	tenv=[x=int,v=int,i=int]	
285	+ exp=-(x,1)	
286	tenv=[x=int,x=int,v=int,i=int]	
287	+ exp=x	
288	tenv=[x=int,x=int,v=int,i=int]	
289	- type=int	
290	+ exp=1	
291	tenv=[x=int,x=int,v=int,i=int]	
292	- type=int	
293	type=int	
294	- type=(int->int)	
295	+ exp=30	
296	tenv=[x=int,v=int,i=int]	
297	- type=int	
298	type=int	
299	int	
300	29	
301	>	
302	dshin@acacia:~\$ checked -t /home/pl/hw09/tests/proc-currying-1	
303	Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.	
304	> % =====	
305	% multiple arguments with proc that returns proc. (value=-1)	
306	% (This is called Currying.)	
307		
308	((proc(x:int) proc(y:int) -(x,y) 5) 6)	
309		
310	% =====	
311	+ exp=((proc(x:int)proc(y:int)-(x,y) 5) 6)	
312	tenv=[x=int,v=int,i=int]	
313	+ exp=(proc(x:int)proc(y:int)-(x,y) 5)	
314	tenv=[x=int,v=int,i=int]	

11iM-Λ[M-ΛT 24, 16 21:34	hw09-CHECKED-output-trace	Page 6/8
315	+ exp=proc(x:int)proc(y:int)-(x,y)	
316	tenv=[x=int,v=int,i=int]	
317	+ exp=proc(y:int)-(x,y)	
318	tenv=[x=int,x=int,v=int,i=int]	
319	+ exp=-(x,y)	
320	tenv=[y=int,x=int,x=int,v=int,i=int]	
321	+ exp=x	
322	tenv=[y=int,x=int,x=int,v=int,i=int]	
323	- type=int	
324	+ exp=y	
325	tenv=[y=int,x=int,x=int,v=int,i=int]	
326	- type=int	
327	- type=int	
328	- type=(int->int)	
329	- type=(int->(int->int))	
330	+ exp=5	
331	tenv=[x=int,v=int,i=int]	
332	- type=int	
333	- type=(int->int)	
334	+ exp=6	
335	tenv=[x=int,v=int,i=int]	
336	- type=int	
337	type=int	
338	int	
339	-1	
340	>	
341	dshin@acacia:~\$ checked -t /home/pl/hw09/tests/proc-type-error-1	
342	Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.	
343	> % =====	
344	% proc. (value=type-error)	
345		
346	(proc(x:(int->int)) -(x,1) 30)	
347		
348	% =====	
349	+ exp=(proc(x:(int->int))-(x,1) 30)	
350	tenv=[x=int,v=int,i=int]	
351	+ exp=proc(x:(int->int))-(x,1)	
352	tenv=[x=int,v=int,i=int]	
353	+ exp=-(x,1)	
354	tenv=[x=(int->int),x=int,v=int,i=int]	
355	+ exp=x	
356	tenv=[x=(int->int),x=int,v=int,i=int]	
357	- type=(int->int)	
358	+ exp=1	
359	tenv=[x=(int->int),x=int,v=int,i=int]	
360	- type=int	
361	type-error: exp=x, expected type=(int->int), current type=int	
362	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:8	
5:4:	(eopl:error (quote type-error) "exp=~a, expected type=~a, current type=~a"	
	(exp->string exp) (type->string ty1) (type->string ty2))	
363	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:2	
9:21:	(let-values (((ty1) (type-of exp1 tenv)) ((ty2) (type-of exp2 tenv))) (che	
	ck-equal-type! ty1 (int-type) exp1) (check-equal-type! ty2 (int-type) exp2) (int	
	-type))	
364	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1	
5:4:	(let-values (((val) (type-of-1 exp tenv))) (trace-check-exit val) val)	
365	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:5	
1:21:	(let-values (((result-type) (type-of body (extend-tenv var ...)))) (proc-	
	type var-type result-type))	
366	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1	
5:4:	(let-values (((val) (type-of-1 exp tenv))) (trace-check-exit val) val)	
367	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:5	
6:21:	(let-values (((rator-type) (type-of rator tenv)) ((rand-type) (type-of ran	
	d tenv))) (let-values (((v) rator-type)) (if (not (...)) (error ...) (...)))	
368	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1	
5:4:	(let-values (((val) (type-of-1 exp tenv))) (trace-check-exit val) val)	

