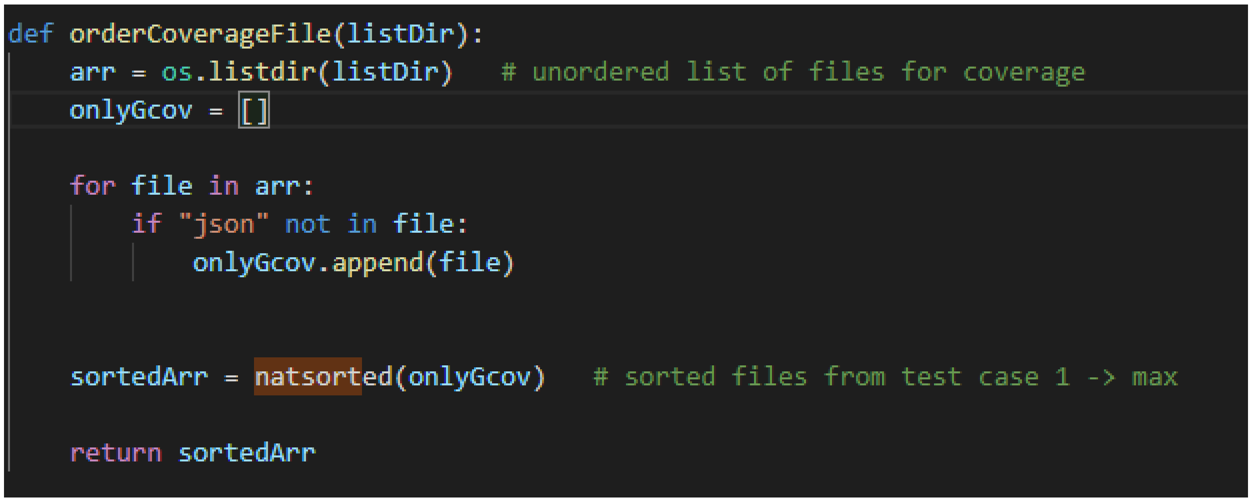
Marcus Martin 862026991
Reuben D’cunha 862034259

CS206 Project

Additional library used: natsort

- In order to be able to list test cases with associated test case and associated coverage in the folder, we needed to get a list of the gcov files ordered from least to greatest test case number so that it would correspond with the test cases from universe.txt.
- For example:
    - “statementCoverage_1_replace.c.gcov” corresponds to the first test case (line 1) in “universe.txt”
    - However, os.listdir(“benchmarks/[benchmark]/[branch/statement]Coverage”) returns the files in the directory in an arbitrary order
    - natsort allowed us to sort the list of gcov files from least to greatest according to the “_#_” in the name of the file
- 

```
def orderCoverageFile(listDir):
    arr = os.listdir(listDir)   # unordered list of files for coverage
    onlyGcov = []

    for file in arr:
        if "json" not in file:
            onlyGcov.append(file)


    sortedArr = natsorted(onlyGcov)   # sorted files from test case 1 -> max

    return sortedArr
```

createGcovsJsons.py: creates gcov and json files for each benchmark in folder “benchmarks/[benchmark]/[statement/branch]Coverage”
statementBranchCoverage.py: lists test cases with associated coverage for each benchmark in folder “benchmarks/[benchmark]/[statement/branch]CoverageLineNumbers/lineNumbers.txt”
orderedLineNumberTestCases.py: lists test cases with associated test case and associated coverage in folder “benchmarks/[benchmark]/[statement/branch]CoverageLineNumbers/2ordered[Statement/Branch]LineNumberTestCases.txt”
testSuites.py: lists coverage for random, total, and additional test prioritization methods in folder “benchmarks/[benchmark]/[statement/branch]CoverageLineNumbers/[randomized/total/additional][Statement/Branch]TestSuite.txt”

faultyTests.py: lists number of faults exposed by a test suite in folder "benchmarks/[benchmark]/[statement/branch]CoverageLineNumbers/[randomized/total/additional]Fails.txt"
datatable.py: lists the test cases, the size of the test suite, and the number of faults exposed for each test suite in folder "benchmarks/[benchmark]/[statement/branch]CoverageLineNumbers/[randomized/total/additional][Statement/Branch]Datatable.txt"

How small are your test suites as compared to the original number of available test cases?
Generated test suites are substantially smaller than the original universe.txt file.

