Summary

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

Table 1: Pulse Analysis Summary

Classes	Methods	States	Unsatisfiable Clauses	Unreachable States	Possible concurrent Methods	Total. no. of pairs	No. of concurrent pairs	Percentage of concurrent Methods
JGFMonteCarloBenchSizeA	2	1	0	0	1	3	1	33
JGFMonteCarloBench	6	1	0	0	5	21	15	71
JGFInstrumentor	4	1	0	0	0	10	0	0
CallAppDemo	3	1	0	0	2	6	3	50
AppDemo	15	1	0	0	14	120	105	88
Universal	11	1	0	0	10	66	49	74
JGFTimer	2	1	0	0	0	3	0	0
RatePath	11	1	0	0	10	66	55	83
PathId	11	1	0	0	10	66	10	15
PriceStock	3	1	0	0	2	6	3	50
MonteCarloPath	18	1	0	0	17	171	153	89
ToInitAllTasks	20	1	0	0	19	210	135	64
ReturnPath	21	1	0	0	20	231	20	9
DemoException	1	1	0	0	0	1	0	0
ToResult	14	1	0	0	13	105	70	67
ToTask	5	1	0	0	4	15	7	47
Utilities	4	1	0	0	3	10	6	60
Total Classes=17	151	17	0	0	130	1110	632	57

Contents

1 JGFMonteCarloBenchSizeA 3

2 JGFMonteCarloBench 4

3 JGFInstrumentor 5

4 CallAppDemo 6

5 AppDemo 7

6 Universal 8

7 JGFTimer 9

8 RatePath 10

9 PathId 11

10 PriceStock 12

11 MonteCarloPath 13

12 ToInitAllTasks 15

13 ReturnPath 17

14 DemoException 19

15 ToResult 20

16 ToTask 21

17 Utilities 22

18 Abbreviation 23

19 Annotated Version of Sequential Java Program generated by Sip4j 24

1 JGFMonteCarloBenchSizeA

Table 2: Methods Requires Clause Satisfiability

Method	Satisfiability
JGFMonteCarloBenchSizeA	√
main	√

Table 3: State Transition Matrix

	alive
alive	↑

Table 4: Methods Concurrency Matrix

	JGFMonteCarloBenchSizeA	main
JGFMonteCarloBenchSizeA	∦	∦
main	∦	‖

2 JGFMonteCarloBench

Table 5: Methods Requires Clause Satisfiability

Method	Satisfiability
JGFMonteCarloBench	$\surd$
JGFrun	$\surd$
JGFinitialise	$\surd$
JGFvalidate	$\surd$
JGFtidyup	$\surd$
JGFsetsize	$\surd$

Table 6: State Transition Matrix

	alive
alive	$\uparrow$

Table 7: Methods Concurrency Matrix

	JGFMonteCarloBench	JGFrun	JGFinitialise	JGFvalidate	JGFtidyup	JGFsetsize
JGFMonteCarloBench	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$
JGFrun	$\nparallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$
JGFinitialise	$\nparallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$
JGFvalidate	$\nparallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$
JGFtidyup	$\nparallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$
JGFsetsize	$\nparallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$

3 JGFInstrumentor

Table 8: Methods Requires Clause Satisfiability

Method	Satisfiability
JGFInstrumentor	√
addTimer	√
addOpsToTimer	√
printTimer	√

Table 9: State Transition Matrix

	alive
alive	↑

Table 10: Methods Concurrency Matrix

	JGFInstrumentor	addTimer	addOpsToTimer	printTimer
JGFInstrumentor	∦	∦	∦	∦
addTimer	∦	∦	∦	∦
addOpsToTimer	∦	∦	∦	∦
printTimer	∦	∦	∦	∦

4 CallAppDemo

Table 11: Methods Requires Clause Satisfiability

Method	Satisfiability
CallAppDemo	√
initialise	√
runiters	√

Table 12: State Transition Matrix

	alive
alive	↑

Table 13: Methods Concurrency Matrix

	CallAppDemo	initialise	runiters
CallAppDemo	∦	∦	∦
initialise	∦	‖	‖
runiters	∦	‖	‖

