University College Cork, Ireland
Coláiste na hOllscoile Corcaigh

MuTester

A Mutation Testing Tool for Academic Demonstration of Mutation Testing

INSTRUCTIONS MANUAL

Version 1

30th July, 2020

CONTENTS

ABOUT

MuTester is a Mutation Testing tool for the academic demonstration of mutation testing. Mutation testing necessitates automatic mutant generation. It is a type of software testing in which we mutate certain statements in the source code and check if the errors can be found in the test cases. It is a method to validate the quality of your test suite. The quality of the test suite can be determined by the percentage of killed mutants. MuTester achieves this by taking as input your java program and the corresponding test suite that tests the program, generates mutants of the program and runs the test suite against each mutant thereby checking the quality of the test suite.

SYSTEM REQUIREMENTS

- Windows (Minimum Requirements):
  - Edition: Windows 7
  - Processor: Core 2 Duo
  - RAM: 2 GB
  - Type: 32-bit
  - Java: Version 8

- MAC (Minimum Requirements):
  - Edition: MAC OS Mojave
  - Processor: i3
  - RAM: 2 GB
  - Type: 32-bit
  - Java: Version 8

INSTALLATION STEPS

MuTester is available as an open source software and is free to download from below link:

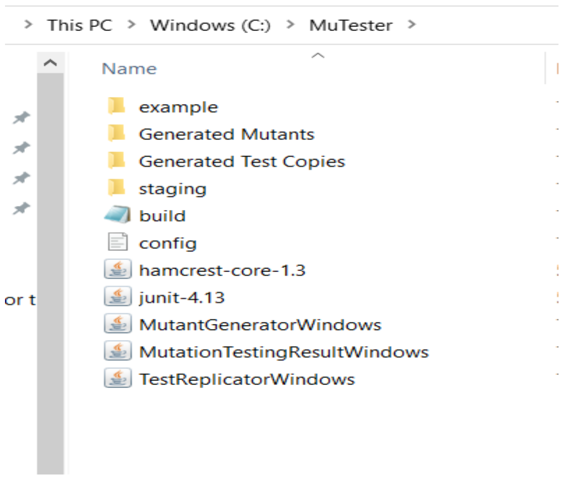
- For Windows OS: https://github.com/harshalk994/MuTester-for-Windows-OS
- For MAC/Linux OS: https://github.com/harshalk994/MuTester-for-MAC-OS

MuTester is licensed under the Apache License 2.0. You can find a copy of the license in the MuTester folder. You can also view the license at below link:

- https://www.apache.org/licenses/LICENSE-2.0

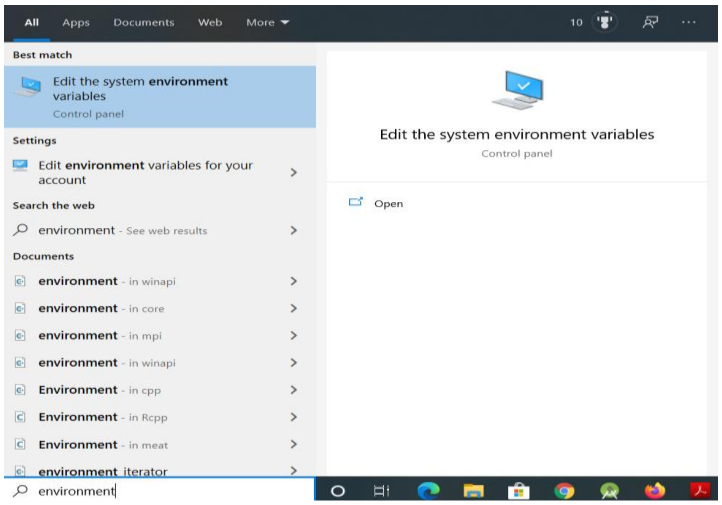
➢ Steps for Windows OS:

1. After the tool is downloaded, unzip it to the desired directory.
   - Eg: **C:\MuTester**



2. The tool folder (MuTester) will consist of the Junit and Hamcrest JARs. You need to add the Junit and Hamcrest JARs to the environment variables under the CLASSPATH variable.
   (If you want to download Junit from the web use the below link:
   https://github.com/junit-team/junit4/wiki/Download-and-Install)

   - Click on the Windows icon in the bottom left corner of your desktop and type "**environment**".
   - Click on "**Edit the system environment variables**" to open it.



Best match

Edit the system **environment** variables
Control panel

Settings

Edit **environment** variables for your account

Search the web

environment - See web results

Documents

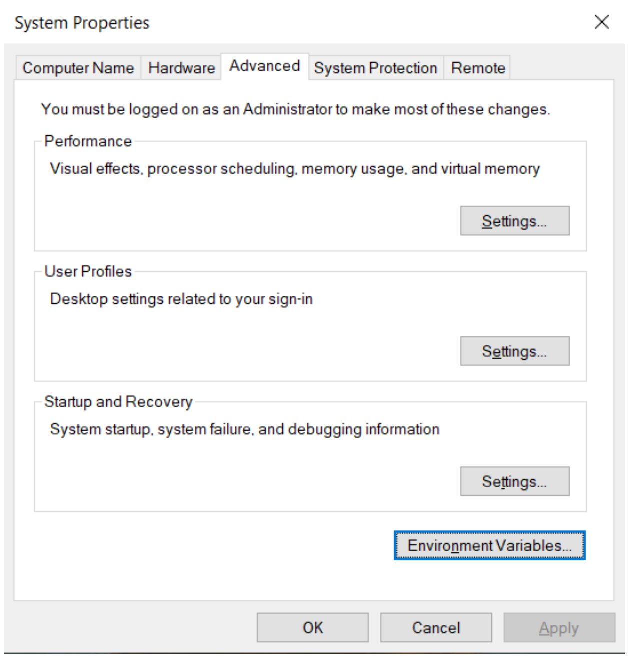
environment - in winapi
environment - in core
environment - in mpi
environment - in winapi
Environment - in cpp
Environment - in Rcpp
Environment - in meat
environment iterator

environment

Edit the system environment variables
Control panel

Open

- The system properties window should open on your screen.


System Properties

You must be logged on as an Administrator to make most of these changes.

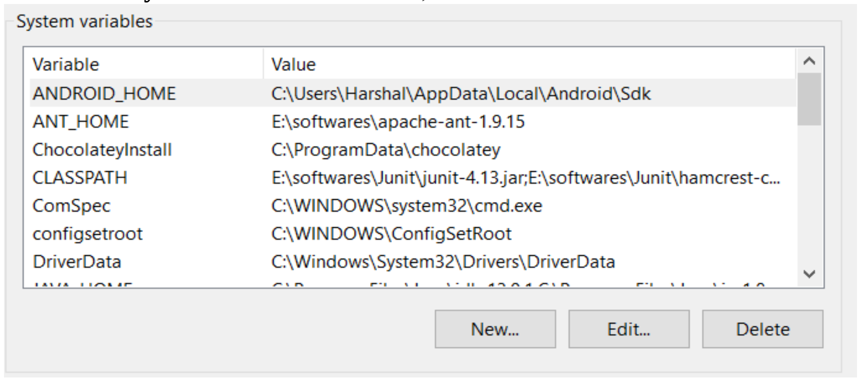
Performance
Visual effects, processor scheduling, memory usage, and virtual memory
Settings...

User Profiles
Desktop settings related to your sign-in
Settings...

Startup and Recovery
System startup, system failure, and debugging information
Settings...

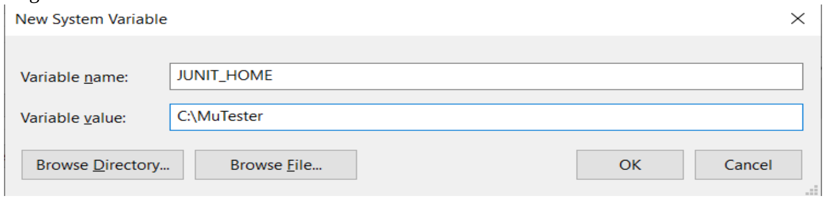
Environment Variables...

