Summary

Sink States: $0(0 \times 10^0)$

Table 1: Sip4J Analysis Summary

Classes	Methods	States	Unreachable clauses	Unreachable states	Possible concurrent methods	Total. no. of method pairs	No. of concurrent method pairs	Percentage of concurrent methods pairs
JGFMonteCarloBenchSizeA	2	1	0	0	0	3	0	0
JGFMonteCarloBench	7	1	0	0	1	28	1	4
CallAppDemo	4	1	0	0	0	10	0	0
AppDemo	18	1	0	0	6	171	21	12
Universal	11	1	0	0	7	66	28	42
PathId	11	1	0	0	5	66	15	23
RatePath	11	1	0	0	4	66	10	15
ReturnPath	23	1	0	0	9	276	45	16
MonteCarloPath	21	1	0	0	20	231	56	24
ToInitAllTasks	21	1	0	0	10	231	55	24
ToResult	14	1	0	0	7	105	28	27
PriceStock	5	1	0	0	1	15	1	7
ToTask	5	1	0	0	2	15	3	20
DemoException	1	1	0	0	0	1	0	0
JGFInstrumentor	13	1	0	0	12	91	12	13
JGFTimer	9	1	0	0	3	45	6	13
test	4	1	0	0	1	10	1	10
Utilities	5	1	0	0	4	15	8	53
Total Classes=18	185	18	0	0	92	1445	290	20

Contents

1	JGFMonteCarloBenchSizeA	3
2	JGFMonteCarloBench	4
3	CallAppDemo	5
4	AppDemo	6
5	Universal	8
6	PathId	9
7	RatePath	10
8	ReturnPath	11
9	MonteCarloPath	13
10	ToInitAllTasks	15
11	ToResult	17
12	PriceStock	18
13	ToTask	19
14	DemoException	20
15	JGFInstrumentor	21
16	JGFTimer	22
17	test	23
18	Utilities	24
19	Abbreviation	25
20	Annotated version of the input program generated by Sip4J	26

1 JGFMonteCarloBenchSizeA

 ${\it Table 2: Method's Satisfiability} ({\it Code Reachability Analysis}$

Method	Satisfiability
JGFMonteCarloBenchSizeA	$\sqrt{}$
main	

Table 3: State Transition Matrix

	alive
alive	1

Table 4: Methods Concurrency Matrix

	${\tt JGFMonteCarloBenchSizeA}$	main
JGFMonteCarloBenchSizeA	#	\parallel
main	#	\parallel

2 JGFMonteCarloBench

 ${\it Table 5: Method's Satisfiability} ({\it Code Reachability Analysis}$

Method	Satisfiability
JGFMonteCarloBench	\checkmark
JGFrun	
JGFsetsize	
JGFinitialise	
JGFapplication	\checkmark
JGFvalidate	
JGFtidyup	

Table 6: State Transition Matrix

	alive
alive	↑

Table 7: Methods Concurrency Matrix

	JGFMonteCarloBench	JGFrun	JGFsetsize	JGFinitialise	JGFapplication	JGFvalidate	JGFtidyup
JGFMonteCarloBench	#	#	#	#	#	#	*
JGFrun	#	#	#	#	#	#	*
JGFsetsize	#	#	#	#	#	#	*
JGFinitialise	#	#	#	#	#	#	*
JGFapplication	#	#	#	#	#	#	*
JGFvalidate	#	#	#	#	#		*
JGFtidyup	1	1	1	¥	#	¥	1

3 CallAppDemo

Table 8: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
CallAppDemo	$\sqrt{}$
initialise	$\sqrt{}$
presults	$\sqrt{}$
runiters	

Table 9: State Transition Matrix

	alive
alive	

Table 10: Methods Concurrency Matrix

	CallAppDemo	initialise	presults	runiters
CallAppDemo	ł	#	#	#
initialise	#	#	#	#
presults	#	#	#	#
runiters	#	#	#	#

4 AppDemo

Table 11: Method's Satisfiability(Code Reachability Analysis

Method	Satisfiability
AppDemo	\checkmark
initSerial	\checkmark
initTasks	$\sqrt{}$
processSerial	
processResults	
runSerial	$\sqrt{}$
getdataDirname	
setdataDirname	
getdataFilename	
setdataFilename	
getnTimeStepsMC	
setnTimeStepsMC	
getnRunsMC	
setnRunsMC	
gettasks	
settasks	$\sqrt{}$
getresults	
setresults	$\sqrt{}$

Table 12: State Transition Matrix



Table 13: Methods Concurrency Matrix

	AppDemo	initSerial	initTasks	processSerial	processResults	runSerial	getdataDirname	setdataDirname	getdataFilename	setdataFilename	${\rm getnTimeStepsMC}$	${\rm setnTimeStepsMC}$	getnRunsMC	setnRunsMC	gettasks	settasks	getresults	setresults
AppDemo	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	\parallel
initSerial	#	#	#	#	#	#	#	#	#	#	\parallel	\parallel	#	#	#	#	#	\parallel
initTasks	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	\parallel
processSerial	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#
processResults	#	#	\parallel	#	#	#	#	#	\parallel	#	\parallel	#	#	#	#	#	#	\parallel
runSerial	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	\parallel
getdataDirname	#	#	\parallel	#	#	#		#		#		#		#		#		\parallel
setdataDirname	#	#	#	#	#	#	#	#	#	#	\parallel	#	#	#	#	#	#	\parallel
getdataFilename	\parallel	\parallel	\downarrow	#	#	#		#		#		\parallel		#		\parallel		\parallel
setdataFilename	\parallel	 	\parallel	#	#	#	#	#	#	#	#	\parallel	#	#	#	#	#	\parallel
getnTimeStepsMC	#	#	#	#	#	#		#		#		#		#		#		\parallel
setnTimeStepsMC	∦	∦	#		#	#	#	 	\parallel	 	#	#	∦	 	 	\parallel	#	\parallel

getnRunsMC	#	#	#	#	#	#		#		#		#		#		#		#
setnRunsMC	#	#	\parallel	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#
gettasks	#	#	\parallel	#	#	#		#		#		#		#		#		#
settasks	#	#	\parallel	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#
getresults	#	#	\parallel	#	#	#		#		#		#		#		#		#
setresults	#	1	#	1	#	#	\parallel	#	#	#	#	#	#	#	#	#	#	#

