**Program Analysis and Testing**

# Numpy Testing Report

Analysing large software projects and provide assessment of testing

[Project GitHub Link]

**Keerthi Pogula -** 2145674
**Usha Rani Nalla -** 2144301
**Sai Krishna Bhopathi -** 2086040

Table of Contents

## Introduction

Testing is critical because it identifies errors and bugs before the software is delivered to the client, ensuring the software's quality. While clever software can make our lives easier, a glitch can be disastrous. Computer bug disasters have killed people and ruined lives on a significant scale in the previous few decades.

A military plane crashed in Seville in May 2015 after making a failed emergency landing on its first flight, killing four air force crew members and injuring two more. Engineers have discovered abnormalities in the aircraft's data logs that may be related to the crash, according to the planemaker.
Last year, a computer glitch almost bankrupted an investment firm when a software mistake allowed computers to buy and sell millions of shares without human control.

Hence, rigorous testing in Software Design is critical because our daily lives involve a lot of software usage, from banking applications to automated metro trains and flights that we take, and a faulty product could result in financial or human casualties.

## Numpy

Numerical Python (Numpy) is a library in Python used for working with mathematical and numerical calculations. It is an open-source project that we can use for free. It includes a large library of high-level mathematical functions for working with arrays and matrices.

In this project, we tested the Numpy open source library to test its code coverage, analysed the results and generated the plots to explain the same.

## Testing

Software testing is the process of evaluating the functionality of a software program with the goal of determining whether the developed software met the defined requirements and identifying defects in order to deliver a quality product. Software failures can occur due to ambiguities, errors, coding flaws, insufficient testing, and other unforeseen issues.

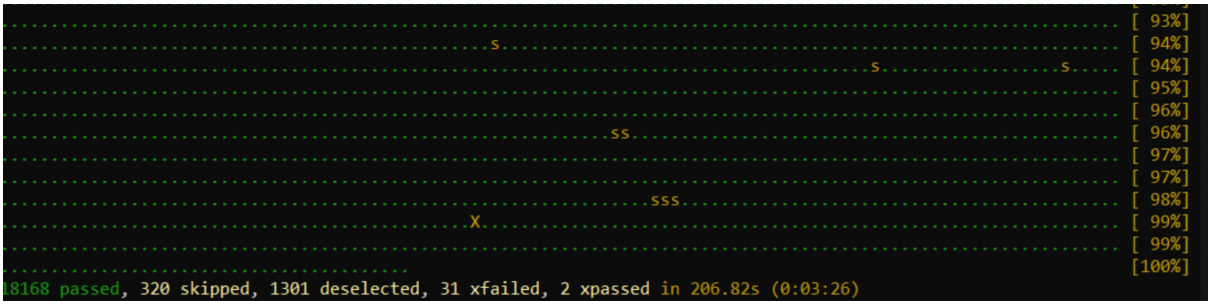
The entire functionality of a function should be tested, as well as its robustness to erroneous or unexpected input parameters.

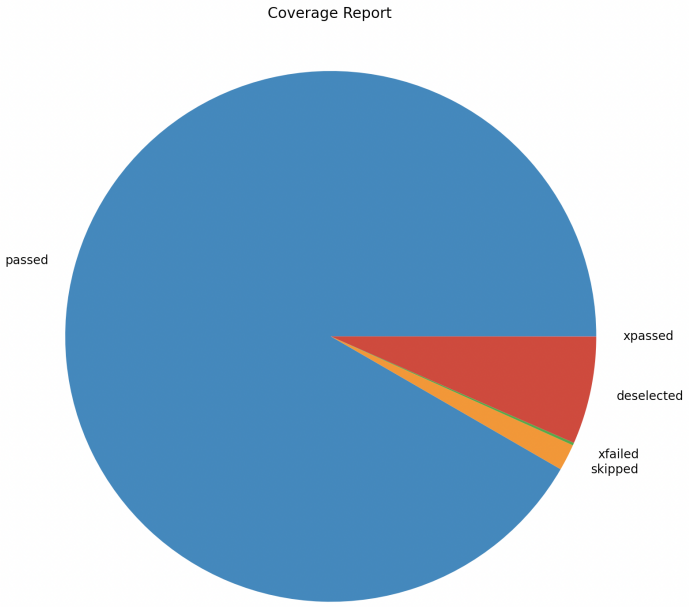
### Agile Testing

Agile testing is the process where both development and testing takes place concurrently. A developer wants to work quickly, but they also need the assistance of tests to detect errors in their code as quickly as possible. Detecting the bugs early in the development reduces defects and also saves the stakeholders costly bugs. Hence, the collaboration between developers and testers is very important as it helps in quicker identification, and correction of bugs which will help them to achieve better code quality and faster delivery expectations.