- Click on "**Environment Variables**" button.
- Under the "**System Variables**" section, click on "**New**"


System variables

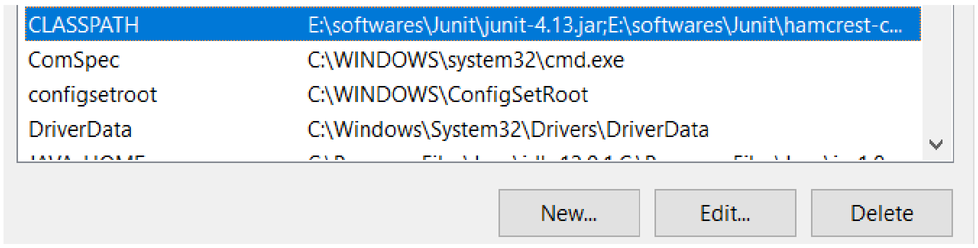
Variable	Value
ANDROID_HOME	C:\Users\Harshal\AppData\Local\Android\Sdk
ANT_HOME	E:\softwares\apache-ant-1.9.15
ChocolateyInstall	C:\ProgramData\chocolatey
CLASSPATH	E:\softwares\Junit\junit-4.13.jar;E:\softwares\Junit\hamcrest-c...
ComSpec	C:\WINDOWS\system32\cmd.exe
configsetroot	C:\WINDOWS\ConfigSetRoot
DriverData	C:\Windows\System32\Drivers\DriverData

- You have to create a new system variable for Junit.
  - In the "**Variable name:**" field enter JUNIT_HOME.
  - In the "**Variable value:**" field enter the path to where the Junit JAR is stored in your system.
    - Eg: **C:\MuTester**


New System Variable

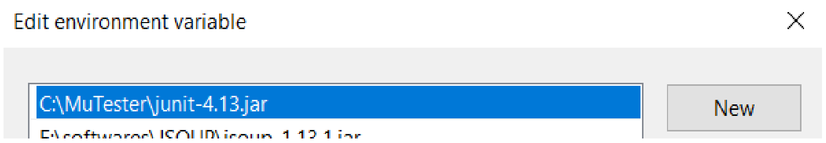
Variable name:	JUNIT_HOME
Variable value:	C:\MuTester

- Next, add the Junit and Hamcrest JAR paths to the CLASSPATH variable.
  - Click on **CLASSPATH** and then click on **Edit** button:



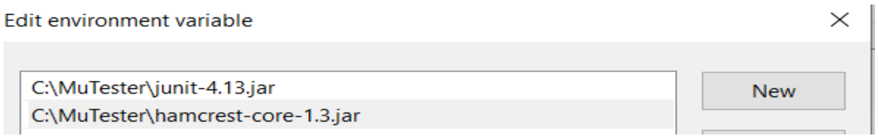
CLASSPATH	E:\softwares\Junit\junit-4.13.jar;E:\softwares\Junit\hamcrest-c...
ComSpec	C:\WINDOWS\system32\cmd.exe
configsetroot	C:\WINDOWS\ConfigSetRoot
DriverData	C:\Windows\System32\Drivers\DriverData

  - Click on **New** button and add the path for Junit JAR:


Edit environment variable

C:\MuTester\junit-4.13.jar

  - Similarly add the path for Hamcrest JAR, click on **New** button and add the path for Hamcrest JAR:


Edit environment variable

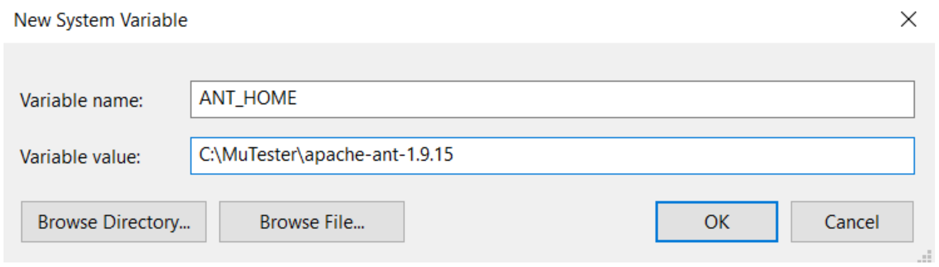
C:\MuTester\junit-4.13.jar
C:\MuTester\hamcrest-core-1.3.jar

  - Click on the **Ok** button.

3. Now we have to install ANT in the system. Apache ANT v1.9.15 is also downloaded when you download the tool from the Git repository. Unzip the Apache ANT zip file and store it anywhere in your system. Then we have to add ANT to the environment variables.
(If you want to download it from the web, use the below link: https://downloads.apache.org//ant/binaries/apache-ant-1.9.15-bin.zip)

- Under the "**System Variables**" section, click on "**New**"
- You have to create a new system variable for ANT.
  - In the "**Variable name:**" field enter ANT_HOME
  - In the "**Variable value:**" field enter the path to where the Apache ANT folder is stored in your system.

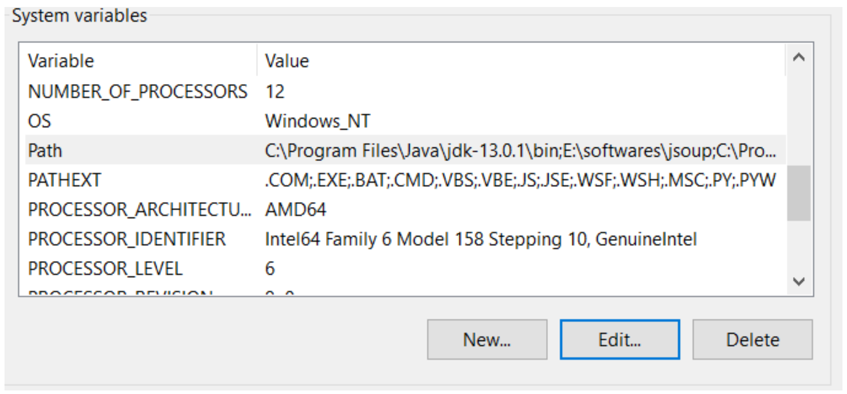
  - Eg: **C:\MuTester\apache-ant-1.9.15**



New System Variable

Variable name:	ANT_HOME
Variable value:	C:\MuTester\apache-ant-1.9.15

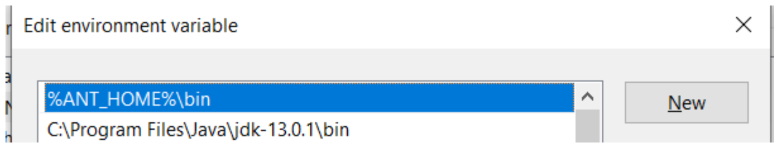
- Next, you need to add the ANT bin path to the System Variables.
  - Click on **Path** and then click on **Edit**.



System variables

Variable	Value
NUMBER_OF_PROCESSORS	12
OS	Windows_NT
Path	C:\Program Files\Java\jdk-13.0.1\bin;E:\softwares\jsoup;C:\Pro...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC;.PY;.PYW
PROCESSOR_ARCHITECTU...	AMD64
PROCESSOR_IDENTIFIER	Intel64 Family 6 Model 158 Stepping 10, GenuineIntel
PROCESSOR_LEVEL	6

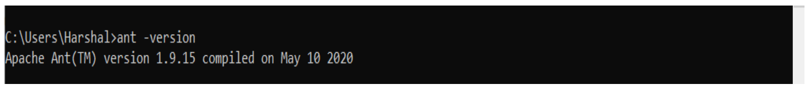
  - Click on **New** button and add the below path:
    - **%ANT_HOME%\bin**



Edit environment variable

%ANT_HOME%\bin
C:\Program Files\Java\jdk-13.0.1\bin

- To verify is ANT is successfully installed in your system, type the below command in command prompt. It should display the ant version installed in the system:
    - **ant -version**

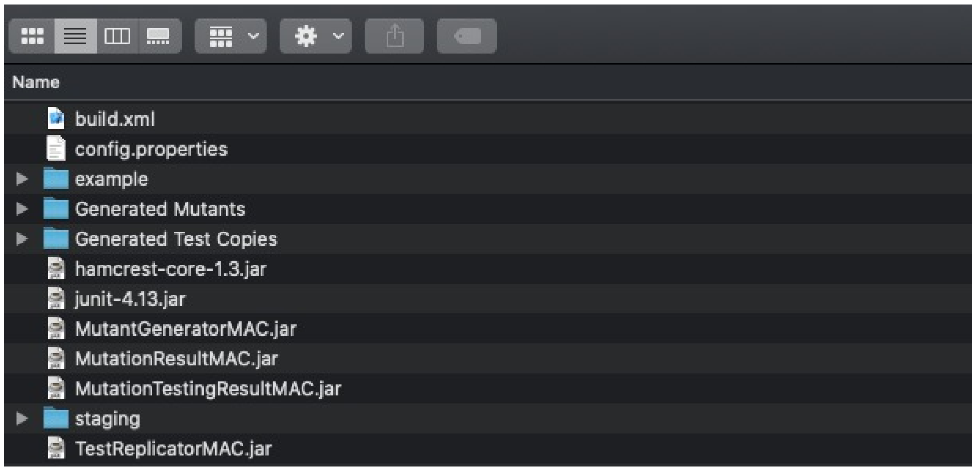


```
C:\Users\Harshal>ant -version
Apache Ant(TM) version 1.9.15 compiled on May 10 2020
```

---END OF SETUP STEPS FOR WINDOWS OS---------------------------------------

➢ Steps for MAC OS:

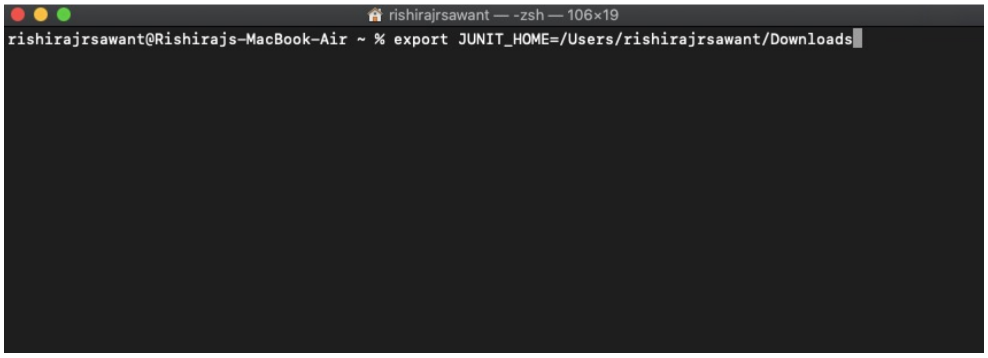
1. After the tool is downloaded, unzip it to the desired directory.
   a. **/Users/rishirajrsawant/Downloads/MuTester**



```
build.xml
config.properties
example
Generated Mutants
Generated Test Copies
hamcrest-core-1.3.jar
junit-4.13.jar
MutantGeneratorMAC.jar
MutationResultMAC.jar
MutationTestingResultMAC.jar
staging
TestReplicatorMAC.jar
```

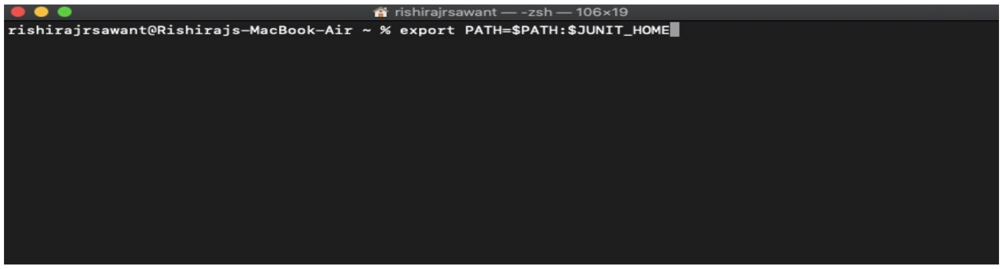
2. The tool folder (MuTester) will consist of the Junit and Hamcrest JARs. You need to add the Junit and Hamcrest JARs to the environment variables.
   (If you want to download Junit from the web use the below link: https://github.com/junit-team/junit4/wiki/Download-and-Install)

   a. **Copy** and **paste** the Junit and Hamcrest JARs in any other directory than the MuTester directory.
   b. Open the terminal and navigate to the directory where you have pasted the JARs.
   c. Type the below commands one after the other in the terminal to setup Junit in the system:
      - **export JUNIT_HOME=/Users/rishirajrsawant/Downloads**



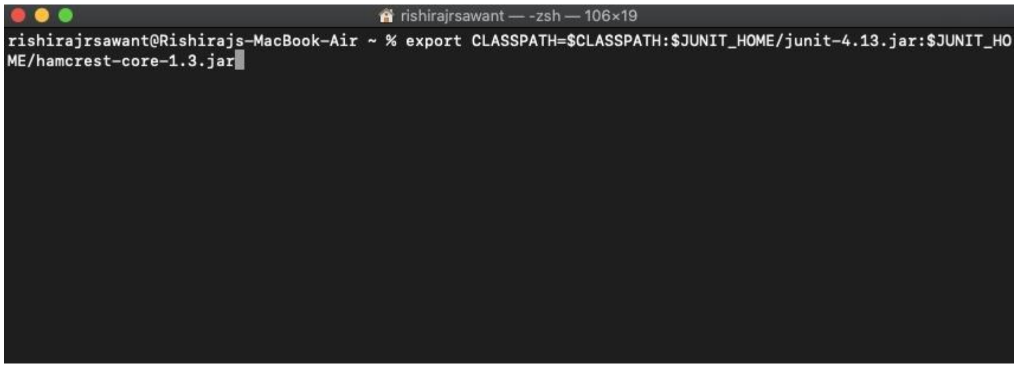
```
rishirajrsawant@Rishirajs-MacBook-Air ~ % export JUNIT_HOME=/Users/rishirajrsawant/Downloads
```

- **export PATH=$PATH:$JUNIT_HOME**



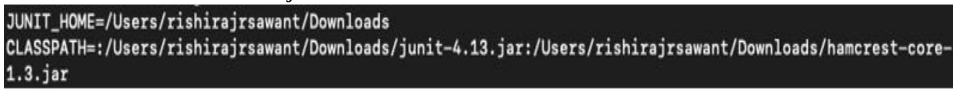
```
rishirajrsawant@Rishirajs-MacBook-Air ~ % export PATH=$PATH:$JUNIT_HOME
```

- **export CLASSPATH=$CLASSPATH:$JUNIT_HOME/junit-4.13.jar:$JUNIT_HOME/hamcrest-core-1.3.jar**



```
rishirajrsawant@Rishirajs-MacBook-Air ~ % export CLASSPATH=$CLASSPATH:$JUNIT_HOME/junit-4.13.jar:$JUNIT_HO
ME/hamcrest-core-1.3.jar
```

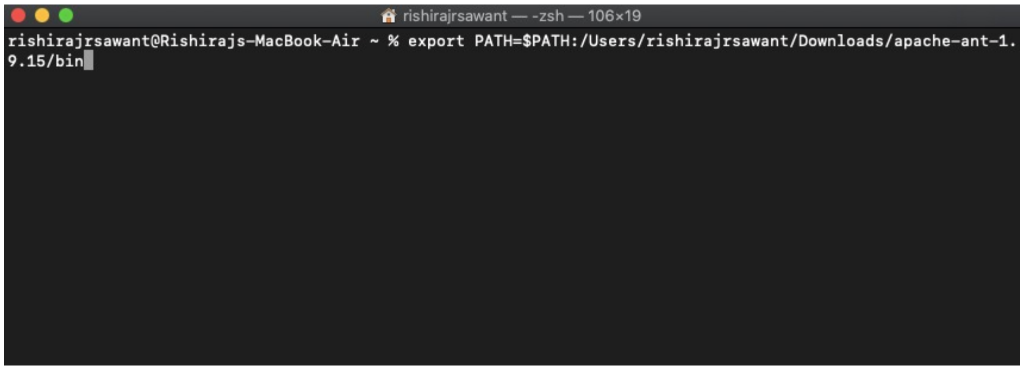
- To verify if the variable is successfully added, type below command in terminal:
  - **printenv**
    a. Check if JUNIT_HOME variable is added and also check if the CLASSPATH variable is appended with the junit and hamcrest jars.



```
JUNIT_HOME=/Users/rishirajrsawant/Downloads
CLASSPATH=:/Users/rishirajrsawant/Downloads/junit-4.13.jar:/Users/rishirajrsawant/Downloads/hamcrest-core-
1.3.jar
```

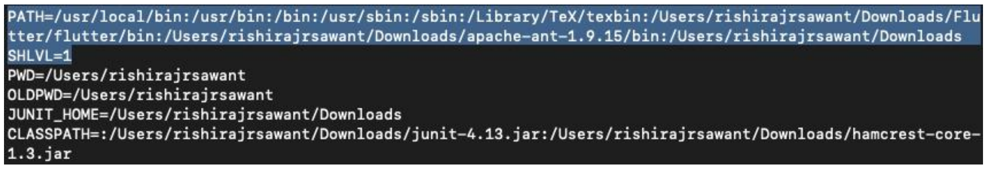
d. Next, you have to install ANT in your system. Apache ANT v1.9.15 is also downloaded when you download the tool from the Git repository. Unzip the Apache ANT zip file and store it anywhere in your system. Then we have to add ANT to the PATH variable.
(If you want to download it from the web, use the below link:
https://downloads.apache.org//ant/binaries/apache-ant-1.9.15-bin.zip)

- Open the terminal and navigate to the directory where you have pasted the ANT folder.
- Type the below command in the terminal to setup ANT in the system:
  - **export PATH=$PATH:/Users/rishirajrsawant/Downloads/Downloads/apache-ant-1.9.15/bin**



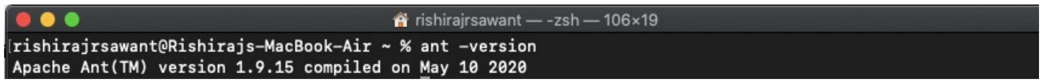
```
rishirajrsawant@Rishirajs-MacBook-Air ~ % export PATH=$PATH:/Users/rishirajrsawant/Downloads/apache-ant-1.
9.15/bin
```

- To verify if the variable is successfully added, type below command in terminal:
  - **printenv**
    a. Check if Path variable is appended with apache ant bin folder.



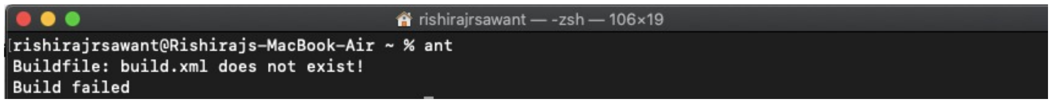
```
PATH=/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin:/Library/TeX/texbin:/Users/rishirajrsawant/Downloads/Flu
tter/flutter/bin:/Users/rishirajrsawant/Downloads/apache-ant-1.9.15/bin:/Users/rishirajrsawant/Downloads
SHLVL=1
PWD=/Users/rishirajrsawant
OLDPWD=/Users/rishirajrsawant
JUNIT_HOME=/Users/rishirajrsawant/Downloads
CLASSPATH=:/Users/rishirajrsawant/Downloads/junit-4.13.jar:/Users/rishirajrsawant/Downloads/hamcrest-core-
1.3.jar
```

- To check if ANT is successfully installed, type below command:
  - **ant -version**



```
rishirajrsawant@Rishirajs-MacBook-Air ~ % ant -version
Apache Ant(TM) version 1.9.15 compiled on May 10 2020
```

  - **ant**



```
rishirajrsawant@Rishirajs-MacBook-Air ~ % ant
Buildfile: build.xml does not exist!
Build failed
```

- If you see the above messages, ANT is successfully installed in the system.