5 Universal

Table 14: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
Universal	\checkmark
setprompt	\checkmark
setDEBUG	
dbgPrintln	\checkmark
errPrintln	\checkmark
getDEBUG	\checkmark
getUNIVERSALDEBUG	\checkmark
setUNIVERSALDEBUG	\checkmark
getprompt	$\sqrt{}$
dbgPrint	
errPrint	

Table 15: State Transition Matrix



Table 16: Methods Concurrency Matrix

	Universal	setprompt	setDEBUG	dbgPrintln	errPrintln	getDEBUG	${\tt getUNIVERSALDEBUG}$	setUNIVERSALDEBUG	getprompt	dbgPrint	errPrint
Universal	#	#	#	#	#	#	#	#	#	#	
setprompt	#	#	#	ł	#	#	#	#	#	#	#
setDEBUG	#	#	#	#	#	#	#	#	#	#	#
dbgPrintln	#	#	#					#			
errPrintln	#	#	#					#			
getDEBUG	#	#	#					#			
getUNIVERSALDEBUG	#	#	#					#			
setUNIVERSALDEBUG	#	#	#	ł	#	#	#	#	#	#	*
getprompt	#	#	#					#			
dbgPrint	#	#	#					#			
errPrint	#	#	#					#			

6 PathId

Table 17: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
PathId	$\sqrt{}$
dbgDumpFields	
copyInstanceVariables	$$
getdTime	
getname	
getstartDate	
getendDate	
setname	
setstartDate	
setendDate	$\sqrt{}$
setdTime	

Table 18: State Transition Matrix



Table 19: Methods Concurrency Matrix

	PathId	dbgDumpFields	${\rm copyInstanceVariables}$	getdTime	getname	getstartDate	getendDate	setname	setstartDate	setendDate	setdTime
PathId	#	#	#	#	#	#	#	#	#	#	
dbgDumpFields	\parallel		\parallel					#	#	#	#
copyInstanceVariables	#	#	#	 	#	#	#	#	#	#	#
getdTime	#		#					#	#	#	∦
getname	#		#					#	#	#	
getstartDate	#		#					#	#	#	∦
getendDate	#		#					#	#	#	#
setname	#	#	\parallel	#	#	#	#	#	#	#	#
setstartDate	#	#	\parallel	#	\parallel	#	#	#	#	#	#
setendDate	\parallel	#	#	#	\parallel	#	#	#		#	∦
setdTime	\parallel	#	#	#	#	#	#	#	#	#	∦

7 RatePath

Table 20: Method's Satisfiability (Code Reachabiity Analysis

Method	Satisfiability
RatePath	\vee
getReturnCompounded	\checkmark
getReturnNonCompounded	
readRatesFile	\checkmark
getEndPathValue	
getPathValue	$\sqrt{}$
incpathValue	
getpathValue	$\sqrt{}$
setpathValue	
getpathDate	$\sqrt{}$
setpathDate	

Table 21: State Transition Matrix



Table 22: Methods Concurrency Matrix

	RatePath	getReturnCompounded	getReturnNonCompounded	readRatesFile	getEndPathValue	getPathValue	incpathValue	getpathValue	setpathValue	getpathDate	setpathDate
RatePath	#	#	#	#	#	\parallel	#	\parallel	\parallel	#	\parallel
getReturnCompounded	#	#	#	#	#	#	#	#	#	#	\parallel
getReturnNonCompounded	#	#	#	#	#	#	#	#	#	 	\parallel
readRatesFile	#	#	#	#	#	\parallel	#	#	#	#	\parallel
getEndPathValue	#	#	#	#			#		#		\parallel
getPathValue	#	#	#	#			#		#		\parallel
incpathValue	#	#	#	#	#	#	#	#	#	#	#
getpathValue	#	#	#	\parallel			#		#		#
setpathValue	#	#	#	#	#	#	#	#	#	#	#
getpathDate	#	#	#	#			#		#		#
setpathDate	#	#	#	#	#	#	#	#	#	#	#

8 ReturnPath

Table 23: Method's Satisfiability (Code Reachabiity Analysis

Method	Satisfiability
ReturnPath	\checkmark
estimatePath	\checkmark
computeMean	\checkmark
computeVariance	\checkmark
compute Expected Return Rate	$\sqrt{}$
computeVolatility	\checkmark
dbgDumpFields	\checkmark
getexpectedReturnRate	\checkmark
getvolatility	\checkmark
getreturnDefinition	\checkmark
getvolatility2	\checkmark
getpathValue	\checkmark
setpathValue	\checkmark
getnPathValue	$\sqrt{}$
setnPathValue	\checkmark
setreturnDefinition	\checkmark
setexpectedReturnRate	\checkmark
setvolatility	\checkmark
setvolatility2	$\sqrt{}$
getmean	
setmean	
getvariance	$\sqrt{}$
setvariance	$\sqrt{}$