5 AppDemo

Table 14: Methods Requires Clause Satisfiability

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

Table 15: State Transition Matrix

	alive
alive	↑

Table 16: Methods Concurrency Matrix

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

6 Universal

Table 17: Methods Requires Clause Satisfiability

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

Table 18: State Transition Matrix

	alive
alive	↑

Table 19: Methods Concurrency Matrix

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

7 JGFTimer

Table 20: Methods Requires Clause Satisfiability

Method	Satisfiability
JGFTimer	√
addops	√

Table 21: State Transition Matrix

	alive
alive	↑

Table 22: Methods Concurrency Matrix

	JGFTimer	addops
JGFTimer	∦	∦
addops	∦	∦

8 RatePath

Table 23: Methods Requires Clause Satisfiability

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

Table 24: State Transition Matrix

	alive
alive	↑

Table 25: Methods Concurrency Matrix

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

9 PathId

Table 26: Methods Requires Clause Satisfiability

Method	Satisfiability
PathId	$\surd$
setname	$\surd$
setendDate	$\surd$
setdTime	$\surd$
getname	$\surd$
setstartDate	$\surd$
getstartDate	$\surd$
getendDate	$\surd$
getdTime	$\surd$
copyInstanceVariables	$\surd$
dbgDumpFields	$\surd$

Table 27: State Transition Matrix

	alive
alive	$\uparrow$

Table 28: Methods Concurrency Matrix

	PathId	setname	setendDate	setdTime	getname	setstartDate	getstartDate	getendDate	getdTime	copyInstanceVariables	dbgDumpFields
PathId	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$
setname	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\parallel$
setendDate	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\parallel$
setdTime	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\parallel$
getname	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\parallel$
setstartDate	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\parallel$
getstartDate	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\parallel$
getendDate	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\parallel$
getdTime	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\parallel$
copyInstanceVariables	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\nparallel$	$\parallel$
dbgDumpFields	$\nparallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$	$\parallel$

10 PriceStock

Table 29: Methods Requires Clause Satisfiability

Method	Satisfiability
PriceStock	√
setInitAllTasks	√
run	√

Table 30: State Transition Matrix

	alive
alive	↑

Table 31: Methods Concurrency Matrix

	PriceStock	setInitAllTasks	run
PriceStock	⫽	⫽	⫽
setInitAllTasks	⫽	‖	‖
run	⫽	‖	‖

11 MonteCarloPath

Table 32: Methods Requires Clause Satisfiability

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

Table 33: State Transition Matrix

	alive
alive	↑

Table 34: Methods Concurrency Matrix

	MonteCarloPath	copyInstanceVariables	setpathValue	setfluctuations	getnTimeSteps	computePathValue	getpathValue	getreturnDefinition	setreturnDefinition	getexpectedReturnRate	setexpectedReturnRate	getvolatility	setvolatility	setnTimeSteps	getpathStartValue	setpathStartValue	getRatePath	computeFluctuationsGaussian
MonteCarloPath	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
copyInstanceVariables	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
setpathValue	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
setfluctuations	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
getnTimeSteps	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
computePathValue	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
getpathValue	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
getreturnDefinition	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖

setreturnDefinition	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
getexpectedReturnRate	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
setexpectedReturnRate	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
getvolatility	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
setvolatility	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
setnTimeSteps	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
getpathStartValue	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
setpathStartValue	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
getRatePath	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
computeFluctuationsGaussian	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖

12 ToInitAllTasks

Table 35: Methods Requires Clause Satisfiability

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

Table 36: State Transition Matrix

	alive
alive	↑

Table 37: Methods Concurrency Matrix

	ToInitAllTasks	getname	getnTimeSteps	setheader	setname	getstartDate	setstartDate	getendDate	setendDate	getdTime	setDTime	getreturnDefinition	setReturnDefinition	getexpectedReturnRate	setExpectedReturnRate	getvolatility	setVolatility	setnTimeSteps	getpathStartValue	setpathStartValue
ToInitAllTasks	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
getname	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
getnTimeSteps	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
setheader	∦	‖	‖	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	∦	‖	∦
setname	∦	‖	‖	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	∦	‖	∦
getstartDate	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
setstartDate	∦	‖	‖	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	∦	‖	∦
getendDate	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖

setendDate	∦	‖	‖	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	∦	‖	∦
getdTime	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
setDTime	∦	‖	‖	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	∦	‖	∦
getreturnDefinition	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
setReturnDefinition	∦	‖	‖	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	∦	‖	∦
getexpectedReturnRate	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
setExpectedReturnRate	∦	‖	‖	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	∦	‖	∦
getvolatility	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
setVolatility	∦	‖	‖	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	∦	‖	∦
setnTimeSteps	∦	‖	‖	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	∦	‖	∦
getpathStartValue	∦	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖	‖
setpathStartValue	∦	‖	‖	∦	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	‖	∦	∦	‖	∦

