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

Table 2: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
JGFMonteCarloBenchSizeA	√
main	√

Table 3: State Transition Matrix

	alive
alive	↑

Table 4: Methods Concurrency Matrix

	JGFMonteCarloBenchSizeA	main
JGFMonteCarloBenchSizeA	∦	∦
main	∦	∦

2 JGFMonteCarloBench

Table 5: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
JGFMonteCarloBench	√
JGFrun	√
JGFsetsize	√
JGFinitialise	√
JGFapplication	√
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	∦	∦	∦	∦	∦	∦	∦

3 CallAppDemo

Table 8: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
CallAppDemo	√
initialise	√
presults	√
runiters	√

Table 9: State Transition Matrix

	alive
alive	↑

Table 10: Methods Concurrency Matrix

	CallAppDemo	initialise	presults	runiters
CallAppDemo	∦	∦	∦	∦
initialise	∦	∦	∦	∦
presults	∦	∦	∦	∦
runiters	∦	∦	∦	∦

4 AppDemo

Table 11: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
AppDemo	√
initSerial	√
initTasks	√
processSerial	√
processResults	√
runSerial	√
getdataDirname	√
setdataDirname	√
getdataFilename	√
setdataFilename	√
getnTimeStepsMC	√
setnTimeStepsMC	√
getnRunsMC	√
setnRunsMC	√
gettasks	√
settasks	√
getresults	√
setresults	√

Table 12: State Transition Matrix

	alive
alive	↑

Table 13: Methods Concurrency Matrix

	AppDemo	initSerial	initTasks	processSerial	processResults	runSerial	getdataDirname	setdataDirname	getdataFilename	setdataFilename	getnTimeStepsMC	setnTimeStepsMC	getnRunsMC	setnRunsMC	gettasks	settasks	getresults	setresults
AppDemo	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
initSerial	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
initTasks	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
processSerial	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
processResults	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
runSerial	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getdataDirname	∦	∦	∦	∦	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦
setdataDirname	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getdataFilename	∦	∦	∦	∦	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦
setdataFilename	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getnTimeStepsMC	∦	∦	∦	∦	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦
setnTimeStepsMC	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦

getnRunsMC	∦	∦	∦	∦	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦
setnRunsMC	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
gettasks	∦	∦	∦	∦	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦
settasks	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getresults	∦	∦	∦	∦	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦
setresults	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦

5 Universal

Table 14: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
Universal	√
setprompt	√
setDEBUG	√
dbgPrintln	√
errPrintln	√
getDEBUG	√
getUNIVERSALDEBUG	√
setUNIVERSALDEBUG	√
getprompt	√
dbgPrint	√
errPrint	√

Table 15: State Transition Matrix

	alive
alive	↑

Table 16: Methods Concurrency Matrix

	Universal	setprompt	setDEBUG	dbgPrintln	errPrintln	getDEBUG	getUNIVERSALDEBUG	setUNIVERSALDEBUG	getprompt	dbgPrint	errPrint
Universal	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setprompt	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setDEBUG	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
dbgPrintln	∦	∦	∦	∥	∥	∥	∥	∦	∥	∥	∥
errPrintln	∦	∦	∦	∥	∥	∥	∥	∦	∥	∥	∥
getDEBUG	∦	∦	∦	∥	∥	∥	∥	∦	∥	∥	∥
getUNIVERSALDEBUG	∦	∦	∦	∥	∥	∥	∥	∦	∥	∥	∥
setUNIVERSALDEBUG	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getprompt	∦	∦	∦	∥	∥	∥	∥	∦	∥	∥	∥
dbgPrint	∦	∦	∦	∥	∥	∥	∥	∦	∥	∥	∥
errPrint	∦	∦	∦	∥	∥	∥	∥	∦	∥	∥	∥

6 PathId

Table 17: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
PathId	√
dbgDumpFields	√
copyInstanceVariables	√
getdTime	√
getname	√
getstartDate	√
getendDate	√
setname	√
setstartDate	√
setendDate	√
setdTime	√

Table 18: State Transition Matrix

	alive
alive	↑

Table 19: Methods Concurrency Matrix

	PathId	dbgDumpFields	copyInstanceVariables	getdTime	getname	getstartDate	getendDate	setname	setstartDate	setendDate	setdTime
PathId	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
dbgDumpFields	∦	∥	∦	∥	∥	∥	∥	∦	∦	∦	∦
copyInstanceVariables	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getdTime	∦	∥	∦	∥	∥	∥	∥	∦	∦	∦	∦
getname	∦	∥	∦	∥	∥	∥	∥	∦	∦	∦	∦
getstartDate	∦	∥	∦	∥	∥	∥	∥	∦	∦	∦	∦
getendDate	∦	∥	∦	∥	∥	∥	∥	∦	∦	∦	∦
setname	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setstartDate	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setendDate	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setdTime	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦

7 RatePath

Table 20: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
RatePath	√
getReturnCompounded	√
getReturnNonCompounded	√
readRatesFile	√
getEndPathValue	√
getPathValue	√
incpathValue	√
getpathValue	√
setpathValue	√
getpathDate	√
setpathDate	√

Table 21: State Transition Matrix

	alive
alive	↑

Table 22: Methods Concurrency Matrix

	RatePath	getReturnCompounded	getReturnNonCompounded	readRatesFile	getEndPathValue	getPathValue	incpathValue	getpathValue	setpathValue	getpathDate	setpathDate
RatePath	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getReturnCompounded	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getReturnNonCompounded	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
readRatesFile	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getEndPathValue	∦	∦	∦	∦	‖	‖	∦	‖	∦	‖	∦
getPathValue	∦	∦	∦	∦	‖	‖	∦	‖	∦	‖	∦
incpathValue	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getpathValue	∦	∦	∦	∦	‖	‖	∦	‖	∦	‖	∦
setpathValue	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getpathDate	∦	∦	∦	∦	‖	‖	∦	‖	∦	‖	∦
setpathDate	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦

8 ReturnPath

Table 23: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
ReturnPath	√
estimatePath	√
computeMean	√
computeVariance	√
computeExpectedReturnRate	√
computeVolatility	√
dbgDumpFields	√
getexpectedReturnRate	√
getvolatility	√
getreturnDefinition	√
getvolatility2	√
getpathValue	√
setpathValue	√
getnPathValue	√
setnPathValue	√
setreturnDefinition	√
setexpectedReturnRate	√
setvolatility	√
setvolatility2	√
getmean	√
setmean	√
getvariance	√
setvariance	√

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	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦

computeVariance	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
computeExpectedReturnRate	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
computeVolatility	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
dbgDumpFields	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	∦	‖	∦	∦	∦	∦	∦	‖
getexpectedReturnRate	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	∦	‖	∦	∦	∦	∦	∦	‖
getvolatility	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	∦	‖	∦	∦	∦	∦	∦	‖
getreturnDefinition	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	∦	‖	∦	∦	∦	∦	∦	‖
getvolatility2	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	∦	‖	∦	∦	∦	∦	∦	‖
getpathValue	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	∦	‖	∦	∦	∦	∦	∦	‖
setpathValue	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getnPathValue	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	∦	‖	∦	∦	∦	∦	∦	‖
setnPathValue	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setreturnDefinition	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setexpectedReturnRate	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setvolatility	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setvolatility2	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getmean	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	∦	‖	∦	∦	∦	∦	∦	‖
setmean	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getvariance	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	∦	‖	∦	∦	∦	∦	∦	‖
setvariance	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦

9 MonteCarloPath

Table 26: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
MonteCarloPath	√
copyInstanceVariables	√
setreturnDefinition	√
setexpectedReturnRate	√
setvolatility	√
setnTimeSteps	√
setpathStartValue	√
setpathValue	√
setfluctuations	√
computeFluctuationsGaussian	√
computePathValue	√
getpathValue	√
getnTimeSteps	√
getfluctuations	√
getreturnDefinition	√
getexpectedReturnRate	√
getvolatility	√
getpathStartValue	√
writeFile	√
getRatePath	√
computeFluctuationsGaussianOverload	√

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	getpathStartValue
MonteCarloPath	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
copyInstanceVariables	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦

setreturnDefinition	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setexpectedReturnRate	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setvolatility	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setnTimeSteps	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setpathStartValue	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setpathValue	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setfluctuations	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
computeFluctuationsGaussian	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
computePathValue	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getpathValue	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	‖
getnTimeSteps	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	‖
getfluctuations	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	‖
getreturnDefinition	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	‖
getexpectedReturnRate	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	‖
getvolatility	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	‖
getpathStartValue	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	‖
writeFile	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	‖	‖	‖	‖	‖	‖
getRatePath	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
computeFluctuationsGaussianOverload	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦

10 ToInitAllTasks

Table 29: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
ToInitAllTasks	√
getname	√
getstartDate	√
getendDate	√
getdTime	√
getreturnDefinition	√
getexpectedReturnRate	√
getvolatility	√
getnTimeSteps	√
getpathStartValue	√
getheader	√
setheader	√
setname	√
setstartDate	√
setendDate	√
setDTime	√
setReturnDefinition	√
setExpectedReturnRate	√
setVolatility	√
setnTimeSteps	√
setpathStartValue	√

Table 30: State Transition Matrix

	alive
alive	↑

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	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getname	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getstartDate	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getendDate	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getdTime	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getreturnDefinition	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getexpectedReturnRate	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦

getvolatility	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getnTimeSteps	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getpathStartValue	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getheader	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setheader	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setname	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setstartDate	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setendDate	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setDTime	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setReturnDefinition	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setExpectedReturnRate	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setVolatility	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setnTimeSteps	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setpathStartValue	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦

11 ToResult

Table 32: Method's Satisfiability(Code Reachabiity Analysis

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

Table 33: State Transition Matrix

	alive
alive	↑

Table 34: Methods Concurrency Matrix

	ToResult	getexpectedReturnRate	getvolatility	toString	getheader	setheader	setexpectedReturnRate	setvolatility	getVolatility2	setvolatility2	getfinalStockPrice	setfinalStockPrice	getpathValue	setpathValue
ToResult	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getexpectedReturnRate	∦	‖	‖	‖	‖	∦	∦	∦	‖	∦	‖	∦	‖	∦
getvolatility	∦	‖	‖	‖	‖	∦	∦	∦	‖	∦	‖	∦	‖	∦
toString	∦	‖	‖	‖	‖	∦	∦	∦	‖	∦	‖	∦	‖	∦
getheader	∦	‖	‖	‖	‖	∦	∦	∦	‖	∦	‖	∦	‖	∦
setheader	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setexpectedReturnRate	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
setvolatility	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getVolatility2	∦	‖	‖	‖	‖	∦	∦	∦	‖	∦	‖	∦	‖	∦
setvolatility2	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getfinalStockPrice	∦	‖	‖	‖	‖	∦	∦	∦	‖	∦	‖	∦	‖	∦
setfinalStockPrice	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getpathValue	∦	‖	‖	‖	‖	∦	∦	∦	‖	∦	‖	∦	‖	∦
setpathValue	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦

12 PriceStock

Table 35: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
PriceStock	√
setInitAllTasks	√
setTask	√
run	√
getResult	√

Table 36: State Transition Matrix

	alive
alive	↑

Table 37: Methods Concurrency Matrix

	PriceStock	setInitAllTasks	setTask	run	getResult
PriceStock	∦	∦	∦	∦	∦
setInitAllTasks	∦	∦	∦	∦	∦
setTask	∦	∦	∦	∦	∦
run	∦	∦	∦	∦	∦
getResult	∦	∦	∦	∦	‖

13 ToTask

Table 38: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
ToTask	√
getheader	√
getrandomSeed	√
setheader	√
setrandomSeed	√

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	√

Table 42: State Transition Matrix

	alive
alive	↑

15 JGFInstrumentor

Table 43: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
JGFInstrumentor	√
addTimer	√
startTimer	√
stopTimer	√
addOpsToTimer	√
readTimer	√
resetTimer	√
printTimer	√
printperfTimer	√
storeData	√
retrieveData	√
printHeader	√
main	√

Table 44: State Transition Matrix

	alive
alive	↑

Table 45: Methods Concurrency Matrix

	JGFInstrumentor	addTimer	startTimer	stopTimer	addOpsToTimer	readTimer	resetTimer	printTimer	printperfTimer	storeData	retrieveData	printHeader	main
JGFInstrumentor	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
addTimer	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	∦
startTimer	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	∦
stopTimer	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	∦
addOpsToTimer	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	∦
readTimer	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	∦
resetTimer	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	∦
printTimer	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	∦
printperfTimer	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	∦
storeData	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	∦
retrieveData	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	∦
printHeader	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
main	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	‖	∦

16 JGFTimer

Table 46: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
JGFTimer	√
start	√
stop	√
addops	√
reset	√
print	√
perf	√
printperf	√
longprint	√

Table 47: State Transition Matrix

	alive
alive	↑

Table 48: Methods Concurrency Matrix

	JGFTimer	start	stop	addops	reset	print	perf	printperf	longprint
JGFTimer	∦	∦	∦	∦	∦	∦	∦	∦	∦
start	∦	∦	∦	∦	∦	∦	∦	∦	∦
stop	∦	∦	∦	∦	∦	∦	∦	∦	∦
addops	∦	∦	∦	∦	∦	∦	∦	∦	∦
reset	∦	∦	∦	∦	∦	∦	∦	∦	∦
print	∦	∦	∦	∦	∦	∦	∦	∦	∦
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	↑

Table 51: Methods Concurrency Matrix

	test	createObject	readA	main
test	∦	∦	∦	∦
createObject	∦	∦	∦	∦
readA	∦	∦	‖	∦
main	∦	∦	∦	∦

18 Utilities

Table 52: Method's Satisfiability(Code Reachabiity Analysis

Method	Satisfiability
Utilities	√
which	√
splitString	√
joinString	√
joinStringOverloaded	√