Table 24: State Transition Matrix

	alive
alive	↑

Table 25: Methods Concurrency Matrix

	ReturnPath	estimatePath	computeMean	computeVariance	computeExpectedReturnRate	computeVolatility	dbgDumpFields	getexpectedReturnRate	getvolatility	getreturnDefinition	getvolatility2	getpathValue	setpathValue	getnPathValue	setnPathValue	setreturnDefinition	setexpectedReturnRate	setvolatility	setvolatility2	getmean
ReturnPath	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#
estimatePath	#	#	#	#	#	#	#	#	#	 	#	#	#	#	#	#	#	#	#	#
computeMean	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	\parallel

computeVariance	#	#	#	 	#	#	#		#	 	#	#	#	#	\parallel	#	#	#	#	\parallel
compute Expected Return Rate	#	#	\parallel	#	#	#	#	#	#	#	#	#	#	#	#	ł	#	#	#	1
computeVolatility	#	#	\parallel	#	#	#	#	#	#	#	#	\parallel	#	#	#	#	#	#	#	\parallel
dbgDumpFields	#	#	\parallel	#	#	#							#		#	#	#	#	#	
getexpectedReturnRate	#	#	\parallel	#	#	#							#		#	#	#	#	#	\Box
getvolatility	#	#	\parallel	#	#	#							#		#	#	#	#	#	
getreturnDefinition	#	#	\parallel	#	\parallel	#							#		#	#	#	#	#	
getvolatility2	#	#	\parallel	#	#	#							#		#	#	#	#	#	
getpathValue	#	#	\parallel	#	#	#							#		#	#	#	#	#	\prod
setpathValue	#	#	\parallel	#	#	#	#		#	#	#	#	#	#	#	#	#	#	#	1
getnPathValue	#	#	\parallel	#	#	#							#		#	#	#	#	#	\prod
setnPathValue	#	#	\parallel	#	#	#	#		#	#	#	#	#	#	#	#	#	#	#	1
setreturnDefinition	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	1
setexpectedReturnRate	#	#	\parallel	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	\parallel
setvolatility	#	#	\parallel	#	 	#	#	 	\parallel		#	#	#	#	#	#	#	#	#	1
setvolatility2	#	#	\parallel	#	#	#	#		#	#	#	#	#	#	#	#	#	#	#	1
getmean	#	#	\parallel	#	 	#							#		#	#	#	#	#	
setmean	#	#	#	#	#	#	#	1	#	#	#	#	#	#	#	#	#	#	#	1
getvariance	#	#	#	#	#	#							#		 	#	#	#	#	
setvariance	#	#	\parallel	#	#	#	#	#	#	\parallel	#	#		#	1	1	1	#	#	1

9 MonteCarloPath

Table 26: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
MonteCarloPath	\checkmark
copyInstanceVariables	$\sqrt{}$
setreturnDefinition	\checkmark
setexpectedReturnRate	\checkmark
setvolatility	$\sqrt{}$
setnTimeSteps	\checkmark
setpathStartValue	$\sqrt{}$
setpathValue	\checkmark
setfluctuations	\checkmark
computeFluctuationsGaussian	\checkmark
computePathValue	$\sqrt{}$
getpathValue	
getnTimeSteps	\checkmark
getfluctuations	
getreturnDefinition	\checkmark
getexpectedReturnRate	\checkmark
getvolatility	$\sqrt{}$
getpathStartValue	$\sqrt{}$
writeFile	$\sqrt{}$
getRatePath	$\sqrt{}$
compute Fluctuations Gaussian Overload	\checkmark

Table 27: State Transition Matrix

	alive
alive	↑

Table 28: Methods Concurrency Matrix

	MonteCarloPath	copyInstanceVariables	setreturnDefinition	setexpectedReturnRate	= setvolatility	= setnTimeSteps	setpathStartValue	setpathValue	= setfluctuations	computeFluctuationsGaussian	computePathValue	getpathValue	= getnTimeSteps	= getfluctuations	getreturnDefinition	getexpectedReturnRate	getvolatility	= antrath Ctont Valua
MonteCarloPath	<u> </u>	#	<u> </u>	#	<u> </u>	<u> </u>	<u> </u>	<u> </u>	<u> </u>	<u> </u>	#	<u> </u>	#	<u> </u>		<u> </u>	<u> </u>	1
copyInstanceVariables	 	∦	 		 		∦	∦	∦	ł	∦	∦	\parallel	∦		 	 	

setreturnDefinition	\parallel	 	#	#	#	#	 	#	#	#	 	#	#	#	#	#	#	1
setexpectedReturnRate	#	#	#	#	#	#	#	#	#	#	#	#	\parallel	#	#	#	#	H
setvolatility	#	 	#	#	\parallel	1		#	#	#	#	#	\parallel	#	#	\parallel	\parallel	1
setnTimeSteps	#	#	#	#	#	#	#	#	#	#	#	#	\parallel	#	#	\parallel	#	∦
setpathStartValue	#	#	#	#	#	#	#	#	#	\parallel	#	#	#	#	#	\parallel	#	╢
setpathValue	#		#	#	#	#		#	#	#	#	#	#	#	#	#		1
setfluctuations	*		#	#	\parallel	#		\parallel	#	\Rightarrow	#	#	\parallel	#		\parallel	\parallel	\parallel
computeFluctuationsGaussian	#	∦	#	#	#	#		#	#	#	#	#	#	#	#	#	$ \parallel $	\parallel
computePathValue	*		#	#	\parallel	#		\parallel	#	*	#	#	\parallel	#		\parallel	\parallel	\parallel
getpathValue	#	 	#	#	#	#	∦	#	#	#	#							Ш
getnTimeSteps	*		#	#	\parallel	#		\parallel	#	*	#							\prod
getfluctuations	#	 	#	#	#	#	#	#	#	#	#							Ш
getreturnDefinition	*		#	#	\parallel	#		\parallel	#	*	#							\prod
getexpectedReturnRate	#	#	#	#	#	#	#	#	#	*	#							
getvolatility	#		#	#	\parallel			\parallel	#	*	#							\prod
getpathStartValue	#	∦	#	#	\parallel	#	∦	\parallel	#	*	\parallel							Ш
writeFile	#	#	#	#	#	#	#	#	#	#	#							
getRatePath	#																	
compute Fluctuations Gaussian Overload	\parallel		#	#	#	#		#	#	#	#	#	#	\parallel		\parallel	∦	ł

