

STEM (Spatio-temporal Epidemiological Modeler)

...

Katherine Sweeney, Ashanti Long, Clare Clever

What is STEM?

Why We Chose STEM

STEM Installation Guide



Contents [\[hide\]](#)

- 1 STEM Video on YouTube™
- 2 Getting Started
 - 2.1 Prerequisites
 - 2.2 Which File To Download?
 - 2.2.1 32-bit vs. 64-bit
 - 2.2.2 Release, Milestone, or Integration Builds
- 3 Downloading and Installing STEM
- 4 Security Settings in Mac OSx Mojave
- 5 Launching and Running STEM
- 6 Configure STEM Preference
- 7 Installing Additional STEM Features
- 8 Updating STEM
 - 8.1 Checking the Version of STEM Installed
 - 8.2 Updating to Latest Integration or Milestone Build
- 9 Setting up the STEM Development Environment in Eclipse

Method Selection

lnFactorial()

```
/**
 * compute the log(n!)
 * @param n
 * @return log(n!)
 */
static double lnFactorial(int n) {
    double retVal = 0.0;

    for (int i = 1; i <= n; i++) {
        retVal += Math.log((double)i);
    }

    return retVal;
}
```

calculateSlope()

```
/**
 * Calculate the slope
 * @return the slope
 */
public double calculateSlope(List<Double> xList, List<Double> yList) {
    int npts = xList.size();
    assert(npts == yList.size());

    //x, y for least squares fitting to line y = alpha*x - beta
    double[] x = new double[npts];
    double[] y = new double[npts];

    for(int j = 0; j < npts; j++){
        x[j] = xList.get(j).doubleValue();
        y[j] = yList.get(j).doubleValue();
    }

    // do the fit
    double sumx = 0;
    double sumy = 0;
    double sumxy = 0;
    double sumx2 = 0;
    double sumy2 = 0;
    double sum = 0;

    for(int j = 0; j < npts; j++){
        sumx += x[j];
        sumy += y[j];
        sumxy += x[j] * y[j];
        sumx2 += x[j] * x[j];
        sumy2 += y[j] * y[j];
        sum += 1;
    }

    double delta = -(sum * sumx2 - sumx * sumx);

    // get the slope
    slope = (sumx * sumy - sum * sumxy) / delta;

    return slope;
}
```

compareTo()

```
/**
 * for sorting coordinates by their x value
 * @see java.lang.Comparable#compareTo(java.lang.Object)
 */
public int compareTo(Object otherCoord) throws ClassCastException {
    if (!(otherCoord instanceof PhaseSpaceCoordinate)) throw new ClassCastException("A Person object expected.");
    double otherX = ((PhaseSpaceCoordinate)otherCoord).xValue;
    if (this.xValue < otherX) return -1;
    if (this.xValue > otherX) return 1;
    return 0;
}
```

formatLatLngValue()

```
/**
 * Converts double value into a fractional string with fracDigits
 * number of decimal places. Should be locale agnostic.
 * @param value value to convert
 * @param fracDigits number of digits after decimal point to hold
 * @return String containing new value
 */
static String formatLatLngValue(double value, double fracDigits) {

    double power = Math.pow(10, fracDigits);
    return String.valueOf(((long)(value*power))/power);
}
```


getDistance()

```
/**
 *
 * @param otherPoint
 * @return the distance between this point and other point in phase space
 */
public double getDistance(PhaseSpaceCoordinate otherPoint) {
    double dist2 = ((this.xValue - otherPoint.xValue)*(this.xValue - otherPoint.xValue))+
                    ((this.yValue - otherPoint.yValue)*(this.yValue - otherPoint.yValue))
    ;
    double dist = Math.sqrt(dist2);

    return dist;
}
```

Initial

Test Case Progression

```
1 {  
2   "id": 1,  
3   "requirement": "Method computes the log(n!)",  
4   "component": "../project/BinomialDistributionUtil.java",  
5   "method": "lnFactorial",  
6   "driver": "../testCasesExecutables/lnFactorial/testCase1.java",  
7   "input": "0",  
8   "output": "0"  
9 }
```



Final

```
{  
  "id": 21,  
  "requirement": "Method computes the log(n!)",  
  "component": "project/BinomialDistributionUtil.java",  
  "method": "lnFactorial",  
  "driver": "testCasesExecutables/lnFactorialTestCase.java",  
  "input": "0",  
  "output": "0"  
}
```

Bad Driver

```
public class testCase1 {  
    public static void main(String[] args) {  
  
        // Instantiate the Binomial Distribution Utility class  
        BinomialDistributionUtil BinomialDistributionUtil = new BinomialDistributionUtil();  
  
        // Test 1: Normal numerical value in range  
        int testOne = Integer.parseInt(args[0]);  
  
        // Run the actual method we are testing  
        double value = BinomialDistributionUtil.lnFactorial(testOne);  
  
        // Print test number  
        System.out.println("Test One:");  
        System.out.println("ln(" + testOne + "!): " + value);  
  
        // Print out test result  
        double testOracle = Double.parseDouble(args[1]);  
  
        // Test passed  
        if (value == testOracle) {  
            System.out.println("Oracle: " + testOracle);  
            System.out.println("Test one passed!");  
        }  
        // Test failed  
        else if (value != testOracle) {  
            System.out.println("Oracle: " + testOracle);  
            System.out.println("Test one failed...");  
        }  
        // Test ERROR  
        else {  
            System.out.println("Test one ERROR");  
        }  
    }  
}
```



