```
7)  logging.error('error Information')
8)  logging.critical('critical Information')
```

log.txt:

```
DEBUG:root:Debug Information
INFO:root:info Information
WARNING:root:warning Information
ERROR:root:error Information
CRITICAL:root:critical Information
```

How to configure log file in over writing mode:

In the above program by default data will be appended to the log file.i.e append is the default mode. Instead of appending if we want to over write data then we have to use filemode property.

```
logging.basicConfig(filename='log786.txt',level=logging.WARNING)
```
meant for appending

```
logging.basicConfig(filename='log786.txt',level=logging.WARNING,filemode='a')
```
explicitly we are specifying appending.

```
logging.basicConfig(filename='log786.txt',level=logging.WARNING,filemode='w')
```
meant for over writing of previous data.

Note:

```
logging.basicConfig(filename='log.txt',level=logging.DEBUG)
```

If we are not specifying level then the default level is WARNING(30)

If we are not specifying file name then the messages will be printed to the console.

test.py:

```
1)  import logging
2)  logging.basicConfig()
3)  print('Logging Demo')
4)  logging.debug('Debug Information')
5)  logging.info('info Information')
6)  logging.warning('warning Information')
7)  logging.error('error Information')
8)  logging.critical('critical Information')
```

```
D:\durgaclasses>py test.py
Logging Demo
WARNING:root:warning Information
ERROR:root:error Information
CRITICAL:root:critical Information
```