--------------------------------------END OF SETUP STEPS FOR MAC OS--

MuTester SPECIFICATIONS

The tool consists of below specifications:

1) **Generated Mutants folder:**
    - This folder will store all the mutants that are generated for the original program. It will also consist of a temporary copy of the original program in a file named Temp.java.

2) **Generate Test Copies folder:**
    - This folder will store the copies of the test suite that will be invoked to test each individual mutant.

3) **Staging folder:**
    - This folder will store the class files generated after compiling the mutants and test suite copies.

4) **Build file:**
    - This is the ANT build file script that will be used to run the tool.

5) **Reports folder:**
    - A reports folder will be generated after the ANT script is run. This folder will store the detailed reports of the Junit tests run by the tool. You can view the generated reports in the reports\html subfolder.

6) **HTML report:**
    - An html report will be generated after running the ANT script. The report will give you detailed information on the mutations that took place for the original program.

7) **Config File:**
    - The config file is the file that will configure the user inputs for the tool.
    - The below user inputs will be recorded in the config file:
        i. **originalprogrampath**: Store the path to where the original program is stored in the user’s system. (User should not include the program/class name in the path, Eg: C:\Java Programs)
        ii. **mutantdestination**: Store the path to where the generated mutants will be stored. (User should not edit this path as its value is already stored for the tool)
        iii. **originaltestpath**: Store the path to where the original test suite is stored in the user’s system. (User should include the filename in the path, Eg: C:\Java Programs\TestSuite.java)
        iv. **testcopypath**: Store the path to where the generated test suite copies will be stored. (User should not edit this path as its value is already stored for the tool)
        v. **originalclassname**: Store the name of the original program, Eg: Triangle
        vi. **testclassname**: Store the name of the class that is being tested by the test suite, Eg: Triangle.
        vii. **dependentclassname**: If you are testing a dependent class, ie a class that is dependent on the original class, provide its name in the dependentclassname field.
        viii. **originalprogrampackagename**: Store the package name of the original program, Eg: com.hsk.code
        ix. **testpackagename**: Store the package name of the test suite, Eg: com.hsk.test
        x. **reportspath**: Store the path to where the generated test reports will be stored. (User should not edit this path as its value is already stored for the tool)
        xi. **testdependentclass(y/n)**: If you are testing a dependent class, enter “**y**” (w/o the inverted commas) for this field. The default value for this field is “**n**” (w/o the inverted commas).

xii. **arithop(y/n)**: If you want to apply the arithmetic operator mutation rule, enter "**y**" (w/o the inverted commas) for this field. The default value for this field is "**y**" (w/o the inverted commas). If you do not want to apply the rule, specify "**n**" (w/o the inverted commas) in this field.

xiii. **assignmentop(y/n)**: If you want to apply the assignment operator mutation rule, enter "**y**" (w/o the inverted commas) for this field. The default value for this field is "**y**" (w/o the inverted commas). If you do not want to apply the rule, specify "**n**" (w/o the inverted commas) in this field.

xiv. **bitwiseop(y/n)**: If you want to apply the bitwise operator mutation rule, enter "**y**" (w/o the inverted commas) for this field. The default value for this field is "**y**" (w/o the inverted commas). If you do not want to apply the rule, specify "**n**" (w/o the inverted commas) in this field.

xv. **conditionalop(y/n)**: If you want to apply the conditional operator mutation rule, enter "**y**" (w/o the inverted commas) for this field. The default value for this field is "**y**" (w/o the inverted commas). If you do not want to apply the rule, specify "**n**" (w/o the inverted commas) in this field.

xvi. **incrementdecrementop(y/n)**: If you want to apply the increment/decrement operator mutation rule, enter "**y**" (w/o the inverted commas) for this field. The default value for this field is "**y**" (w/o the inverted commas). If you do not want to apply the rule, specify "**n**" (w/o the inverted commas) in this field.

xvii. **relationalop(y/n)**: If you want to apply the relational operator mutation rule, enter "**y**" (w/o the inverted commas) for this field. The default value for this field is "**y**" (w/o the inverted commas). If you do not want to apply the rule, specify "**n**" (w/o the inverted commas) in this field.

xviii. **shiftop(y/n)**: If you want to apply the shift operator mutation rule, enter "**y**" (w/o the inverted commas) for this field. The default value for this field is "**y**" (w/o the inverted commas). If you do not want to apply the rule, specify "**n**" (w/o the inverted commas) in this field.

xix. **deletekilledmutants(y/n)**: If you want to delete the mutants if they are killed, enter "**y**" (w/o the inverted commas) for this field. The default value for this field is "**n**" (w/o the inverted commas). If the value is set to "**y**", it will also automatically delete the test suite copies that killed the respective mutants.

8) MutantGeneratorWindows/ MutantGeneratorMAC JAR:

- The Mutant Generator JAR file will be invoked by the ANT script to generate the mutants of the original program and store them in the Generated Mutants folder.

9) Test ReplicatorWindows/ TestReplicatorMAC JAR:

- The Test Replicator JAR file will be invoked by the ANT script to generate copies of the test suite and store them in the Generated Test Copies folder.

10) MutationTestingResultWindows / MutationTestingResultMAC JAR:

- The Mutation Testing Result JAR file will be invoked by the ANT script to generate the mutation score results. It will also run a deletion mechanism after calculating the mutation score to delete the killed mutants and test suite copies that killed the mutants if the deletekilledmutants(y/n) flag is set to "**y**" in the config file.

---------------------------------------END OF MuTester Specifications---

HOW TO RUN THE TOOL

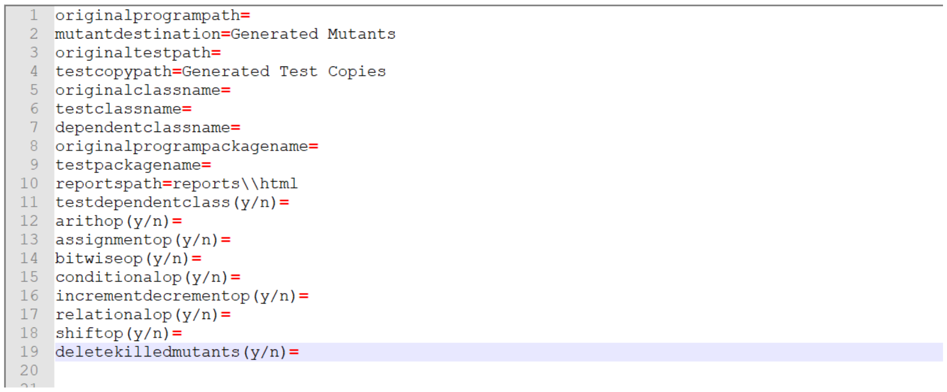
1) Steps for Windows OS:

Notes:

- ✓ **In this manual, the program Triangle.java stored in the examples folder of the tool will be used for demonstration.**
- ✓ **For every "\" in the path, make sure to append another "\" to it. For eg, if the path is C:\MuTester\examples, then enter it in the config file as C:\\MuTester\\examples**

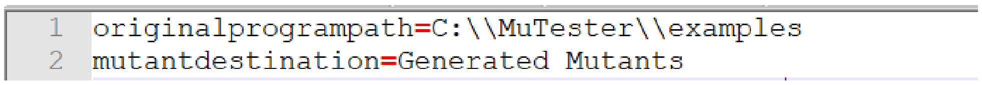
a. The first step is to configure the user inputs in the config file.