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
$\surd$	requires clause of the method is satisfiable
$\times$	requires clause of the method is unsatisfiable
$\uparrow$	The row-state can be transitioned to the column-state
$\times$	The row-state cannot be transitioned to the column-state
$\parallel$	The row-method can be possibly executed parallel with the column-method
$\nparallel$	The row-method cannot be executed parallel with the column-method

20 Annotated version of the input program generated by Sip4J

```
1  package outputs;
2  import edu.cmu.cs.plural.annot.*;

4  @ClassStates({@State(name = "alive")})
5  class JGFMonteCarloBenchSizeA {
6  @Perm(ensures="unique(this) in alive")
7  JGFMonteCarloBenchSizeA() {   }

9  @Perm(requires="unique(this) in alive",
10 ensures="unique(this) in alive")
11   void main(String argv[]) {

13 }

15 }ENDOFCLASS

17 @ClassStates({@State(name = "alive")})

19 class JGFMonteCarloBench {
20 @Perm(ensures="unique(this) in alive")
21 JGFMonteCarloBench() {   }

23 @Perm(requires="unique(this) in alive",
24 ensures="unique(this) in alive")
25  public void JGFrun(int size) {

27 }
28 @Perm(requires="share(this) in alive",
29 ensures="share(this) in alive")
30  public void JGFsetsize(int size) {

32 }
33 @Perm(requires="unique(this) in alive",
34 ensures="unique(this) in alive")
35  public void JGFinitialise() {

37 }
38 @Perm(requires="unique(this) in alive",
39 ensures="unique(this) in alive")
40  public void JGFapplication() {

42 }
43 @Perm(requires="pure(this) in alive",
44 ensures="pure(this) in alive")
45  public void JGFvalidate() {

47 }
48 @Perm(requires="unique(this) in alive",
49 ensures="unique(this) in alive")
50  public void JGFtidyup() {

52 }

54 }ENDOFCLASS

56 @ClassStates({@State(name = "alive")})

58 class CallAppDemo {
59 @Perm(ensures="unique(this) in alive")
60 CallAppDemo() {   }

62 @Perm(requires="unique(this) in alive",
63 ensures="unique(this) in alive")
64  public void initialise() {

66 }
67 @Perm(requires="share(this) in alive",
68 ensures="share(this) in alive")
69  public void presults() {

71 }
72 @Perm(requires="unique(this) in alive",
73 ensures="unique(this) in alive")
74  public void runiters() {

76 }

78 }ENDOFCLASS
```

```
80 @ClassStates({@State(name = "alive")})

82 class AppDemo {
83 @Perm(ensures="unique(this) in alive")
84 AppDemo() {   }

86 @Perm(requires="unique(this) in alive",
87 ensures="unique(this) in alive")
88  public void initSerial() {

90 }
91 @Perm(requires="unique(this) in alive",
92 ensures="unique(this) in alive")
93  private void initTasks(int nRunsMC) {

95 }
96 @Perm(requires="share(this) in alive",
97 ensures="share(this) in alive")
98  public void processSerial() {

100 }
101 @Perm(requires="share(this) in alive",
102 ensures="share(this) in alive")
103  private void processResults() {

105 }
106 @Perm(requires="unique(this) in alive",
107 ensures="unique(this) in alive")
108  public void runSerial() {

110 }
111 @Perm(requires="pure(this) in alive",
112 ensures="pure(this) in alive")
113  public String getdataDirname() {
114  return null;

116 }
117 @Perm(requires="full(this) in alive",
118 ensures="full(this) in alive")
119  public void setdataDirname(String dataDirname) {

121 }
122 @Perm(requires="pure(this) in alive",
123 ensures="pure(this) in alive")
124  public String getdataFilename() {
125  return null;

127 }
128 @Perm(requires="full(this) in alive",
129 ensures="full(this) in alive")
130  public void setdataFilename(String dataFilename) {

132 }
133 @Perm(requires="pure(this) in alive",
134 ensures="pure(this) in alive")
135  public int getnTimeStepsMC() {
136  return 0;

138 }
139 @Perm(requires="full(this) in alive",
140 ensures="full(this) in alive")
141  public void setnTimeStepsMC(int nTimeStepsMC) {

143 }
144 @Perm(requires="pure(this) in alive",
145 ensures="pure(this) in alive")
146  public int getnRunsMC() {
147  return 0;

149 }
150 @Perm(requires="full(this) in alive",
151 ensures="full(this) in alive")
152  public void setnRunsMC(int nRunsMC) {

154 }
155 @Perm(requires="pure(this) in alive",
156 ensures="pure(this) in alive")
157  public Vector gettasks() {
158  return null;
```

```
160  }
161  @Perm(requires="share(this) in alive",
162  ensures="share(this) in alive")
163   public void settasks(Vector tasks) {

165  }
166  @Perm(requires="pure(this) in alive",
167  ensures="pure(this) in alive")
168   public Vector getresults() {
169   return null;

171  }
172  @Perm(requires="share(this) in alive",
173  ensures="share(this) in alive")
174   public void setresults(Vector results) {

176  }

178  }ENDOFCLASS

180  @ClassStates({@State(name = "alive")})

182  class Universal {
183  @Perm(ensures="unique(this) in alive")
184  Universal() {   }

186  @Perm(requires="full(this) in alive",