Improved Driver

```
public class lnFactorialTestCase {  
    public static void main(String[] args) {  
        try {  
            // Instantiate the Binomial Distribution Utility class  
            BinomialDistributionUtil BinomialDistributionUtil = new BinomialDistributionUtil();  
  
            // Test 1: Normal numerical value in range  
            int testOne = Integer.parseInt(args[0]);  
  
            // Run the actual method we are testing  
            double value = BinomialDistributionUtil.lnFactorial(testOne);  
  
            // Print test number  
            System.out.println("Test:");  
            System.out.println("Calculate ln(" + testOne + "!)" );  
  
            System.out.println("Result: " + value);  
  
            // Print out test result  
            double testOracle = Double.parseDouble(args[1]);  
  
            // Test passed  
            if (value == testOracle) {  
                System.out.println("Oracle: " + testOracle);  
                System.out.println("Pass");  
            }  
            // Test failed  
            else {  
                System.out.println("Oracle: " + testOracle);  
                System.out.println("Fail");  
            }  
        } catch (Exception e) {  
            System.out.println("ERROR");  
        }  
    }  
}
```

Script Mistakes

Incorrect Script Attempt

```
def main():

#####

    # Get the method names
    methodNames = []

    os.system("ls testCases > temp.txt")

    tempFile = open("temp.txt", "r")

    for line in tempFile:
        methodNames.append(line.strip())

    os.system("rm temp.txt")

    for method in methodNames:
        testMethod(method)

#####

    # Construct the HTML file and open it in the browser
    constructReport(methodNames)

    # Open the html file in the browser
    new = 2 # open in a new tab, if possible
    print("Opening the html file")
    webbrowser.open("reports/testReport.html", new=new)

#####
#####
#####

main()
```

Final Script Implementation (Main)

```
testCaseNames = []

# Call the ls command on the testCases directory
os.system("ls testCases > temp.txt")

# Open the temp file
tempFile = open("temp.txt", "r")
```

```
# Run each test case
for testCase in testCaseNames:
    jsonFile = readJsonAtLocation("testCases/" + testCase)
    print("Running test " + str(jsonFile["id"]))
    moveProjectFileandCompile(jsonFile)
    runTestCase(jsonFile)
    cleanUpTestCaseExe(jsonFile)
    writeTestResults("temp/" + jsonFile["method"] + "TestCase" + str(jsonFile["id"]) + "results.txt", jsonFile)
    os.system("rm ./temp/" + jsonFile["method"] + "TestCase" + str(jsonFile["id"]) + "results.txt")
```

Test Result Report Progression

`lnFactorial()`

Test One:

`ln(0!):` 0.0

Oracle: 0.0

Test one *passed!*

Test Two:

`ln(-5!):` 0.0

Oracle: 0.0

Test two *passed!*

Test Three:

`ln(2000000000!):` 4.083282604664613E10

Oracle: 4.083282604664613E10

Test three *passed!*

Test Four:

`ln(1!):` 0.0

Oracle: 0.0

Test four *passed!*

Test Five:

`ln(3!):` 1.791759469228055

Oracle: 1.791759469228055

Test five *passed!*

Test Results

`calculateSlope()`

Method calculates the slope of a line

ID	Calculation	Input	Oracle	Output	Result
1	Calculate slope between (5.000000, 3.000000) and (4.000000, 7.000000)	5 3 4 7	-4.0	-4.0	Pass
2	Calculate slope between (-3.000000, 3.000000) and (3.000000, -3.000000)	-3 3 3 -3	-1.0	-1.0	Pass
3	Calculate slope between (136.000000, -38.000000) and (17.000000, -32.000000)	136 -38 17 -32	-0.05042016806722689	-0.05042016806722689	Pass
4	Calculate slope between (35943.000000, 4037823.000000) and (132894.000000, 650983.000000)	35943 4037823 132894 650983	-34.93352311992656	-34.93352311992656	Pass
5	Calculate slope between (64.238763, 64.590870) and (64.240898, 64.590654)	64.238763 64.590870 64.240898 64.590654	-0.10123448901101095	-0.10123448901101095	Pass

`compareTo()`

Method sorts coordinates by their x value

ID	Calculation	Input	Oracle	Output	Result
1	Compare points (5.000000, -63.000000) and (72.000000, 38.000000)	5 -63 72 38	-1.0	-1.0	Pass
2	Compare points (0.000000, 0.000000) and (1.000000, 0.000000)	0 0 1 0	-1.0	-1.0	Pass
3	Compare points (136.000000, -38.000000) and (17.000000, -32.000000)	136 -38 17 -32	1.0	1.0	Pass
4	Compare points (1.000000, 0.000000) and (0.000000, 0.000000)	1 0 0 0	1.0	1.0	Pass
5	Compare points (92.000000, 92.000000) and (92.000000, 92.000000)	92 92 92 92	0.0	0.0	Pass

`formatLatLngValue()`

Method converts double value into a fractional string with default number of decimal places