- Open the config file in any text editor. (Notepad++ is used in the demonstration)



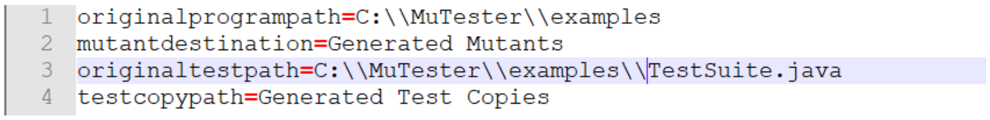
```
originalprogrampath=
mutantdestination=Generated Mutants
originaltestpath=
testcopypath=Generated Test Copies
originalclassname=
testclassname=
dependentclassname=
originalprogrampackagename=
testpackagename=
reportspath=reports\\html
testdependentclass(y/n)=
arithop(y/n)=
assignmentop(y/n)=
bitwiseop(y/n)=
conditionalop(y/n)=
incrementdecrementop(y/n)=
relationalop(y/n)=
shiftop(y/n)=
deletekilledmutants(y/n)=
```

- In the **originalprogrampath** field, enter the path of the program whose mutants you want to generate, **Eg: C:\\MuTester\\examples**



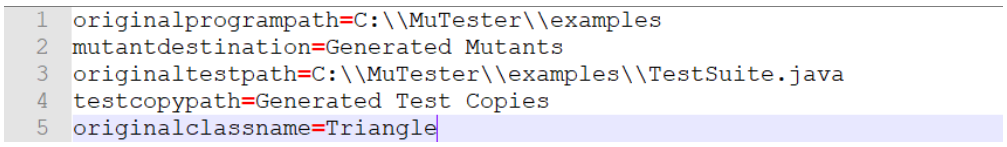
```
originalprogrampath=C:\\MuTester\\examples
mutantdestination=Generated Mutants
```

- In the **originaltestpath** field, enter the path of the test suite whose copies need to be generated to test the mutants, **Eg: C:\\MuTester\\examples\\TestSuite.java**



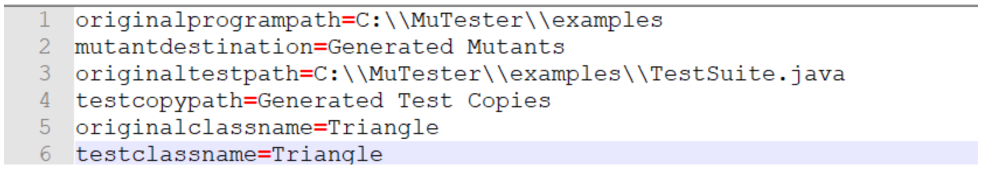
```
originalprogrampath=C:\\MuTester\\examples
mutantdestination=Generated Mutants
originaltestpath=C:\\MuTester\\examples\\TestSuite.java
testcopypath=Generated Test Copies
```

- In the **originalclassname** field, enter the name of the original class whose mutants are being generated, **Eg: Triangle**

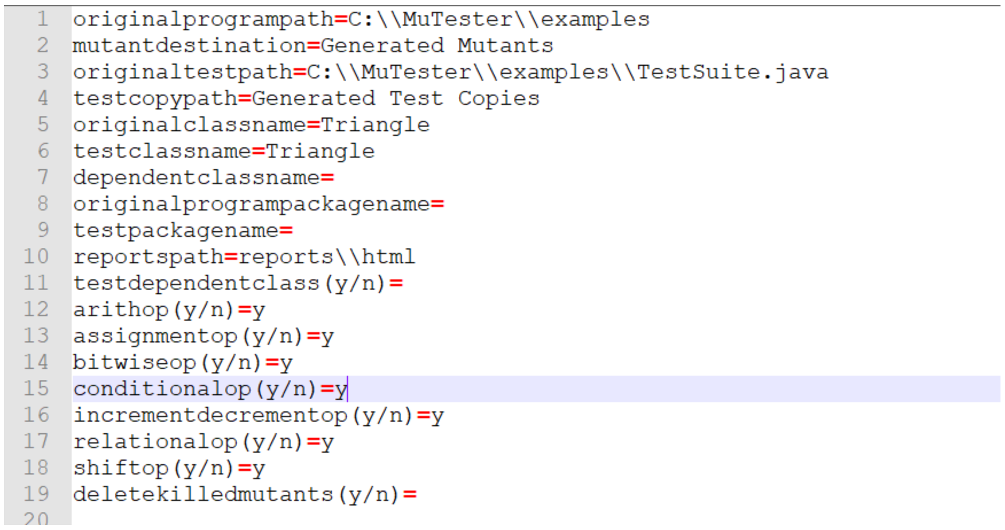


```
originalprogrampath=C:\\MuTester\\examples
mutantdestination=Generated Mutants
originaltestpath=C:\\MuTester\\examples\\TestSuite.java
testcopypath=Generated Test Copies
originalclassname=Triangle
```

- In the **testclassname** field, enter the name of the class that is being tested by the test suite, **Eg: Triangle**


```
originalprogrampath=C:\\MuTester\\examples
mutantdestination=Generated Mutants
originaltestpath=C:\\MuTester\\examples\\TestSuite.java
testcopypath=Generated Test Copies
originalclassname=Triangle
testclassname=Triangle
```

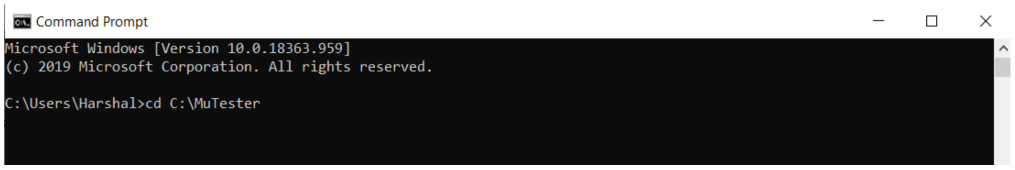
- In the **dependentclassname** field, enter the name of the dependent class if it is being tested by the test suite. In the demonstration, there is no dependent class, so this field is left blank.
- In the **originalprogrampackagename** field, enter the name of the package of the original program. In the demonstration, there is no dependent class, so this field is left blank.
- In the **testpackagename** field, enter the name of the package of the test suite. In the demonstration, there is no dependent class, so this field is left blank.
- In the **testdependentclass(y/n)** field, set the flag to **y** if the dependent class is being tested by the test suite. The default value for the field is **n.**
- In the **arithop(y/n)** field, set the flag to **n** if you do not want to apply the arithmetic operator mutation rule. If you enter the value as **y** then the arithmetic operator mutation rule will be applied.
- In the **assignmentop(y/n)** field, set the flag to **n** if you do not want to apply the assignment operator mutation rule. If you enter the value as **y** then the assignment operator mutation rule will be applied.
- In the **bitwiseop(y/n)** field, set the flag to **n** if you do not want to apply the bitwise operator mutation rule. If you enter the value as **y** then the bitwise operator mutation rule will be applied.
- In the **conditionalop(y/n)** field, set the flag to **n** if you do not want to apply the conditional operator mutation rule. If you enter the value as **y** then the conditional operator mutation rule will be applied.
- In the **incrementdecrementop(y/n)** field, set the flag to **n** if you do not want to apply the increment/decrement operator mutation rule. If you enter the value as **y** then the increment/decrement operator mutation rule will be applied.
- In the **relationalop(y/n)** field, set the flag to **n** if you do not want to apply the relational operator mutation rule. If you enter the value as **y** then the relational operator mutation rule will be applied.
- In the **shiftop(y/n)** field, set the flag to **n** if you do not want to apply the shift operator mutation rule. If you enter the value as **y** then the shift operator mutation rule will be applied.
- In the **deletekilledmutants(y/n)** field, set the flag to **y** if you want to delete the killed mutants and their respective test suites that killed these mutants from the Generated Mutants and Generated Test Copies folders. If the field is left blank, the flag is set to **n** and the deletion mechanism will not be implemented.



```
originalprogrampath=C:\\MuTester\\examples
mutantdestination=Generated Mutants
originaltestpath=C:\\MuTester\\examples\\TestSuite.java
testcopypath=Generated Test Copies
originalclassname=Triangle
testclassname=Triangle
dependentclassname=
originalprogrampackagename=
testpackagename=
reportspath=reports\\html
testdependentclass(y/n)=
arithop(y/n)=y
assignmentop(y/n)=y
bitwiseop(y/n)=y
conditionalop(y/n)=y
incrementdecrementop(y/n)=y
relationalop(y/n)=y
shiftop(y/n)=y
deletekilledmutants(y/n)=
```

b. Open the command prompt and navigate to the directory where you unzipped the MuTester tool.

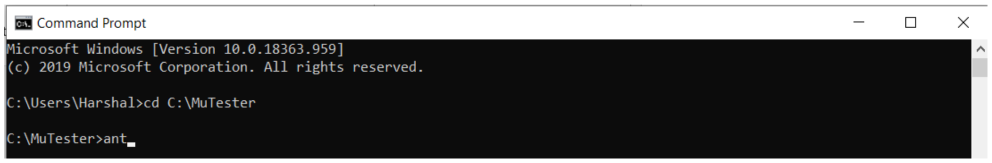
- **cd C:\MuTester**



```
Microsoft Windows [Version 10.0.18363.959]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Harshal>cd C:\MuTester
```

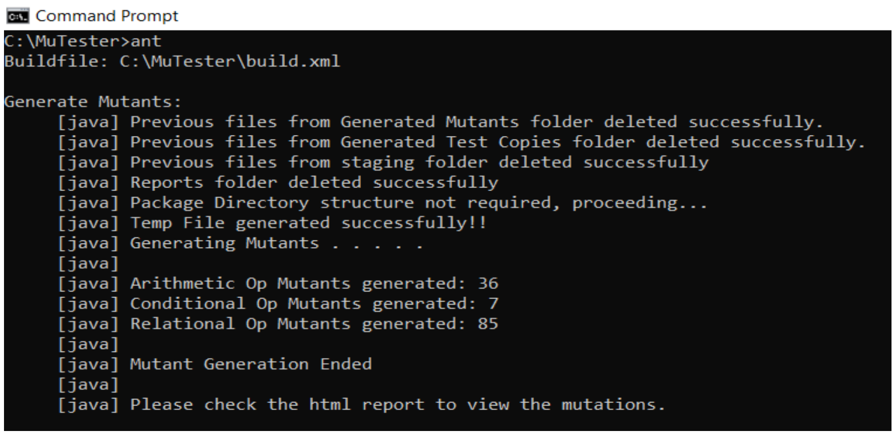
c. Next, we have to invoke the ANT script to run the tool. The entire process from generating mutants, testing each mutant and calculating the mutation score will be executed one after the other on the command prompt screen. On the command prompt type the below command:

- **ant**



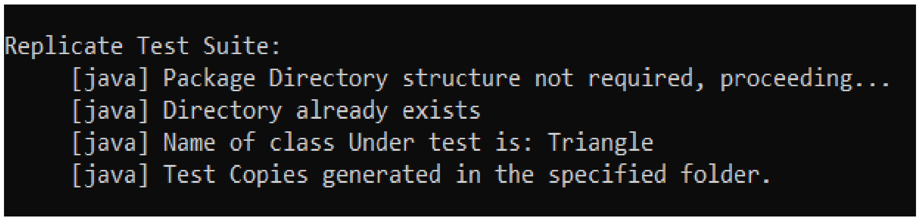
```
Microsoft Windows [Version 10.0.18363.959]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\Harshal>cd C:\MuTester

C:\MuTester>ant_
```

- You will see the below messages on the command prompt after running the script:
  - For Mutant Generation:



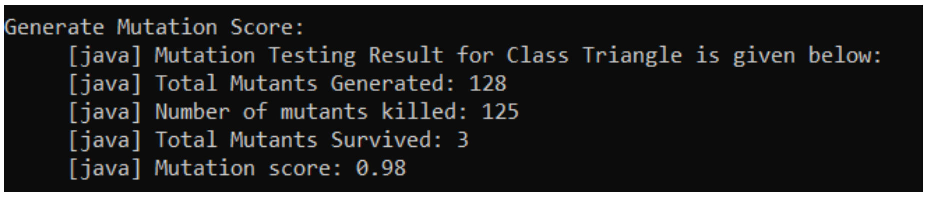
```
C:\MuTester>ant
Buildfile: C:\MuTester\build.xml

Generate Mutants:
     [java] Previous files from Generated Mutants folder deleted successfully.
     [java] Previous files from Generated Test Copies folder deleted successfully.
     [java] Previous files from staging folder deleted successfully
     [java] Reports folder deleted successfully
     [java] Package Directory structure not required, proceeding...
     [java] Temp File generated successfully!!
     [java] Generating Mutants . . . . .
     [java]
     [java] Arithmetic Op Mutants generated: 36
     [java] Conditional Op Mutants generated: 7
     [java] Relational Op Mutants generated: 85
     [java]
     [java] Mutant Generation Ended
     [java]
     [java] Please check the html report to view the mutations.
```

- For Test Suite copies:



```
Replicate Test Suite:
     [java] Package Directory structure not required, proceeding...
     [java] Directory already exists
     [java] Name of class Under test is: Triangle
     [java] Test Copies generated in the specified folder.
```

- For Mutation Score:



```
Generate Mutation Score:
     [java] Mutation Testing Result for Class Triangle is given below:
     [java] Total Mutants Generated: 128
     [java] Number of mutants killed: 125
     [java] Total Mutants Survived: 3
     [java] Mutation score: 0.98
```

Note:

- ✓ You can view the detailed test report in the **reports\html** folder.
- ✓ You can view the mutations for each mutated line in the **htmlreport.html** file.
- ✓ You can view the generated mutant programs in the Generated Mutants folder.
- ✓ You can view the generated test suite copies in the Generated Test Copies folder.

---END OF STEPS FOR WINDOWS OS---

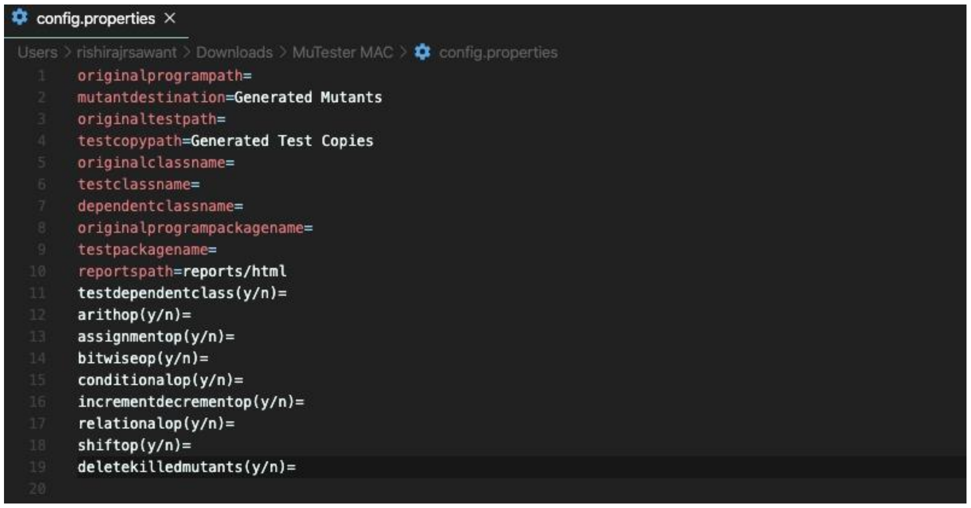
2) Steps for MAC OS:

Note:

✓ **In this manual, the program Triangle.java stored in the examples folder of the tool will be used for demonstration.**

a. The first step is to configure the user inputs in the config file.

- Open the config file in any text editor. (Using VSCode editor in the demo)

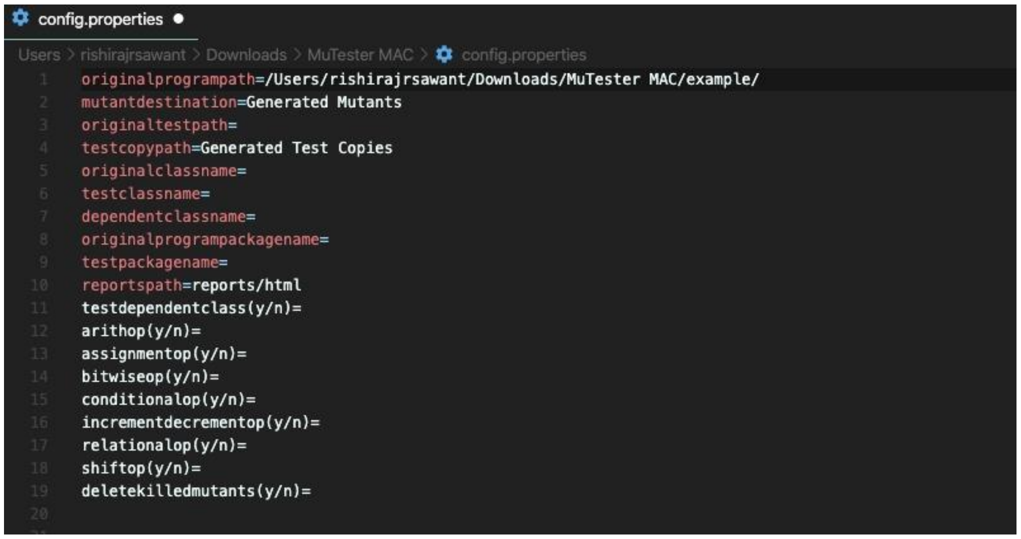


config.properties ×

Users > rishirajrsawant > Downloads > MuTester MAC > config.properties

```
originalprogrampath=
mutantdestination=Generated Mutants
originaltestpath=
testcopypath=Generated Test Copies
originalclassname=
testclassname=
dependentclassname=
originalprogrampackagename=
testpackagename=
reportspath=reports/html
testdependentclass(y/n)=
arithop(y/n)=
assignmentop(y/n)=
bitwiseop(y/n)=
conditionalop(y/n)=
incrementdecrementop(y/n)=
relationalop(y/n)=
shiftop(y/n)=
deletekilledmutants(y/n)=
```

- In the **originalprogrampath** field, enter the path of the program whose mutants you want to generate, **Eg: /Users/rishirajrsawant/Downloads/MuTester MAC/example/**