13 ReturnPath

Table 38: Methods Requires Clause Satisfiability

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

Table 39: State Transition Matrix

	alive
alive	↑

Table 40: Methods Concurrency Matrix

	ReturnPath	estimatePath	computeMean	computeVariance	computeExpectedReturnRate	getexpectedReturnRate	getvolatility	getvolatility2	setpathValue	getnPathValue	setnPathValue	getreturnDefinition	setreturnDefinition	setexpectedReturnRate	setvolatility	setvolatility2	getmean	setmean	getvariance	setvariance
ReturnPath	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
estimatePath	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
computeMean	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
computeVariance	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦
computeExpectedReturnRate	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦	∦

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

14 DemoException

Table 41: Methods Requires Clause Satisfiability

Method	Satisfiability
DemoException	√

Table 42: State Transition Matrix

	alive
alive	↑

15 ToResult

Table 43: Methods Requires Clause Satisfiability

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

Table 44: State Transition Matrix

	alive
alive	↑

Table 45: 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	∦	‖	‖	‖	‖	∦	∦	∦	‖	∦	‖	∦	‖	∦

16 ToTask

Table 46: Methods Requires Clause Satisfiability

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

Table 47: State Transition Matrix

	alive
alive	↑

Table 48: Methods Concurrency Matrix

	ToTask	getheader	setheader	getrandomSeed	setrandomSeed
ToTask	∦	∦	∦	∦	∦
getheader	∦	‖	‖	‖	‖
setheader	∦	‖	∦	‖	∦
getrandomSeed	∦	‖	‖	‖	‖
setrandomSeed	∦	‖	∦	‖	∦

17 Utilities

Table 49: Methods Requires Clause Satisfiability

Method	Satisfiability
Utilities	√
which	√
splitString	√
joinString	√

Table 50: State Transition Matrix

	alive
alive	↑

Table 51: Methods Concurrency Matrix

	Utilities	which	splitString	joinString
Utilities	∦	∦	∦	∦
which	∦	∥	∥	∥
splitString	∦	∥	∥	∥
joinString	∦	∥	∥	∥

18 Abbreviation

Table 52: 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

19 Annotated Version of Sequential Java Program generated by Sip4j

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

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

@Perm(requires="none(this) in alive",
ensures="unique(this) in alive")
 void main(String argv[]) {
}

}ENDOFCLASS

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

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

@Perm(requires="none(this) in alive",
ensures="unique(this) in alive")
public void JGFrun(int size) {
}
@Perm(requires="none(this) in alive",
ensures="unique(this) in alive")
public void JGFinitialise() {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public void JGFvalidate() {
}
@Perm(requires="unique(this) in alive",
ensures="unique(this) in alive")
public void JGFtidyup() {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void JGFsetsize(int size) {
}

}ENDOFCLASS

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

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

@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 void addTimer(String name, String opname, int size) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 void addOpsToTimer(String name, double count) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
 void printTimer(String name) {
}

}ENDOFCLASS

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

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

@Perm(requires="none(this) in alive",
ensures="unique(this) in alive")
public void initialise() {
}
```

```
@Perm(requires="none(this) in alive",
ensures="unique(this) in alive")
public void runiters() {
}

}ENDOFCLASS

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

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

@Perm(requires="none(this) in alive",
ensures="unique(this) in alive")
public void runSerial() {
}
@Perm(requires="none(this) in alive",
ensures="unique(this) in alive")
private void initTasks(int nRunsMC) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
private void processResults() {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setdataDirname(String dataDirname) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public String getdataFilename() {
 return null;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setdataFilename(String dataFilename) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public int getnTimeStepsMC() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setnTimeStepsMC(int nTimeStepsMC) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public int getnRunsMC() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setnRunsMC(int nRunsMC) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public Vector gettasks() {
 return null;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void settasks(Vector tasks) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public Vector getresults() {
 return null;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setresults(Vector results) {
}