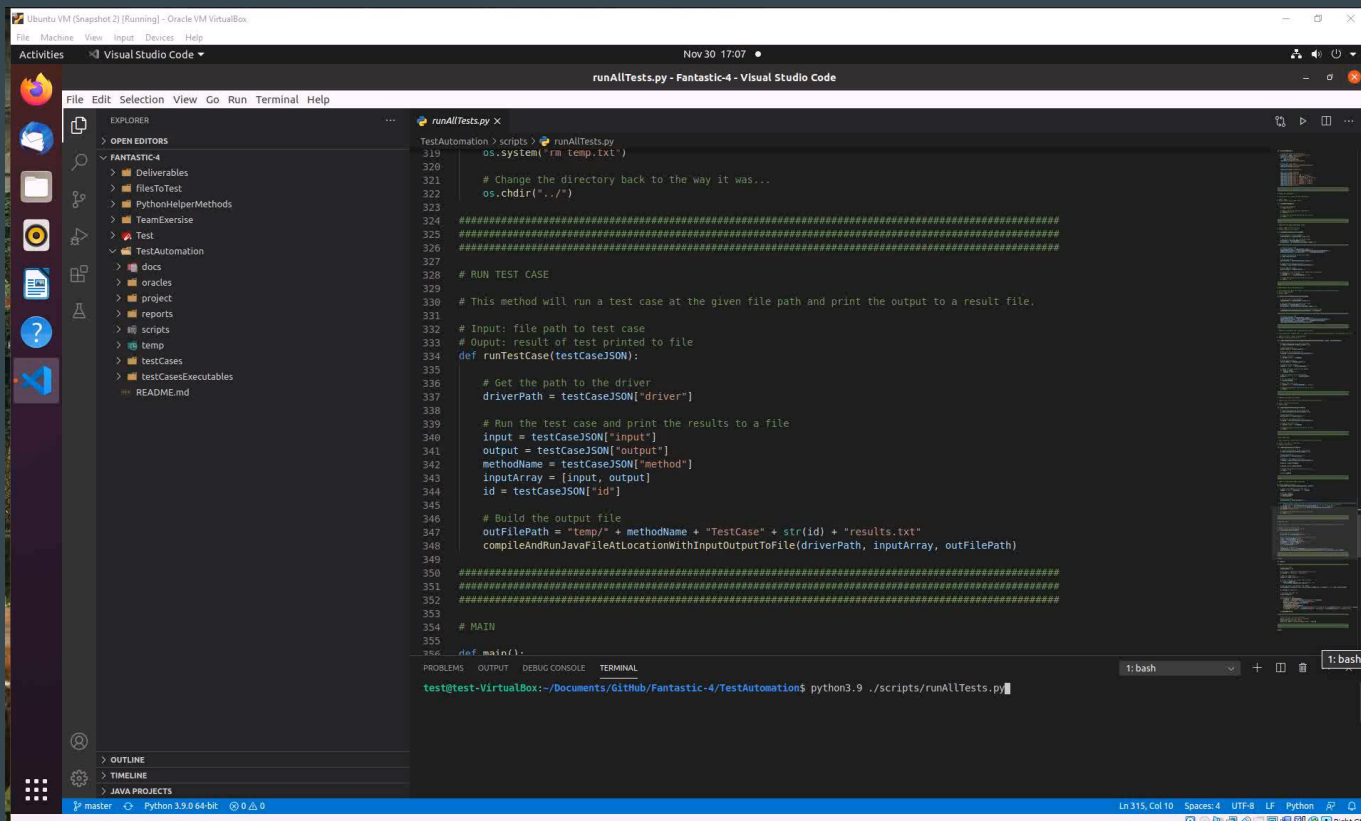
ID	Calculation	Input	Oracle	Output	Result
1	Format 1253404.47262174	1253404.47262174 3	1253404.472	1253404.472	Pass
2	Format 0.128427	0.128427 4	0.1284	0.1284	Pass
3	Format -126.1253799	-126.1253799 0	-126.0	-126.0	Pass
4	Format -5.47271430234885E7	-54727143.0234885 2	-5.472714302E7	-5.472714302E7	Pass
5	Format 9.120370981409123E12	9120370981409.12309714287894513 8	9.223372036854776E10	9.223372036854776E10	Pass

Final Test Results Report

Test Results

ID	Method	Requirement	Input	Oracle	Output	Result
1	calculateSlope	Method calculates the slope of a line	5 3 4 7	-4.0	-4.0	Pass
2	calculateSlope	Method calculates the slope of a line	-3 3 3 -3	-1.0	-1.0	Pass
3	calculateSlope	Method calculates the slope of a line	136 -38 17 -32	-0.05042016806722689	-0.05042016806722689	Pass
4	calculateSlope	Method calculates the slope of a line	35943 4037823 132894 650983	-34.93352311992656	-34.93352311992656	Pass
5	calculateSlope	Method calculates the slope of a line	64.2387634 64.5908703477 64.24089753 64.5906543	-0.10123448901101095	-0.10123448901101095	Pass
6	compareTo	Method sorts coordinates by their x value	5 -63 72 38	-1.0	-1.0	Pass
7	compareTo	Method sorts coordinates by their x value	0 0 1 0	-1.0	-1.0	Pass
8	compareTo	Method sorts coordinates by their x value	136 -38 17 -32	1.0	1.0	Pass
9	compareTo	Method sorts coordinates by their x value	1 0 0 0	1.0	1.0	Pass
10	compareTo	Method sorts coordinates by their x value	92 92 92 92	0.0	0.0	Pass
11	formatLatLngValue	Method converts double value into a fractional string with default number of decimal places	1253404.47262174 3	1253404.472	1253404.472	Pass
12	formatLatLngValue	Method converts double value into a fractional string with default number of decimal places	0.128427 4	0.1284	0.1284	Pass
13	formatLatLngValue	Method converts double value into a fractional string with default number of decimal places	-126.1253799 0	-126.0	-126.0	Pass
14	formatLatLngValue	Method converts double value into a fractional string with default number of decimal places	-54727143.0234885 2	-5.472714302E7	-5.472714302E7	Pass
15	formatLatLngValue	Method converts double value into a fractional string with default number of decimal places	9120370981409.12309714287894513 8	9.223372036854776E10	9.223372036854776E10	Pass
16	getDistance	Method returns the distance between this point and other point in phase space	5 5 5 5	0.0	0.0	Pass
17	getDistance	Method returns the distance between this point and other point in phase space	7 3 4 9	6.708203932499369	6.708203932499369	Pass
18	getDistance	Method returns the distance between this point and other point in phase space	-52 -3 -23 -45	51.03920062069938	51.03920062069938	Pass
19	getDistance	Method returns the distance between this point and other point in phase space	0 0 0 0	0.0	0.0	Pass
20	getDistance	Method returns the distance between this point and other point in phase space	120983 12349078 487094 430803248	4.18454330157609E8	4.18454330157609E8	Pass
21	lnFactorial	Method computes the log(n!)	0	0.0	0.0	Pass
22	lnFactorial	Method computes the log(n!)	-5	0.0	0.0	Pass
23	lnFactorial	Method computes the log(n!)	2000000	2.701732365031526E7	2.701732365031526E7	Pass
24	lnFactorial	Method computes the log(n!)	1	0.0	0.0	Pass
25	lnFactorial	Method computes the log(n!)	3	1.791759469228055	1.791759469228055	Pass

Script Demo



Ubuntu VM (Snapshot 2) [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Activities Visual Studio Code Nov 30 17:07

runAllTests.py - Fantastic-4 - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