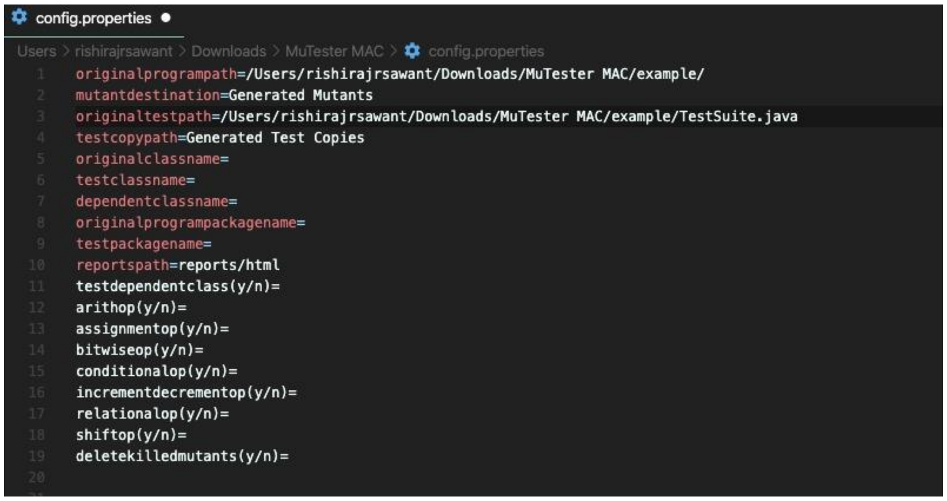


config.properties ●

Users > rishirajrsawant > Downloads > MuTester MAC > config.properties

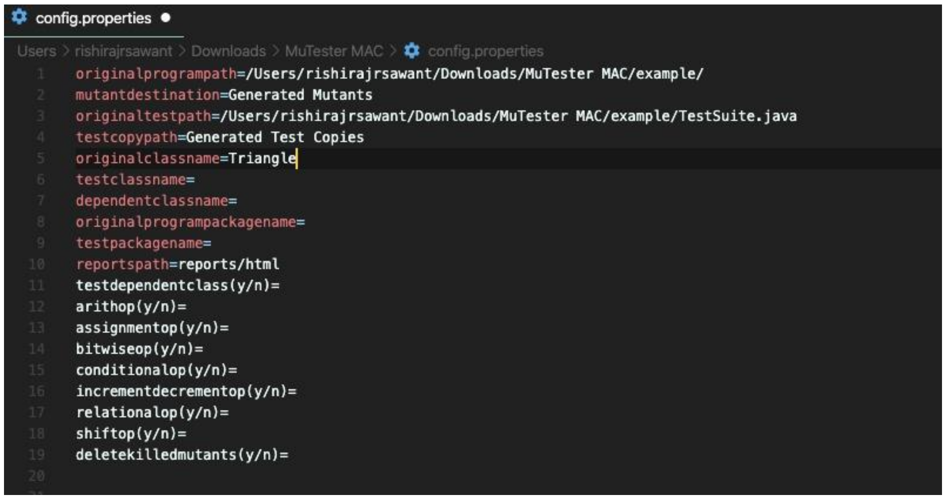
```
originalprogrampath=/Users/rishirajrsawant/Downloads/MuTester MAC/example/
mutantdestination=Generated Mutants
originaltestpath=
testcopypath=Generated Test Copies
originalclassname=
testclassname=
dependentclassname=
originalprogrampackagename=
testpackagename=
reportspath=reports/html
testdependentclass(y/n)=
arithop(y/n)=
assignmentop(y/n)=
bitwiseop(y/n)=
conditionalop(y/n)=
incrementdecrementop(y/n)=
relationalop(y/n)=
shiftop(y/n)=
deletekilledmutants(y/n)=
```

- In the **originaltestpath** field, enter the path of the test suite whose copies need to be generated to test the mutants, **Eg: /Users/rishirajrsawant/Downloads/MuTester MAC/example/TestSuite.java**


config.properties
Users > rishirajrsawant > Downloads > MuTester MAC > config.properties

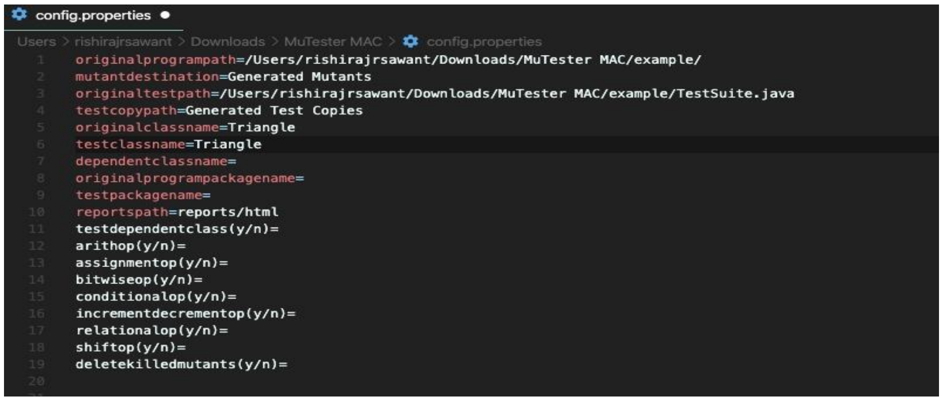
```
originalprogrampath=/Users/rishirajrsawant/Downloads/MuTester MAC/example/
mutantdestination=Generated Mutants
originaltestpath=/Users/rishirajrsawant/Downloads/MuTester MAC/example/TestSuite.java
testcopypath=Generated Test Copies
originalclassname=
testclassname=
dependentclassname=
originalprogrampackagename=
testpackagename=
reportspath=reports/html
testdependentclass(y/n)=
arithop(y/n)=
assignmentop(y/n)=
bitwiseop(y/n)=
conditionalop(y/n)=
incrementdecrementop(y/n)=
relationalop(y/n)=
shiftop(y/n)=
deletekilledmutants(y/n)=
```

- In the **originalclassname** field, enter the name of the original class whose mutants are being generated, **Eg: Triangle**


config.properties
Users > rishirajrsawant > Downloads > MuTester MAC > config.properties

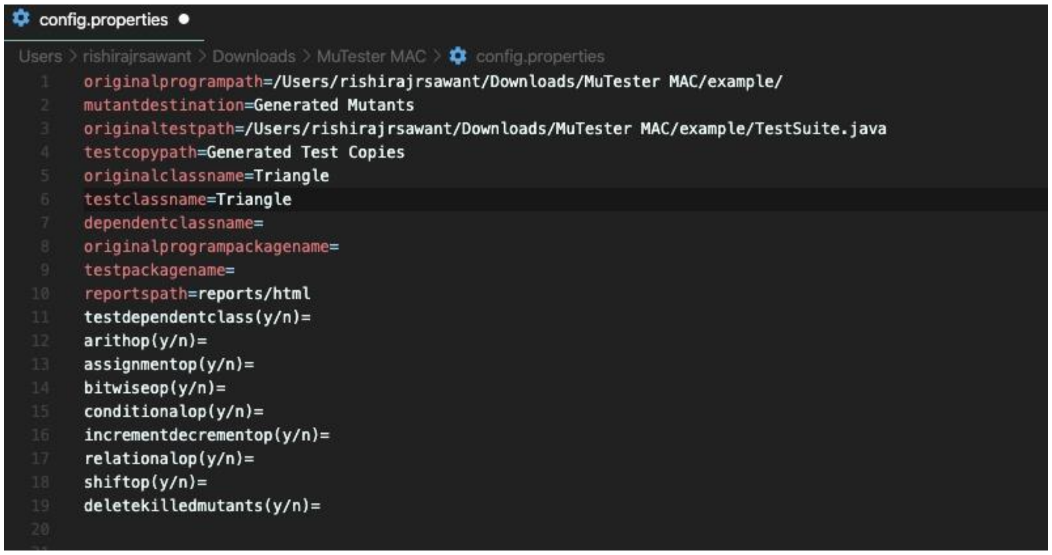
```
originalprogrampath=/Users/rishirajrsawant/Downloads/MuTester MAC/example/
mutantdestination=Generated Mutants
originaltestpath=/Users/rishirajrsawant/Downloads/MuTester MAC/example/TestSuite.java
testcopypath=Generated Test Copies
originalclassname=Triangle
testclassname=
dependentclassname=
originalprogrampackagename=
testpackagename=
reportspath=reports/html
testdependentclass(y/n)=
arithop(y/n)=
assignmentop(y/n)=
bitwiseop(y/n)=
conditionalop(y/n)=
incrementdecrementop(y/n)=
relationalop(y/n)=
shiftop(y/n)=
deletekilledmutants(y/n)=
```

- In the **testclassname** field, enter the name of the class that is being tested by the test suite, **Eg: Triangle**


config.properties
Users > rishirajrsawant > Downloads > MuTester MAC > config.properties

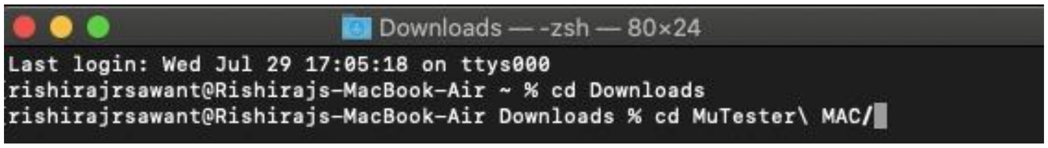
```
originalprogrampath=/Users/rishirajrsawant/Downloads/MuTester MAC/example/
mutantdestination=Generated Mutants
originaltestpath=/Users/rishirajrsawant/Downloads/MuTester MAC/example/TestSuite.java
testcopypath=Generated Test Copies
originalclassname=Triangle
testclassname=Triangle
dependentclassname=
originalprogrampackagename=
testpackagename=
reportspath=reports/html
testdependentclass(y/n)=
arithop(y/n)=
assignmentop(y/n)=
bitwiseop(y/n)=
conditionalop(y/n)=
incrementdecrementop(y/n)=
relationalop(y/n)=
shiftop(y/n)=
deletekilledmutants(y/n)=
```