}ENDOFCLASS

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

class Universal {
@Perm(ensures="unique(this) in alive")
```

```
157 Universal() {   }

159 @Perm(requires="full(this) in alive",
160 ensures="full(this) in alive")
161 public void setprompt(String prompt) {
162 }
163 @Perm(requires="full(this) in alive",
164 ensures="full(this) in alive")
165 public void setDEBUG(boolean DEBUG) {
166 }
167 @Perm(requires="pure(this) in alive",
168 ensures="pure(this) in alive")
169 public void dbgPrintln(String s) {
170 }
171 @Perm(requires="pure(this) in alive",
172 ensures="pure(this) in alive")
173 public boolean getDEBUG() {
174  return 0;
175 }
176 @Perm(requires="pure(this) in alive",
177 ensures="pure(this) in alive")
178 public boolean getUNIVERSALDEBUG() {
179  return 0;
180 }
181 @Perm(requires="full(this) in alive",
182 ensures="full(this) in alive")
183 public void setUNIVERSALDEBUG(boolean UNIVERSAL_DEBUG) {
184 }
185 @Perm(requires="pure(this) in alive",
186 ensures="pure(this) in alive")
187 public String getprompt() {
188  return null;
189 }
190 @Perm(requires="pure(this) in alive",
191 ensures="pure(this) in alive")
192 public void dbgPrint(String s) {
193 }
194 @Perm(requires="pure(this) in alive",
195 ensures="pure(this) in alive")
196 public void errPrintln(String s) {
197 }
198 @Perm(requires="pure(this) in alive",
199 ensures="pure(this) in alive")
200 public void errPrint(String s) {
201 }

203 }ENDOFCLASS

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

207 class JGFTimer {
208 @Perm(ensures="unique(this) in alive")
209 JGFTimer() {   }

211 @Perm(requires="full(this) in alive",
212 ensures="full(this) in alive")
213 public void addops(double count) {
214 }

216 }ENDOFCLASS

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

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

224 @Perm(requires="full(this) in alive",
225 ensures="full(this) in alive")
226 private void readRatesFile(String dirName, String filename) {
227 }
228 @Perm(requires="full(this) in alive",
229 ensures="full(this) in alive")
230 public void incpathValue(double scale) {
231 }
232 @Perm(requires="pure(this) in alive",
233 ensures="pure(this) in alive")
234 public double[] getpathValue() {
235  return null;
236 }
237 @Perm(requires="full(this) in alive",
```

```
ensures="full(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="full(this) in alive",
ensures="full(this) in alive")
public void setpathDate(int[] pathDate) {
}
@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="none(this) in alive",
ensures="unique(this) in alive")
public ReturnPath getReturnCompounded() {
 return null;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public ReturnPath getReturnNonCompounded() {
 return null;
}