187  ensures="full(this) in alive")
188   public void setprompt(String prompt) {

190  }
191  @Perm(requires="full(this) in alive",
192  ensures="full(this) in alive")
193   public void setDEBUG(boolean DEBUG) {

195  }
196  @Perm(requires="pure(this) in alive",
197  ensures="pure(this) in alive")
198   public void dbgPrintln(String s) {

200  }
201  @Perm(requires="pure(this) in alive",
202  ensures="pure(this) in alive")
203   public void errPrintln(String s) {

205  }
206  @Perm(requires="pure(this) in alive",
207  ensures="pure(this) in alive")
208   public boolean getDEBUG() {
209   return 0;

211  }
212  @Perm(requires="pure(this) in alive",
213  ensures="pure(this) in alive")
214   public boolean getUNIVERSALDEBUG() {
215   return 0;

217  }
218  @Perm(requires="full(this) in alive",
219  ensures="full(this) in alive")
220   public void setUNIVERSALDEBUG(boolean UNIVERSAL_DEBUG) {

222  }
223  @Perm(requires="pure(this) in alive",
224  ensures="pure(this) in alive")
225   public String getprompt() {
226   return null;

228  }
229  @Perm(requires="pure(this) in alive",
230  ensures="pure(this) in alive")
231   public void dbgPrint(String s) {

233  }
234  @Perm(requires="pure(this) in alive",
235  ensures="pure(this) in alive")
236   public void errPrint(String s) {

238  }

240  }ENDOFCLASS
```

```
@ClassStates({@State(name = "alive")})

class PathId {
@Perm(ensures="unique(this) in alive")
PathId() {   }

@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
 public void dbgDumpFields() {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void copyInstanceVariables(PathId obj) {

}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
 public double getdTime() {
 return 0;

}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
 public String getname() {
 return null;

}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
 public int getstartDate() {
 return 0;

}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
 public int getendDate() {
 return 0;

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void setname(String name) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void setstartDate(int startDate) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void setendDate(int endDate) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void setdTime(double dTime) {

}

}ENDOFCLASS

@ClassStates({@State(name = "alive")})

class RatePath {
@Perm(ensures="unique(this) in alive")
RatePath() {   }

@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public ReturnPath getReturnCompounded() {
 return null;

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public ReturnPath getReturnNonCompounded() {
 return null;
```

```
 }
@Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
 private void readRatesFile(String dirName, String filename) {

 }
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
 public double getEndPathValue() {
 return 0;

 }
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
 public double getPathValue(int index) {
 return 0;

 }
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void incpathValue(double[] operandPath) {

 }
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
 public double[] getpathValue() {
 return null;

 }
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void setpathValue(double[] pathValue) {

 }
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
 public int[] getpathDate() {
 return null;

 }
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void setpathDate(int[] pathDate) {

 }

}ENDOFCLASS

@ClassStates({@State(name = "alive")})

class ReturnPath {
@Perm(ensures="unique(this) in alive")
ReturnPath() {   }

@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void estimatePath() {

 }
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void computeMean() {

 }
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void computeVariance() {

 }
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void computeExpectedReturnRate() {

 }
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void computeVolatility() {

 }
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
```

```
403   public void dbgDumpFields() {

405  }
406  @Perm(requires="pure(this) in alive",
407  ensures="pure(this) in alive")
408   public double getexpectedReturnRate() {
409   return 0;

411  }
412  @Perm(requires="pure(this) in alive",
413  ensures="pure(this) in alive")
414   public double getvolatility() {
415   return 0;

417  }
418  @Perm(requires="pure(this) in alive",
419  ensures="pure(this) in alive")
420   public int getreturnDefinition() {
421   return 0;

423  }
424  @Perm(requires="pure(this) in alive",
425  ensures="pure(this) in alive")
426   public double getvolatility2() {
427   return 0;

429  }
430  @Perm(requires="pure(this) in alive",
