

For Windows OS and Linux (e.g. Ubuntu,...):

Step 1: Download the Algorithms.rar file from the link and extract it

Step 2:

+ For Windows: Go to the windows start menu and open the **Command Prompt application**.

+ For Linux: Open **Terminal application**

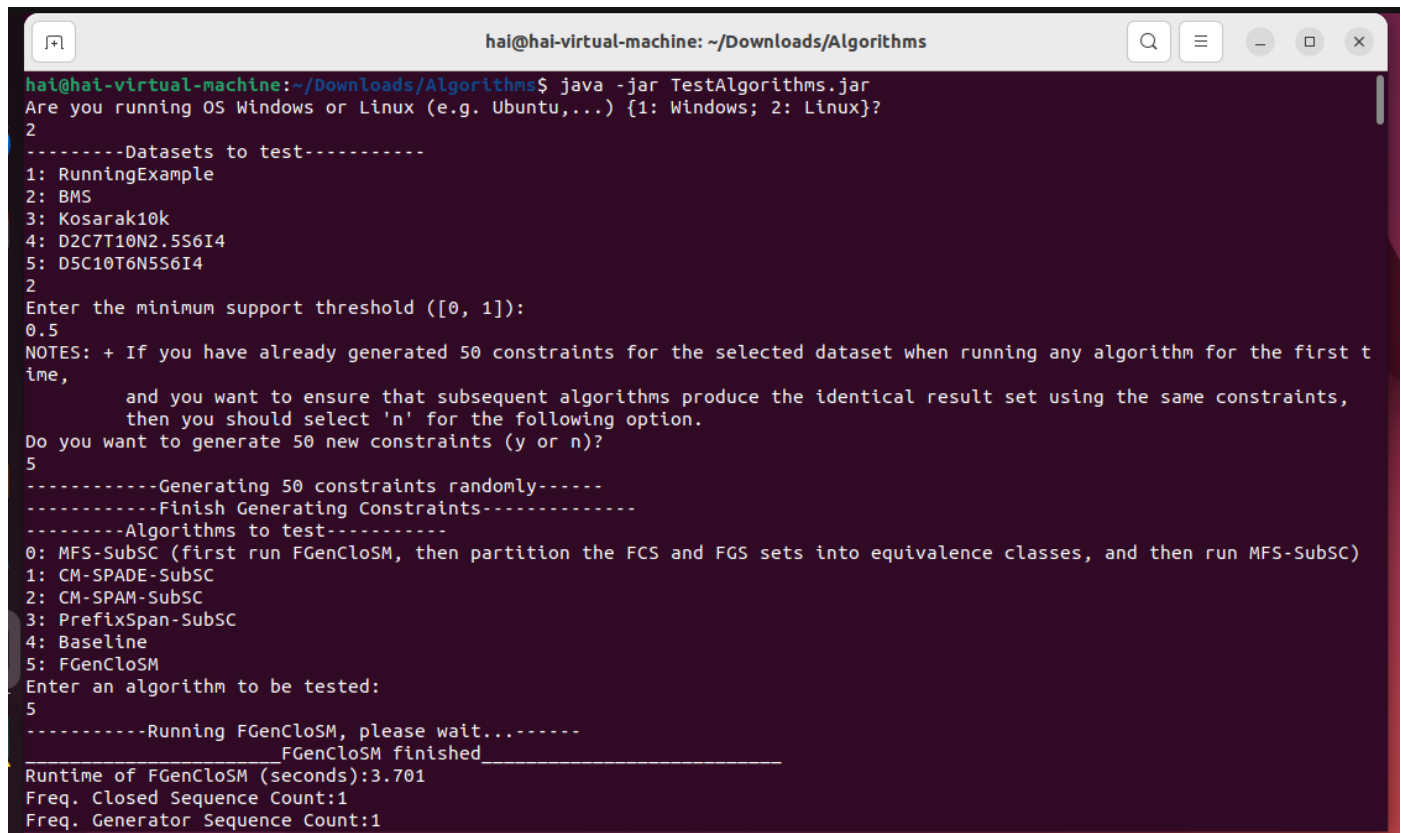
Step 3: Go to the extracted Algorithms folder and type the following command:

```
java -jar TestAlgorithms.jar
```

Notes: To run the program for the low values of *minsup* on datasets (e.g. $\text{minsup} \leq 0.059\%$ on the BMS dataset,...), you should make sure that the heap memory of the program is at least 12GB.