### Test Statistics

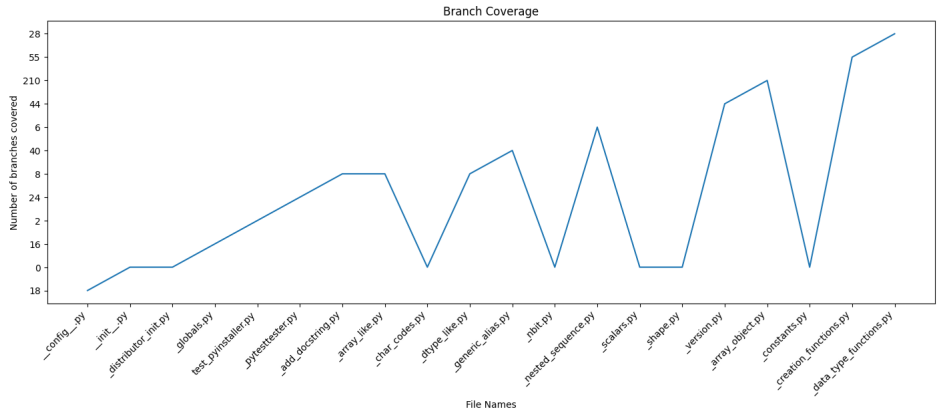
Below is the statistics of running the runtests.py. Both the pie chart and the run statistics indicate the maximum number of tests are passed.



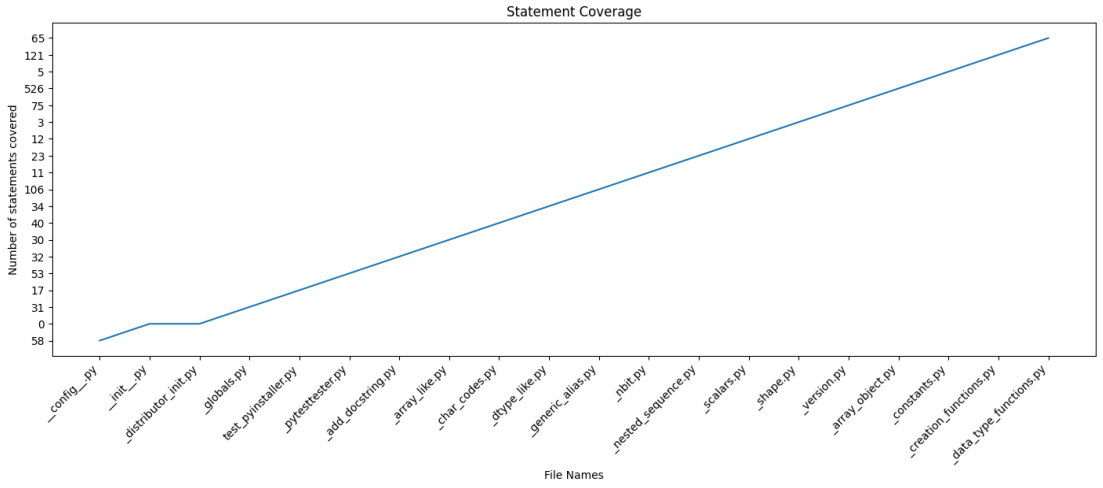
## Branch Coverage

Branch coverage is a metric that indicates whether all branches in a codebase are exercised by tests.
Below is the branch coverage that we obtained for the Numpy project. We have included the top files with maximum branch coverage to show the below plot.