- OPEN EDITORS
- FANTASTIC-4
 - Deliverables
 - FilesToTest
 - PythonHelperMethods
 - TeamExercise
 - Test
 - TestAutomation
 - docs
 - oracles
 - project
 - reports
 - scripts
 - temp
 - testCases
 - testCasesExecutables
 - README.md

runAllTests.py X

```
TestAutomation > scripts > runAllTests.py
319 os.system("rm temp.txt")
320
321 # Change the directory back to the way it was...
322 os.chdir("../")
323
324 #####
325 #####
326 #####
327
328 # RUN TEST CASE
329
330 # This method will run a test case at the given file path and print the output to a result file.
331
332 # Input: file path to test case
333 # Output: result of test printed to file
334 def runTestCase(testCaseJSON):
335
336     # Get the path to the driver
337     driverPath = testCaseJSON["driver"]
338
339     # Run the test case and print the results to a file
340     input = testCaseJSON["input"]
341     output = testCaseJSON["output"]
342     methodName = testCaseJSON["method"]
343     inputArray = [input, output]
344     id = testCaseJSON["id"]
345
346     # Build the output file
347     outFilePath = "temp/" + methodName + "TestCase" + str(id) + "results.txt"
348     compileAndRunJavaFileAtLocationWithInputOutputToFile(driverPath, inputArray, outFilePath)
349
350     #####
351     #####
352     #####
353
354 # MAIN
355
356 def main():
357
358     #####
359     #####
360     #####
361
362     #####
363     #####
364     #####
365
366     #####
367     #####
368     #####
369
370     #####
371     #####
372     #####
373
374     #####
375     #####
376     #####
377
378     #####
379     #####
380     #####
381
382     #####
383     #####
384     #####
385
386     #####
387     #####
388     #####
389
390     #####
391     #####
392     #####
393
394     #####
395     #####
396     #####
397
398     #####
399     #####
400     #####
401
402     #####
403     #####
404     #####
405
406     #####
407     #####
408     #####
409
410     #####
411     #####
412     #####
413
414     #####
415     #####
416     #####
417
418     #####
419     #####
420     #####
421
422     #####
423     #####
424     #####
425
426     #####
427     #####
428     #####
429
430     #####
431     #####
432     #####
433
434     #####
435     #####
436     #####
437
438     #####
439     #####
440     #####
441
442     #####
443     #####
444     #####
445
446     #####
447     #####
448     #####
449
450     #####
451     #####
452     #####
453
454     #####
455     #####
456     #####
457
458     #####
459     #####
460     #####
461
462     #####
463     #####
464     #####
465
466     #####
467     #####
468     #####
469
470     #####
471     #####
472     #####
473
474     #####
475     #####
476     #####
477
478     #####
479     #####
480     #####
481
482     #####
483     #####
484     #####
485
486     #####
487     #####
488     #####
489
490     #####
491     #####
492     #####
493
494     #####
495     #####
496     #####
497
498     #####
499     #####
500     #####
501
502     #####
503     #####
504     #####
505
506     #####
507     #####
508     #####
509
510     #####
511     #####
512     #####
513
514     #####
515     #####
516     #####
517
518     #####
519     #####
520     #####
521
522     #####
523     #####
524     #####
525
526     #####
527     #####
528     #####
529
530     #####
531     #####
532     #####
533
534     #####
535     #####
536     #####
537
538     #####
539     #####
540     #####
541
542     #####
543     #####
544     #####
545
546     #####
547     #####
548     #####
549
550     #####
551     #####
552     #####
553
554     #####
555     #####
556     #####
557
558     #####
559     #####
560     #####
561
562     #####
563     #####
564     #####
565
566     #####
567     #####
568     #####
569
570     #####
571     #####
572     #####
573
574     #####
575     #####
576     #####
577
578     #####
579     #####
580     #####
581
582     #####
583     #####
584     #####
585
586     #####
587     #####
588     #####
589
590     #####
591     #####
592     #####
593
594     #####
595     #####
596     #####
597
598     #####
599     #####
600     #####
601
602     #####
603     #####
604     #####
605
606     #####
607     #####
608     #####
609
610     #####
611     #####
612     #####
613
614     #####
615     #####
616     #####
617
618     #####
619     #####
620     #####
621
622     #####
623     #####
624     #####
625
626     #####
627     #####
628     #####
629
630     #####
631     #####
632     #####
633
634     #####
635     #####
636     #####
637
638     #####
639     #####
640     #####
641
642     #####
643     #####
644     #####
645
646     #####
647     #####
648     #####
649
650     #####
651     #####
652     #####
653
654     #####
655     #####
656     #####
657
658     #####
659     #####
660     #####
661
662     #####
663     #####
664     #####
665
666     #####
667     #####
668     #####
669
670     #####
671     #####
672     #####
673
674     #####
675     #####
676     #####
677
678     #####
679     #####
680     #####
681
682     #####
683     #####
684     #####
685
686     #####
687     #####
688     #####
689
690     #####
691     #####
692     #####
693
694     #####
695     #####
696     #####
697
698     #####
699     #####
700     #####
701
702     #####
703     #####
704     #####
705
706     #####
707     #####
708     #####
709
710     #####
711     #####
712     #####
713
714     #####
715     #####
716     #####
717
718     #####
719     #####
720     #####
721
722     #####
723     #####
724     #####
725
726     #####
727     #####
728     #####
729
730     #####
731     #####
732     #####
733
734     #####
735     #####
736     #####
737
738     #####
739     #####
740     #####
741
742     #####
743     #####
744     #####
745
746     #####
747     #####
748     #####
749
750     #####
751     #####
752     #####
753
754     #####
755     #####
756     #####
757
758     #####
759     #####
760     #####
761
762     #####
763     #####
764     #####
765
766     #####
767     #####
768     #####