Below are two examples to show how to run the program:

1) The first example of running the program on **OS Ubuntu 22** as follows:



```
hai@hai-virtual-machine: ~/Downloads/Algorithms
hai@hai-virtual-machine:~/Downloads/Algorithms$ java -jar TestAlgorithms.jar
Are you running OS Windows or Linux (e.g. Ubuntu,...) {1: Windows; 2: Linux}?
2
-----Datasets to test-----
1: RunningExample
2: BMS
3: Kosarak10k
4: D2C7T10N2.5S6I4
5: D5C10T6N5S6I4
2
Enter the minimum support threshold ([0, 1]):
0.5
NOTES: + If you have already generated 50 constraints for the selected dataset when running any algorithm for the first time,
        and you want to ensure that subsequent algorithms produce the identical result set using the same constraints,
        then you should select 'n' for the following option.
Do you want to generate 50 new constraints (y or n)?
5
-----Generating 50 constraints randomly-----
-----Finish Generating Constraints-----
-----Algorithms to test-----
0: MFS-SubSC (first run FGenCloSM, then partition the FCS and FGS sets into equivalence classes, and then run MFS-SubSC)
1: CM-SPADE-SubSC
2: CM-SPAM-SubSC
3: PrefixSpan-SubSC
4: Baseline
5: FGenCloSM
Enter an algorithm to be tested:
5
-----Running FGenCloSM, please wait...-----
FGenCloSM finished
Runtime of FGenCloSM (seconds):3.701
Freq. Closed Sequence Count:1
Freq. Generator Sequence Count:1
```

2) The second example of running the program on **OS Ubuntu 22** as follows:

```
hai@hai-virtual-machine: ~/Downloads/Algorithms
hai@hai-virtual-machine:~/Downloads/Algorithms$ java -jar TestAlgorithms.jar
Are you running OS Windows or Linux (e.g. Ubuntu,...) {1: Windows; 2: Linux}?
2
-----Datasets to test-----
1: RunningExample
2: BMS
3: Kosarak10k
4: D2C7T10N2.5S6I4
5: D5C10T6N5S6I4
2
Enter the minimum support threshold ([0, 1]):
0.5
NOTES: + If you have already generated 50 constraints for the selected dataset when running any algorithm for the first time,
        and you want to ensure that subsequent algorithms produce the identical result set using the same constraints,
        then you should select 'n' for the following option.
Do you want to generate 50 new constraints (y or n)?
y
-----Generating 50 constraints randomly-----
-----Finish Generating Constraints-----
-----Algorithms to test-----
0: MFS-SubSC (first run FGenCloSM, then partition the FCS and FGS sets into equivalence classes, and then run MFS-SubSC)
1: CM-SPADE-SubSC
2: CM-SPAM-SubSC
3: PrefixSpan-SubSC
4: Baseline
5: FGenCloSM
Enter an algorithm to be tested:
0
-----Running FGenCloSM, please wait.-----
FGenCloSM finished
Runtime of FGenCloSM (seconds):1.987
Freq. Closed Sequence Count:1
Freq. Generator Sequence Count:1
```

```
-----The program is partitioning FCS and FGS into equivalence classes, please wait.-----
Runtime for partitioning FCS and FGS into equivalence Classes (seconds):57.817
Finish partitioning
-----Testing the constraint sequence 1 per 50 constraints-----
-----MFS-SubSC-----
Constraint sequence = MC: {18427}
Runtime for discovering frequent sequences with this constraint: 0.009 (seconds)
Memory usage for discovering frequent sequences with this constraint: 0.0 (MB)
Number of frequent sequences statisfying this constraint: 0

-----Testing the constraint sequence 2 per 50 constraints-----
-----MFS-SubSC-----
Constraint sequence = MC: {18659}
Runtime for discovering frequent sequences with this constraint: 0.001 (seconds)
Memory usage for discovering frequent sequences with this constraint: 0.0 (MB)
Number of frequent sequences statisfying this constraint: 0

-----Testing the constraint sequence 3 per 50 constraints-----
-----MFS-SubSC-----
Constraint sequence = MC: {18635}
Runtime for discovering frequent sequences with this constraint: 0.001 (seconds)
Memory usage for discovering frequent sequences with this constraint: 0.0 (MB)
Number of frequent sequences statisfying this constraint: 0
```

.....

-----Testing the constraint sequence 49 per 50 constraints-----

-----MFS-SubSC-----

Constraint sequence = MC: {12743}

Runtime for discovering frequent sequences with this constraint: 0.001 (seconds)

Memory usage for discovering frequent sequences with this constraint: 0.0 (MB)

Number of frequent sequences satisfying this constraint: 0

-----Testing the constraint sequence 50 per 50 constraints-----

-----MFS-SubSC-----

Constraint sequence = MC: {12603}

Runtime for discovering frequent sequences with this constraint: 0.0 (seconds)

Memory usage for discovering frequent sequences with this constraint: 0.0 (MB)

Number of frequent sequences satisfying this constraint: 0

-----Statistical information for 50 constraint sequences-----

(a) Dataset	(b) Algorithm	(c) Total Runtime (second)	(d) Average Runtime (second)	(e) Peak Memory (MB)
-------------	---------------	----------------------------	------------------------------	----------------------

(a) BMS	(b) FGenCloSM + Partition + MFS-SubSC	(c) 59.865	(d) 1.197	(e) 0.0
---------	---------------------------------------	------------	-----------	---------

Press any key to finish

☐