```
# Python program to
# demonstrate private members

# Creating a Base class


class Base:
  def __init__(self):
      self.a = "GeeksforGeeks"
      self.__c = "GeeksforGeeks"

# Creating a derived class
class Derived(Base):
  def __init__(self):

      # Calling constructor of
      # Base class
      Base.__init__(self)
      print("Calling private member of base class: ")
      print(self.__c)


# Driver code
obj1 = Base()
print(obj1.a)

# Python program to
# demonstrate protected members

# Creating a base class
class Base:
  def __init__(self):

      # Protected member
      self._a = 2

# Creating a derived class
class Derived(Base):
  def __init__(self):

      # Calling constructor of
      # Base class
      Base.__init__(self)
      print("Calling protected member of base class: ",
          self._a)

      # Modify the protected variable:
      self._a = 3
      print("Calling modified protected member outside class: ",
          self._a)


obj1 = Derived()
```

```python
obj2 = Base()

# Calling protected member
# Can be accessed but should not be done due to convention
print("Accessing protected member of obj1: ", obj1._a)

# Accessing the protected variable outside
print("Accessing protected member of obj2: ", obj2._a)
```