10 ToInitAllTasks

Table 29: Method's Satisfiability (Code Reachabiity Analysis

Method	Satisfiability
ToInitAllTasks	\checkmark
getname	$\sqrt{}$
getstartDate	$\sqrt{}$
getendDate	$\sqrt{}$
getdTime	\checkmark
getreturnDefinition	$\sqrt{}$
getexpectedReturnRate	$\sqrt{}$
getvolatility	$\sqrt{}$
getnTimeSteps	$\sqrt{}$
getpathStartValue	$\sqrt{}$
getheader	$\sqrt{}$
setheader	$\sqrt{}$
setname	$\sqrt{}$
setstartDate	$\sqrt{}$
setendDate	$\sqrt{}$
setDTime	$\sqrt{}$
setReturnDefinition	$\sqrt{}$
${\bf set Expected Return Rate}$	$\sqrt{}$
setVolatility	$\sqrt{}$
setnTimeSteps	√ ·
setpathStartValue	√

Table 30: State Transition Matrix



Table 31: Methods Concurrency Matrix

	ToInitAllTasks	getname	getstartDate	getendDate	getdTime	getreturnDefinition	getexpectedReturnRate	getvolatility	getnTimeSteps	getpathStartValue	getheader	setheader	setname	setstartDate	setendDate	setDTime	setReturnDefinition	setExpectedReturnRate	setVolatility	setnTimeSteps	setpathStartValue
ToInitAllTasks	#	#	#	#	#	#	#	#	#	#	#	#	#	\parallel	\parallel	#	#	#	#	#	
getname												#	#	#	#	#	#	#	#	#	
getstartDate												#	#	#	#	#	#	#	#	#	\
getendDate	#											#	*	#	#	#	¥	#	#	#	#
getdTime	#											#	#	#	#	#	#	#	#	#	
getreturnDefinition	#											#	*	¥	¥	#	#	#	#	#	#
getexpectedReturnRate	#											#	$ \downarrow $	#	$ \downarrow $		\parallel	#	#	#	1

getvolatility	#											#	#	#	\parallel	\parallel	\parallel	 	#	 	#
getnTimeSteps	#											#	#	#	#	ł	#	#	#	#	#
getpathStartValue	#											#	#	#	#	#	#		\parallel		#
getheader	#											#	#	#	\parallel	#	#	#	#		#
setheader	#	#	#	#	\parallel	#	#	#	#	#	#	#	#	#	#	#	#	#	\parallel	#	#
setname	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#
setstartDate	#	#	#	#	\parallel	#	#	#		#	#	#	#	#	#	#	\parallel	 	\parallel	 	#
setendDate	#	#	#	#	#	#	#	#	#	#	#	#	#	#	¥	ł	#	#	#	#	#
setDTime	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#
setReturnDefinition	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#
setExpectedReturnRate	#	#	#	#	\parallel	#	#	#	#	#	#	#	#	#	#	#	\parallel	#	\parallel	 	#
setVolatility	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#
setnTimeSteps	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#	#
setpathStartValue	#	#	#	#	#	#	#	#	#	#	#	#	#	#	\parallel	#	#	#	#	#	#

11 ToResult

Table 32: Method's Satisfiability (Code Reachabiity Analysis

Method	Satisfiability
ToResult	\checkmark
getexpectedReturnRate	\checkmark
getvolatility	\checkmark
toString	\checkmark
getheader	\checkmark
setheader	\checkmark
setexpectedReturnRate	\checkmark
setvolatility	\checkmark
getVolatility2	\checkmark
setvolatility2	\checkmark
getfinalStockPrice	\checkmark
setfinalStockPrice	
getpathValue	\checkmark
setpathValue	

Table 33: State Transition Matrix



Table 34: Methods Concurrency Matrix

	ToResult	getexpectedReturnRate	getvolatility	toString	getheader	setheader	${\bf setexpectedReturnRate}$	setvolatility	getVolatility2	setvolatility2	getfinalStockPrice	setfinalStockPrice	$\operatorname{getpathValue}$	setpathValue
ToResult	\parallel	#	#	#	#	#	#	\parallel	#	#	#	#	*	#
getexpectedReturnRate	#					#	#	#		\parallel		#		*
getvolatility	#					#	#	#		#		#		#
toString	#					#	#	#		#		#		∦
getheader	¥					#	#	#		#		¥		#
setheader	¥	#	#	#	#	#	#	#	#	#	#	#	#	#
setexpectedReturnRate	\parallel	#	#	#	#	#	#	#	#	#	#	¥	#	#
setvolatility	#	#	#	#	#	#	#	#	#	#	#	#	#	#
getVolatility2	\parallel					#	#	#		#		¥		#
setvolatility2	#	#	#	#	#	#	#	#	#	#	#	#	#	#
getfinalStockPrice	#					\parallel	#	#		#		¥		#
setfinalStockPrice	#	#	#	#	#	#	#	#	#	#	#	#	#	#
getpathValue	\parallel					#	#	#		#		¥		#
setpathValue	\parallel	#	#	#	#	\parallel	#	#	#	#	#	#	¥	#

12 PriceStock

Table 35: Method's Satisfiability (Code Reachabiity Analysis

Method	Satisfiability
PriceStock	
setInitAllTasks	$\sqrt{}$
setTask	
run	$\sqrt{}$
getResult	\checkmark

Table 36: State Transition Matrix

	alive
alive	↑

Table 37: Methods Concurrency Matrix

	PriceStock	setInitAllTasks	setTask	run	getResult
PriceStock	#	#	#	#	#
setInitAllTasks	#	#	#	ł	#
setTask	#	#	#	#	#
run	#	#	\parallel	¥	#
getResult	\parallel	#	\parallel	#	