How do the suite sizes change according to different coverage criteria?
Statement and branch coverage, both have shown to exhibit substantially greater size when random tests are taken into account as opposed to additional and total test coverage. The general trend is that size(random) >= size(total) >= size(additional) and statement coverage generally leads to more test cases than branch coverage.

How many faults are exposed by your test suites as compared to the total number of available faults?
For the most part, about 25-75% of the total available faults are able to be exposed by the test suites, there are some exceptions (such as random branch coverage being able to expose 100%) but those are far and few.

Which coverage criteria seem to be the least/most effective at being able to expose faults?
Statement additional, based on collected data, seems to be the least effective at exposing faults while having the smallest test case suite (in most cases).
Branch total, based on collected data, seems to be the most effective at exposing faults while having moderate to smaller test case suites.

What other conclusions can you draw from your observations?
Tcas has a statement (132) which never gets executed based on provided test cases.
Schedule2 has some cases which achieve 100% statement coverage.
Given more time, we could probably construct a concrete answer on which testing method is better but for now, branch total seems to be the most consistent at killing mutants.

SR: statement, random
ST: statement, total
SA: statement, additional
BR: branch, random
BT: branch, total

BA: branch, additional

Tcas (avg mutant killed: 28%)

Suite	Size (max: 1590)	Number of faults exposed (max: 41)	% of mutants killed
SR	5	11	26.8%
ST	4	10	24.3%
SA	4	10	24.3%
BR	13	10	24.3%
BT	13	13	31.7%
BA	11	15	36.6%

Totinfo (avg mutant killed: 61.6%)

Suite	Size (max: 1026)	Number of faults exposed (max: 23)	% of mutants killed
SR	8	18	78.2%
ST	5	14	60.9%
SA	5	14	60.9%
BR	11	11	47.8%
BT	5	11	47.8%
BA	5	17	73.9%

Schedule1 (avg mutant killed: 42.6%)

Suite	Size (max: 2634)	Number of faults exposed (max: 9)	% of mutants killed
SR	7	7	77.8%
ST	3	2	22.2%
SA	3	2	22.2%
BR	11	2	22.2%
BT	8	5	55.6%
BA	7	5	55.6%

Schedule2 (avg mutant killed: 31.4%)

Suite	Size (max: 2679)	Number of faults exposed (max: 9)	% of mutants killed
SR	5	2	22.2%
ST	1	3	33.3%
SA	1	3	33.3%
BR	13	2	22.2%
BT	7	5	55.6%
BA	5	2	22.2%

Print1 (avg mutant killed: 64.3%)

Suite	Size (max: 4072)	Number of faults exposed (max: 7)	% of mutants killed
SR	12	3	42.8%
ST	6	6	85.7%
SA	5	4	57.1%
BR	15	5	71.4%
BT	7	5	71.4%
BA	6	4	57.1%

Print2 (avg mutant killed: 81.4%)

Suite	Size (max: 4057)	Number of faults exposed (max: 9)	% of mutants killed
SR	10	6	66.7%
ST	4	7	77.8%
SA	4	7	77.9%
BR	12	9	100%
BT	6	8	88.9%
BA	4	7	77.8%

Replace (avg mutants killed: 41.3%)

Suite	Size (max: 5542)	Number of faults exposed (max: 31)	% of mutants killed
SR	19	12	38.7%

ST	12	10	32.2%
SA	9	6	19.3%
BR	27	14	45.1%
BT	21	19	61.2%
BA	11	16	51.6%