Interfaces In Python:

In general if an abstract class contains only abstract methods such type of abstract class is considered as interface.

Demo program:

```
1)  from abc import *
2)  class DBInterface(ABC):
3)      @abstractmethod
4)      def connect(self):pass
5)
6)      @abstractmethod
7)      def disconnect(self):pass
8)
9)  class Oracle(DBInterface):
10)     def connect(self):
11)         print('Connecting to Oracle Database...')
12)     def disconnect(self):
13)         print('Disconnecting to Oracle Database...')
14)
15) class Sybase(DBInterface):
16)     def connect(self):
17)         print('Connecting to Sybase Database...')
18)     def disconnect(self):
19)         print('Disconnecting to Sybase Database...')
20)
21) dbname=input('Enter Database Name:')
22) classname=globals()[dbname]
23) x=classname()
24) x.connect()
25) x.disconnect()
```

```
D:\durga_classes>py test.py
Enter Database Name:Oracle
Connecting to Oracle Database...
Disconnecting to Oracle Database...

D:\durga_classes>py test.py
Enter Database Name:Sybase
Connecting to Sybase Database...
Disconnecting to Sybase Database...
```

Note: **The inbuilt function globals()[str] converts the string 'str' into a class name and returns the classname.**