## Line/ Statement Coverage

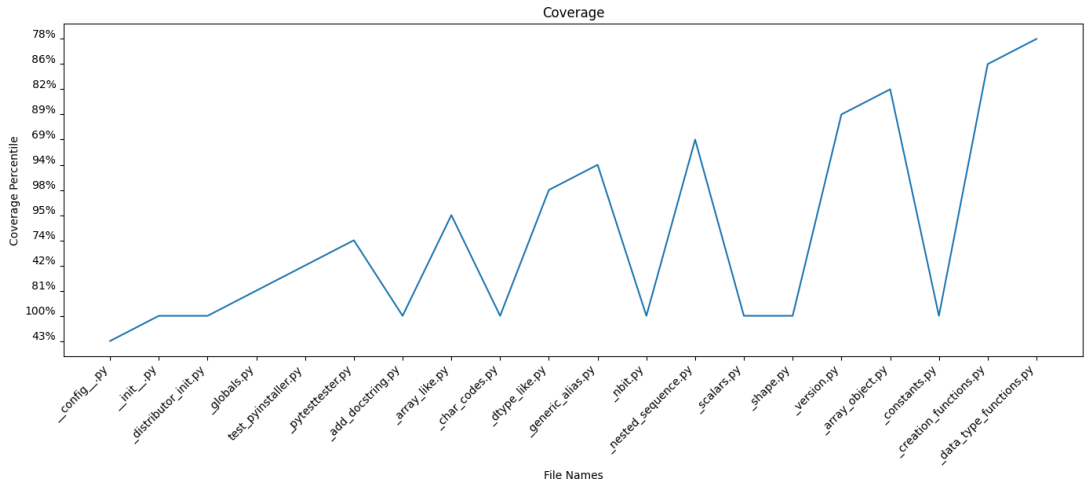
Statement Coverage is a white box testing technique in which all the executable statements in the source code are executed at least once. It is used for calculation of the number of statements in source code which have been executed.
Below is the line coverage that we achieved for the Numpy project. We have considered the top files with maximum line coverage.




## Total Coverage

Below is the total coverage that we achieved for different files in the Numpy project. It is sorted in ascending order of coverage.

There are very few files that have achieved 100% coverage. Hence, the test cases have to be written more efficiently.
https://github.com/kpogula/NumpyTest/blob/main/Outputs/output.csv




### Number of Test Files in Numpy Folder

Below is the total number of test files that are present in the Numpy project.

```
import glob

# Regular Expression to Identify Test Files
test_files = glob.glob("numpy/numpy/" + r"/**/*test*.py", recursive=True)
print("Total Number of Test Files:", len(test_files))

```

```
Terminal:  Local
(venv) (base) keerthipogula@Keerthis-MacBook-Pro PAT % python test.py
Total Number of Test Files: 172
```

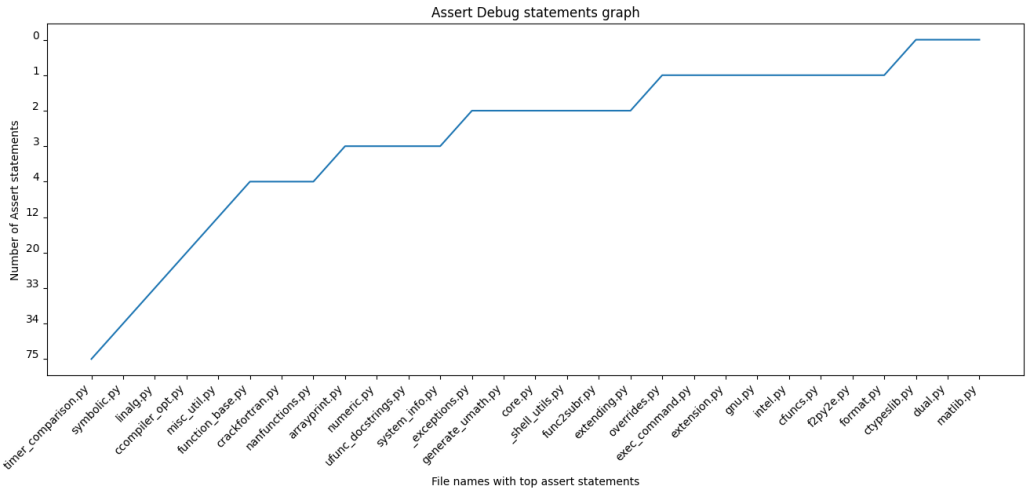
### Asserts Statements Count

Below table shows the test files with maximum number of Assert Statements.

| File Name | Assert Statements Count |
|---|---|
| test_core.py | 1607 |
| test_function_base.py | 932 |

| File Name | Assert Statements Count |
|---|---|
| test_umath.py | 913 |
| test_datetime.py | 839 |
| test_nditer.py | 632 |
| test_ufunc.py | 532 |
| test_generator_mt19937.py | 532 |
| test_extras.py | 497 |
| test_randomstate.py | 436 |
| test_linalg.py | 401 |

Below plot shows the number of assert statements found in test files. We have considered the top files with the maximum number of assert statements.



