How to execute the code:

We have created an abstract class named “freqeuntPatterns” inside the “abstractFrequentPatterns.py” python file. Therefore, every program has to import this file and needs to extend the abstract class as follows:

- ***from traditional.abstractClass.abstractFrequentPatterns import ****
- ***class Fpgrowth(frequentPatterns):***
  - ***Complete code along with the implementation of the given abstract methods and variables available in the abstract class ‘frequentPatterns’.***

1. **Frequent Pattern Mining (FPM) Process:**
   1.1. Import our package and initialize the method called '**Fpgrowth**' using the input file path/input file and minimum support (It has to be given in terms of count of total number of transactions in the input database/file).
   1.2. Then call the method '**startMine**' using the following command

```
import Fpgrowth as Myap
fp= Myap.Fpgrowth(r"filepath or filename", minimum
      support)
fp.startMine()
```

output is displayed as follows:

- Frequent patterns were generated successfully using Fpgrowth algorithm.

For example:
If we execute the following command:

```
import Fpgrowth as Myap
fp = Myap.Fpgrowth(r" transactional_T10I4D100K.csv", 1000)
fp.startMine()
```

output is displayed as follows:

- Frequent patterns were generated successfully using Fpgrowth algorithm.

2. To get the frequent patterns along with their support count:
   2.1. Complete the FPM Process mentioned in **(1)**
   2.2. Then call the method **'getFrequentPatterns'** using the following command:

```
import Fpgrowthas Myap
fp= Myap.Fpgrowth(r"filepath or filename", minimum
      support)
fp.startMine()
variable = fp.getFrequentPatterns()
```

   output is displayed as follows:
   - Frequent patterns were generated successfully using Fpgrowth algorithm.
   - All the Frequent patterns will be stored in a dictionary, with patterns as keys and support count as value and returned to the called function.

   For example:
   If we execute the following command:

```
import Fpgrowth as Myap
fp = Myap.Fpgrowth(r" transactional_T10I4D100K.csv", 1000)
fp.startMine()
frequentPatterns = fp.getFrequentPatterns()
```

   output is displayed as follows:
   - Frequent patterns were generated successfully using Fpgrowth algorithm.
   - All the Frequent patterns will be stored in a dictionary, with patterns as keys and support count as value and assigned to the variable called **'frequentPatterns.'**

3. To get the frequent patterns along with their support count in a file:
   3.1. Complete the FPM Process mentioned in **(1)**
   3.2. Then call the method **'storePatternsInFile'** using the following command:

```
import Fpgrowth as Myap
fp= Myap.Fprgowth(r"filepath or filename", minimum
      support)
fp.startMine()
fp.storePatternsInFile("output file")
```

output is displayed as follows:

- Frequent patterns were generated successfully using Fpgrowth algorithm.
- All the Frequent patterns will be stored in a file named as “output file"

For example:
If we execute the following command:

```
import Fpgrowth as Myap
fp = Myap.Fpgrowth(r" transactional_T10I4D100K.csv", 1000)
fp.startMine()
fp.storePatternsInFile("sampleoutput")
```

output is displayed as follows:

- Frequent patterns were generated successfully using Fpgrowth algorithm.
- All the Frequent patterns will be stored in a file named as ‘sampleoutput.’

4. To get the frequent patterns along with their support count in a DataFrame:
   4.1. Complete the FPM Process mentioned in **(1)**
   4.2. Then call the method **'getPatternsInDataFrame'** using the following command:

```
import Fpgrowth as Myap
fp = Myap.Fpgrowth(r"filepath or filename", minimum
      support)
fp.startMine()
variable =fp.getPatternsInDataFrame()
```

output is displayed as follows:

- Frequent patterns were generated successfully using fpgrowth algorithm.
- All the Frequent patterns will be stored in a data frame, their columns named as 'Patterns' and 'Support' and returned to the called function.

For example:
If we execute the following command:

```
import Fpgrowth as Myap
fp = Myap.Fpgrowth(r" transactional_T10I4D100K.csv", 1000)
fp.startMine()
dataFrame= fp.getPatternsInDataFrame()
```

output is displayed as follows:

- Frequent patterns were generated successfully using fpgrowth algorithm.
- All the Frequent patterns will be stored in a data frame, their columns named as 'Patterns' and 'Support' and stored in a variable called 'dataFrame.'

5. If we want to know the amount of USS memory consumed by the fpgrowth algorithm:
   5.1. Complete the FPM Process mentioned in **(1)**
   5.2. Then call the method **'getMemoryUSS'** using the following command:

```
import Fpgrowth  as Myap
fp = Myap.Fpgrowth(r"filepath or filename", minimum
      support)
fp.startMine()
variable = fp.getMemoryUSS()
```

   output is displayed as follows:

   - Frequent patterns were generated successfully using Fpgrowth algorithm.
   - Total amount of USS memory consumed by the program will be computed and returned to the called function.

   For example:
   If we execute the following command:

```
import Fpgrowth as Myap
fp= Myap.Fpgrowth(r" transactional_T10I4D100K.csv", 1000)
fp.startMine()
memoryUSS = fp.getMemoryUSS()
```

   output is displayed as follows:

   - Frequent patterns were generated successfully using fpgrowth algorithm.
   - Total amount of USS memory consumed by the program will be computed and returned to the variable called ' **memoryUSS.**'

6. If we want to know the amount of RSS memory consumed by the fpgrowth algorithm:
   6.1. Complete the FPM Process mentioned in **(1)**
   6.2. Then call the method **'getMemoryRSS'** using the following command:

```
import Fpgrowth as Myap
fp = Myap.Fpgrowth(r"filepath or filename", minimum
      support)
```

fp.startMine()
variable = fp.getMemoryRSS()

output is displayed as follows:

- Frequent patterns were generated successfully using Fpgrowth algorithm.
- Total amount of RSS memory consumed by the program will be computed and returned to the called function.

For example:
If we execute the following command:
import Fpgrowth as Myap
fp = Myap.Fpgrowth(r" transactional_T10I4D100K.csv", 1000)
fp.startMine()
memoryRSS = fp.getMemoryRSS()

output is displayed as follows:

- Frequent patterns were generated successfully using Fpgrowth algorithm.
- Total amount of RSS memory consumed by the program will be computed and returned to the variable called ' **memoryRSS.**'

7. If we want to know the runtime taken by the fpgrowth algorithm created by us:
   7.1. Complete the FPM Process mentioned in **(1)**
   7.2. Then call the method **'getRuntime'** using the following command:

import Fpgrowth as Myap
fp = Myap.Fpgrowth(r"filepath or filename",
minimumsupport)
fp.startMine()
variable = fp.getRuntime()

output is displayed as follows:

- Frequent patterns were generated successfully using Fpgrowth algorithm.
- Total runtime taken by the program in seconds will be computed and returned to the called function.

For example:
If we execute the following command:
import Fpgrowth as Myap
fp= Myap.Fpgrowth(r" transactional_T10I4D100K.csv", 1000)
fp.startMine()
run = fpgetRuntime()

output is displayed as follows:

- Frequent patterns were generated successfully using Fpgrowth algorithm.
- Total runtime taken by the program in seconds will be computed and returned to the variable called 'run.'