769
770     #####
771     #####
772     #####
773
774     #####
775     #####
776     #####
777
778     #####
779     #####
780     #####
781
782     #####
783     #####
784     #####
785
786     #####
787     #####
788     #####
789
790     #####
791     #####
792     #####
793
794     #####
795     #####
796     #####
797
798     #####
799     #####
800     #####
801
802     #####
803     #####
804     #####
805
806     #####
807     #####
808     #####
809
810     #####
811     #####
812     #####
813
814     #####
815     #####
816     #####
817
818     #####
819     #####
820     #####
821
822     #####
823     #####
824     #####
825
826     #####
827     #####
828     #####
829
830     #####
831     #####
832     #####
833
834     #####
835     #####
836     #####
837
838     #####
839     #####
840     #####
841
842     #####
843     #####
844     #####
845
846     #####
847     #####
848     #####
849
850     #####
851     #####
852     #####
853
854     #####
855     #####
856     #####
857
858     #####
859     #####
860     #####
861
862     #####
863     #####
864     #####
865
866     #####
867     #####
868     #####
869
870     #####
871     #####
872     #####
873
874     #####
875     #####
876     #####
877
878     #####
879     #####
880     #####
881
882     #####
883     #####
884     #####
885
886     #####
887     #####
888     #####
889
890     #####
891     #####
892     #####
893
894     #####
895     #####
896     #####
897
898     #####
899     #####
900     #####
901
902     #####
903     #####
904     #####
905
906     #####
907     #####
908     #####
909
910     #####
911     #####
912     #####
913
914     #####
915     #####
916     #####
917
918     #####
919     #####
920     #####
921
922     #####
923     #####
924     #####
925
926     #####
927     #####
928     #####
929
930     #####
931     #####
932     #####
933
934     #####
935     #####
936     #####
937
938     #####
939     #####
940     #####
941
942     #####
943     #####
944     #####
945
946     #####
947     #####
948     #####
949
950     #####
951     #####
952     #####
953
954     #####
955     #####
956     #####
957
958     #####
959     #####
960     #####
961
962     #####
963     #####
964     #####
965
966     #####
967     #####
968     #####
969
970     #####
971     #####
972     #####
973
974     #####
975     #####
976     #####
977
978     #####
979     #####
980     #####
981
982     #####
983     #####
984     #####
985
986     #####
987     #####
988     #####
989
990     #####
991     #####
992     #####
993
994     #####
995     #####
996     #####
997
998     #####
999     #####
1000     #####
1001
1002     #####
1003     #####
1004     #####
1005
1006     #####
1007     #####
1008     #####
1009
1010     #####
1011     #####
1012     #####
1013
1014     #####
1015     #####
1016     #####
1017
1018     #####
1019     #####
1020     #####
1021
1022     #####
1023     #####
1024     #####
1025
1026     #####
1027     #####
1028     #####
1029
1030     #####
1031     #####
1032     #####
1033
1034     #####
1035     #####
1036     #####
1037
1038     #####
1039     #####
1040     #####
1041
1042     #####
1043     #####
1044     #####
1045
1046     #####
1047     #####
1048     #####
1049
1050     #####
1051     #####
1052     #####
1053
1054     #####
1055     #####
1056     #####
1057
1058     #####
1059     #####
1060     #####
1061
1062     #####
1063     #####
1064     #####
1065
1066     #####
1067     #####
1068     #####
1069
1070     #####
1071     #####
1072     #####
1073
1074     #####
1075     #####
1076     #####
1077
1078     #####
1079     #####
1080     #####
1081
1082     #####
1083     #####
1084     #####
1085
1086     #####
1087     #####
1088     #####
1089
1090     #####
1091     #####
1092     #####
1093
1094     #####
1095     #####
1096     #####
1097
1098     #####
1099     #####
1100     #####
1101
1102     #####
1103     #####
1104     #####
1105
1106     #####
1107     #####
1108     #####
1109
1110     #####
1111     #####
1112     #####
1113
1114     #####
1115     #####
1116     #####
1117
1118     #####
1119     #####
1120     #####
1121
1122     #####
1123     #####
1124     #####
1125
1126     #####
1127     #####
1128     #####
1129
1130     #####
1131     #####
1132     #####
1133
1134     #####
1135     #####
1136     #####
1137
1138     #####
1139     #####
1140     #####
1141
1142     #####
1143     #####
1144     #####
1145
1146     #####
1147     #####
1148     #####
1149
1150     #####
1151     #####
1152     #####
1153
1154     #####
1155     #####
1156     #####
1157
1158     #####
1159     #####
1160     #####
1161
1162     #####
1163     #####
1164     #####
1165
1166     #####
1167     #####
1168     #####
1169
1170     #####
1171     #####
1172     #####
1173
1174     #####
1175     #####
1176     #####
1177
1178     #####
1179     #####
1180     #####
1181
1182     #####
1183     #####
1184     #####
1185
1186     #####
1187     #####
1188     #####
1189
1190     #####
1191     #####
1192     #####
1193
1194     #####
1195     #####
1196     #####
1197
1198     #####
1199     #####
1200     #####
1201
1202     #####
1203     #####
1204     #####
1205
1206     #####
1207     #####
1208     #####
1209
1210     #####
1211     #####
1212     #####
1213
1214     #####
1215     #####
1216     #####
1217
1218     #####
1219     #####
1220     #####
1221
1222     #####
1223     #####
1224     #####
1225
1226     #####
1227     #####
1228     #####
1229
1230     #####
1231     #####
1232     #####
1233
1234     #####
1235     #####
1236     #####
1237
1238     #####
1239     #####
1240     #####
1241
1242     #####
1243     #####
1244     #####
1245
1246     #####
1247     #####
1248     #####
1249
1250     #####
1251     #####
1252     #####
1253
1254     #####
1255     #####
1256     #####
1257
1258     #####
1259     #####
1260     #####
1261
1262     #####
1263     #####
1264     #####
1265
1266     #####
1267     #####
1268     #####
1269
1270     #####
1271     #####