13 ToTask

Table 38: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
ToTask	$\sqrt{}$
getheader	
getrandomSeed	
setheader	$\sqrt{}$
setrandomSeed	\checkmark

Table 39: State Transition Matrix

	alive
alive	↑

Table 40: Methods Concurrency Matrix

	ToTask	getheader	getrandomSeed	setheader	setrandomSeed
ToTask	#	#	#	#	*
getheader	#			#	#
getrandomSeed	#			#	#
setheader	#	#	 	#	#
setrandomSeed	#	#	 	#	#

14 DemoException

Table 41: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
DemoException	\checkmark

Table 42: State Transition Matrix

	alive
alive	↑

15 JGFInstrumentor

Table 43: Method's Satisfiability (Code Reachabiity Analysis

Method	Satisfiability
JGFInstrumentor	$$
addTimer	
startTimer	\checkmark
stopTimer	
addOpsToTimer	\checkmark
readTimer	
resetTimer	$$
printTimer	
printperfTimer	\checkmark
storeData	\checkmark
retrieveData	
printHeader	$\sqrt{}$
main	

Table 44: State Transition Matrix



Table 45: Methods Concurrency Matrix

	JGFInstrumentor	addTimer	startTimer	stopTimer	addOpsToTimer	readTimer	resetTimer	printTimer	printperfTimer	storeData	retrieveData	printHeader	main
JGFInstrumentor	#	#	#	#	#	#	#	#	#	#	#	#	\parallel
addTimer	#	#	#	#	#	#	#	#	#	#	#		\parallel
startTimer	#	#	#	#	#	#	#	#	#	#	#		\parallel
stopTimer	#	#	#	#	#	#	#	#	#	#	¥		\parallel
addOpsToTimer	#	#	#	#	#	#	#	#	#	#	#		\parallel
readTimer	#	¥	#	#	#	#	¥	#	#	#	¥		\parallel
resetTimer	#	#	#	#	#	#	#	#	#	#	#		\parallel
printTimer	#	#	#	#	#	#	#	#	#	#	¥		\parallel
printperfTimer	#	#	#	#	#	#	#	#	#	#	#		\parallel
storeData	#	#	#	#	#	#	#	#	#	#	#		\parallel
retrieveData	#	¥	#	#	#	#	¥	#	#	#	#		\parallel
printHeader	#												
main	#	#	#	#	#	#	#	#	#	#	#		\parallel

16 JGFTimer

Table 46: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
JGFTimer	\checkmark
start	$\sqrt{}$
stop	
addops	
reset	
print	
perf	
printperf	$\sqrt{}$
longprint	

Table 47: State Transition Matrix



Table 48: Methods Concurrency Matrix

	JGFTimer	start	stop	addops	reset	print	perf	printperf	longprint
JGFTimer	#	#	#	#	#	#	#	#	#
start	#	#	#	#	#	#	#	#	\parallel
stop	#	#	#	#	#	#	#	#	#
addops	#	#	#	#	#	#	#	#	\parallel
reset	#	#	#	#	#	#	#	#	#
print	#	#	#	#	#	#	#	#	\parallel
perf	#	#	#	#	#	 			
printperf	#	#	#	#	#	#			
longprint	#	#	#	#	#	 			

17 test

Table 49: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
test	
createObject	
readA	
main	

Table 50: State Transition Matrix

	alive
alive	\uparrow

Table 51: Methods Concurrency Matrix

	test	createObject	readA	main
test	#	#	\parallel	#
createObject	#	#	#	#
readA	#	#		#
main	#	#	\parallel	

18 Utilities

Table 52: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
Utilities	$\sqrt{}$
which	
splitString	
joinString	
joinStringOverloaded	$\sqrt{}$

Table 53: State Transition Matrix

	alive
alive	↑

Table 54: Methods Concurrency Matrix

	Utilities	which	splitString	joinString	joinStringOverloaded
Utilities	#	#	#	#	#
which	#	#	#		
splitString	#	#			
joinString	#				
joinStringOverloaded	#				

19 Abbreviation

Table 55: Used Abbreviation

Symbol	Meaning
	requires clause of the method is satisfiable
X	requires clause of the method is unsatisfiable
↑	The row-state can be transitioned to the column-state
×	The row-state cannot be transitioned to the column-state
	The row-method can be possibly executed parallel with the column-method
H	The row-method cannot be executed parallel with the column-method

