第 6 章作业

1. 封装，继承，多态

2. __pow__(),__floordiv__()

3. a._A__value=3

5.

```
class Vector_3D:
    def __init__(self, x, y, z):
        self.__x = x
        self.__y = y
        self.__z = z

    def add(self, other):
        x = self.__x + other.__x
        y = self.__y + other.__y
        z = self.__z + other.__z
        return Vector_3D(x, y, z)

    def Minus(self, other):
        x = self.__x - other.__x
        y = self.__y - other.__y
        z = self.__z - other.__z
        return Vector_3D(x, y, z)

    def Multiply(self, a):
        x = self.__x * a
        y = self.__y * a
        z = self.__z * a
        return Vector_3D(x, y, z)

    def Except(self, a):
        x = self.__x / a
        y = self.__y / a
        z = self.__z / a
        return Vector_3D(x, y, z)

    def show(self):
        print('X:{0}, Y:{1}, Z:{2}'.format(self.__x,self.__y,self.__z))

    def lenght(self):
```

```
        return (self.__x ** 2 + self.__y ** 2 + self.__z ** 2) ** 0.5


b = Vector_3D(1, 2, 3)
b1 = b.Multiply(4)
b1.show()

b2 = b1.add(b)
b2.show()
print(b2.lenght())
```

6.

在面向对象编程中，以两个下画线开始的成员为私有成员，这样的成员不建议在类的外部直接访问，虽然可以通过“对象名._类名__私有成员名”的形式访问。类的定义中前后各有两个下画线的成员表示特殊成员，这样的成员一般与某个运算符或内置函数对应

7.

```
class SecInfo(object):
    def __init__(self, my_set):
        self.sett = my_set

    def add_setinfo(self, keyname):
        self.sett.add(keyname)
        return self.sett

    def get_intersection(self, unioninfo):
        if isinstance(unioninfo, set):
            return self.sett & unioninfo
        else:
            return "你传入的不是 set"
    def get_union(self, unioninfo):
        return self.sett |   unioninfo

    def del_difference(self, unioninfo):
        return self.sett - unioninfo
A = set([1,2,3,4,5])
B = set([5,6,3])
#my_set = SecInfo((1,2,3,4,5))
#print(my_set.add_setinfo(6))
my_set = SecInfo(A)
print(my_set.get_intersection(B))
print(my_set.del_difference(B))
```

8.

```
class Deque(object):
    def __init__(self):
        self.items = []

    def isEmpty(self):
        return self.items == []

    def addFront(self, item):
        self.items.append(item)

    def removeFront(self):
        return self.items.pop()

    def addRear(self, item):
        self.items.insert(0, item)

    def removeRear(self):
        return self.items.pop(0)

    def size(self):
        return len(self.items)
```