- In the **dependentclassname** field, enter the name of the dependent class if it is being tested by the test suite. In the demonstration, there is no dependent class, so this field is left blank.
- In the **originalprogrampackagename** field, enter the name of the package of the original program. In the demonstration, there is no dependent class, so this field is left blank.
- In the **testpackagename** field, enter the name of the package of the test suite. In the demonstration, there is no dependent class, so this field is left blank.
- In the **testdependentclass(y/n)** field, set the flag to **y** if the dependent class is being tested by the test suite. The default value for the field is **n.**
- In the **arithop(y/n)** field, set the flag to **n** if you do not want to apply the arithmetic operator mutation rule. If you enter the value as **y** then the arithmetic operator mutation rule will be applied.
- In the **assignmentop(y/n)** field, set the flag to **n** if you do not want to apply the assignment operator mutation rule. If you enter the value as **y** then the assignment operator mutation rule will be applied.
- In the **bitwiseop(y/n)** field, set the flag to **n** if you do not want to apply the bitwise operator mutation rule. If you enter the value as **y** then the bitwise operator mutation rule will be applied.
- In the **conditionalop(y/n)** field, set the flag to **n** if you do not want to apply the conditional operator mutation rule. If you enter the value as **y** then the conditional operator mutation rule will be applied.
- In the **incrementdecrementop(y/n)** field, set the flag to **n** if you do not want to apply the increment/decrement operator mutation rule. If you enter the value as **y** then the increment/decrement operator mutation rule will be applied.
- In the **relationalop(y/n)** field, set the flag to **n** if you do not want to apply the relational operator mutation rule. If you enter the value as **y** then the relational operator mutation rule will be applied.
- In the **shiftop(y/n)** field, set the flag to **n** if you do not want to apply the shift operator mutation rule. If you enter the value as **y** then the shift operator mutation rule will be applied.
- In the **deletekilledmutants(y/n)** field, set the flag to **y** if you want to delete the killed mutants and their respective test suites that killed these mutants from the Generated Mutants and Generated Test Copies folders. If the field is left blank, the flag is set to **n** and the deletion mechanism will not be implemented.


config.properties
Users > rishirajrsawant > Downloads > MuTester MAC > config.properties

```
originalprogrampath=/Users/rishirajrsawant/Downloads/MuTester MAC/example/
mutantdestination=Generated Mutants
originaltestpath=/Users/rishirajrsawant/Downloads/MuTester MAC/example/TestSuite.java
testcopypath=Generated Test Copies
originalclassname=Triangle
testclassname=Triangle
dependentclassname=
originalprogrampackagename=
testpackagename=
reportspath=reports/html
testdependentclass(y/n)=
arithop(y/n)=
assignmentop(y/n)=
bitwiseop(y/n)=
conditionalop(y/n)=
incrementdecrementop(y/n)=
relationalop(y/n)=
shiftop(y/n)=
deletekilledmutants(y/n)=
```

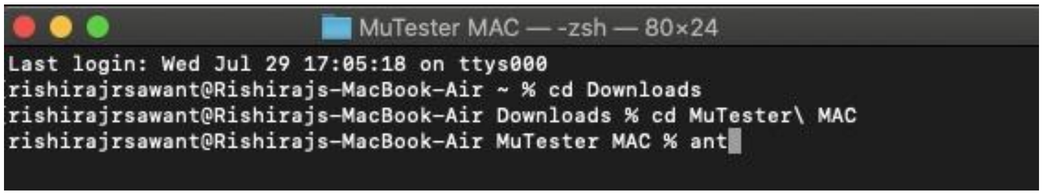
b. Open the command prompt and navigate to the directory where you unzipped the MuTester tool.



```
Last login: Wed Jul 29 17:05:18 on ttys000
rishirajrsawant@Rishirajs-MacBook-Air ~ % cd Downloads
rishirajrsawant@Rishirajs-MacBook-Air Downloads % cd MuTester\ MAC/
```

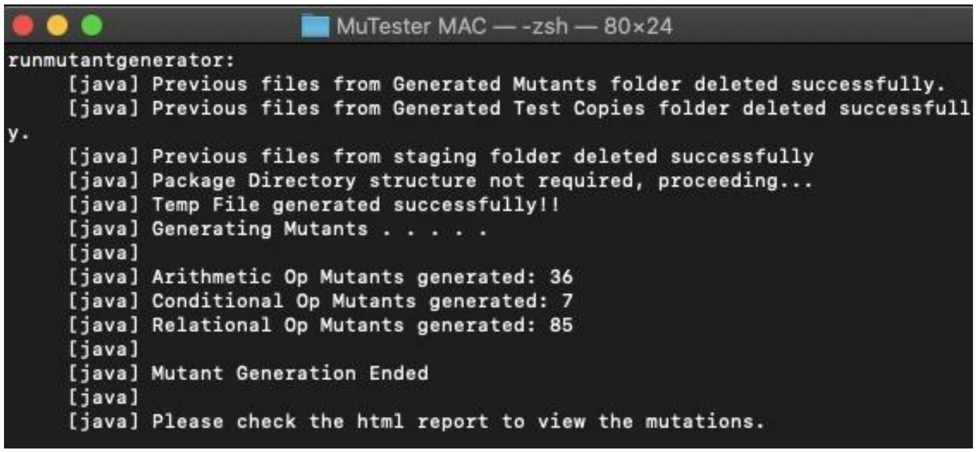
c. Next, we have to invoke the ANT script to run the tool. The entire process from generating mutants, testing each mutant and calculating the mutation score will be executed one after the other on the command prompt screen. On the command prompt type the below command:

- **ant**



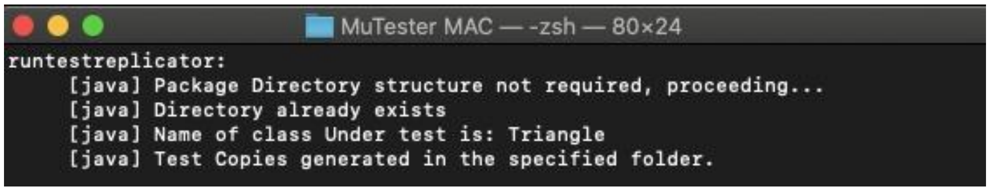
```
Last login: Wed Jul 29 17:05:18 on ttys000
rishirajrsawant@Rishirajs-MacBook-Air ~ % cd Downloads
rishirajrsawant@Rishirajs-MacBook-Air Downloads % cd MuTester\ MAC
rishirajrsawant@Rishirajs-MacBook-Air MuTester MAC % ant
```

- You will see the below messages on the command prompt after running the script:
  - For Mutant Generation:



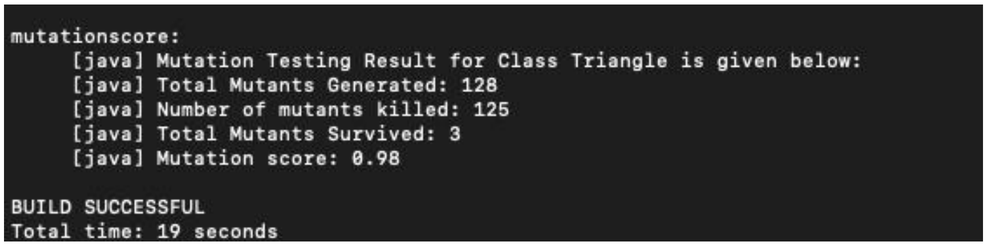
```
runmutantgenerator:
     [java] Previous files from Generated Mutants folder deleted successfully.
     [java] Previous files from Generated Test Copies folder deleted successfull
y.
     [java] Previous files from staging folder deleted successfully
     [java] Package Directory structure not required, proceeding...
     [java] Temp File generated successfully!!
     [java] Generating Mutants . . . . .
     [java]
     [java] Arithmetic Op Mutants generated: 36
     [java] Conditional Op Mutants generated: 7
     [java] Relational Op Mutants generated: 85
     [java]
     [java] Mutant Generation Ended
     [java]
     [java] Please check the html report to view the mutations.
```

  - For Test Suite copies:



```
runtestreplicator:
     [java] Package Directory structure not required, proceeding...
     [java] Directory already exists
     [java] Name of class Under test is: Triangle
     [java] Test Copies generated in the specified folder.
```

- For Mutation Score:



```
mutationscore:
     [java] Mutation Testing Result for Class Triangle is given below:
     [java] Total Mutants Generated: 128
     [java] Number of mutants killed: 125
     [java] Total Mutants Survived: 3
     [java] Mutation score: 0.98

BUILD SUCCESSFUL
Total time: 19 seconds
```

Note:

- ✓ You can view the detailed test report in the **reports/html** folder.
- ✓ You can view the mutations for each mutated line in the **htmlreport.html** file.
- ✓ You can view the generated mutant programs in the Generated Mutants folder.
- ✓ You can view the generated test suite copies in the Generated Test Copies folder.

---END OF MANUAL--