20 Annotated version of the input program generated by Sip4J

```
package outputs;
import edu.cmu.cs.plural.annot.*;
    @ClassStates({@State(name = "alive")})
class JGFMonteCarloBenchSizeA {
    JGFMonteCarloBenchSizeA() { }
    @Perm(requires="unique(this) in alive",
       nsures="unique(this) in alive")
void main(String argv[]) {
    ensures=
   }
13
15 }ENDOFCLASS
   @ClassStates({@State(name = "alive")})
    class JGFMonteCarloBench {
   @Perm(ensures="unique(this) in alive")
JGFMonteCarloBench() {
    }
   @Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
    public void JGFrun(int size) {
   @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
public void JGFsetsize(int size) {
   @Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
public void JGFinitialise() {
    @Perm(requires="unique(this) in alive",
    ensures="unique(this) in alive";
public void JGFapplication() {
   OPerm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public void JGFvalidate() {
   OPerm(requires="unique(this) in alive",
ensures="unique(this) in alive")
public void JGFtidyup() {
52
   }
54 }ENDOFCLASS
   @ClassStates({@State(name = "alive")})
    class CallAppDemo {
   @Perm(ensures="unique(this) in alive")
CallAppDemo() { }
   @Perm(requires="unique(this) in alive",
   ensures="unique(this) in alive")
public void initialise() {
   OPerm(requires="share(this) in alive",
ensures="share(this) in alive")
public void presults() {
    @Perm(requires="unique(this) in alive",
     ensures="unique(this) in alive")
public void runiters() {
78 }ENDOFCLASS
```

```
80 @ClassStates({@State(name = "alive")})
     class AppDemo {
     @Perm(ensures="unique(this) in alive")
     AppDemo() { }
     @Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
public void initSerial() {
    }
@Perm(requires="unique(this) in alive",
     ensures="unique(this) in alive")
private void initTasks(int nRunsMC) {
     @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 96
       public void processSerial() {
     @Perm(requires="share(this) in alive",
101
     ensures="share(this) in alive")
private void processResults() {
102
105
    @Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
public void runSerial() {
106
107
110 }
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public String getdataDirname() {
112
113
114
       return null;
    @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
117
118
      public void setdataDirname(String dataDirname) {
122
     @Perm(requires="pure(this) in alive",
     ensures="pure(this) in alive")
public String getdataFilename() {
123
125
       return null;
     @Perm(requires="full(this) in alive",
128
     ensures="full(this) in alive")
129
       public void setdataFilename(String dataFilename) {
132
    @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public int getnTimeStepsMC() {
133
134
        return 0;
136
     Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setnTimeStepsMC(int nTimeStepsMC) {
139
    @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public int getnRunsMC() {
  return 0;
144
145
147
    @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setnRunsMC(int nRunsMC) {
150
151
152
    GPerm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public Vector gettasks() {
return null;
155
156
158
```

```
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
public void settasks(Vector tasks) {
161
163
     Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public Vector getresults() {
166
167
        return null:
169
171
     OPerm(requires="share(this) in alive",
ensures="share(this) in alive")
public void setresults(Vector results) {
172
174
176 }
178 }ENDOFCLASS
180 @ClassStates({@State(name = "alive")})
      class Universal {
     @Perm(ensures="unique(this) in alive")
Universal() { }
183
      @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setprompt(String prompt) {
188
190
     @Perm(requires="full(this) in alive",
193
      ensures="full(this) in alive")
public void setDEBUG(boolean DEBUG) {
193
195
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public void dbgPrintln(String s) {
196
198
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
201
202
       public void errPrintln(String s) {
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public boolean getDEBUG() {
return 0;
206
207
209
211
     GPerm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public boolean getUNIVERSALDEBUG() {
return 0;
212
213
215
217
     @Perm(requires="full(this) in alive",
218
     ensures="full(this) in alive")
public void setUNIVERSALDEBUG(boolean UNIVERSAL_DEBUG) {
220
222
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public String getprompt() {
223
225
        return null;
226
228
     @Perm(requires="pure(this) in alive",
229
      ensures="pure(this) in alive")
public void dbgPrint(String s) {
23
     Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public void errPrint(String s) {
234
236
238 }
240 }ENDOFCLASS
```

```
242 @ClassStates({@State(name = "alive")})
     class PathId {
244
     @Perm(ensures="unique(this) in alive")
245
     PathId() { }
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public void dbgDumpFields() {
250
252
     @Perm(requires="share(this) in alive",
253
      public void copyInstanceVariables(PathId obj) {
255
257
    Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getdTime() {
  return 0;
258
260
26
263
     @Perm(requires="pure(this) in alive",
264
     ensures="pure(this) in alive")
public String getname() {
  return null;
266
267
269
     @Perm(requires="pure(this) in alive",
     ensures="pure(this) in alive")
public int getstartDate() {
27
272
273
      return 0;
    @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public int getendDate() {
276
277
       return 0;
279
    @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
282
283
      public void setname(String name) {
    Perm(requires="share(this) in alive",
ensures="share(this) in alive")
public void setstartDate(int startDate) {
287
288
291
    @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
292
293
      public void setendDate(int endDate) {
29
296
     @Perm(requires="share(this) in alive",
     ensures="share(this) in alive")
public void setdTime(double dTime) {
298
299
    }
301
303 }ENDOFCLASS
    @ClassStates({@State(name = "alive")})
    class RatePath {
@Perm(ensures="unique(this) in alive")
RatePath() {
    }
307
309
     @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
311
312
      public ReturnPath getReturnCompounded() {
  return null;
31
31
    @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
317
318
      public ReturnPath getReturnNonCompounded() {
       return null:
320
```

```
@Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
private void readRatesFile(String dirName, String filename) {
323
325
     Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getEndPathValue() {
return 0;
328
329
331
333
     OPerm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getPathValue(int index) {
334
336
       return 0;
337
339
     @Perm(requires="share(this) in alive",
340
     ensures="share(this) in alive")
public void incpathValue(double[] operandPath) {
341
342
344
     GPerm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double[] getpathValue() {
  return null;
345
347
350
     Perm(requires="share(this) in alive",
ensures="share(this) in alive")
public void setpathValue(double[] pathValue) {
352
353
355
     @Perm(requires="pure(this) in alive",
356
     ensures="pure(this) in alive")
public int[] getpathDate() {