}ENDOFCLASS

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

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

@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setname(String name) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setendDate(int endDate) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setdTime(double dTime) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public String getname() {
 return null;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setstartDate(int startDate) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public int getstartDate() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public int getendDate() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public double getdTime() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void copyInstanceVariables(PathId obj) {
}
```

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

}ENDOFCLASS

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

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

@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setInitAllTasks(Object obj) {
}
@Perm(requires="none(this) in alive",
ensures="unique(this) in alive")
public void run() {
}

}ENDOFCLASS

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

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

@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
private void copyInstanceVariables(ReturnPath obj) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setpathValue(double[] pathValue) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setfluctuations(double[] fluctuations) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public int getnTimeSteps() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void computePathValue(double startValue) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public double[] getpathValue() {
 return null;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public int getreturnDefinition() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setreturnDefinition(int returnDefinition) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public double getexpectedReturnRate() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setexpectedReturnRate(double expectedReturnRate) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public double getvolatility() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
```

```
public void setvolatility(double volatility) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setnTimeSteps(int nTimeSteps) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public double getpathStartValue() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setpathStartValue(double pathStartValue) {
}
@Perm(requires="none(this) in alive",
ensures="unique(this) in alive")
public RatePath getRatePath() {
 return null;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void computeFluctuationsGaussian(long randomSeed) {
}

}ENDOFCLASS

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

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

@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 getnTimeSteps() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setheader(String header) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setname(String name) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public int getstartDate() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setstartDate(int startDate) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public int getendDate() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setendDate(int endDate) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getdTime() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setDTime(double dTime) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public int getreturnDefinition() {
```

```
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setReturnDefinition(int returnDefinition) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getexpectedReturnRate() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setExpectedReturnRate(double expectedReturnRate) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getvolatility() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setVolatility(double volatility) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setnTimeSteps(int nTimeSteps) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getpathStartValue() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setpathStartValue(double pathStartValue) {
}

}ENDOFCLASS

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

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

@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void estimatePath() {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void computeMean() {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void computeVariance() {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void computeExpectedReturnRate() {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public double getexpectedReturnRate() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public double getvolatility() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public double getvolatility2() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setpathValue(double[] pathValue) {
}
```

```
562 @Perm(requires="full(this) in alive",
563 ensures="full(this) in alive")
564 public int getnPathValue() {
565  return 0;
566 }
567 @Perm(requires="full(this) in alive",
568 ensures="full(this) in alive")
569 public void setnPathValue(int nPathValue) {
570 }
571 @Perm(requires="full(this) in alive",
572 ensures="full(this) in alive")
573 public int getreturnDefinition() {
574  return 0;
575 }
576 @Perm(requires="full(this) in alive",
577 ensures="full(this) in alive")
578 public void setreturnDefinition(int returnDefinition) {
579 }
580 @Perm(requires="full(this) in alive",
581 ensures="full(this) in alive")
582 public void setexpectedReturnRate(double expectedReturnRate) {
583 }
584 @Perm(requires="full(this) in alive",
585 ensures="full(this) in alive")
586 public void setvolatility(double volatility) {
587 }
588 @Perm(requires="full(this) in alive",
589 ensures="full(this) in alive")
590 public void setvolatility2(double volatility2) {
591 }
592 @Perm(requires="full(this) in alive",
593 ensures="full(this) in alive")
594 public double getmean() {
595  return 0;
596 }
597 @Perm(requires="full(this) in alive",
598 ensures="full(this) in alive")
599 public void setmean(double mean) {
600 }
601 @Perm(requires="full(this) in alive",
602 ensures="full(this) in alive")
603 public double getvariance() {
604  return 0;
605 }
606 @Perm(requires="full(this) in alive",
607 ensures="full(this) in alive")
608 public void setvariance(double variance) {
609 }
610 @Perm(requires="pure(this) in alive",
611 ensures="pure(this) in alive")
612 public void dbgDumpFields() {
613 }

615 }ENDOFCLASS

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

619 class DemoException {
620 @Perm(ensures="unique(this) in alive")
621 DemoException() {   }


624 }ENDOFCLASS

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

628 class ToResult {
629 @Perm(ensures="unique(this) in alive")
630 ToResult() {   }

632 @Perm(requires="pure(this) in alive",
633 ensures="pure(this) in alive")
634 public double getexpectedReturnRate() {
635  return 0;
636 }
637 @Perm(requires="pure(this) in alive",
638 ensures="pure(this) in alive")
639 public double getvolatility() {
640  return 0;
641 }
642 @Perm(requires="pure(this) in alive",
```

```
ensures="pure(this) in alive")
public String toString() {
 return null;
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public String getheader() {
 return null;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setheader(String header) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setexpectedReturnRate(double expectedReturnRate) {
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setvolatility(double volatility) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getVolatility2() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setvolatility2(double volatility2) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double getfinalStockPrice() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setfinalStockPrice(double finalStockPrice) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public double[] getpathValue() {
 return null;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setpathValue(double[] pathValue) {
}

}ENDOFCLASS

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

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

@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public String getheader() {
 return null;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setheader(String header) {
}
@Perm(requires="pure(this) in alive",
ensures="pure(this) in alive")
public long getrandomSeed() {
 return 0;
}
@Perm(requires="full(this) in alive",
ensures="full(this) in alive")
public void setrandomSeed(long randomSeed) {
}

}ENDOFCLASS

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

class Utilities {
```

```
724 @Perm(ensures="unique(this) in alive")
725 Utilities() {   }

727 @Perm(requires="pure(this) in alive",
728 ensures="pure(this) in alive")
729  String which(String executable, String pathEnv) {
730  return null;
731 }
732 @Perm(requires="pure(this) in alive",
733 ensures="pure(this) in alive")
734  String[] splitString(String splitChar, String arg) {
735  return null;
736 }

738  String joinString(String joinChar, String stringArray[]) {
739  return null;
740 }

742 }ENDOFCLASS
```