1272     #####
1273
1274     #####
1275     #####
1276     #####
1277
1278     #####
1279     #####
1280     #####
1281
1282     #####
1283     #####
1284     #####
1285
1286     #####
1287     #####
1288     #####
1289
1290     #####
1291     #####
1292     #####
1293
1294     #####
1295     #####
1296     #####
1297
1298     #####
1299     #####
1300     #####
1301
1302     #####
1303     #####
1304     #####
1305
1306     #####
1307     #####
1308     #####
1309
1310     #####
1311     #####
1312     #####
1313
1314     #####
1315     #####
1316     #####
1317
1318     #####
1319     #####
1320     #####
1321
1322     #####
1323     #####
1324     #####
1325
1326     #####
1327     #####
1328     #####
1329
1330     #####
1331     #####
1332     #####
1333
1334     #####
1335     #####
1336     #####
1337
1338     #####
1339     #####
1340     #####
1341
1342     #####
1343     #####
1344     #####
1345
1346     #####
1347     #####
1348     #####
1349
1350     #####
1351     #####
1352     #####
1353
1354     #####
1355     #####
1356     #####
1357
1358     #####
1359     #####
1360     #####
1361
1362     #####
1363     #####
1364     #####
1365
1366     #####
1367     #####
1368     #####
1369
1370     #####
1371     #####
1372     #####
1373
1374     #####
1375     #####
1376     #####
1377
1378     #####
1379     #####
1380     #####
1381
1382     #####
1383     #####
1384     #####
1385
1386     #####
1387     #####
1388     #####
1389
1390     #####
1391     #####
1392     #####
1393
1394     #####
1395     #####
1396     #####
1397
1398     #####
1399     #####
1400     #####
1401
1402     #####
1403     #####
1404     #####
1405
1406     #####
1407     #####
1408     #####
1409
1410     #####
1411     #####
1412     #####
1413
1414     #####
1415     #####
1416     #####
1417
1418     #####
1419     #####
1420     #####
1421
1422     #####
1423     #####
1424     #####
1425
1426     #####
1427     #####
1428     #####
1429
1430     #####
1431     #####
1432     #####
1433
1434     #####
1435     #####
1436     #####
1437
1438     #####
1439     #####
1440     #####
1441
1442     #####
1443     #####
1444     #####
1445
1446     #####
1447     #####
1448     #####
1449
1450     #####
1451     #####
1452     #####
1453
1454     #####
1455     #####
1456     #####
1457
1458     #####
1459     #####
1460     #####
1461
1462     #####
1463     #####
1464     #####
1465
1466     #####
1467     #####
1468     #####
1469
1470     #####
1471     #####
1472     #####
1473
1474     #####
1475     #####
1476     #####
1477
1478     #####
1479     #####
1480     #####
1481
1482     #####
1483     #####
1484     #####
1485
1486     #####
1487     #####
1488     #####
1489
1490     #####
1491     #####
1492     #####
1493
1494     #####
1495     #####
1496     #####
1497
1498     #####
1499     #####
1500     #####
1501
1502     #####
1503     #####
1504     #####
1505
1506     #####
1507     #####
1508     #####
1509
1510     #####
1511     #####
1512     #####
1513
1514     #####
1515     #####
1516     #####
1517
1518     #####
1519     #####
1520     #####
1521
1522     #####
1523     #####
1524     #####
1525
1526     #####
1527     #####
1528     #####
1529
1530     #####
1531     #####
1532     #####
1533
1534     #####
1535     #####
1536     #####
1537
1538     #####
1539     #####
1540     #####
1541
1542     #####
1543     #####
1544     #####
1545
1546     #####
1547     #####
1548     #####
1549
1550     #####
1551     #####
1552     #####
1553
1554     #####
1555     #####
1556     #####
1557
1558     #####
1559     #####
1560     #####
1561
1562     #####
1563     #####
1564     #####
1565
1566     #####
1567     #####
1568     #####
1569
1570     #####
1571     #####
1572     #####
1573
1574     #####
1575     #####
1576     #####
1577
1578     #####
1579     #####
1580     #####
1581
1582     #####
1583     #####
1584     #####
1585
1586     #####
1587     #####
1588     #####
1589
1590     #####
1591     #####
1592     #####
1593
1594     #####
1595     #####
1596     #####
1597
1598     #####
1599     #####
1600     #####
1601
1602     #####
1603     #####
1604     #####
1605
1606     #####
1607     #####
1608     #####
1609
1610     #####
1611     #####
1612     #####
1613
1614     #####
1615     #####
1616     #####
1617
1618     #####
1619     #####
1620     #####
1621
1622     #####
1623     #####
1624     #####
1625
1626     #####
1627     #####
1628     #####
1629
1630     #####
1631     #####
1632     #####
1633
1634     #####
1635     #####
1636     #####
1637
1638     #####
1639     #####
1640     #####
1641
1642     #####
1643     #####
1644     #####
1645
1646     #####
1647     #####
1648     #####
1649
1650     #####
1651     #####
1652     #####
1653
1654     #####
1655     #####
1656     #####
1657
1658     #####
1659     #####
1660     #####
1661
1662     #####
1663     #####
1664     #####
1665
1666     #####
1667     #####
1668     #####
1669
1670     #####
1671     #####
1672     #####
1673
1674     #####
1675     #####
1676     #####
1677
1678     #####
1679     #####
1680     #####
1681
1682     #####
1683     #####
1684     #####
1685
1686     #####
1687     #####
1688     #####
1689
1690     #####
1691     #####
1692     #####
1693
1694     #####
1695     #####
1696     #####
1697
1698     #####
1699     #####
1700     #####
1701
1702     #####
1703     #####
1704     #####
1705
1706     #####
1707     #####
1708     #####
1709
1710     #####
1711     #####
1712     #####
1713
1714     #####
1715     #####
1716     #####
1717
1718     #####
1719     #####
1720     #####
1721
1722     #####
1723     #####
1724     #####
1725
1726     #####
1727     #####
1728     #####
1729
1730     #####
1731     #####
1732     #####
1733
1734     #####
1735     #####
1736     #####
1737
1738     #####
1739     #####
1740     #####
1741
1742     #####
1743     #####
1744     #####
1745
1746     #####
1747     #####
1748     #####
1749
1750     #####
1751     #####
1752     #####
1753
1754     #####
1755     #####
1756     #####