  return null;
357
358
36
     @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
363
       public void setpathDate(int[] pathDate) {
366
     }
368 }ENDOFCLASS
370 @ClassStates({@State(name = "alive")})
     class ReturnPath {
     @Perm(ensures="unique(this) in alive")
ReturnPath() { }
374
     @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
376
377
      public void estimatePath() {
380
     @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
382
       public void computeMean() {
385
     Perm(requires="share(this) in alive",
ensures="share(this) in alive")
public void computeVariance() {
387
388
390
     Perm(requires="share(this) in alive",
ensures="share(this) in alive")
public void computeExpectedReturnRate() {
39
392
393
395
     @Perm(requires="share(this) in alive",
396
     ensures="share(this) in alive")
public void computeVolatility() {
398
     Perm(requires="pure(this) in alive",
400
401
     ensures="pure(this) in alive")
```

```
40$ public void dbgDumpFields() {
     Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getexpectedReturnRate() {
return 0;
406
407
409
411
     GPerm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getvolatility() {
return 0;
412
414
415
417
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public int getreturnDefinition() {
return 0;
418
419
420
423
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getvolatility2() {
425
426
        return 0:
429
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double[] getpathValue() {
430
43
433
        return null:
435
     @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
436
437
438
       public void setpathValue(double[] pathValue) {
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
44
     ensures="pure(this) in
ensures="pure(this) in alive"
public int getnPathValue() {
return 0;
442
444
446
     OPerm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setnPathValue(int nPathValue) {
447
449
     @Perm(requires="full(this) in alive",
452
     ensures="full(this) in alive")
453
45
       public void setreturnDefinition(int returnDefinition) {
456
     @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
457
458
      public void setexpectedReturnRate(double expectedReturnRate) {
461
     @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
463
       public void setvolatility(double volatility) {
    }
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
public void setvolatility2(double volatility2) {
466
468
469
471
     @Perm(requires="pure(this) in alive",
472
     ensures="pure(this) in alive")
public double getmean() {
473
474
        return 0;
477
     QPerm(requires="share(this) in alive",
ensures="share(this) in alive")
public void setmean(double mean) {
479
480
     OPerm(requires="pure(this) in alive",
```

```
ensures="pure(this) in alive")
public double getvariance() {
return 0;
485
488
     OPerm(requires="share(this) in alive",
ensures="share(this) in alive")
490
      public void setvariance(double variance) {
49
493 }
495 FENDOFCLASS
    @ClassStates({@State(name = "alive")})
    class MonteCarloPath {
    @Perm(ensures="unique(this) in alive")
MonteCarloPath() {
    }
501
    @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
503
504
      private void copyInstanceVariables(ReturnPath obj) {
507
    OPerm(requires="share(this) in alive",
ensures="share(this) in alive")
public void setreturnDefinition(int returnDefinition) {
508
509
512
    Perm(requires="share(this) in alive",
ensures="share(this) in alive")
public void setexpectedReturnRate(double expectedReturnRate) {
514
515
517
    @Perm(requires="share(this) in alive",
518
     ensures="share(this) in alive")
public void setvolatility(double volatility) {
520
522
    Perm(requires="share(this) in alive",
ensures="share(this) in alive")
public void setnTimeSteps(int nTimeSteps) {
523
525
527
    @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
528
      public void setpathStartValue(double pathStartValue) {
    @Perm(requires="share(this) in alive",
533
    ensures="share(this) in alive")
534
      public void setpathValue(double[] pathValue) {
    @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
538
539
     public void setfluctuations(double[] fluctuations) {
542
    @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
544
      public void computeFluctuationsGaussian(long randomSeed) {
    @Perm(requires="share(this) in alive",
    ensures="share(this) in alive")
public void computePathValue(double startValue) {
549
550
552
    @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double[] getpathValue() {
553
554
555
      return null;
558
    @Perm(requires="pure(this) in alive",
    ensures="pure(this) in alive")
public int getnTimeSteps() {
560
      return 0;
```

```
566 @Perm(requires="pure(this) in alive",
566 ensures="pure(this) in alive")
567 public double[] getfluctuations() {
        return null:
568
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public int getreturnDefinition() {
return 0;
57
57
576
     QPerm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getexpectedReturnRate() {
577
579
        return 0;
580
582
      @Perm(requires="pure(this) in alive",
583
      ensures="pure(this) in alive")
public double getvolatility() {
584
585
        return 0;
588
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getpathStartValue() {
590
592
        return 0;
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
595
596
597
       public void writeFile(String dirName, String filename) {
599 }
       public RatePath getRatePath() {
601
        return null;
602
604
     @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
606
       public void computeFluctuationsGaussianOverload() {
609 }
611 }ENDOFCLASS
613 @ClassStates({@State(name = "alive")})
     class ToInitAllTasks {
     @Perm(ensures="unique(this) in alive")
ToInitAllTasks() { }
617
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public String getname() {
619
620
622
        return null;
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public int getstartDate() {
return 0;
625
626
627
628
630
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public int getendDate() {
  return 0;
63
633
634
636
     Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getdTime() {
637
638
639
        return 0;
642
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public int getreturnDefinition() {
644
```

```
646 return 0;
    @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getexpectedReturnRate() {
return 0;
649
650
652
654
    @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getvolatility() {
return 0;
655
657
658
660
    Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public int getnTimeSteps() {
663
662
663
       return 0;
664
666
    @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getpathStartValue() {
668
669
       return 0:
672
    @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public String getheader() {
673
674
676
       return null:
678
    @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
679
680
683
      public void setheader(String header) {
    @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
684
685
      public void setname(String name) {
688
689
     @Perm(requires="full(this) in alive",
     ensures="full(this) in alive")
public void setstartDate(int startDate) {
690
693
    @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
695
      public void setendDate(int endDate) {
696
698
     @Perm(requires="full(this) in alive",
699
     ensures="full(this) in alive")
public void setDTime(double dTime) {