431  ensures="pure(this) in alive")
432   public double[] getpathValue() {
433   return null;

435  }
436  @Perm(requires="full(this) in alive",
437  ensures="full(this) in alive")
438   public void setpathValue(double[] pathValue) {

440  }
441  @Perm(requires="pure(this) in alive",
442  ensures="pure(this) in alive")
443   public int getnPathValue() {
444   return 0;

446  }
447  @Perm(requires="full(this) in alive",
448  ensures="full(this) in alive")
449   public void setnPathValue(int nPathValue) {

451  }
452  @Perm(requires="full(this) in alive",
453  ensures="full(this) in alive")
454   public void setreturnDefinition(int returnDefinition) {

456  }
457  @Perm(requires="share(this) in alive",
458  ensures="share(this) in alive")
459   public void setexpectedReturnRate(double expectedReturnRate) {

461  }
462  @Perm(requires="share(this) in alive",
463  ensures="share(this) in alive")
464   public void setvolatility(double volatility) {

466  }
467  @Perm(requires="share(this) in alive",
468  ensures="share(this) in alive")
469   public void setvolatility2(double volatility2) {

471  }
472  @Perm(requires="pure(this) in alive",
473  ensures="pure(this) in alive")
474   public double getmean() {
475   return 0;

477  }
478  @Perm(requires="share(this) in alive",
479  ensures="share(this) in alive")
480   public void setmean(double mean) {

482  }
483  @Perm(requires="pure(this) in alive",
```

```
ensures="pure(this) in alive")
 public double getvariance() {
 return 0;

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void setvariance(double variance) {

}

}ENDOFCLASS

@ClassStates({@State(name = "alive")})

class MonteCarloPath {
@Perm(ensures="unique(this) in alive")
MonteCarloPath() {   }

@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 private void copyInstanceVariables(ReturnPath obj) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void setreturnDefinition(int returnDefinition) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void setexpectedReturnRate(double expectedReturnRate) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void setvolatility(double volatility) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void setnTimeSteps(int nTimeSteps) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void setpathStartValue(double pathStartValue) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void setpathValue(double[] pathValue) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void setfluctuations(double[] fluctuations) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void computeFluctuationsGaussian(long randomSeed) {

}
@Perm(requires="share(this) in alive",
ensures="share(this) in alive")
 public void computePathValue(double startValue) {

}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
 public double[] getpathValue() {
 return null;

}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
 public int getnTimeSteps() {
 return 0;

}
```

```
565 @Perm(requires="pure(this) in alive",
566 ensures="pure(this) in alive")
567  public double[] getfluctuations() {
568  return null;

570 }
571 @Perm(requires="pure(this) in alive",
572 ensures="pure(this) in alive")
573  public int getreturnDefinition() {
574  return 0;

576 }
577 @Perm(requires="pure(this) in alive",
578 ensures="pure(this) in alive")
579  public double getexpectedReturnRate() {
580  return 0;

582 }
583 @Perm(requires="pure(this) in alive",
584 ensures="pure(this) in alive")
585  public double getvolatility() {
586  return 0;

588 }
589 @Perm(requires="pure(this) in alive",
590 ensures="pure(this) in alive")
591  public double getpathStartValue() {
592  return 0;

594 }
595 @Perm(requires="pure(this) in alive",
596 ensures="pure(this) in alive")
597  public void writeFile(String dirName, String filename) {

599 }

601  public RatePath getRatePath() {
602  return null;

604 }
605 @Perm(requires="share(this) in alive",
606 ensures="share(this) in alive")
607  public void computeFluctuationsGaussianOverload() {

609 }

611 }ENDOFCLASS

613 @ClassStates({@State(name = "alive")})

615 class ToInitAllTasks {
616 @Perm(ensures="unique(this) in alive")
617 ToInitAllTasks() {   }

619 @Perm(requires="pure(this) in alive",
620 ensures="pure(this) in alive")
621  public String getname() {
622  return null;

624 }
625 @Perm(requires="pure(this) in alive",
626 ensures="pure(this) in alive")
627  public int getstartDate() {
628  return 0;

630 }
631 @Perm(requires="pure(this) in alive",
632 ensures="pure(this) in alive")
633  public int getendDate() {
634  return 0;

636 }
637 @Perm(requires="pure(this) in alive",
638 ensures="pure(this) in alive")
639  public double getdTime() {
640  return 0;

642 }
643 @Perm(requires="pure(this) in alive",
644 ensures="pure(this) in alive")
645  public int getreturnDefinition() {