11iM-Λ[M-ΛT 24, 16 21:34	hw09-CHECKED-output-trace	Page 7/8
369	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:7:3	
370	0: (type->string (check string))	
371	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:7:1	
372	5: (format "~a~n" (type->string (check string)))	
373	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:7:6	
374	: (display (format "~a~n" (type->string (check string))))	
375	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:5:4	
376	: (let-values (((string) (readfile file))) (display string) (display (format "~a~n" ((....)))) (set! trace-flag #f) (display (format (....))) (newline)))	
377	=== context ===	
378	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:2	
379	0:2: type-of-1	
380	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1	
381	3:2: type-of	
382	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:2	
383	0:2: type-of-1	
384	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1	
385	3:2: type-of	
386	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:2	
387	0:2: type-of-1	
388	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1	
389	3:2: type-of	
390	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1	
391	3:2: type-of	
392	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:4:2	
393	: runfile	
394	>	
395	dshin@acacia:checked\$ checked -t /home/pl/hw09/tests/proc-type-error-3	
396	Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.	
397	> % =====	
398	% proc. (value=type-error)	
399		
400	proc (x:int) proc (f:(int->(int->bool))) (f zero?(x))	
401		
402	% =====	
403	+ exp=proc(x:int)proc(f:(int->(int->bool))) (f zero?(x))	
404	tenv=[x=int,v=int,i=int]	
405	+ exp=proc(f:(int->(int->bool))) (f zero?(x))	
406	tenv=[x=int,x=int,v=int,i=int]	
407	+ exp=(f zero?(x))	
408	tenv=[f=(int->(int->bool)),x=int,x=int,v=int,i=int]	
409	- type=(int->(int->bool))	
410	+ exp=zero?(x)	
411	tenv=[f=(int->(int->bool)),x=int,x=int,v=int,i=int]	
412	+ exp=x	
413	tenv=[f=(int->(int->bool)),x=int,x=int,v=int,i=int]	
414	- type=int	
415	- type=bool	
416	type-error: exp=zero?(x), expected type=int, current type=bool	
417	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:8	
418	5:4: (eopl:error (quote type-error) "exp=~a, expected type=~a, current type=~a"	
419	(exp->string exp) (type->string ty1) (type->string ty2))	
420	[unknown source]: (let-values (((arg-type) (proc-type-accessor v 0)) ((result-type) (proc-type-accessor v 1))) (check-equal-type! arg-type rand-type rand) result-type)	
421	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1	
422	5:4: (let-values (((val) (type-of-1 exp tenv))) (trace-check-exit val) val)	
423	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:5	
424	1:21: (let-values (((result-type) (type-of body (extend-tenv var)))) (proc-type var-type result-type))	
425	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1	
426	5:4: (let-values (((val) (type-of-1 exp tenv))) (trace-check-exit val) val)	

11iM-Λ[M-ΛT 24, 16 21:34	hw09-CHECKED-output-trace	Page 8/8
414	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:5	
415	1:21: (let-values (((result-type) (type-of body (extend-tenv var)))) (proc-type var-type result-type))	
416	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1	
417	5:4: (let-values (((val) (type-of-1 exp tenv))) (trace-check-exit val) val)	
418	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:7:3	
419	0: (type->string (check string))	
420	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:7:1	
421	5: (format "~a~n" (type->string (check string)))	
422	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:7:6	
423	: (display (format "~a~n" (type->string (check string))))	
424	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:5:4	
425	: (let-values (((string) (readfile file))) (display string) (display (format "~a~n" ((....)))) (set! trace-flag #f) (display (format (....))) (newline)))	
426	=== context ===	
427	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:2	
428	0:2: type-of-1	
429	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1	
430	3:2: type-of	
431	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:2	
432	0:2: type-of-1	
433	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1	
434	3:2: type-of	
435	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:2	
436	0:2: type-of-1	
437	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1	
438	3:2: type-of	
439	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:2	
440	0:2: type-of-1	
441	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/check.scm:1	
442	3:2: type-of	
443	/home/dshin/2010-ea0011-pl/eopl3-programs/eopl3-interpreters/checked/run.scm:4:2	
444	: runfile	
445	>	
446	dshin@acacia:checked\$ checked -t /home/pl/hw09/tests/variable	
447	Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.	
448	> % =====	
449	% variable, where env=[i=1,v=5,x=10]. (value=10)	
450		
451	x	
452	% =====	
453	+ exp=x	
454	tenv=[x=int,v=int,i=int]	
455	- type=int	
456	int	
457	10	
458	>	
459	dshin@acacia:checked\$ checked -t /home/pl/hw09/tests/zero-true	
460	Welcome to MzScheme v372 [3m], Copyright (c) 2004-2007 PLT Scheme Inc.	
461	> % =====	
462	% zero? (value=#t)	
463		
464	zero?(0)	
465	% =====	
466	+ exp=zero?(0)	
467	tenv=[x=int,v=int,i=int]	
468	+ exp=0	
469	tenv=[x=int,v=int,i=int]	
470	- type=int	
471	- type=bool	
472	bool	
473	#t	
474	>	
475	dshin@acacia:checked\$	