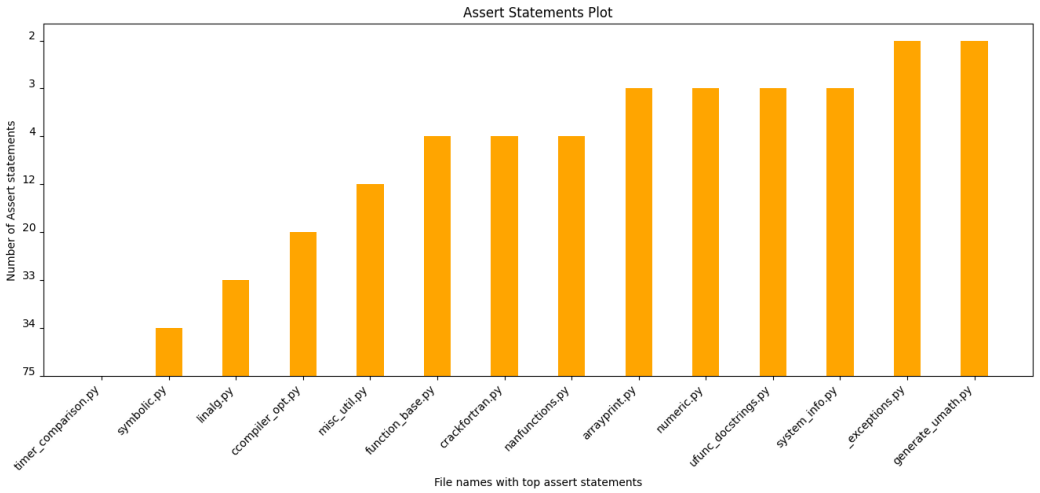

### Assert and Debug Statements in Production Files

Below plots show the Assert and Debug statements count in the production files. The production files don't include test files.

- **Assert Statements:**

  Below bar graph containing the top files with maximum number of assert statements indicates that there are a good number of production files with assert statements.

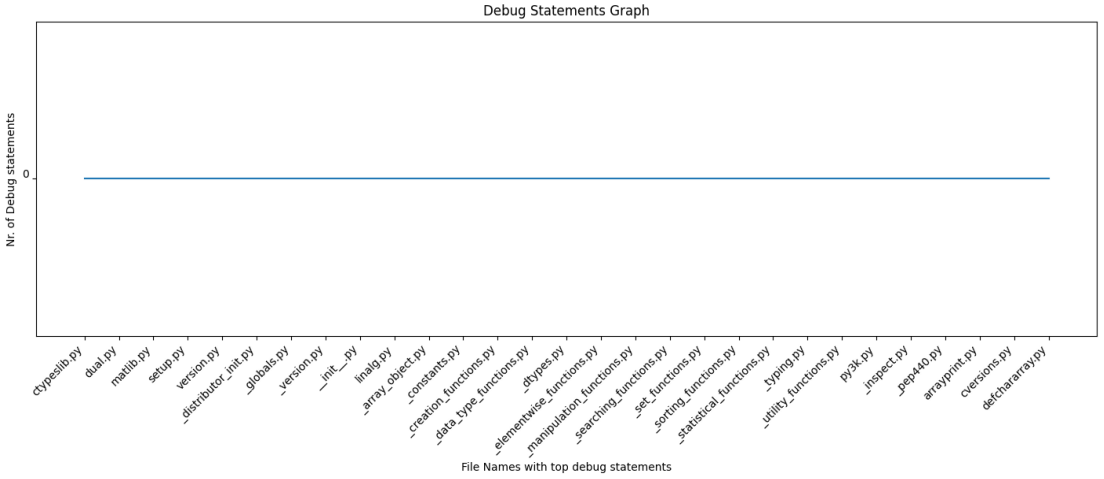
  https://github.com/kpogula/NumpyTest/blob/main/Outputs/asserts_debug_count1.csv