700
70
703
     @Perm(requires="full(this) in alive",
704
     ensures="full(this) in alive")
public void setReturnDefinition(int returnDefinition) {
706
708
    @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
709
      public void setExpectedReturnRate(double expectedReturnRate) {
71
    @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
714
715
      public void setVolatility(double volatility) {
716
    @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
719
720
      public void setnTimeSteps(int nTimeSteps) {
723
724
    @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
725
      public void setpathStartValue(double pathStartValue) {
```

```
728 }
730 PENDOFCLASS
732 @ClassStates({@State(name = "alive")})
     class ToResult {
     @Perm(ensures="unique(this) in alive")
ToResult() {
    }
735
736
     @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getexpectedReturnRate() {
738
739
74
      return 0;
    GPerm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getvolatility() {
return 0;
744
746
747
     @Perm(requires="pure(this) in alive",
750
     ensures="pure(this) in alive")
public String toString() {
  return null;
752
755
     @Perm(requires="pure(this) in alive",
     ensures="pure(this) in alive")
public String getheader() {
757
758
759
       return null;
762
    @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
763
      public void setheader(String header) {
766
    @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
768
      public void setexpectedReturnRate(double expectedReturnRate) {
     Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setvolatility(double volatility) {
773
77
    Perm(requires="pure(this) in alive",
776
777
     ensures="pure(this) in alive")
public double getVolatility2() {
779
      return 0;
782
     Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setvolatility2(double volatility2) {
784
     @Perm(requires="pure(this) in alive",
     ensures="pure(this) in alive",
public double getfinalStockPrice() {
return 0;
789
790
    Perm(requires="full(this) in alive",
ensures="full(this) in alive")
793
795
      public void setfinalStockPrice(double finalStockPrice) {
796
798
     Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double[] getpathValue() {
799
800
803
       return null;
    @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setpathValue(double[] pathValue) {
806
```

```
809 }
    }ENDOFCLASS
811
    @ClassStates({@State(name = "alive")})
    class PriceStock {
815
    @Perm(ensures="unique(this) in alive")
PriceStock() {
    }
817
    @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
819
820
     public void setInitAllTasks(Object obj) {
823
    @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
824
825
     public void setTask(Object obj) {
826
828
    Perm(requires="share(this) in alive",
ensures="share(this) in alive")
public void run() {
830
83
833
     @Perm(requires="pure(this) in alive",
834
    public Object getResult() {
  return null;
835
836
839 }
841 }ENDOFCLASS
    @ClassStates({@State(name = "alive")})
    class ToTask {
    @Perm(ensures="unique(this) in alive")
ToTask() { }
846
847
    @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
850
85
     public String getheader() {
852
     return null;
854
    @Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
855
    ensures="pure(this) in alive")
public long getrandomSeed() {
  return 0;
857
858
860
    @Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setheader(String header) {
863
862
863
865
    @Perm(requires="full(this) in alive",
866
    ensures="full(this) in alive")
public void setrandomSeed(long randomSeed) {
868
870 }
    }ENDOFCLASS
    @ClassStates({@State(name = "alive")})
874
    class DemoException {
876
    @Perm(ensures="unique(this) in alive")
DemoException() { }
87
881 }ENDOFCLASS
    @ClassStates({@State(name = "alive")})
    class JGFInstrumentor {
    @Perm(ensures="unique(this) in alive")
JGFInstrumentor() {
}
887
```

```
void addTimer(String name) {
893
    OPerm(requires="share(this) in alive",
ensures="share(this) in alive")
895
       void startTimer(String name) {
896
898
    @Perm(requires="share(this) in alive",
    ensures="share(this) in alive")
void stopTimer(String name) {
900
901
903
    Perm(requires="share(this) in alive",
ensures="share(this) in alive")
904
905
      void addOpsToTimer(String name, double count) {
906
908
    @Perm(requires="share(this) in alive",
909
    ensures="share(this) in alive")
double readTimer(String name) {
91
     return 0;
912
914
    @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
915
916
      void resetTimer(String name) {
91
919
    @Perm(requires="share(this) in alive",
920
    ensures="share(this) in alive")
       void printTimer(String name) {
922
924
    @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
925
927
      void printperfTimer(String name) {
    @Perm(requires="share(this) in alive",
930
93
    ensures=
       void storeData(String name, Object obj) {
    @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
935
936
       void retrieveData(String name, Object obj) {
939 }
      void printHeader(int section, int size) {
941
    @Perm(requires="unique(this) in alive",
944
    ensures="unique(this) in alive")
void main(String argv[]) {
946
   }
   }ENDOFCLASS
    @ClassStates({@State(name = "alive")})
952
954
    class JGFTimer {
    @Perm(ensures="unique(this) in alive")
JGFTimer() {
955
    @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
public void start() {
958
959
960
962
    @Perm(requires="share(this) in alive",
963
    ensures="share(this) in alive")
public void stop() {
965
967
   @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
968
```

```
97¢ public void addops(double count) {
     @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 973
 974
      public void reset() {
 977
     @Perm(requires="share(this) in alive",
ensures="share(this) in alive")
public void print() {
 978
 979
 982
     Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double perf() {
 984
 98
 986
       return 0;
     Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public void printperf() {
 989
 990
 993
     @Perm(requires="pure(this) in alive",
     ensures="pure(this) in alive")
public void longprint() {
 995
 996
     }
 998
1000 }ENDOFCLASS
1002
     @ClassStates({@State(name = "alive")})
     class test {
@Perm(ensures="unique(this) in alive")
test() {
}
1004
1005
1006
    @Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
1008
1009
      public void createObject() {
1012
1013
     @Perm(requires="pure(this) in alive",
     ensures="pure(this) in alive")
public void readA() {
1014
1015
1017
     @Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
void main(String[] arg) {
1019
1020
1022
1024 }ENDOFCLASS
1026 @ClassStates({@State(name = "alive")})
     class Utilities {
1028
     @Perm(ensures="unique(this) in alive")
Utilities() {
    }
1030
1032
     @Perm(requires="unique(this) in alive",
1033
     ensures="unique(this) in alive")
        String which (String executable, String pathEnv) {
1035
      return null;
1037
     OPerm(requires="immutable(this) in alive",
ensures="immutable(this) in alive")
1038
1039
        String[] splitString(String splitChar, String arg) {
1040
      return null;
104
1043
        String joinString(String joinChar, String stringArray[]) {
1046
       return null;
1048 }
       String joinStringOverloaded(String joinChar, String stringArray[], int index) {
```

```
return null;
1053 }
1055 }ENDOFCLASS
```