1757
1758     #####
1759     #####
1760     #####
1761
1762     #####
1763     #####
1764     #####
1765
1766     #####
1767     #####
1768     #####
1769
1770     #####
1771     #####
1772     #####
1773
1774     #####
1775     #####
1776     #####
1777
1778     #####
1779     #####
1780     #####
1781
1782     #####
1783     #####
1784     #####
1785
1786     #####
1787     #####
1788     #####
1789
1790     #####
1791     #####
1792     #####
1793
1794     #####
1795     #####
1796     #####
1797
1798     #####
1799     #####
1800     #####
1801
1802     #####
1803     #####
1804     #####
1805
1806     #####
1807     #####
1808     #####
1809
1810     #####
1811     #####
1812     #####
1813
1814     #####
1815     #####
1816     #####
1817
1818     #####
1819     #####
1820     #####
1821
1822     #####
1823     #####
1824     #####
1825
1826     #####
1827     #####
1828     #####
1829
1830     #####
1831     #####
1832     #####
1833
1834     #####
1835     #####
1836     #####
1837
1838     #####
1839     #####
1840     #####
1841
1842     #####
1843     #####
1844     #####
1845
1846     #####
1847     #####
1848     #####
1849
1850     #####
1851     #####
1852     #####
1853
1854     #####
1855     #####
1856     #####
1857
1858     #####
1859     #####
1860     #####
1861
1862     #####
1863     #####
1864     #####
1865
1866     #####
1867     #####
1868     #####
1869
1870     #####
1871     #####
1872     #####
1873
1874     #####
1875     #####
1876     #####
1877
1878     #####
1879     #####
1880     #####
1881
1882     #####
1883     #####
1884     #####
1885
1886     #####
1887     #####
1888     #####
1889
1890     #####
1891     #####
1892     #####
1893
1894     #####
1895     #####
1896     #####
1897
1898     #####
1899     #####
1900     #####
1901
1902     #####
1903     #####
1904     #####
1905
1906     #####
1907     #####
1908     #####
1909
1910     #####
1911     #####
1912     #####
1913
1914     #####
1915     #####
1916     #####
1917
1918     #####
1919     #####
1920     #####
1921
1922     #####
1923     #####
1924     #####
1925
1926     #####
1927     #####
1928     #####
1929
1930     #####
1931     #####
1932     #####
1933
1934     #####
1935     #####
1936     #####
1937
1938     #####
1939     #####
1940     #####
1941
1942     #####
1943     #####
1944     #####
1945
1946     #####
1947     #####
1948     #####
1949
1950     #####
1951     #####
1952     #####
1953
1954     #####
1955     #####
1956     #####
1957
1958     #####
1959     #####
1960     #####
1961
1962     #####
1963     #####
1964     #####
1965
1966     #####
1967     #####
1968     #####
1969
1970     #####
1971     #####
1972     #####
1973
1974     #####
1975     #####
1976     #####
1977
1978     #####
1979     #####
1980     #####
1981
1982     #####
1983     #####
1984     #####
1985
1986     #####
1987     #####
1988     #####
1989
1990     #####
1991     #####
1992     #####
1993
1994     #####
1995     #####
1996     #####
1997
1998     #####
1999     #####
2000     #####
2001
2002     #####
2003     #####
2004     #####
2005
2006     #####
2007     #####
2008     #####
2009
2010     #####
2011     #####
2012     #####
2013
2014     #####
2015     #####
2016     #####
2017
2018     #####
2019     #####
2020     #####
2021
2022     #####
2023     #####
2024     #####
2025
2026     #####
2027     #####
2028     #####
2029
2030     #####
2031     #####
2032     #####
2033
2034     #####
2035     #####
2036     #####
2037
2038     #####
2039     #####
2040     #####
2041
2042     #####
2043     #####
2044     #####
2045
2046     #####
2047     #####
2048     #####
2049
2050     #####
2051     #####
2052     #####
2053
2054     #####
2055     #####
2056     #####
2057
2058     #####
2059     #####
2060     #####
2061
2062     #####
2063     #####
2064     #####
2065
2066     #####
2067     #####
2068     #####
2069
2070     #####
2071     #####
2072     #####
2073
2074     #####
2075     #####
2076     #####
2077
2078     #####
2079     #####
2080     #####
2081
2082     #####
2083     #####
2084     #####
2085
2086     #####
2087     #####
2088     #####
2089
2090     #####
2091     #####
2092     #####
2093
2094     #####
2095     #####
2096     #####
2097
2098     #####
2099     #####
2100     #####
2101
2102     #####
2103     #####
2104     #####
2105
2106     #####
2107     #####
2108     #####
2109
2110     #####
2111     #####
2112     #####
2113
2114     #####
2115     #####
2116     #####
2117
2118     #####
2119     #####
2120     #####
2121
2122     #####
2123     #####
2124     #####
2125
2126     #####
2127     #####
2128     #####
2129
2130     #####
2131     #####
2132     #####
2133
2134     #####
2135     #####
2136     #####
2137
2138     #####
2139     #####
2140     #####
2141
2142     #####
2143     #####
2144     #####
2145
2146     #####
2147     #####
2148     #####
2149
2150     #####
2151     #####
2152     #####
2153
2154     #####
2155     #####
2156     #####
2157
2158     #####
2159     #####
2160     #####
2161
2162     #####
2163     #####
2164     #####
2165
2166     #####
2167     #####
2168     #####
2169
2170     #####
2171     #####
2172     #####
2173
2174     #####
2175     #####
2176     #####
2177
2178     #####
2179     #####
2180     #####
2181
2182     #####
2183     #####
2184     #####
2185
2186     #####
2187     #####
2188     #####
2189
2190     #####
2191     #####
2192     #####
2193
2194     #####
2195     #####
2196     #####
2197
2198     #####
2199     #####
2200     #####
2201
2202     #####
2203     #####
2204     #####
2205
2206     #####
2207     #####
2208     #####
2209
2210     #####
2211     #####
2212     #####
2213
2214     #####
2215     #####
2216     #####
2217
2218     #####
2219     #####
2220     #####
2221
2222     #####
2223     #####
2224     #####
2225
2226     #####
2227     #####
2228     #####
2229
2230     #####
2231     #####
2232     #####
2233
2234     #####
2235     #####
2236     #####
2237
2238     #####
2239     #####
2240     #####
2241
```