- **Debug Statements:**

  Below plot shows that there are no debug statements in the production files. The CSV file below has the list of files from the Numpy project that shows that the number of debug statements in all of them are zero.

  https://github.com/kpogula/NumpyTest/blob/main/Outputs/asserts_debug_count2.csv

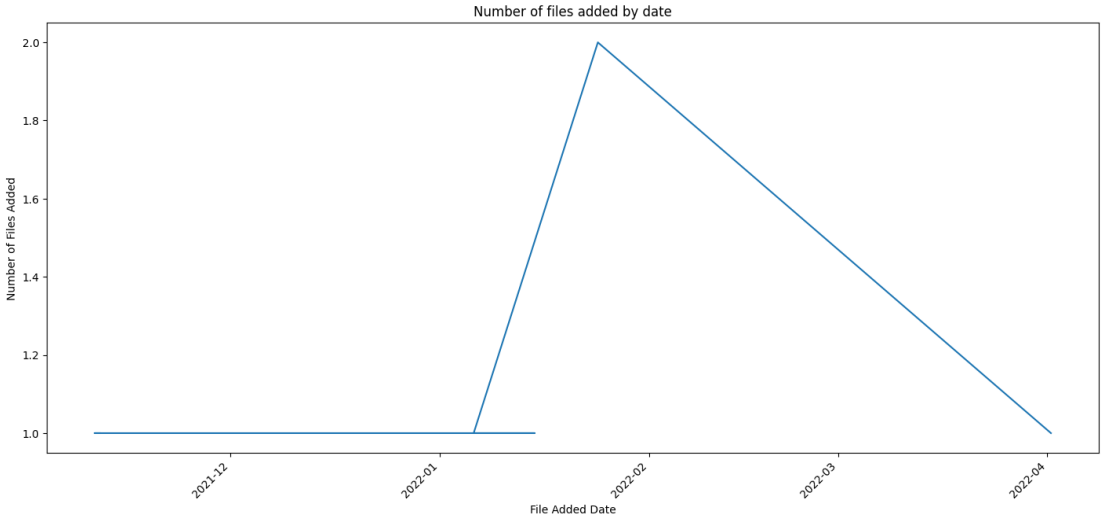


**Pydriller**

PyDriller is a Python framework that helps developers on mining software repositories. With PyDriller you can easily extract information from any Git repository, such as commits, developers, modifications, diffs, and source codes, and quickly export CSV files.

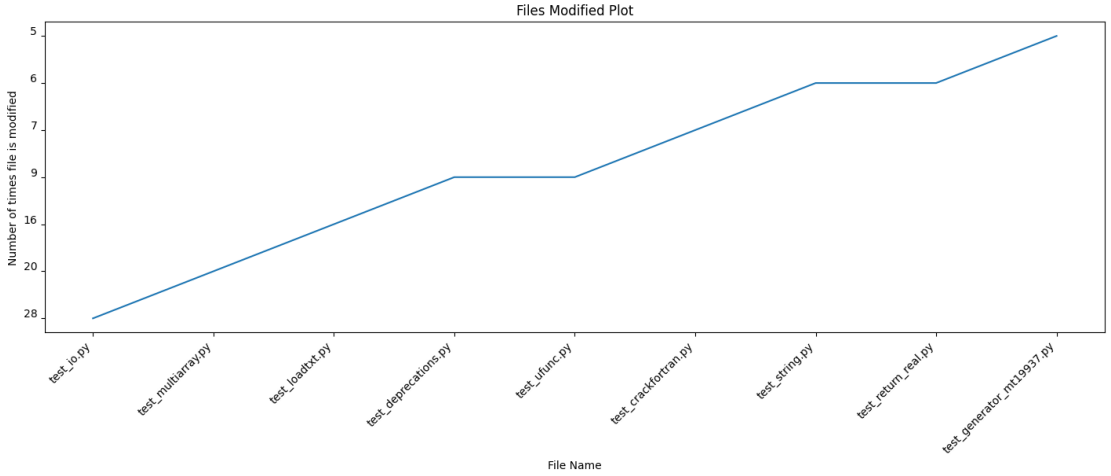
a. **When the test files have been added:**
   Below is the plot that shows the number of files that are added to the Numpy project over the last year. The CSV file mentioned below has all the files that are being added in the span of a year. It shows that a good number of files are added in the given time interval.

   https://github.com/kpogula/NumpyTest/blob/main/Outputs/FilesAdded.csv




b. **How often they are modified**
   Below is the plot that shows the number of times the files are modified in the Numpy project. To show the below graph, we have considered the files that are modified most frequently. It shows that the files are modified frequently. It's a good sign from a developer's perspective.
   The file below is the file containing all the files from the Numpy project that are being modified over the span of a year.