```

```
646   return 0;

648  }
649  @Perm(requires="pure(this) in alive",
650  ensures="pure(this) in alive")
651   public double getexpectedReturnRate() {
652   return 0;

654  }
655  @Perm(requires="pure(this) in alive",
656  ensures="pure(this) in alive")
657   public double getvolatility() {
658   return 0;

660  }
661  @Perm(requires="pure(this) in alive",
662  ensures="pure(this) in alive")
663   public int getnTimeSteps() {
664   return 0;

666  }
667  @Perm(requires="pure(this) in alive",
668  ensures="pure(this) in alive")
669   public double getpathStartValue() {
670   return 0;

672  }
673  @Perm(requires="pure(this) in alive",
674  ensures="pure(this) in alive")
675   public String getheader() {
676   return null;

678  }
679  @Perm(requires="full(this) in alive",
680  ensures="full(this) in alive")
681   public void setheader(String header) {

683  }
684  @Perm(requires="full(this) in alive",
685  ensures="full(this) in alive")
686   public void setname(String name) {

688  }
689  @Perm(requires="full(this) in alive",
690  ensures="full(this) in alive")
691   public void setstartDate(int startDate) {

693  }
694  @Perm(requires="full(this) in alive",
695  ensures="full(this) in alive")
696   public void setendDate(int endDate) {

698  }
699  @Perm(requires="full(this) in alive",
700  ensures="full(this) in alive")
701   public void setDTime(double dTime) {

703  }
704  @Perm(requires="full(this) in alive",
705  ensures="full(this) in alive")
706   public void setReturnDefinition(int returnDefinition) {

708  }
709  @Perm(requires="full(this) in alive",
710  ensures="full(this) in alive")
711   public void setExpectedReturnRate(double expectedReturnRate) {

713  }
714  @Perm(requires="full(this) in alive",
715  ensures="full(this) in alive")
716   public void setVolatility(double volatility) {

718  }
719  @Perm(requires="full(this) in alive",
720  ensures="full(this) in alive")
721   public void setnTimeSteps(int nTimeSteps) {

723  }
724  @Perm(requires="full(this) in alive",
725  ensures="full(this) in alive")
726   public void setpathStartValue(double pathStartValue) {
```

```
728 }

730 }ENDOFCLASS

732 @ClassStates({@State(name = "alive")})

734 class ToResult {
735 @Perm(ensures="unique(this) in alive")
736 ToResult() {   }

738 @Perm(requires="pure(this) in alive",
739 ensures="pure(this) in alive")
740  public double getexpectedReturnRate() {
741  return 0;

743 }
744 @Perm(requires="pure(this) in alive",
745 ensures="pure(this) in alive")
746  public double getvolatility() {
747  return 0;

749 }
750 @Perm(requires="pure(this) in alive",
751 ensures="pure(this) in alive")
752  public String toString() {
753  return null;

755 }
756 @Perm(requires="pure(this) in alive",
757 ensures="pure(this) in alive")
758  public String getheader() {
759  return null;

761 }
762 @Perm(requires="full(this) in alive",
763 ensures="full(this) in alive")
764  public void setheader(String header) {

766 }
767 @Perm(requires="full(this) in alive",
768 ensures="full(this) in alive")
769  public void setexpectedReturnRate(double expectedReturnRate) {

771 }
772 @Perm(requires="full(this) in alive",
773 ensures="full(this) in alive")
774  public void setvolatility(double volatility) {

776 }
777 @Perm(requires="pure(this) in alive",
778 ensures="pure(this) in alive")
779  public double getVolatility2() {
780  return 0;

782 }
783 @Perm(requires="full(this) in alive",
784 ensures="full(this) in alive")
785  public void setvolatility2(double volatility2) {

787 }
788 @Perm(requires="pure(this) in alive",
789 ensures="pure(this) in alive")
790  public double getfinalStockPrice() {
791  return 0;

793 }
794 @Perm(requires="full(this) in alive",
795 ensures="full(this) in alive")
796  public void setfinalStockPrice(double finalStockPrice) {

798 }
799 @Perm(requires="pure(this) in alive",
800 ensures="pure(this) in alive")
801  public double[] getpathValue() {
802  return null;

804 }
805 @Perm(requires="full(this) in alive",
806 ensures="full(this) in alive")
807  public void setpathValue(double[] pathValue) {
```

```
809 }

811 }ENDOFCLASS

813 @ClassStates({@State(name = "alive")})

815 class PriceStock {
816 @Perm(ensures="unique(this) in alive")
817 PriceStock() {   }

819 @Perm(requires="share(this) in alive",
820 ensures="share(this) in alive")
821  public void setInitAllTasks(Object obj) {

823 }
824 @Perm(requires="share(this) in alive",
825 ensures="share(this) in alive")
826  public void setTask(Object obj) {

828 }
829 @Perm(requires="share(this) in alive",
830 ensures="share(this) in alive")
831  public void run() {

833 }
834 @Perm(requires="pure(this) in alive",
835 ensures="pure(this) in alive")
836  public Object getResult() {
837  return null;

839 }

841 }ENDOFCLASS