Faults

lnFactorial()

```
/**
 * compute the log(n!)
 * @param n
 * @return log(n!)
 */
static double lnFactorial(int n) {
    double retVal = 0.0;
    if (injectFault) {
        for (int i = 1; i <= retVal; i++) {
            retVal += Math.log((double)i);
        }
    }
    else {
        for (int i = 1; i <= n; i++) {
            retVal += Math.log((double)i);
        }
    }
    return retVal;
}
```

calculateSlope()

```
double delta = -(sum * sumx2 - sumx * sumx);

if (injectFault) {
    // get the slope
    slope = (sumx * sumy + sum * sumxy )/delta;
}
else {
    // get the slope
    slope = (sumx * sumy - sum * sumxy )/delta;
}

return slope;
}
```

compareTo()

```
/**
 * for sorting coordinates by their x value
 * @see java.lang.Comparable#compareTo(java.lang.Object)
 */
public int compareTo(Object otherCoord) throws ClassCastException {
    if (!(otherCoord instanceof PhaseSpaceCoordinate)) throw new ClassCastException("A Person object expected.");
    double otherX = ((PhaseSpaceCoordinate)otherCoord).xValue;
    if (injectFault) {
        if (this.xValue < otherX) return 1;
        if (this.xValue > otherX) return -1;
    }
    else {
        if (this.xValue < otherX) return -1;
        if (this.xValue > otherX) return 1;
    }
    return 0;
}
```

formatLatLngValue()

```
/**
 * Converts double value into a fractional string with fracDigits
 * number of decimal places. Should be locale agnostic.
 * @param value value to convert
 * @param fracDigits number of digits after decimal point to hold
 * @return String containing new value
 */
static String formatLatLngValue(double value, double fracDigits) {
    double power;
    if (injectFault){
        power = fracDigits * fracDigits;
    }
    else{
        power = Math.pow(10, fracDigits);
    }
    return String.valueOf(((long)(value*power))/power);
}
```

getDistance()

```
/**
 *
 * @param otherPoint
 * @return the distance between this point and other point in phase space
 */
public double getDistance(PhaseSpaceCoordinate otherPoint) {
    double dist2 = ((this.xValue - otherPoint.xValue)*(this.xValue - otherPoint.xValue))+
                   ((this.yValue - otherPoint.yValue)*(this.yValue - otherPoint.yValue));
    if (injectFault) {
        return dist2;
    }
    else {
        double dist = Math.sqrt(dist2);
        return dist;
    }
}
```

Test Result Report with Faults Live

ID	Method	Requirement	Input	Oracle	Output	Result
1	calculateSlope	Method calculates the slope of a line	5 3 4 7	-4.0	-176.0	Fail
2	calculateSlope	Method calculates the slope of a line	-3 3 3 -3	-1.0	1.0	Fail
3	calculateSlope	Method calculates the slope of a line	136 -38 17 -32	-0.05042016806722689	1.5630252100840336	Fail
4	calculateSlope	Method calculates the slope of a line	35943 4037823 132894 650983	-34.93352311992656	-133.5103817126298	Fail
5	calculateSlope	Method calculates the slope of a line	64.2387634 64.5908703477 64.24089753 64.5906543	-0.10123448901101095	-7.288242889080816E9	Fail
6	compareTo	Method sorts coordinates by their x value	5 -63 72 38	-1.0	1.0	Fail
7	compareTo	Method sorts coordinates by their x value	0 0 1 0	-1.0	1.0	Fail
8	compareTo	Method sorts coordinates by their x value	136 -38 17 -32	1.0	-1.0	Fail
9	compareTo	Method sorts coordinates by their x value	1 0 0 0	1.0	-1.0	Fail
10	compareTo	Method sorts coordinates by their x value	92 92 92 92	0.0	0.0	Pass
11	formatLatLngValue	Method converts double value into a fractional string with default number of decimal places	1253404.47262174 3	1253404.472	1253404.4444444445	Fail
12	formatLatLngValue	Method converts double value into a fractional string with default number of decimal places	0.128427 4	0.1284	0.125	Fail
13	formatLatLngValue	Method converts double value into a fractional string with default number of decimal places	-126.1253799 0	-126.0	NaN	Fail
14	formatLatLngValue	Method converts double value into a fractional string with default number of decimal places	-54727143.0234885 2	-5.472714302E7	-5.4727143E7	Fail
15	formatLatLngValue	Method converts double value into a fractional string with default number of decimal places	9120370981409.12309714287894513 8	9.223372036854776E10	9.12037098140911E12	Fail
16	getDistance	Method returns the distance between this point and other point in phase space	5 5 5 5	0.0	0.0	Pass
17	getDistance	Method returns the distance between this point and other point in phase space	7 3 4 9	6.708203932499369	45.0	Fail
18	getDistance	Method returns the distance between this point and other point in phase space	-52 -3 -23 -45	51.03920062069938	2605.0	Fail
19	getDistance	Method returns the distance between this point and other point in phase space	0 0 0 0	0.0	0.0	Pass
20	getDistance	Method returns the distance between this point and other point in phase space	120983 12349078 487094 430803248	4.18454330157609E8	1.75104026427653216E17	Fail
21	InFactorial	Method computes the log(n!)	0	0.0	0.0	Pass
22	InFactorial	Method computes the log(n!)	-5	0.0	0.0	Pass
23	InFactorial	Method computes the log(n!)	2000000	2.701732365031526E7	0.0	Fail
24	InFactorial	Method computes the log(n!)	1	0.0	0.0	Pass
25	InFactorial	Method computes the log(n!)	3	1.791759469228055	0.0	Fail

Reflections on Fault Injection

Closing Thoughts

Questions? Comments?