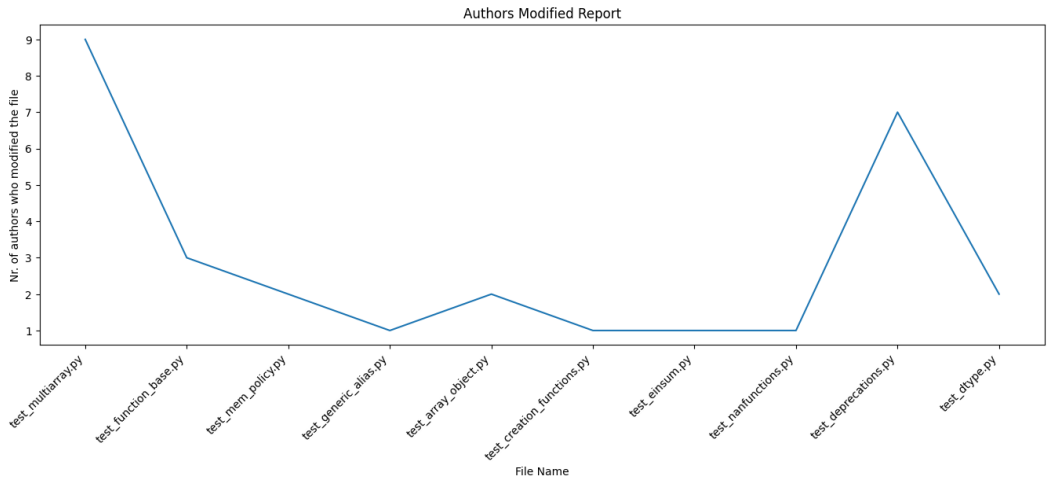
   https://github.com/kpogula/NumpyTest/blob/main/Outputs/FilesModified.csv




**c. How many people were involved for each file**

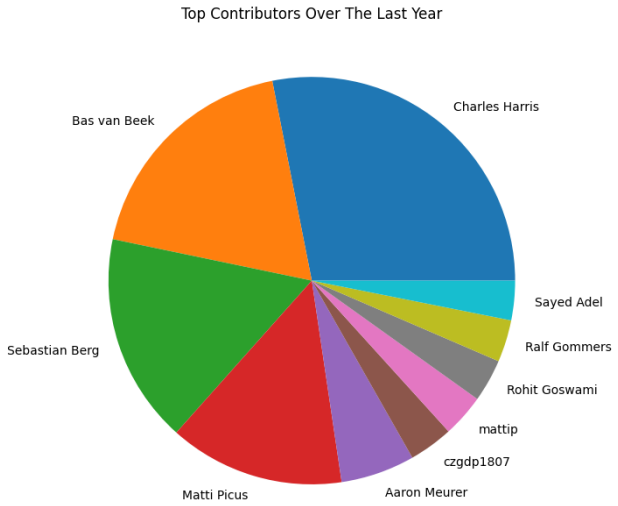
Below plot shows the number of authors for each file that's being modified. We have considered the files that are modified most frequently to show the below graph. It shows a decent number of modifications made for each file in a given period of time. The CSV file mentioned here contains the list of all the python files from the Numpy project with the list of authors who modified them.

https://github.com/kpogula/NumpyTest/blob/main/Outputs/AuthorList.csv




## Total Contributions

Below mentioned pie chart shows the top contributors to the Numpy project in the span of a year. It shows that there are four people who contribute the most to the Numpy project.


Top Contributors Over The Last Year
Charles Harris
Bas van Beek
Sebastian Berg
Sayed Adel
Ralf Gommers
Rohit Goswami
mattip
czgdp1807
Aaron Meurer
Matti Picus