843 @ClassStates({@State(name = "alive")})

845 class ToTask {
846 @Perm(ensures="unique(this) in alive")
847 ToTask() {   }

849 @Perm(requires="pure(this) in alive",
850 ensures="pure(this) in alive")
851  public String getheader() {
852  return null;

854 }
855 @Perm(requires="pure(this) in alive",
856 ensures="pure(this) in alive")
857  public long getrandomSeed() {
858  return 0;

860 }
861 @Perm(requires="full(this) in alive",
862 ensures="full(this) in alive")
863  public void setheader(String header) {

865 }
866 @Perm(requires="full(this) in alive",
867 ensures="full(this) in alive")
868  public void setrandomSeed(long randomSeed) {

870 }

872 }ENDOFCLASS

874 @ClassStates({@State(name = "alive")})

876 class DemoException {
877 @Perm(ensures="unique(this) in alive")
878 DemoException() {   }


881 }ENDOFCLASS

883 @ClassStates({@State(name = "alive")})

885 class JGFInstrumentor {
886 @Perm(ensures="unique(this) in alive")
887 JGFInstrumentor() {   }
```

```
889 @Perm(requires="share(this) in alive",
890 ensures="share(this) in alive")
891   void addTimer(String name) {

893 }
894 @Perm(requires="share(this) in alive",
895 ensures="share(this) in alive")
896   void startTimer(String name) {

898 }
899 @Perm(requires="share(this) in alive",
900 ensures="share(this) in alive")
901   void stopTimer(String name) {

903 }
904 @Perm(requires="share(this) in alive",
905 ensures="share(this) in alive")
906   void addOpsToTimer(String name, double count) {

908 }
909 @Perm(requires="share(this) in alive",
910 ensures="share(this) in alive")
911   double readTimer(String name) {
912  return 0;

914 }
915 @Perm(requires="share(this) in alive",
916 ensures="share(this) in alive")
917   void resetTimer(String name) {

919 }
920 @Perm(requires="share(this) in alive",
921 ensures="share(this) in alive")
922   void printTimer(String name) {

924 }
925 @Perm(requires="share(this) in alive",
926 ensures="share(this) in alive")
927   void printperfTimer(String name) {

929 }
930 @Perm(requires="share(this) in alive",
931 ensures="share(this) in alive")
932   void storeData(String name, Object obj) {

934 }
935 @Perm(requires="share(this) in alive",
936 ensures="share(this) in alive")
937   void retrieveData(String name, Object obj) {

939 }

941   void printHeader(int section, int size) {

943 }
944 @Perm(requires="unique(this) in alive",
945 ensures="unique(this) in alive")
946   void main(String argv[]) {

948 }

950 }ENDOFCLASS

952 @ClassStates({@State(name = "alive")})

954 class JGFTimer {
955 @Perm(ensures="unique(this) in alive")
956 JGFTimer() {   }

958 @Perm(requires="share(this) in alive",
959 ensures="share(this) in alive")
960  public void start() {

962 }
963 @Perm(requires="share(this) in alive",
964 ensures="share(this) in alive")
965  public void stop() {

967 }
968 @Perm(requires="share(this) in alive",
969 ensures="share(this) in alive")
```

```
970   public void addops(double count) {

972  }
973  @Perm(requires="share(this) in alive",
974  ensures="share(this) in alive")
975   public void reset() {

977  }
978  @Perm(requires="share(this) in alive",
979  ensures="share(this) in alive")
980   public void print() {

982  }
983  @Perm(requires="pure(this) in alive",
984  ensures="pure(this) in alive")
985   public double perf() {
986   return 0;

988  }
989  @Perm(requires="pure(this) in alive",
990  ensures="pure(this) in alive")
991   public void printperf() {

993  }
994  @Perm(requires="pure(this) in alive",
995  ensures="pure(this) in alive")
996   public void longprint() {

998  }

1000 }ENDOFCLASS

1002 @ClassStates({@State(name = "alive")})

1004 class test {
1005 @Perm(ensures="unique(this) in alive")
1006 test() {   }

1008 @Perm(requires="unique(this) in alive",
1009 ensures="unique(this) in alive")
1010  public void createObject() {

1012 }
1013 @Perm(requires="pure(this) in alive",
1014 ensures="pure(this) in alive")
1015  public void readA() {

1017 }
1018 @Perm(requires="unique(this) in alive",
1019 ensures="unique(this) in alive")
1020   void main(String[] arg) {

1022 }

1024 }ENDOFCLASS

1026 @ClassStates({@State(name = "alive")})

1028 class Utilities {
1029 @Perm(ensures="unique(this) in alive")
1030 Utilities() {   }

1032 @Perm(requires="unique(this) in alive",
1033 ensures="unique(this) in alive")
1034   String which(String executable, String pathEnv) {
1035  return null;

1037 }
1038 @Perm(requires="immutable(this) in alive",
1039 ensures="immutable(this) in alive")
1040   String[] splitString(String splitChar, String arg) {
1041  return null;

1043 }

1045   String joinString(String joinChar, String stringArray[]) {
1046  return null;

1048 }

1050   String joinStringOverloaded(String joinChar, String stringArray[], int index) {
```

```
1051   return null;
1053  }
1055  }ENDOFCLASS
```